NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MODEL-BASED UAS-UGS IED CLEARANCE MISSION ENGINEERING

by

Robert J. Naquila

September 2021

Thesis Advisor: Oleg A. Yakimenko
Second Reader: Fotis A. Papoulias

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2021	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
MODEL-BASED UAS-UGS IED CLEARANCE MISSION ENGINEERING

5. FUNDING NUMBERS

6. AUTHOR(S) Robert J. Naquila

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

Machine-to-machine teaming is the trend of future warfare aiming at maximizing effectiveness of heterogeneous assets while performing a specific task or mission. Currently, modern militaries employ autonomous systems such as unmanned aircraft systems (UAS) and unmanned ground systems (UGS) independently with separate missions. This study explores the possibility of utilizing an autonomous swarm composed of UAS and UGS working in tandem in the improvised explosive device (IED) clearance mission in the urban environment. In this mission, the UAS serves as the "eyes" to detect IEDs within a specific area assigned by an operator, while the UGS as the "hands and legs" to respond and neutralize the IED detected by the UAS. The thesis uses Cameo Systems Modeler, an industry-leading cross-platform collaborative model-based systems engineering (MBSE) environment, to model the joint UAS and UGS mission and then integrate it with a domain-specific MATLAB/Simulink environment where high-fidelity UAS and UGS models were developed.

14. SUBJECT TERMS
UAS, UGS, machine-machine teaming, IED clearance, Cameo, Simulink, systems engineering, unmanned aircraft systems, unmanned ground systems, improvised explosive device

15. NUMBER OF PAGES
109

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

MODEL-BASED UAS-UGS IED CLEARANCE MISSION ENGINEERING

Robert J. Naquila
Captain, Singapore Army
BSEE, Nanyang Technological University, 2017

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2021

Approved by: Oleg A. Yakimenko
Advisor

Fotis A. Papoulias
Second Reader

Oleg A. Yakimenko
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Machine-to-machine teaming is the trend of future warfare aiming at maximizing effectiveness of heterogeneous assets while performing a specific task or mission. Currently, modern militaries employ autonomous systems such as unmanned aircraft systems (UAS) and unmanned ground systems (UGS) independently with separate missions. This study explores the possibility of utilizing an autonomous swarm composed of UAS and UGS working in tandem in the improvised explosive device (IED) clearance mission in the urban environment. In this mission, the UAS serves as the "eyes" to detect IEDs within a specific area assigned by an operator, while the UGS as the "hands and legs" to respond and neutralize the IED detected by the UAS. The thesis uses Cameo Systems Modeler, an industry-leading cross-platform collaborative model-based systems engineering (MBSE) environment, to model the joint UAS and UGS mission and then integrate it with a domain-specific MATLAB/Simulink environment where high-fidelity UAS and UGS models were developed.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AO	area of operations
ATC	air traffic control
CCW	counterclockwise
CW	clockwise
DOF	degrees of freedom
DVR	digital video recording
FOV	field of view
GCS	ground control station
GPS	Global Positioning System
GUI	graphical user interface
IED	improvised explosive devices
Lidar	light detection and ranging
MBSE	model-based systems engineering
OBC	on-board computer
OPSIT	operational situation
PDB	power distribution box
PID	proportional, integral, derivative
QGC	QGroundControl
Satcom	satellite communication
SysML	Systems Modeling Language
UAS	unmanned aircraft system
UAV	unmanned aerial vehicle
UGS	unmanned ground system
UGV	unmanned ground vehicle
USS	unmanned surface system
UUS	unmanned underwater system
VRX	video receiver
VTOL	vertical take-off and landing

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Machine-to-machine teaming is the trend of future warfare with the aim to maximize effectiveness of heterogeneous assets while performing a specific mission. Currently, modern militaries employ autonomous systems such as unmanned aircraft systems (UAS) and unmanned ground systems (UGS) independently with separate missions. This thesis has explored the possibility of utilizing UAS and UGS working in tandem in performing IED clearance mission in an urban environment. In this mission, the UAS serves as the "eyes" to detect IEDs while the UGS serves as the "hands and legs" to respond and neutralize these detected IEDs.

The development of the models of UAS and UGS missions using model-based systems engineering tools such as Cameo Systems Modeler allowed a deeper understanding of the mission context and system architecture of UAS-UGS teaming. The development of UAS and UGS simulation models using MATLAB/Simulink allowed the analysis of the effects of selected system parameters and visualization of the missions in a 3D simulation environment.

The system parameters that have impact to the mission were identified. With all other parameters remain constant, flying a fully guided UAS at a high altitude generally completes the mission faster than an autonomous UAS that avoids obstacles at a low altitude. However, flying at a higher altitude may incur additional time penalty as more time needs to be catered to take off and land. Obstacle avoidance guidance logic may be suitable for missions with more complex and cluttered environment (i.e., tall buildings and narrow routes). A lower detection range is preferred for such complex environment, although very low detection range may cause the UAS to crash as it would have inadequate safety distance and insufficient time to maneuver away from the obstacle. The mass of the UAS cannot be too light or too heavy such that there is insufficient net thrust for the UAS to take off. It should be noted that the heavier the UAS is, the longer it would take to complete the mission. The higher the velocity of the UGS, the faster it can complete the mission. The UGS would be able to maintain its velocity, even when making sharp turns, through path smoothing.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, Professor Oleg Yakimenko, for his invaluable advice, patience, and continuous support throughout my thesis research. His wide connections and deep knowledge in MATLAB/ Simulink, systems engineering, and unmanned systems have steered my research towards the right direction and given it more depth.

Besides my advisor, I would also like to thank Dr. Saulius Pavalkis and Mr. Arnon Limsatiranan from Dassault Systèmes (3DS) for providing answers to all my Cameo-related queries. Their expertise in Cameo Systems Modeler allowed me to troubleshoot the integration of Cameo with MATLAB/Simulink.

My sincere thanks also go to Dr. Mariano Lizarraga Fernandez and Mr. Jianxin Sun from MathWorks for providing answers to all my Simulink-related queries, more specifically on the UAV Package Delivery Example. Their advice enabled me to explore the Simulink models and tweak them according to the needs of the experiment.

I also extend my appreciation to Mr. Diego Rangel, an alumnus from Naval Postgraduate School, whose thesis work also involved using Cameo-Simulink integration. I appreciate him taking the time to share his personal experience on integrating Cameo and Simulink even after he graduated.

I would also like to express my deepest appreciation to my family, especially to my wife, for the support and encouragement that they have given me throughout my graduate studies. They have really pushed me to achieve things that I thought I never could.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

This chapter provides the background and motivation for this study followed by formulating a research problem and laying out a logical structure of the thesis.

A. BACKGROUND AND MOTIVATION

Currently, detection and removal of mines, especially anti-personnel landmines, is a serious global problem affecting the political, economic, environmental, humanitarian, and military dimensions of the states involved. According to Landmine Monitor (2020), as of October 2020, 60 states remain to be contaminated by mines (see Figure 1) and at least 5,554 casualties – both civilians and military – were recorded since 2019 (see Figure 2).

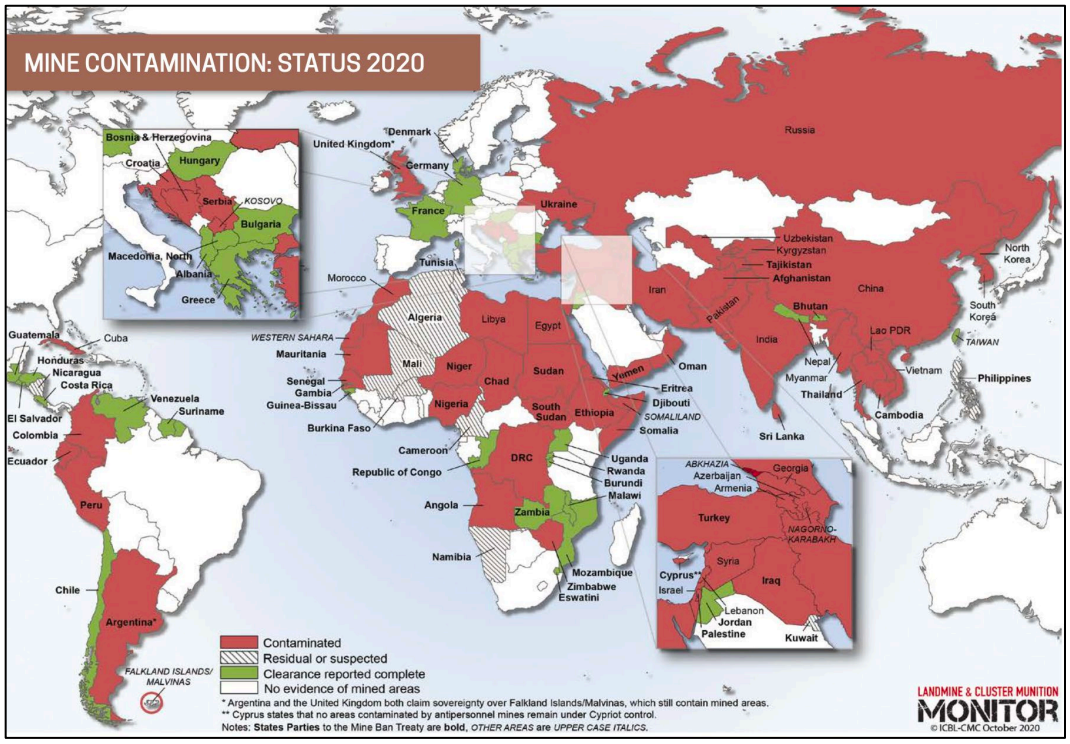


Figure 1. Status of Mine Contamination in 2020. Source: Landmine Monitor (2020).

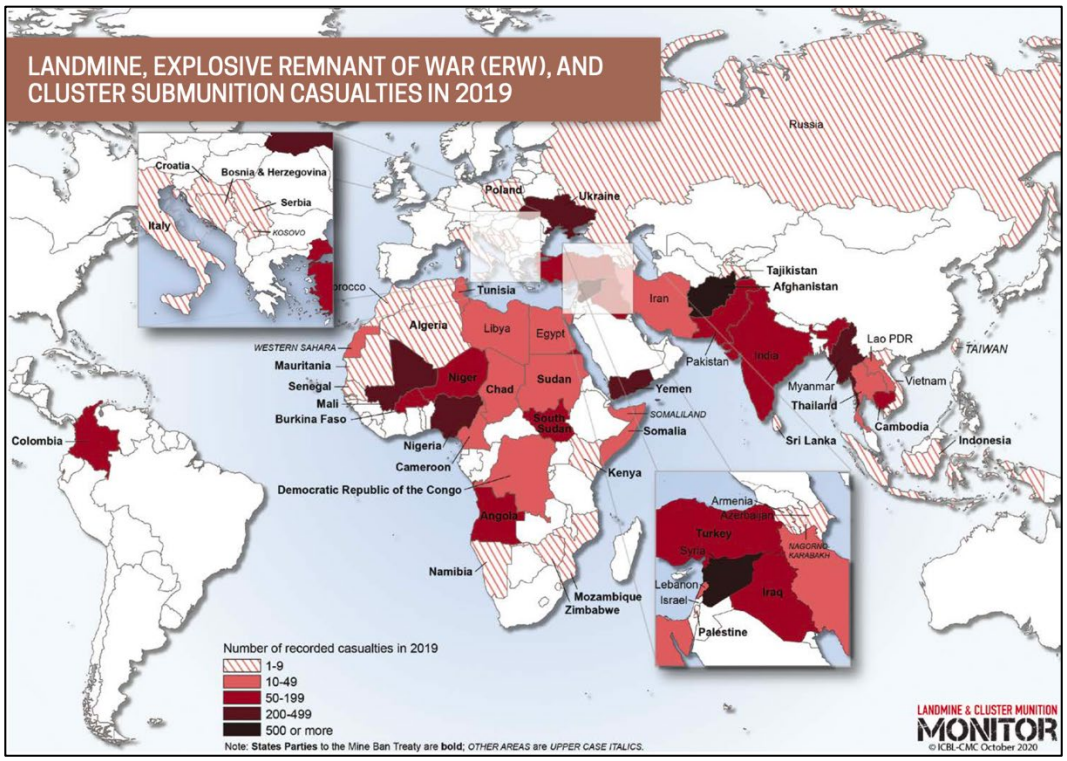


Figure 2. Casualties Caused by Mines in 2019. Source: Landmine Monitor (2020).

When operating in a minefield or any areas suspected with mines, the military combat engineers prioritize the path clearance of path for troops and vehicles to pass through in the most expedient and tactical way by the combat engineers. Mine clearing operations involve high risks, including casualties, especially while being subjected to enemy threats, extreme weather conditions, and tight schedule (see Figure 3). With the help of military intelligence, the combat engineers detect and remove anti-personnel and anti-tank mines using mechanical or explosive means. Modern mine clearing means range from mine flail systems (Tan et al. n.d., 123–128) to unmanned systems (Military Wiki n.d.).

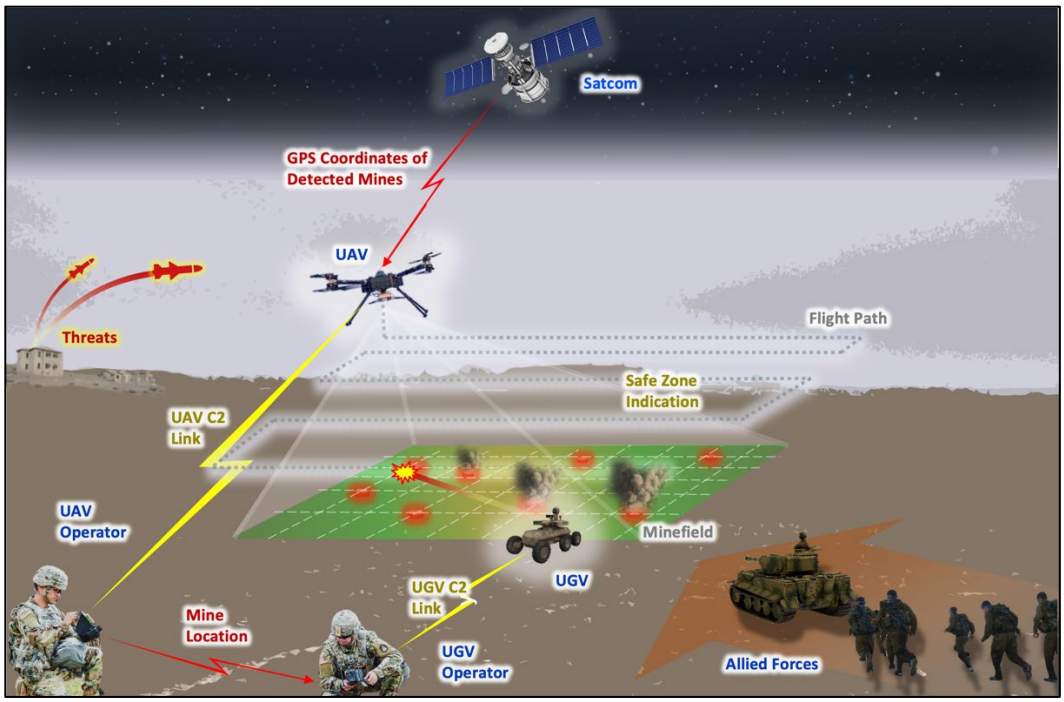


Figure 3. Operational Concept for UAS-UGS Teaming in IED Clearance.

As current nature of conflict evolves, the utilization of IEDs to impede the advancement of military troops and vehicles become more common in urban environment. Militaries face even more challenges on IED clearance in an operating theater with higher complexities (Wilkinson 2019). The presence of multiple routes and blind spots caused by high rise buildings and dense population, coupled with external threats, makes it more difficult to navigate around the area of operations (AO) compared to advancing through a flat minefield.

The external threats that could impact the operations in urban environment includes the environmental conditions in the operational theater: wide-ranging weather factors, such as temperature, precipitation, wind velocity, and humidity, with such a wide range, affect the performance of the system. Furthermore, attacks by enemy forces and terrorists could damage the system and compromise the mission. The network communications within the system could also be threatened by enemy jammers and natural factors such as foliage and rainfall. The impact and debris caused by the detonation of the IED may also damage the system. Inadvertent factors such as extended missions and extreme weather conditions may

also deteriorate the performance of the system. The presence of the threats not only imposes uncertainty, but also compels the military to exploit explosive detection technologies to overcome these challenges.

Using unmanned systems in IED clearance minimizes the risk of casualties in a more expedient and tactical way. Equipping them with the necessary payload, such as sensors and cameras, allows these systems to be able to navigate and overcome the complexity of the urban environment. Unmanned aircraft systems (UAS) are already being used to detect mines within a specific area assigned by an operator. UASs can carry different payloads to remove or neutralize the mines. Unmanned ground systems (UGS) such as the Armtrac 20T Robot (Armtrac 2020) are also being used to neutralize the mines. However, UASs and UGS are still being employed independently with separate missions. Machine-to-machine teaming is the trend of future warfare with the aim of optimizing the assets to perform a specific task or mission. This study explores the possibility of autonomous swarm UASs and UGSs working in tandem in IED clearance.

One approach in analyzing complex systems like UAS-UGS teaming is by modeling the entire mission using model-based systems engineering (MBSE) tools. Cameo Systems Modeler ("Cameo"), a cross-platform collaborative MBSE environment, allows to design and build system engineering models (see Figure 4) to run engineering analysis based on specific requirements and evaluate through simulation (No Magic n.d.). The models developed in Cameo are not only used for analysis and evaluation within Cameo's MBSE environment, but it can also be integrated with external simulation environment such as Simulink to perform even more complex computations and simulation. Simulink is a software in MATLAB which uses block diagrams while incorporating MATLAB codes and algorithms to develop system-level and model-based designs (MathWorks n.d.). Integrating Cameo with Simulink enables the exploration of UAS-UGS teaming in IED clearance.

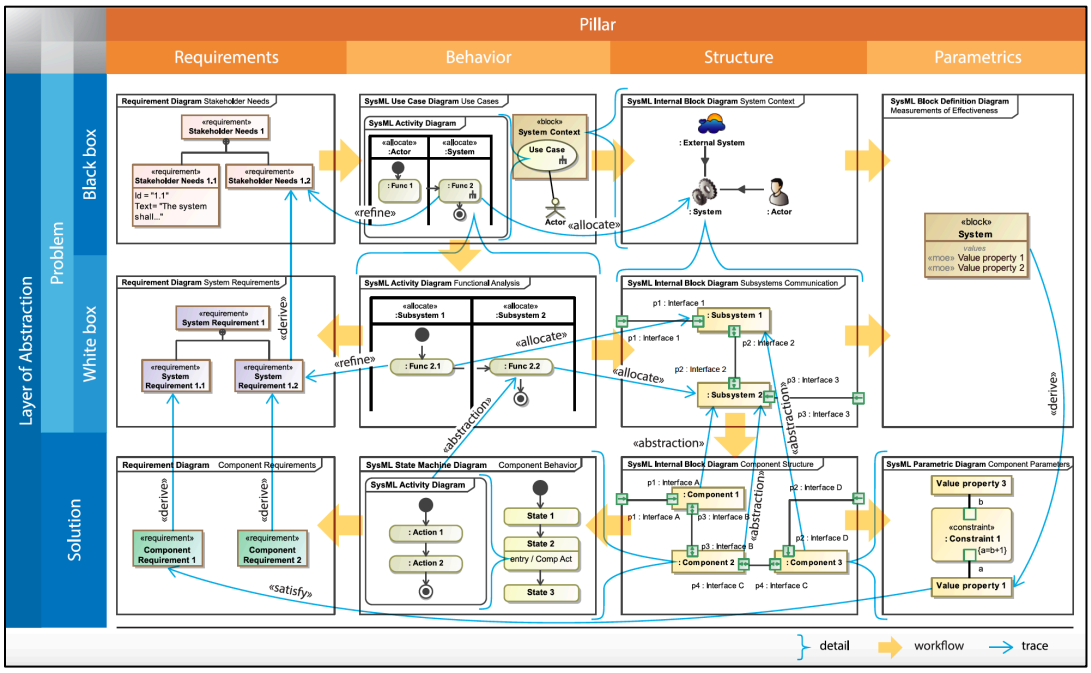


Figure 4. Traceability of MBSE Grid. Source: No Magic (n.d.).

B. RESEARCH QUESTIONS

This thesis explores the possibility of using MBSE environment to architect and explore UAS and UGS teaming in IED clearance while using high-fidelity models developed in MATLAB/Simulink development environment. Specifically, this study is guided by the following research questions:

(1) Can high-fidelity models of UAS and UGS be integrated into the MBSE environment to conduct a study on improving the modern tactics and procedures of IED clearance?

(2) Can the integrated model be used to investigate the optimized configuration of the UAS and UGS in the conduct of IED clearance based on the chosen system parameters?

C. BENEFITS OF THE STUDY

This use of the developed integrated model benefits the Department of Defense (DOD) to analyze the advantages of employing UAS and UGS teaming in optimizing the mine clearing operations with lesser manpower, lower cost, and higher performance. The Navy and systems engineering community would also benefit from this study on showing

the methodology of how integrated Cameo/Simulink models can be built as well as the exploration of other machine-to-machine teaming such as Unmanned Underwater Systems (UUS) and Unmanned Surface Systems (USS) to accomplish a common task or mission.

D. RESEARCH METHODOLOGY

This study builds upon existing high-fidelity models and algorithms developed by domain experts in the MATLAB/Simulink development environment and shows how to integrate them into Cameo MBSE environment where the system architectures of UAS and UGS missions are developed.

A quantitative research method is used to determine and manipulate baseline system parameters of the UAS and UGS. These system parameters were evaluated and optimized in the Cameo simulation to determine if the overall system capability meets the stipulated measures of effectiveness and performance and achieve the mission objectives set for each given operational scenario. All relevant information necessary for the integration of UAS and UGS was obtained through open-source research papers.

E. THESIS OUTLINE

The next few chapters of this thesis are organized as follows.

Chapter II reviews past research works on UAS-UGS teaming in the context of IED clearance as well as the fundamental concepts and kinematics of UAS and UGS as independent systems. This gives the reader an insight on the mathematical principles and physical laws that describes the behavior of a quadcopter and a wheeled UGV. This also highlights the system parameters that can be used as test variables for analysis. This chapter also discusses the purpose and benefits of Cameo as an MBSE tool as well as how it can be integrated with MATLAB/Simulink environment.

Chapter III describes the mission architecture of the UAS-UGS teaming in the context of IED clearance using MBSE diagrams which were developed in Cameo. This chapter demonstrates the creation and implementation of the simulation to be integrated with the Simulink model as discussed in the next chapter.

Chapter IV describes the Simulink models of UAS and UGS which were used to simulate the UAS-UGS teaming in performing IED clearance mission. This chapter breaks down the composition of each model and the various configurations to simulate different test cases.

Chapter V demonstrates the experimentation approach, design, and implementation using the mission architecture and Simulink models as described in Chapters III and IV. This chapter outlines the test cases used to evaluate the test variables based on the parameters described in the literature reviews in Chapter II.

Chapter VI includes the outputs of the simulated test cases and their corresponding analyses on the UAS-UGS teaming system. The analysis of the outputs enabled to understand the effects of each test variable in the mission and produce a set of recommendations for the application of the model.

Chapter VII summarizes the effects of each test variables based on the experiment as well as the recommendations for UAS and UGS configurations to ensure efficiency in terms of the IED clearance mission timing.

THIS PAGE INTENTIONALLY LEFT BLANK

II. LITERATURE REVIEW

This chapter begins with past research studies on UAS-UGS teaming in the context of IED clearance missions. After that, it presents the basic flight dynamics and obstacle avoidance capability of a UAS followed by the basic kinematics and path planning capability of a UGS using mathematical models. Finally, this chapter then proceeds to describe how Cameo as an MBSE tool is used to analyze UAS-UGS teaming as well as how it can be integrated with MATLAB/Simulink environment.

A. UAS-UGS TEAMING

Machine-to-machine teaming is the trend of future warfare with the aim of optimizing the assets to perform a specific task or mission. As seen in Figure 5, the elements of UAS-UGS teaming includes: (1) environment, (2) task (3) UAS, and (4) UGS.

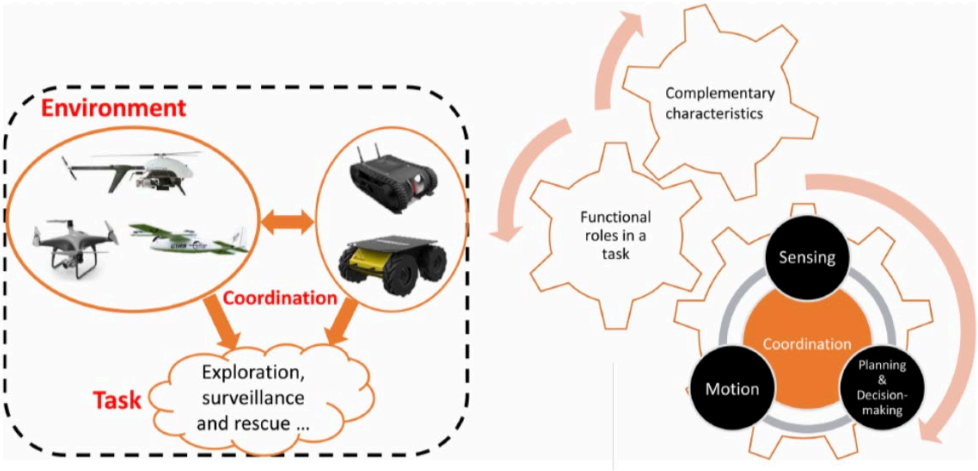


Figure 5. Elements of UAS-UGS Teaming. Source: Ding et al. (2021).

1. Environment

The environment provides the UAS and UGS with external factors to perceive and collaborate. The environment that the UAS confronts may not necessarily be the same as that of the UGS. The ground environment that the UGS interact with is more complex with wide range of obstacles and dynamic objects which forces the UGS to change its behavior.

Positive obstacles (e.g., pavements and rocks) and negative obstacles (e.g., potholes and ditches) compels the UGS to have the ability to detect and avoid them in order to proceed to the desired location. Structured roads with consistent lane width, boundaries, and markings are easier for the UGS to navigate and maneuver across compared to unstructured roads with sudden changes in curvature and road conditions. Dynamic objects (e.g., humans and moving vehicles) that are in motion and could change course without any warning make it more difficult for the UGS to plan its route and maintain its trajectory.

The airspace that the UAS interact with is relatively simpler than that of the UGS, but still pose a set of unique challenges. According to Federal Aviation Agency (FAA), the UAS can either operate on a controlled or uncontrolled airspace. As seen in Figure 6, the controlled airspace (Classes A, B, C, D, and E) is generally free from obstacles as it is within the authority of air traffic control (ATC) while the uncontrolled airspace (Class G) is susceptible to positive obstacles such as buildings and traffic lights (Hu et al. 2019). Other environmental factors (e.g., illumination and foliage) may cause clutter in the background and pose challenges in the ability of both UAS and UGS to detect obstacles.

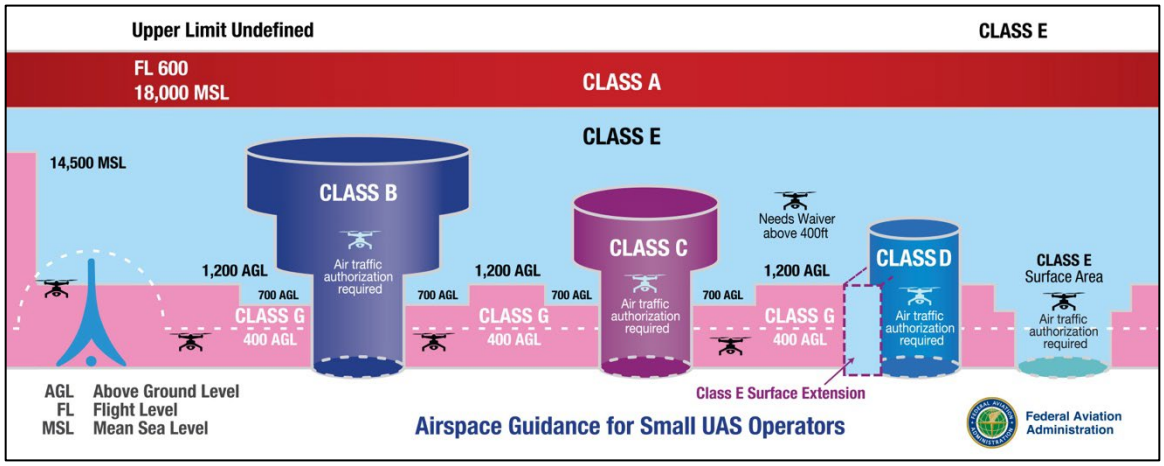


Figure 6. Airspace Classification. Source: FAA (2018).

2. Task

The UAS-UGS teaming can perform a wide array of missions with varying degrees of complexity. The complexity of the mission determines the number and type of vehicle

required to accomplish a task. Tasks that require multiple cooperative vehicles can either be loosely- or tightly-coordinated (Ding et al. 2021). In loosely-coordinated tasks, the operating environment can be partitioned into disjointed areas, and the vehicles operate independently within their specified areas with minimum interaction with each other. Such tasks include large-scale exploration, tracking, and surveillance missions. On the other hand, tightly-coordinated tasks require coordinated execution with significant interaction among vehicles.

As seen in Table 1, UAS and UGS have diverse types and characteristics along with their own advantages and disadvantages. Their limitations may reduce their performance and efficiency to a certain extent (Chen et al. 2016). However, the heterogeneous-nature and complementariness between UAS and UGS in terms of motion, sensing, planning, and decision-making make the UAS-UGS teaming overcome the limitations of each as independent systems to be able to accomplish a wide range of missions.

Table 1. Types of UAS and UGS. Adapted from Ding et al. (2021).

System Type	**Mobility**	**Adaptability**	**Sensing**	**Endurance**	**Payload**
Multirotor UAS	Low speed, VTOL and hover	Can land in very small area and rough surfaces	Precise inspection	Short	Small
Helicopter UAS	Quick speed, VTOL and hover	Can land in small area and rough surfaces	Precise inspection	Long	Middle
Fixed-wing UAS	High speed, No VTOL/hover	Large landing space needed, great stability in wind	Large area coverage	Long	Large
Tracked UGS	Low speed, maneuver in tight places	Can move on rough terrain	Small field of view	Short	Very large
Wheeled UGS	Quick speed, high maneuverability	Easily slide on slopes sink into soil and wet ground	Small field of view	Long	Quite large

The functional roles of UGS and UAS in a coordinated mission are determined by their respective capabilities and specializations. It is important to leverage on the strengths of UGS and UAS to complement with each other's weaknesses. Ding et al. (2021) identified four main functional roles in the UAS-UGS teaming: (1) sensor, (2) actuator, (3) decision-maker, and (4) auxiliary facility. More specifically, for missions like IED clearance, this thesis identified two possible types of UAS-UGS teaming.

a. UAS as Sensor and UGS as Actuator

In this type of UAS-UGS teaming, the UAS acts as a sensor to detect, collect, and transmit the necessary data from the environment to the UGS, which completes the task according to the received data. The UAS's great speed and wider range of vision allow data to be captured swiftly and be transmitted to the UGS to perform the necessary actions to complete the mission.

Earlier technologies relied on manually controlling the flight of the UAS to collect data followed by the deployment of the UGS to perform subsequent actions. The lidar and aerial data collected by the UAS enhances the route planning and navigation capability of the UGS (Stentz et al. 2018). But as autonomous systems advance, the UAS is able to generate, not only in 2D, but also a 3D map of the assigned area using image processing techniques, which helps the UGS to determine its position and orientation without relying on GPS (Kaslin et al. 2016) as well as avoid obstacles and complete the assigned task (Li et al. 2016). The advancement of technology in object-identification paved the way for the UAS to identify simulated mines, localize marked points, and send the UGS to each location (Zawodny MacArthur, MacArthur, and Crane 2005).

b. UAS as Sensor and Decision-Maker and UGS as Actuator

In this type of UAS-UGS teaming, the UAS feeds the UGS with the necessary data from the environment while monitoring and providing guidance for the UGS. The UAS acts as the "eye-in-the-sky" as well as the decision maker for the UGS. The UAS can use vision-based control method (Aranda et al. 2015) to monitor a swarm of UGSs and control their motion, formation, and coordination in an urban environment (Chaimowicz and Kumar n.d.).

B. UAS

Quadcopter is a multirotor UAS (see Table 1) with four rotors. It is primarily used in the military to be deployed in complex contested environment without endangering human pilots. Other than its small size and swift maneuverability, the low production cost of its components enable the quadcopter to be more cost-effective and widely used commercially. It is more preferred than a helicopter UAS as it operates in fixed-rotor propulsion mode, where the two transverse-pairs of rotors are arranged to rotate in opposite directions (Gupte 2012).

1. Physical Decomposition Structure

Figure 7 illustrates the internal and external components of a quadcopter as well as the interfaces that forms the interaction between each component.

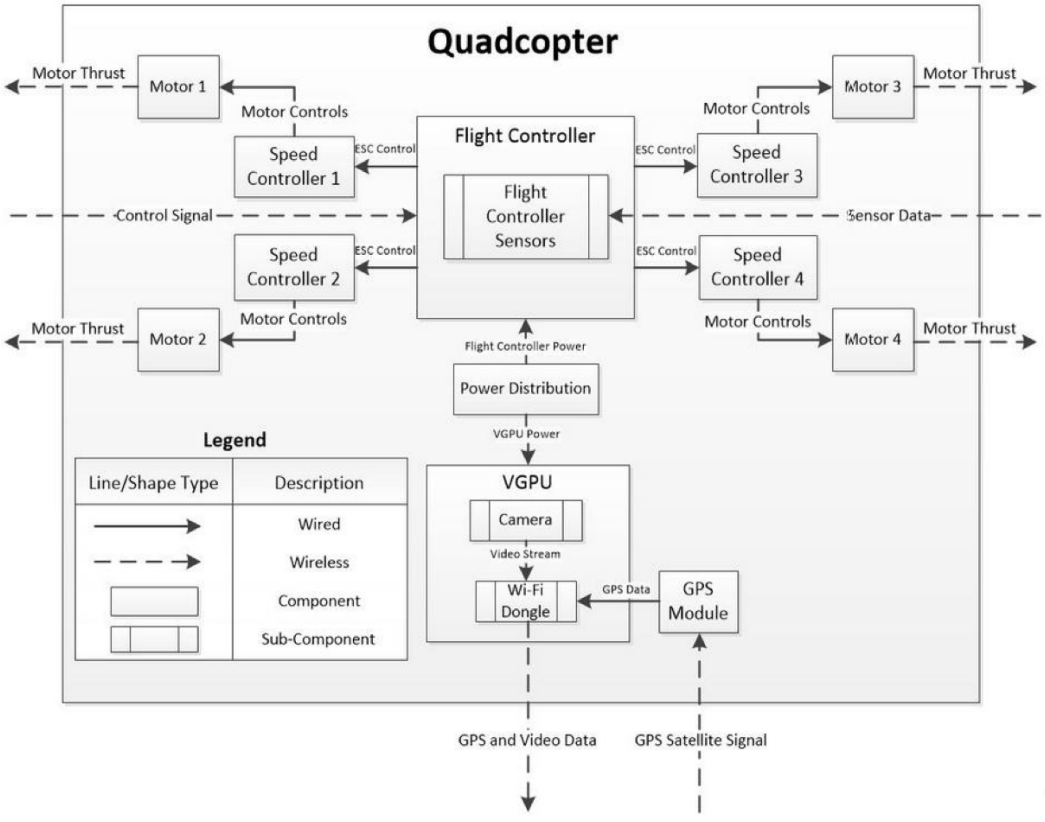


Figure 7. Block Diagram of a Quadcopter. Source: Bhattacharjee (2018).

The flight controller is the brain of the quadcopter, which keeps the quadcopter in the air as well as communicate to other components. The radio transmitter or the remote control sends the operator's commands over to the receiver which specifies the flight controller the corresponding action that the quadcopter is expected to perform. The electronic speed controller (ESC) takes the signal form the flight controller and tells the motor how fast to spin. There are four ESCs and motors. The GPS, which usually has a compass module, sends the location and heading data to the flight controller. first-person camera will take video from the quadcopter in real-time and sends to the flight controller which has a built-in on-screen display. The on-screen display takes the flight data and overlays it on the video for the operator to read it in real-time. It takes this video with overlaid information and sends to the video transmitter, which transmits it through the air and received by a power video receiver (VRX) or "googles" for monitoring. The digital video recording (DVR) allows to save that real-time video footage. The battery supplies the flight controller and other components with power. The flight controller also doubles as a power distribution board (PDB), which to reduce the input voltage to different voltages necessary to power up smaller components. The charger allows to restore the energy of the batteries when they run out of power as well as putting the batteries in storage mode. The HD camera captures the video footages in the quadcopter's point-of-view (POV) while the gimbal stabilizes the HD camera footage. The gimbal can also be controlled via the quadcopter's receiver.

2. Flight Dynamics

The flight dynamics of the quadcopter generally works in two frames: (1) inertial frame and (2) body frame, as seen in Figure 8. The inertial frame is defined with respect to the ground with the gravitational pull in the negative z-axis, while the body frame is defined by the orientation of the quadcopter, with the rotor pointing in the positive z-axis and the arm-extensions pointing in the positive/negative x- and y-axes (Khan 2014).

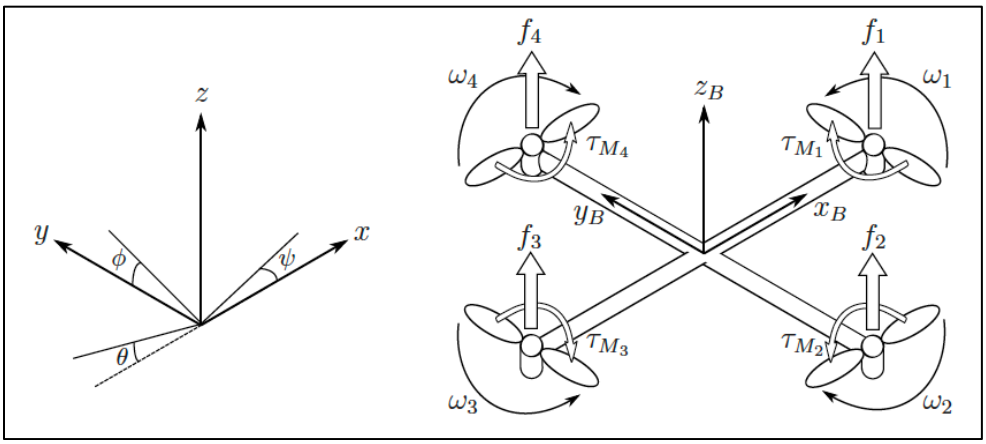


Figure 8. Inertial and Body Frames of a UAS. Source: Luukkonen (2011).

The absolute linear position of the quadcopter is defined in the inertial frame x,y,z-axes with ξ, while the attitude (i.e., angular position) is defined in the inertial frame with three Euler angles η (the attitude provides information about the UAS's orientation with respect to the local level frame (horizontal plane) and true north)

$$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{1}$$

In Equation 1, roll angle ϕ determines the rotation around the x-axis which allows the quadcopter to move left or right, Pitch angle θ determines the rotation around the y-axis which allows the quadcopter to move forward or backward, and Yaw angle ψ determines the rotation around the z-axis which allows rotation with respect to the center of the quadcopter.

The origin of the body frame is at the center of the quadcopter's mass. In the body frame, the linear velocity V_B and the angular velocity ω are vectors directing in axis of rotation

$$V_B = \begin{bmatrix} v_{x,B} \\ v_{y,B} \\ v_{z,B} \end{bmatrix}, \quad \omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{2}$$

Angular velocity components in the body frame are p, q, and r respectively.

The inertial and body frames are related by a rotation matrix R which is derived using ZYZ Euler angle conventions and successively "undoing" the roll, pitch, and yaw

$$R = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\phi\sin\psi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\sin\theta & \sin\psi\sin\theta\sin\phi - \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \sin\psi\cos\phi \\ -\sin\theta & \sin\phi\cos\theta & \cos\theta\cos\psi \end{bmatrix} \tag{3}$$

The Euler angles η dynamics can be derived via angular velocity components p, q, and r

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} p + q\tan\theta\sin\phi + r\tan\theta\cos\phi \\ q\cos\phi - r\sin\phi \\ p + q\sec\phi + r\sec\theta\cos\phi \end{bmatrix} \tag{4}$$

The quadcopter is assumed to have symmetry structure with four arms aligned with the body's x- and y-axes. The inertia matrix I is assumed to be a diagonal matrix whereas the mass products of inertia terms are assumed to be 0

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{5}$$

The thrust forces f_1, f_2, f_3, and f_4 from the four motors are generated by the propellers. The resultant moments about roll, pitch, and yaw axes are given as:

$$\sum M_x = (f_3 - f_1)\frac{L}{2} \tag{6}$$

$$\sum M_y = (f_2 - f_4)\frac{L}{2} \tag{7}$$

$$\sum M_z = (M_1 + M_3) - (M_2 + M_4) \tag{8}$$

Therefore,

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{qr}{I_x}\left(I_y - I_z\right) - \frac{L}{2I_x}\left(f_1 - f_3\right) \\ \frac{pr}{I_y}\left(I_y - I_z\right) - \frac{L}{2I_z}\left(f_2 - f_4\right) \\ \frac{pq}{I_z}\left(I_x - I_y\right) \frac{\left(\left(f_2 + f_4\right) - \left(f_1 + f_3\right)\right)d}{2I_z b} \end{bmatrix} \tag{9}$$

Hence, the state vector considering the roll, pitch, and yaw dynamics of the quadcopter are expressed as

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} p + q\tan\theta\sin\phi + r\tan\theta\cos\phi \\ q\cos\phi - r\sin\phi \\ p + q\sec\phi + r\sec\theta\cos\phi \\ \frac{qr}{I_x}\left(I_y - I_z\right) - \frac{L}{2I_x}\left(f_1 - f_3\right) \\ \frac{pr}{I_y}\left(I_y - I_z\right) - \frac{L}{2I_z}\left(f_2 - f_4\right) \\ \frac{pq}{I_z}\left(I_x - I_y\right) \frac{\left(\left(f_2 + f_4\right) - \left(f_1 + f_3\right)\right)d}{2I_z b} \end{bmatrix} \tag{10}$$

Also, by applying the force and momentum balance laws, the following quadcopter motion formulation can be derived

$$\ddot{x} = u_1\left(\cos\phi\sin\theta + \sin\iota\sin\psi\right) - \frac{K_1\dot{x}}{m} \tag{11}$$

$$\ddot{y} = u_1\left(\sin\phi\sin\theta\cos\psi + \cos\iota\sin\psi\right) - \frac{K_2\dot{y}}{m} \tag{12}$$

$$\ddot{z} = u_1\left(\cos f\cos\psi\right) - g - \frac{K_3\dot{x}}{m} \tag{13}$$

where K_i is the drag coefficient (assumed to be 0 since drag is negligible at low speed). The angle movement of quadcopter is given by

$$\phi_d = \tan^{-1}\frac{y_d - y}{x_d - x} \tag{14}$$

$$\psi_d = \tan^{-1} \frac{z_d - z}{\sqrt{(x_d - x)^2 + (y_d - y)^2}} \tag{15}$$

3. PID Control

The proportional, integral, and derivative (PID) controller, as seen in Figure 9, is a popular way of controlling quadcopters due to its effectiveness, simple implementation, and low cost. With several applications, it has been proven to be robust and adequate; failing only in some applications involving extremely complex, typically nonlinear, systems (Pessoa 2017).

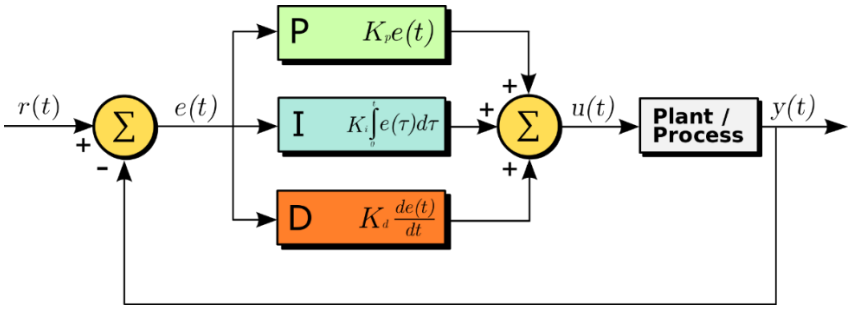


Figure 9. Block Diagram of a PID Controller. Source: Wikipedia (2011).

As shown in Figure 9, the PID controller aims to minimize the error function $e(t)$, which is commonly the difference between the desired states $r(t)$ and the current states $y(t)$, which can be obtained through one or more sensors. It then proceeds to minimize error over time, by adjusting the inputs $u(t)$, which is a weighted sum where each coefficient is positive, as given by

$$\begin{cases} u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \dfrac{de(t)}{dt} \\ e(t) = r(t)\,y - r(t) \end{cases}. \tag{16}$$

The proportional coefficient K_p acts on the present error while the integral coefficient K_i acts on the past error. Increasing both reduces the output's time constant, error in steady state, and general stability. Increasing K_i also increases the amount of overshoot and stabilization time but increasing K_p only increases the amount of overshoot but no effect on stabilization time. The derivative coefficient K_d acts on the future error. Increasing it has no effects on the output's time constant or error in steady state but reduces the amount of overshoot and stabilization time while increasing the overall stability.

4. Flight Operations

The thrust of a quadcopter is measured when it is in full throttle, producing maximum upward force. It changes when the propeller is moving and accelerates in the direction of its force. To be able to take off, the thrust must be greater than the quadcopter's mass. Thrust T is determined by air density ρ, and cross-sectional area A of the propeller, and velocity v of the spinning rotor, as given by

$$\left|\vec{T}_i\right| = \rho A v_i^2 \tag{17}$$

A real-time reading of the air density is necessary as it is correlated to the environmental factors which are always changing. Similarly, the velocity of the spinning rotors required for the quadcopter to ascend, descend, or hover at a constant altitude is also a variable to the environmental factors. While the cross-sectional areas of the propellers remain constant during flight, a constant reading of the air density would significantly compromise the performance of the rotors and therefore, the thrust of the quadcopter.

a. Take-Off and Landing

In takeoff mode, all four rotors spin in clockwise (CW) direction. The CW direction contributes to the positive net thrust (body frame) of the quadcopter, thereby enabling the translation in the positive z-axis (inertial frame). In landing mode, all four rotors spin in counterclockwise (CCW) direction. The CCW direction contributes to the negative net thrust (body frame) of the quadcopter, thereby enabling translation in the negative z-axis

(inertial frame). Provided that all four rotors spin in the same direction and velocity, the net thrust of the quadcopter is given by

$$T_{net} = \rho A \sum_{i=1}^{4} v_i^2 \,. \tag{18}$$

b. Hover

In hover mode, the net thrust of all four rotors is equal to 0, causing the quadcopter to maintain a constant altitude. At any point of time, the direction of rotation of a pair of rotors at each axis is always the same. For the net thrust to be 0, both rotors in the x-axis (body frame) must spin the opposite direction of that in the y-axis (body frame). Regardless which direction the propellers rotate, the velocities of all rotors are equal during hovering.

$$T_{net} = \rho A \sum_{\uparrow i=1}^{2} v_i^2 - \rho A \sum_{\downarrow i=3}^{4} v_i^2 = 0 \tag{19}$$

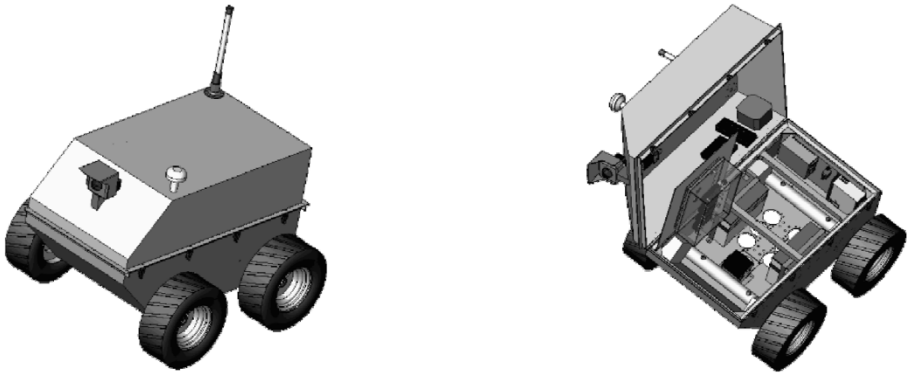
5. Obstacle Avoidance

UAS relies on sensors such as lidar to perceive the environment. Among other kinds of sensors, lidar is less impacted by environmental factors (e.g., cloud and rain) and flight conditions. It also has higher precision and flexibility. Therefore, it is widely used as an obstacle detection capability.

Obstacle detection is generally conducted in three steps: (1) obstacle data collection, (2) point cloud preprocessing, and (3) point cloud clustering (Peng et al. 2015). Lidar collects the obstacle data in real-time and stores them in point clouds. Point cloud preprocessing includes point cloud correction, which restores the distorted point cloud caused by the extrinsic motion of the lidar during the continuous laser ranging, as well as filtering, which removes any unnecessary points. Point cloud clustering involves the extraction of ordered obstacle data from the cluttered point clouds and transforms them into an intuitive set of obstacle data. This set of data may include the angle, position, shape, and size of the obstacle which enables the UAS to obtain full situational awareness and adopt necessary maneuvers to avoid the obstacle.

C. UGS

The primary consideration of employing a UGS in any mission is its mobility. Achieving maximum mobility with minimum energy consumption is a critical constraint in the development of UGS. Other considerations, which include maneuverability, payload capacity, autonomy, and modularity, contribute to the performance of the UGS in its ability to complete the assigned tasks. In comparison to a tracked UGS, a wheeled UGS (see Figure 10) offers higher velocity and maneuverability along with a longer endurance and sufficient payload capacity. Optimal key parameters that best suit the mission requirements must be determined for the UGS to be cost-effective and operationally efficient.

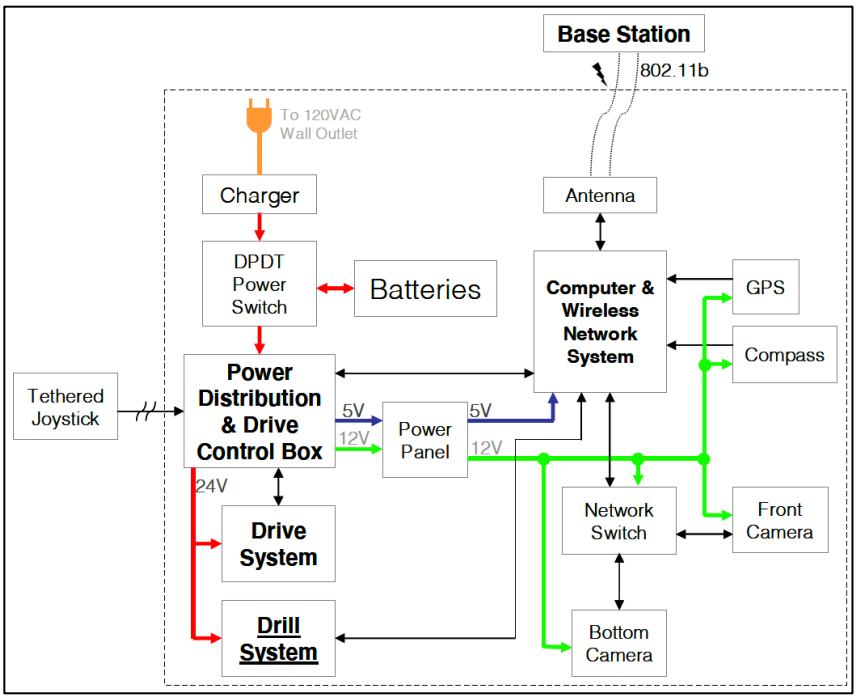


(a) (b)

Figure 10. External and Internal View of a Wheeled UGS. Source: Wasson (2004).

1. Physical Decomposition Structure

A typical UGS is remotely-operated from a base station where the operator can interact with a graphical user interface (GUI). This enables the operator to enter GPS coordinates of a pre-determined or visually-marked target and thereafter send the UGS to autonomously navigate and locate it. When the UGS reaches the proximity of the target, the operator may control the UGS via a tethered joystick on the precise location of the target (Wasson et a. 2004). For mine clearing operations, the UGS may be attached with a payload (i.e., drill system) to neutralize the mine as seen in Figure 11.



The elements within the dashed line are the internal subsystems of a UGS while those that are outside are the external subsystems.

Figure 11. Block Diagram of a UGS. Source: Wasson et al. (2004).

The UGS can be operated in two modes: (1) autonomous or (2) manual. In autonomous mode, the base station sends the command to the UGS's on-board computer (OBC) via wireless connection and the UGS executes the commands autonomously. In manual mode, the tethered joystick overrides the base station and gives the operator full control. This also serves as a fail-safe mode allowing the UGS be recovered should other subsystems lose power. The Power Distribution and Drive Control Box (PDDCB) houses the power source as well as the circuitry that allows OBC and the joystick to interface with the drive system. The drive system enables the UGS to move and deliver enough torque to climb a slope. Steering is achieved when the pairs of motors on opposing sides are driven at different velocities. These motors are driven by servo amplifiers, which are interfaced to the PDDCB, to allow either the OBC or the joystick to control the drive system. The front and bottom cameras stream real-time video. The front camera perceives the frontal view of the UGS while the bottom camera allows the operator to observe the operation at the UGS's undercarriage. The GPS receiver and digital compass enable point-to-point navigation.

2. Kinematics

Kinematics model is a simplistic way to analyze the UGS's motion and predict its dynamic behavior while traveling along a curve at low acceleration. Using a low number of degrees of freedom (DOF), the kinematics equation allows to investigate lateral cornering, longitudinal dynamics, and yaw motion independently. The kinematics model shown in Figure 12 illustrates a UGS in a coordinate system where (x_0, y_0) is the center of the UGS's mass; θ_e is the azimuth angle, which is the angle between the forward direction of UGS and the x-axis; ω_e and v_e are the angular and linear velocities respectively.

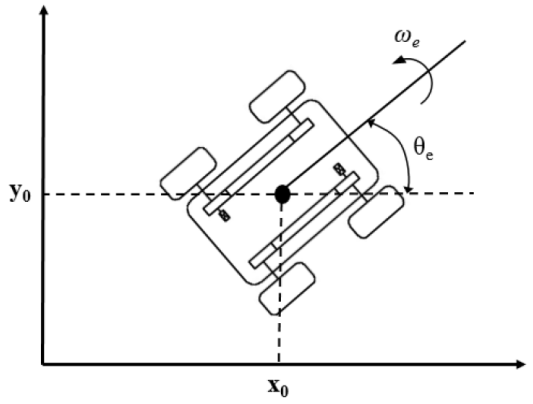


Figure 12. Kinematics Model of a UGS. Source: Jiliang Lv (2021).

The kinematics equation of the UGS is given by

$$\begin{bmatrix} x_e(k+1) \\ y_e(k+1) \\ \theta_e(k+1) \end{bmatrix} = \begin{bmatrix} x_e(k) \\ y_e(k) \\ \theta_e(k) \end{bmatrix} + \begin{bmatrix} \cos\theta_e & 0 \\ \sin\theta_e & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_e \\ \omega_e \end{bmatrix} \Delta t \tag{20}$$

where Δt is the sampling time during the movement of the UGS.

When the rotation speed of the left wheel is equal to that of the right wheel, the turning radius of the UGS is 0, which means that the UGS rotates in place.

3. Path Tracking

Path tracking uses positional data to control the velocity and steering of the UGS to follow a specified path. Theoretically, the path of the UGS is a continuous function (Figure 13a). But, in practice, it is either a piecewise linear path represented by line segments (Figure 13b), or a set of discrete path nodes represented by closely spaced waypoints (Figure 13c). For a piecewise linear path, the algorithm attempts to follow the straight lines, while for a set of discrete path nodes, the algorithm only attempts to reach the next closest node.

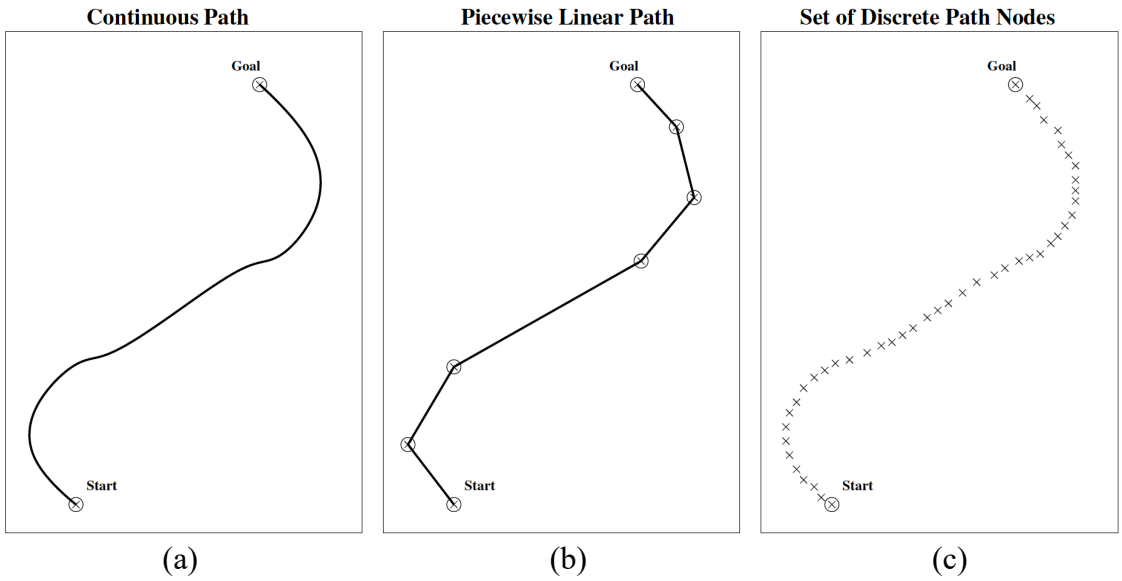
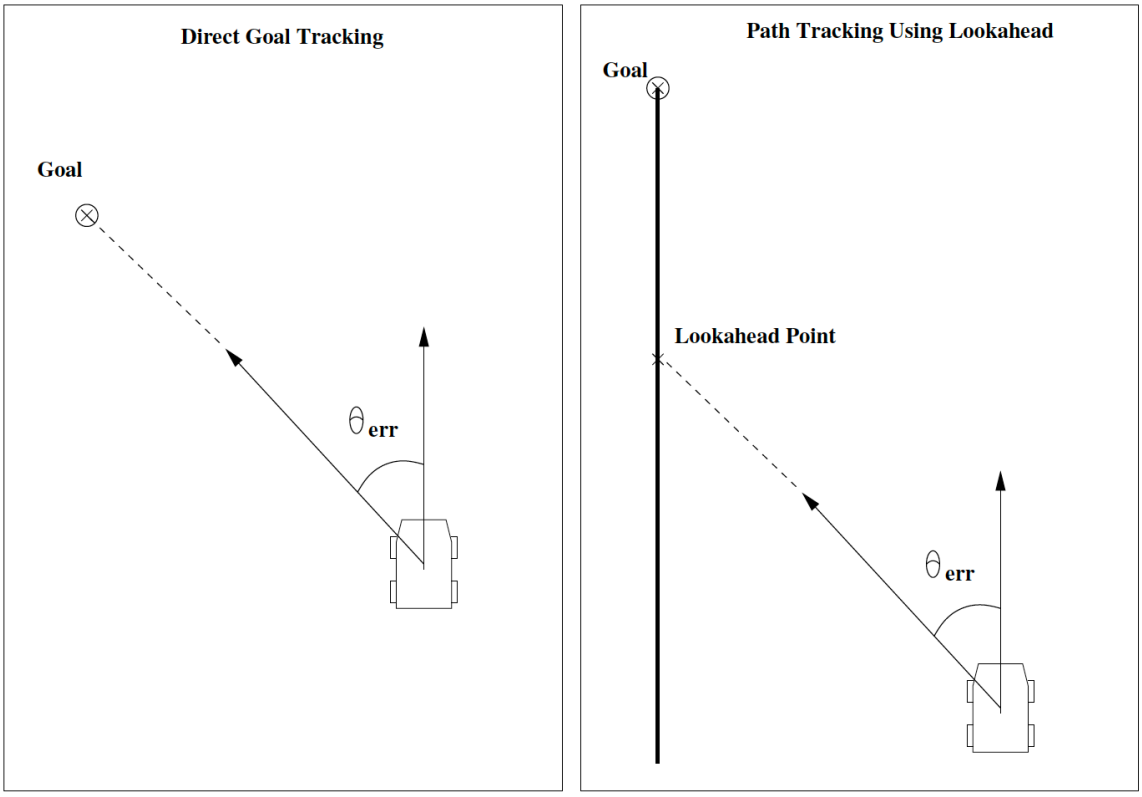


Figure 13. Various Ways of Defining Path. Source: Giesbrecht (2005).

If a path is given by a series of waypoints, then the simplest method to track a path is to use PID control loop to correct the error between the current heading and the heading to the next waypoint (see Figure 14a). This method paved the way to develop basic obstacle avoidance algorithms (Simmons 1996).

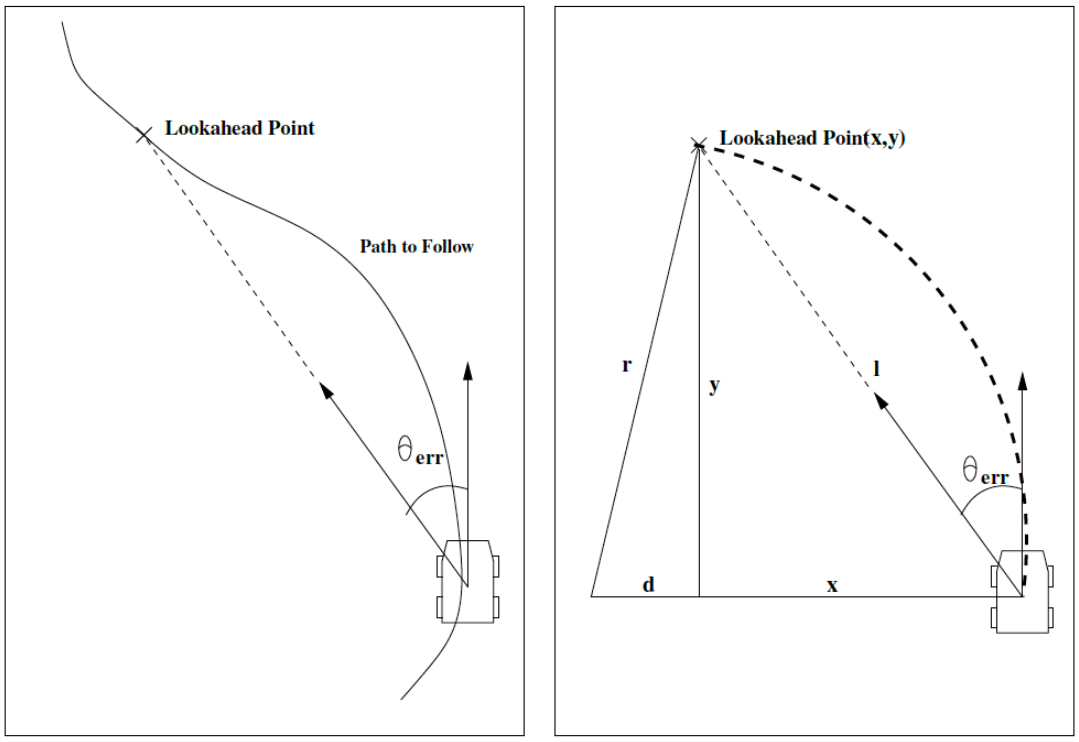


(a) (b)

Figure 14. Direct and Indirect PID Control for Goal Seeking. Source: Giesbrecht (2005).

When the UGS avoids an obstacle and gets directed away from its intended path, the UGS should not move directly towards the next waypoint. Instead, the UGS should steer towards a lookahead point as it slides along the path while maintaining a fixed distance ahead of the UGS (Figure 14b). A PID control loop can then be used to control the error between the current heading and the lookahead point as the UGS gets redirected back to the intended path.

However, using a PID control loop can be time-consuming to tune the proportional (P), integral (I), and derivative (D) components to achieve the desired behaviour (Giesbrecht 2005). Instead, Pure Pursuit algorithm can be used due to its simplicity and effectiveness. This algorithm involves a constant curvature of the arc connecting the current position of the UGS and the lookahead distance l as shown in Figure 15.



(a) (b)

The left image shows the path to be tracked and the right image shows the calculated steering curvature. The curvature of the arc indicates a circle of radius r.

Figure 15. Geometry of the Pure Pursuit Algorithm. Giesbrecht (2005).

By Pythagorean theorem,

$$x^2 + y^2 = l^2 \tag{21}$$

$$d^2 + y^2 = r^2 \tag{22}$$

and from Figure 15(b),

$$d = r - x\,. \tag{23}$$

Substituting Equation 23 into Equation 22 yields

$$(r - x)^2 + y^2 = r^2 \tag{24}$$

$$x^2 + y^2 = 2rx \tag{25}$$

and substituting Equation 25 into Equation 21 yields

$$2rx = l^2 \tag{26}$$

and

$$r = \frac{l^2}{2x}. \tag{27}$$

The curvature of the arc is given as $\gamma = \frac{1}{r}$, hence, Equation 27 can be rewritten as

$$\gamma = \frac{2x}{l^2} \tag{28}$$

In essence, the Pure Pursuit algorithm is a proportional controller which operates on the error between the current heading of the UGS to the next waypoint. This can be seen in Figure 16 where the algorithm only considers a single parameter (i.e., lookahead distance l) making this algorithm easy to implement and tune. Tuning the lookahead distance enables the UGS to track the path more accurately. Furthermore, a smaller lookahead distance allows the UGS to redirect back to its intended path more aggressively.

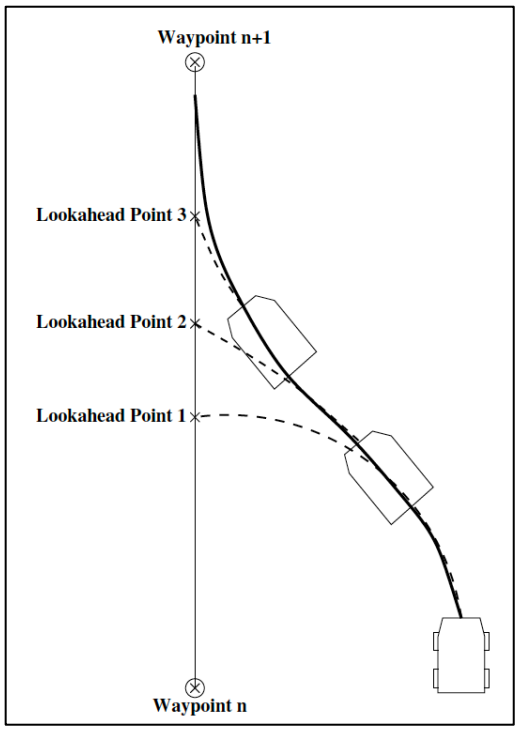
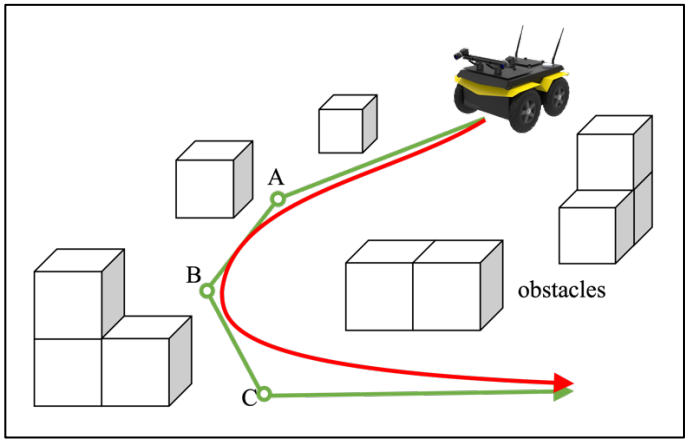


Figure 16. Path Tracking Between Waypoints. Source: Giesbrecht (2005).

4. Path Smoothing

The smoothness of a path is generally expressed in terms of continuity (Ravankar et al. 2018). The continuity ensures that the waypoints meet, and that the tangential vector's direction and magnitudes are both equal. A smooth and continuous path is desirable for UGS navigation as seen in Figure 17 as it allows the UGS to avoid abrupt and sharp turns without significantly reducing its velocity. Path smoothing must satisfy two constraints: (1) geometric continuity (i.e., tangential and curvature); and (2) safety distance, which ensures that the UGS maintains a distance away from any obstacle.



The green path is a piecewise linear path consisting of sharp turns at points A, B, and C while the red path is a smooth and continuous path.

Figure 17. An Illustration of Path Continuity.

A path can be fitted using a smoothing spline, which is a piecewise polynomial function that can have a locally very simple form, yet at the same time be globally flexible and smooth (Weisstein n.d.). Cubic splines yield to C^2 continuity approximation and are sufficiently smooth in the presence of small curvatures.

Considering $n+1$ ordered nodes $a = x_0 < x_1 < \cdots < x_n = b$ and the corresponding evaluations f_i in $[a,b]$, the aim is to construct a cubic spline interpolating those values. Since the it is on the third degree (i.e., cubic), its second-order derivative must be

continuous. By introducing the notation $f_i = s_3(x_i)$, $m_i = s_3'(x_i)$, $M_i = s_3''(x_i)$, the cubic spline interpolation is given by

$$s_{3,i-1}(x) = M_{i-1}\frac{(x_i - x)^3}{6h_i} + M_i\frac{(x - x_{i-1})^3}{6h_i} + C_{i-1}(x - x_{i-1}) + \tilde{C}_{i-1} \tag{29}$$

where $h_i = x_i - x_{i-1}$. Hence, Equation 29 can be rewritten as

$$\tilde{C}_{i-1} = f_{i-1} - M_{i-1}\frac{h_i^2}{6}, \quad C_{i-1} = \frac{f_i - f_{i-1}}{h_i} - \frac{h_i}{6}(M_i - M_{i-1}) \tag{30}$$

and M_i can be calculated from the M-continuity system

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i \tag{31}$$

D. MBSE

Traditionally, large systems adopted a document-based systems engineering approach where system requirements and designs are developed in documents and drawings (i.e., hardcopy or electronic), and exchanged among various stakeholders (i.e., customers, users, developers, and testers). This approach produces several documents making it difficult to assess completeness, consistency, and traceability. But with the rapid advancement of technology and computing power, model-based approach becomes more prominent in systems engineering and is expected to be the standard for systems engineering execution focusing on integrated modeling environment. (Beihoff et al. 2014).

Model-based systems engineering (MBSE) uses system models as part of the systems engineering process starting from the conceptual design phase, throughout the development phase, and later life cycle phases (International Council of Systems Engineering 2007). A system model generally consists of interconnected set of elements that supports system requirements, specification, design, analysis, verification, and validation information (see Figure 18). These elements represent key system parameters that are defined in Systems Modeling Language (SysML) including its structure, behavior, requirements, and parametrics.

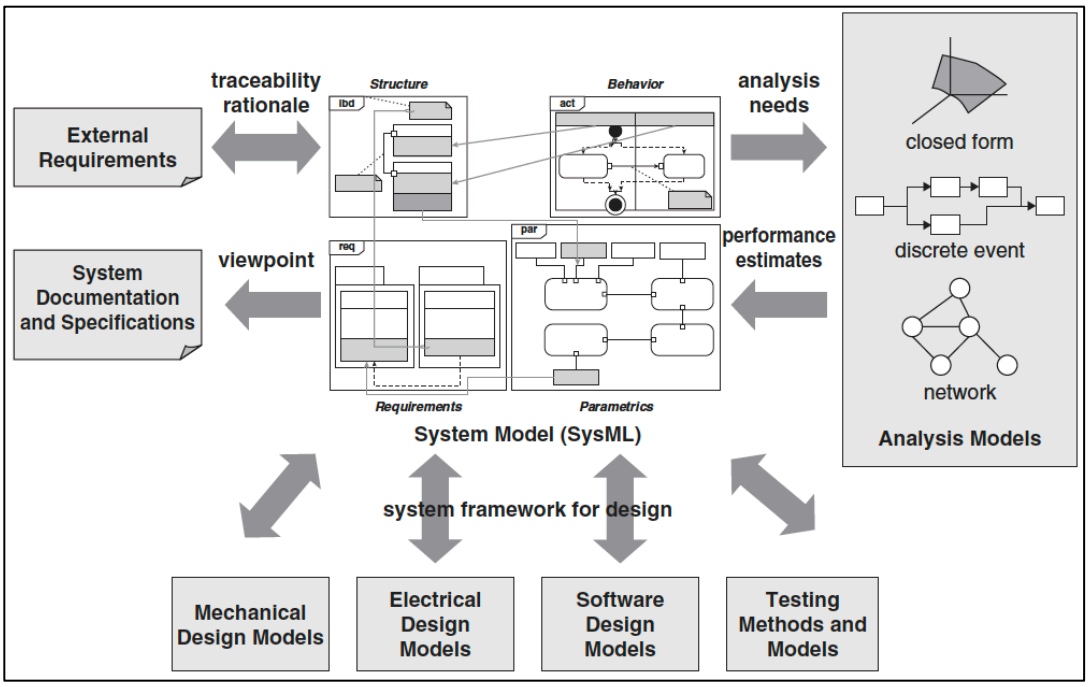


Figure 18. MBSE Integrating Framework. Source: Friedenthal et al. (2008).

System models are generally developed using MBSE tools like Cameo Systems Modeler. It provides a means to specify and integrate components, subsystems, and adjacent systems while maintaining traceability to higher-level requirements. It can also be integrated with engineering analysis and simulation models to perform computation and dynamic execution. If the system model is executed directly, the system modeling environment must be augmented with an execution environment like MATLAB/Simulink.

1. SysML Diagrams

Systems Modeling Language (SysML) is the de facto standard graphical modeling language of MBSE (PivotPoint Technology, n.d.) and has mappings to other frameworks such as Department of Defense Architecture Framework (DoDAF) (zud Muehlen 2012). It provides multiple ways to capture a system model without imposing a specific method (Friedenthal et al. 2008). For instance, system structures are traditionally analyzed using functional decomposition, but an alternative way is to adopt a use case diagram, which shows the system's functions and the interactions of its components based on a scenario. These two methods produce different diagrams but delivers the same intent.

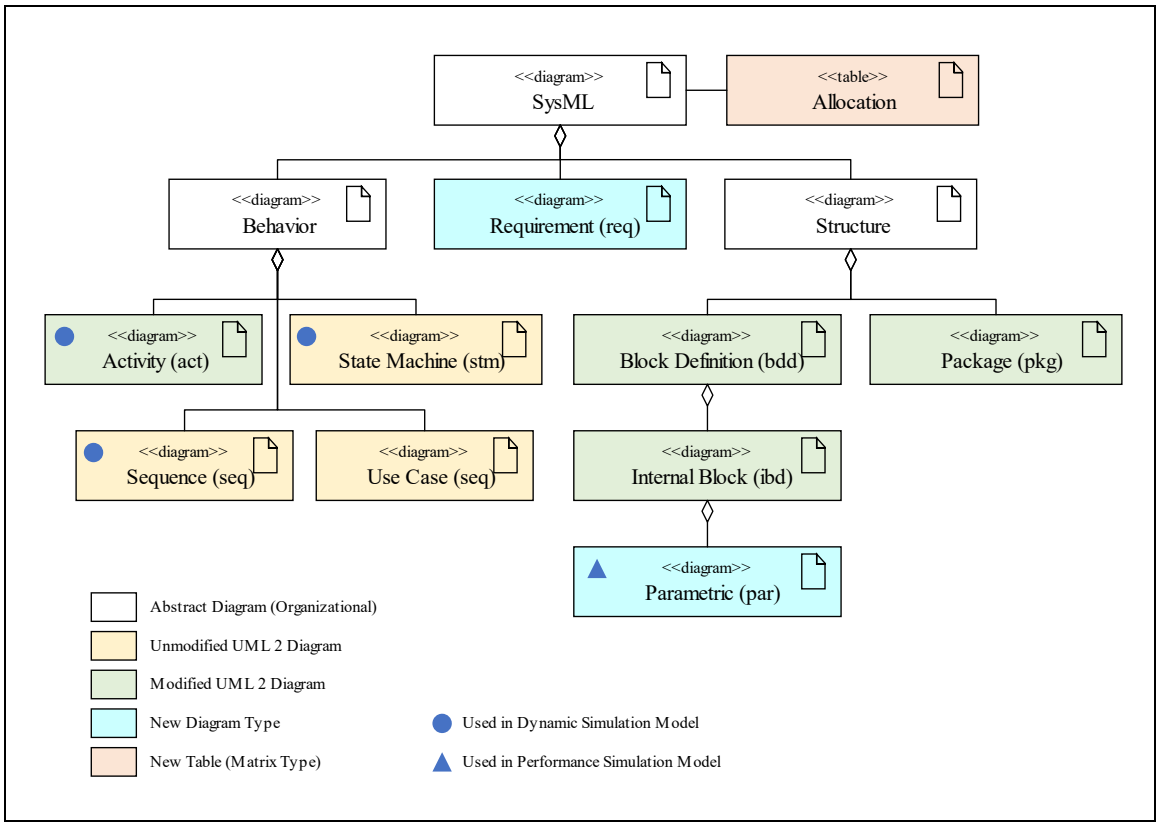


Figure 19. SysML Diagrams. Adapted from PivotPoint Technology (n.d.).

SysML is an extension of the Unified Modeling Language (UML). As shown in Figure 19, there are nine diagrams and one table in SysML with a few adopted from UML. The description of each diagram are as follows:

- **Package diagram** is used to organize the model and its elements. It provides views of a system from multiple levels of abstraction. It describes the dependencies between packages and their inner elements.
- **Block definition diagram** represents structure of the system using blocks as elements and the relationships with each other.
- **Internal block diagram** describes the internal structure (i.e., interaction and interface) of a block using properties and connectors.
- **Use case diagram** describes the functionality provided by the system using external entities (i.e., actors) to accomplish a set of goals.

- **State machine diagram** defines the system behavior as sequence of states that a component experiences in response to the events.
- **Activity diagram** shows a procedural flow of a system behavior based on the inputs, outputs, and control.
- **Sequence diagram** describes how a process is performed by a group of objects through a sequential set of exchanged messages between its parts.
- **Requirements diagram** describes the textual requirements and their relationship with other requirements to support requirements traceability.
- **Parametric diagram** models the system constraints using blocks of equations to support engineering analysis.
- **Allocation table** represents general relationships that map one model element to another, which could be in a form of a diagram, relation map, or a table.

2. Executable Modeling

System models can be used for building executable models by augmenting the static system modeling environment with an execution environment. Executable system models allow to understand and analyze the dynamics and/or performance of the system without manipulating it directly because the actual system may not exist or completely defined, or it cannot be physically tested due to certain constraints (e.g., costs, time, and resources). System models may be simulated using analytical models such as dynamic and performance simulation models (Friedenthal et al. 2008)

- **Dynamic simulation models** are discrete event simulations that are developed using activity, state machine, and sequence diagrams. It can be augmented by animation and other visualizations to step through the system behavior based on a pre-determined scenario or user input manipulation.
- **Performance simulation models** are continuous stimulations that are developed using parametric diagrams and mathematical models to analyze system performance.

3. Cameo-Simulink Integration Applications

Cameo can simulate a system model and validate its functionality or performance in an external simulation environment such as MATLAB/Simulink. When connection is established to a running MATLAB session, Cameo can have access to the variables in the MATLAB shared workspace. When a MATLAB function is called, Cameo passes the input parameters from the system model to MATLAB, executes the MATLAB/Simulink model, and gets the results back to Cameo including any output values and figures (see Figure 20).

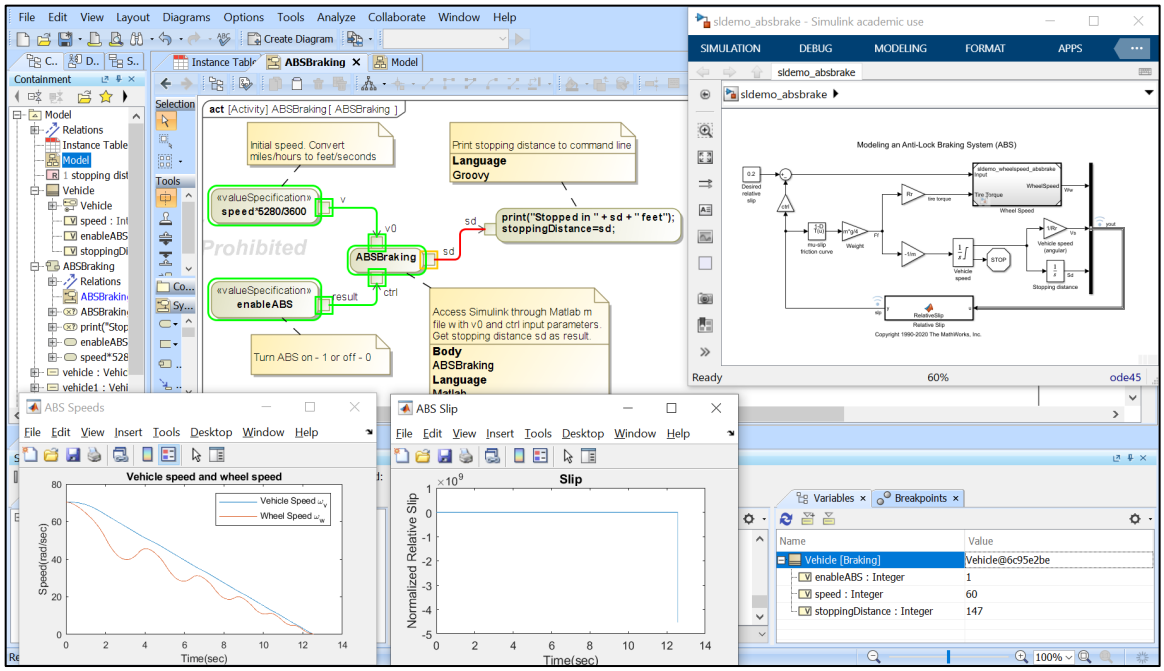


Figure 20. An Example of Cameo Simulation Executing a Simulink Model. Adapted from Pavalkis (2018).

Cameo also supports co-simulation with Simulink where a Simulink model (.slx) is converted into a block (i.e., Simulink block) and works as attached files (see Figure 21).

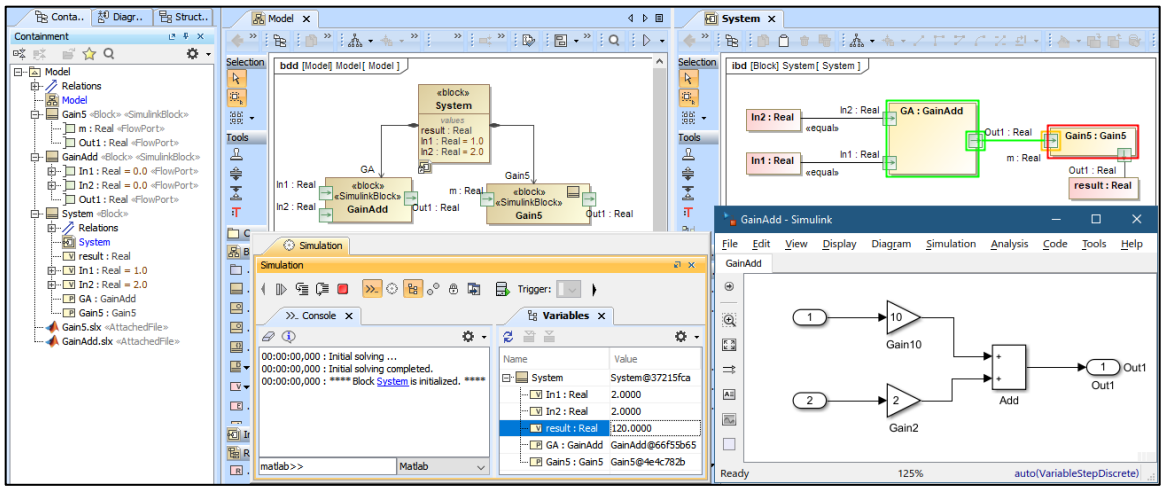


Figure 21. An Example of Co-Simulation in Cameo with Simulink Models. Source: No Magic (2020).

Cameo also provides an extendable model execution framework that provides basic GUI for users to manage the runtime of the executable system model in the context of realistic mockup of the intended user interface (see Figure 22).

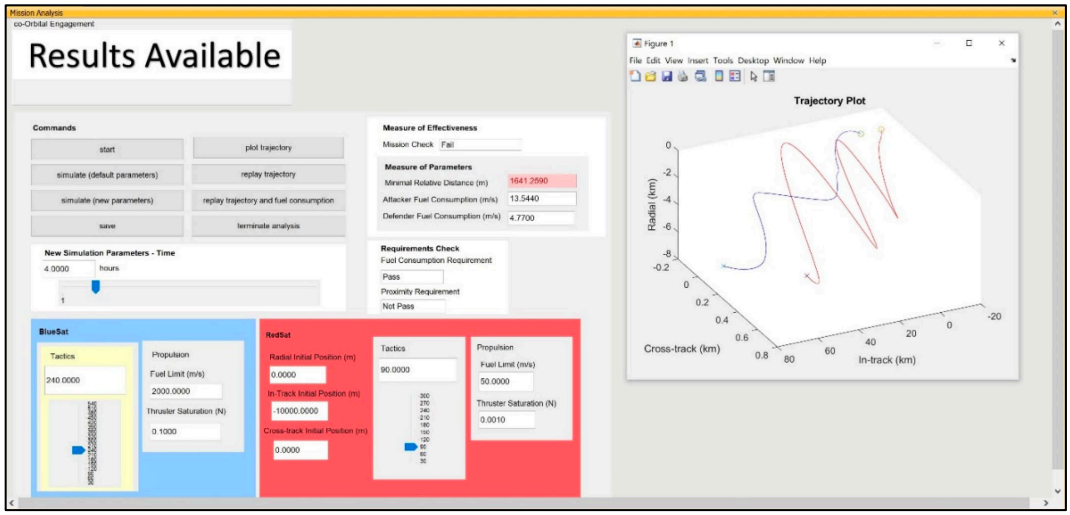


Figure 22. An Example of GUI in Cameo. Source: Rangel (2021).

III. MISSION ARCHITECTURE IN CAMEO

This chapter begins by introducing the MBSE tool used in developing the mission architecture of UAS-UGS teaming in IED clearance. Thereafter, it proceeds to describe the SysML diagrams which were developed. Finally, this chapter also demonstrates the creation and implementation of the executable systems model which will be integrated with MATLAB/Simulink environment.

A. CAMEO SYSTEMS MODELER

The UAS-UGS teaming's system architecture and simulation were modeled using Cameo, which is an MBSE tool that allows to construct SysML models and diagrams for systems engineering applications. The generated models and diagrams allow to graphically describe the concept of a complex system such as the UAS-UGS teaming.

Cameo has a simulation toolkit that allows to dynamically solve parametric models and measurements of effectiveness (MOEs). It also has a capability to integrate with third-party platforms such as MATLAB and Simulink to cross-evaluate system parameters based on specific requirements and visualize through simulation.

B. SYSTEM CONTEXT

The central idea is to utilize UAS and UGS to provide a safer, faster, and cheaper means to detect and neutralize IEDs and mines so as to facilitate the projection of the allied forces in reaching the main objective. Figure 23 illustrates the system context of the UAS-UGS teaming in IED clearance mission. The commander of the allied forces issues the mission to the operator to clear IEDs and create a safe pathway for the troops and vehicles. Using a portable ground control station (GCS), the operator is able to send commands and receive data from the UAS and UGS during the operation. Upon assigning the area of search by the operator, the UAS takes off, maps the area, and plans the mission with a flight path to optimize the IED search and detection. When an IED is detected, the UAS obtains the global positioning system (GPS) coordinates of the detected IED from the

communications satellite (Satcom) for accurate targeting and logging purposes. The UAV then sends the GPS coordinates to the UGV for IED neutralization.

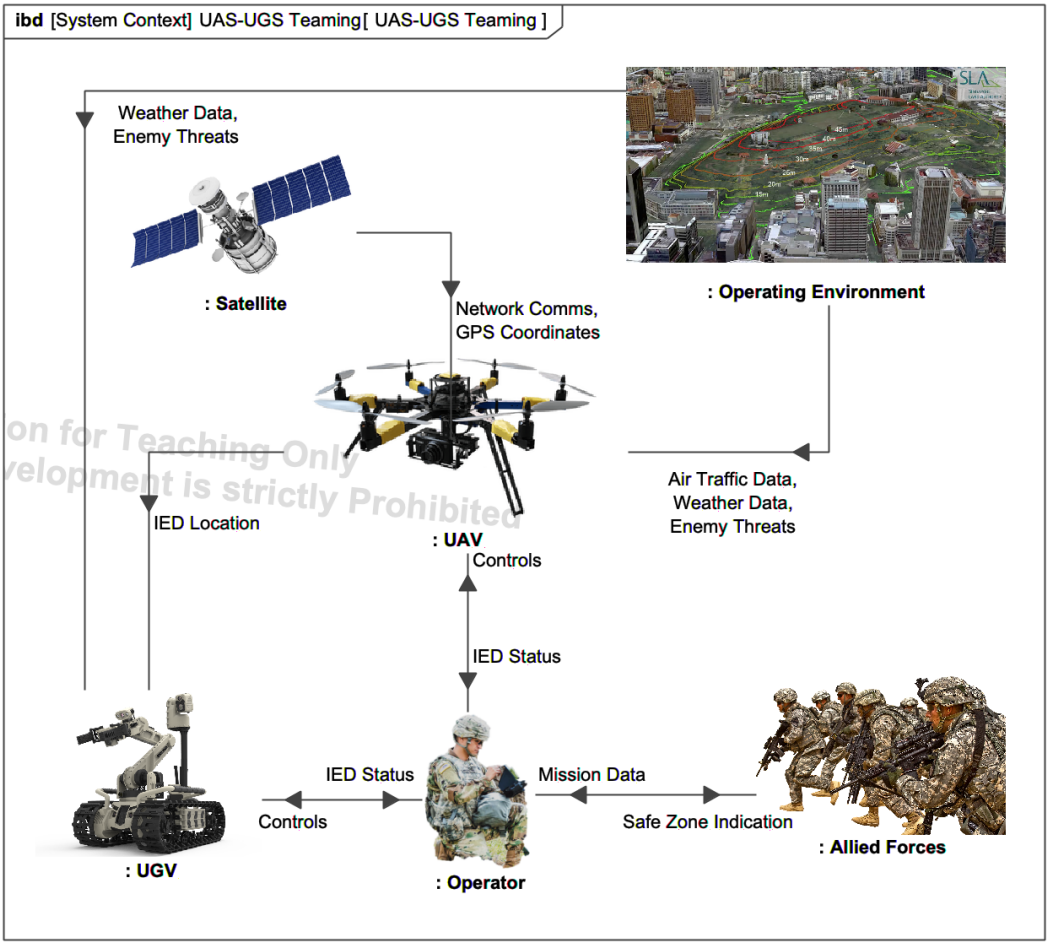


Figure 23. System Context of UAS-UGS Teaming in IED Clearance Mission.

When the UGV obtains the GPS coordinates from the UAV, it navigates to the detected IED for neutralization. Simultaneously, the UAV proceeds on to search for more IEDs, and this goes on until all IEDs are detected within the assigned area. Once all IEDs along the flight path are detonated, indicating that a safe pathway for troops and vehicles has been created, the operator then communicates with the allied forces to initiate their advancement to the next bound of the mission. Along the way, there could be external threats that will try to impede the advancement of allied forces by attacking the UAV or

UGV. Other factors that could affect the operation and performance of the UAV and UGV are air traffic, weather, and atmospheric conditions of the operating environment such as precipitation, wind, air temperature, and visibility.

C. OPERATION SITUATION SCENARIO

The main idea of this operational situation (OPSIT) scenario, as seen in Figure 24, is the information flow between the allied forces, users, and the tandem of UAS and UGS in a typical IED clearance mission.

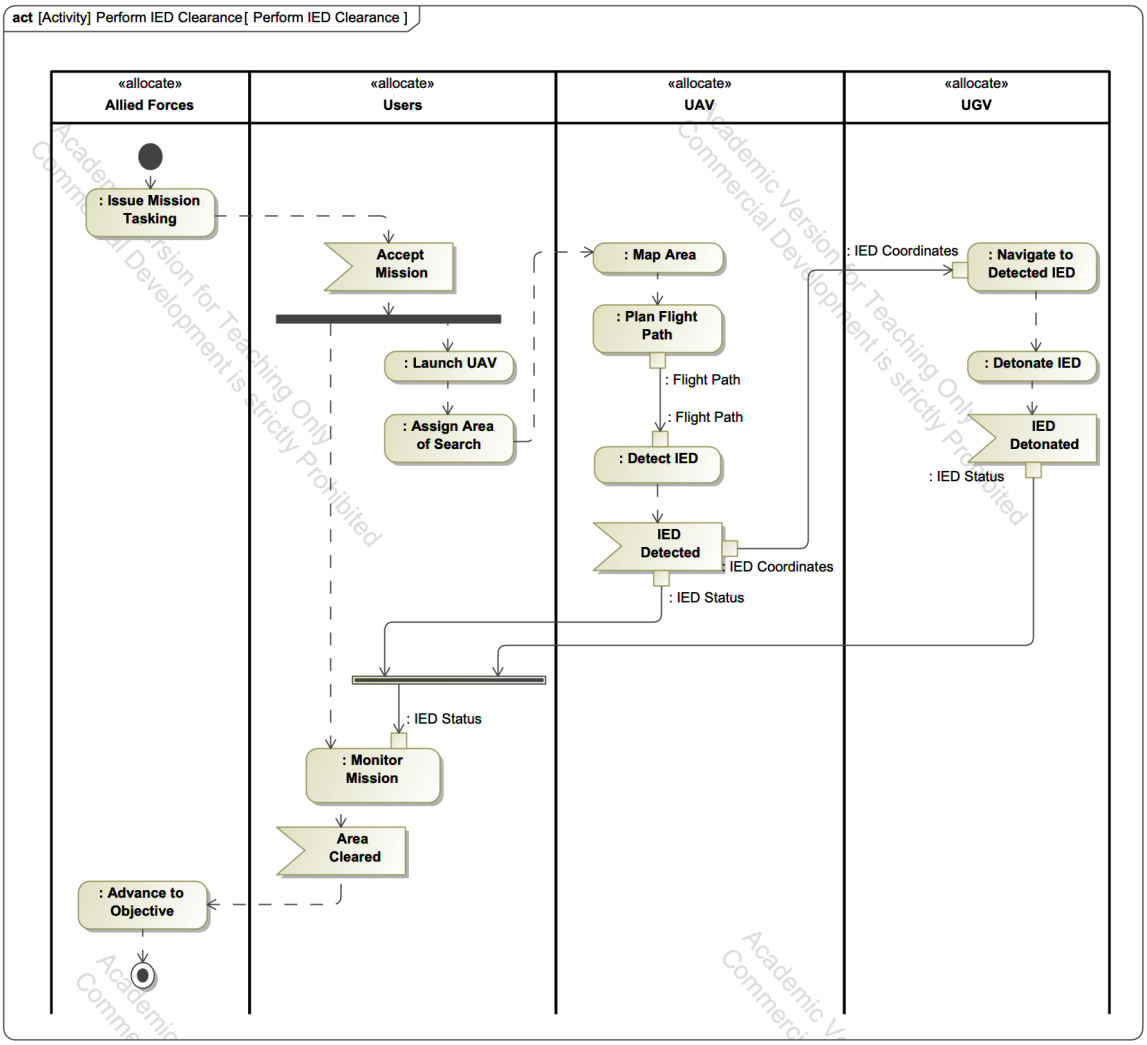


Figure 24. An OPSIT Scenario for UAS-UGS Teaming in a Typical IED Clearance Mission.

The allied forces initiate the mission by issuing the tasking to the operator. Upon acceptance of the mission, the operator launches the UAV and assigns the area of search based on suspected IEDs. The assigned area then triggers the UAV to map the area and plan its flight path based on the mapped area. The flight path then triggers the multirotor component of the UAV to navigate along the flight path and scans the area for IEDs. When an IED is detected, the coordinates of the detected IED are sent to the UGV. The UGV then navigates to the detected IED for detonation. All mission data is being monitored by the operator so that when the area is cleared from IEDs, the allied forces can be informed.

D. USE CASES

1. UAS

During an IED clearance mission, the operator can use the UAS to map the area and perform flight operations within the AO (Figure 25). The UAS is primarily used to perform IED clearance for the advancement of the allied forces. This function includes detecting IEDs and obtaining its location for UGV to detonate. For logistics support, the battery of the UAS can be charged by the maintainer at the charging station.

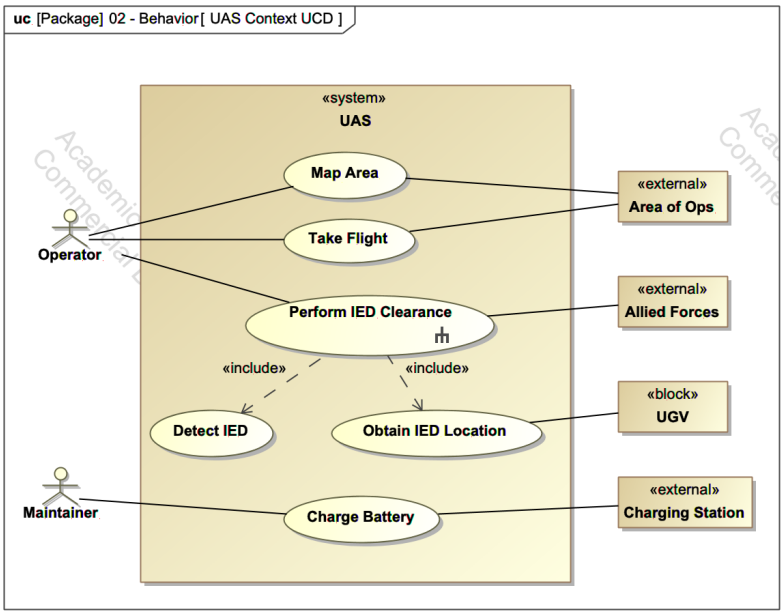


Figure 25. Use Case for UAS in UAS-UGS Teaming.

2. UGS

The UGS is primarily used to perform IED clearance for the advancement of the allied forces (Figure 26). This function includes navigating to the IED's location and neutralizing it. For logistics support, the battery of the UGS can be charged by the maintainer at the charging station.

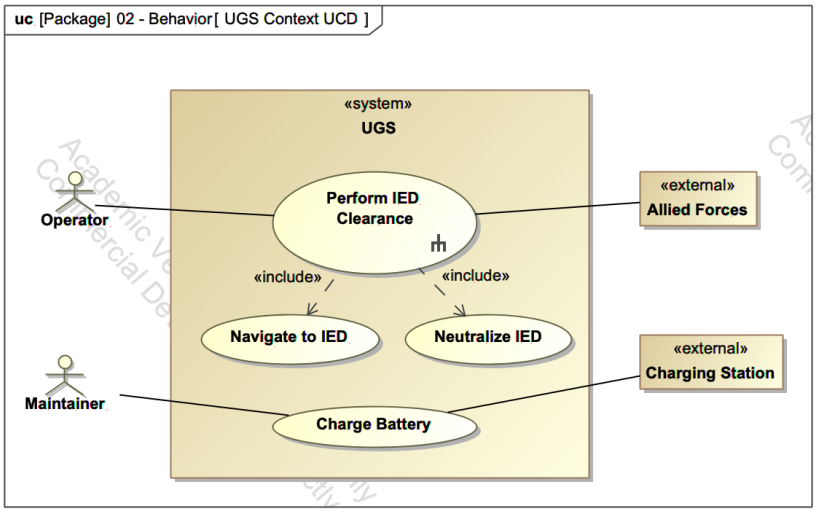


Figure 26. Use Case for UGS in UAS-UGS Teaming.

E. PHYSICAL DECOMPOSITION

The physical decomposition of the UAS-UGS teaming context, along with the external and adjacent systems, are illustrated in Figure 27. This context comprises of at least one UAS and at least one UGS managed by a single or multiple operator(s). Each UAS and UGS can have one or multiple UAV and UGV, depending on the scale of the mission. The direct users of the system are the operators and maintainers while the indirect, but the primary stakeholders, are the allied forces. The external systems include the satcom and the operating environment. The operating environment includes the AO and the base camp where the control station, charging station, and launch/landing site are located.

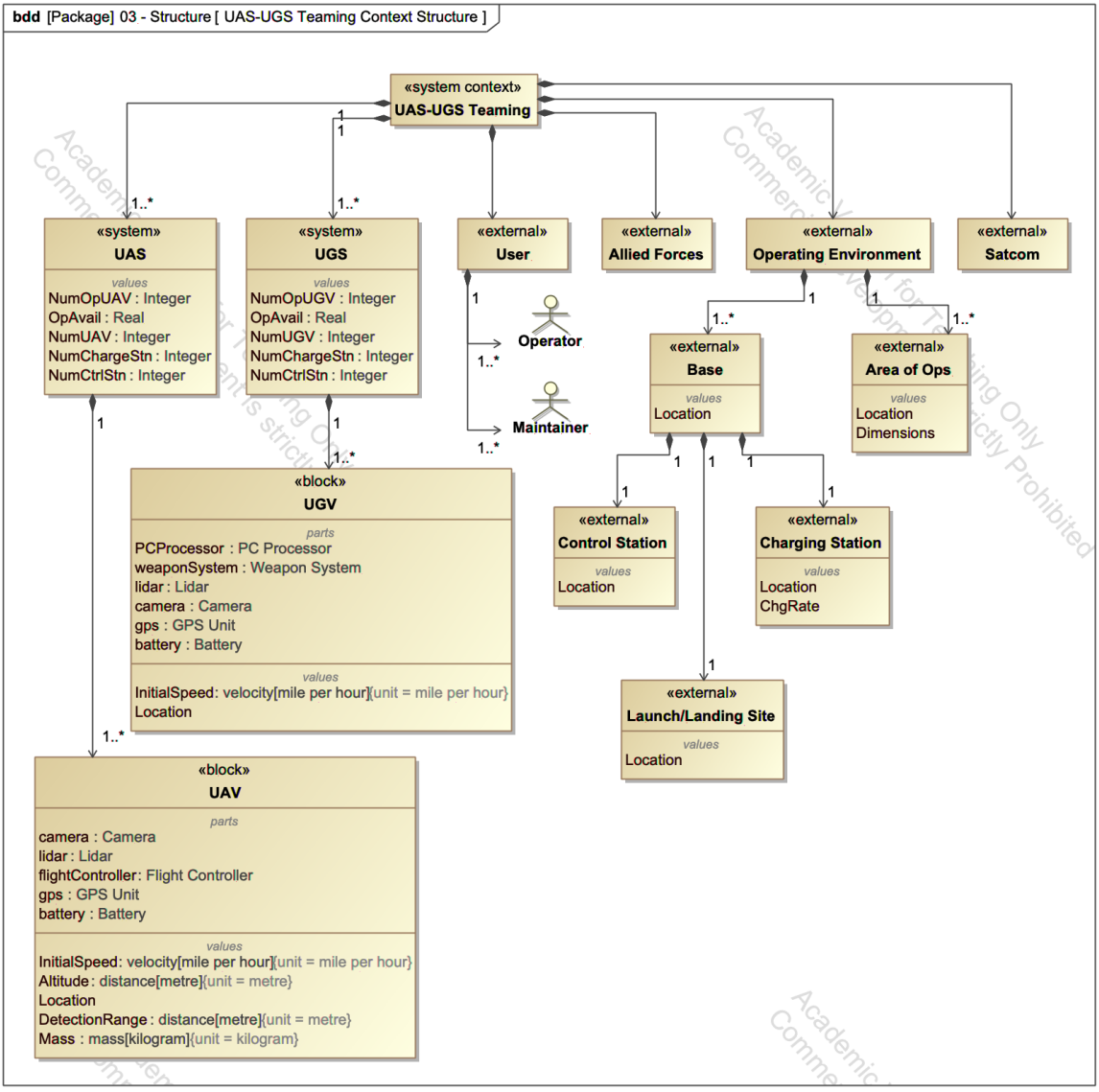


Figure 27. Physical Decomposition of UAS-UGS Teaming.

Within each block, the values of the selected parameters are included within. These parameters can be used for simulation and perform parametric analysis to evaluate the system performance.

F. CAMEO SIMULATION MODEL

A block diagram was developed to facilitate the simulation in Cameo (see Figure 28). Each block contains only the relevant values and states which corresponds to the

primary focus of the experiment. The UAV and UGV blocks are independent vehicles which could run simulations specific to their missions. The vehicle block integrates these independent vehicles and simulates the UAS-UGS teaming. Each block corresponds to a state machine which includes the relevant activities to simulate their missions (Figure 29).

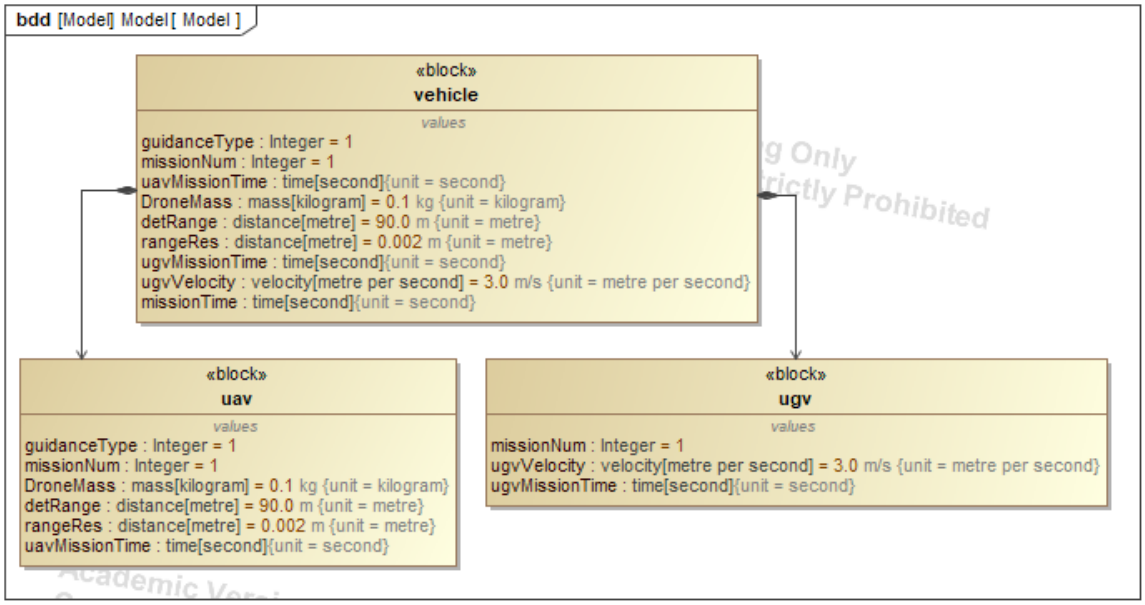
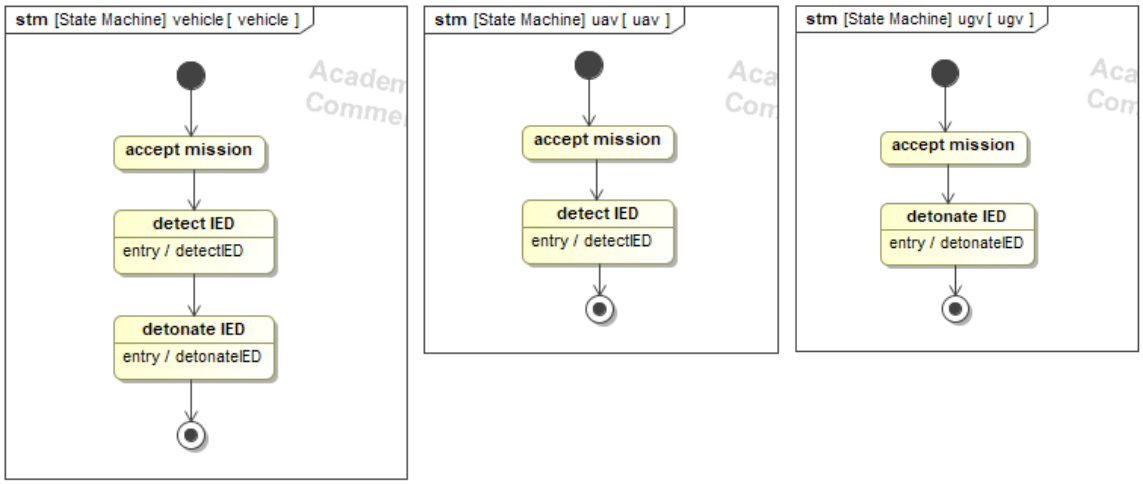


Figure 28. Simulation Model Block Diagram.



(a) (b) (c)

Figure 29. Simulation State Machines.

The simulation model developed in Cameo requires integration with Simulink. After integrating MATLAB in Cameo, the Simulink model can be run using an action block within the activity diagram. This action block accepts input value(s) and returns output value(s) corresponding to the input and output values of the Simulink model. This was incorporated within the IED detection (see Figure 30) and detonation (see Figure 26) activity diagrams that facilitate the missions of the UAV and UGV respectively.

As seen in Figure 30, the UAV mission block inputs the type of mission and UAV parameters, which are the test variables of the experiment, and returns the time it takes for the UAV to detect the IED. More details on the test variables are elaborated in Chapter V.

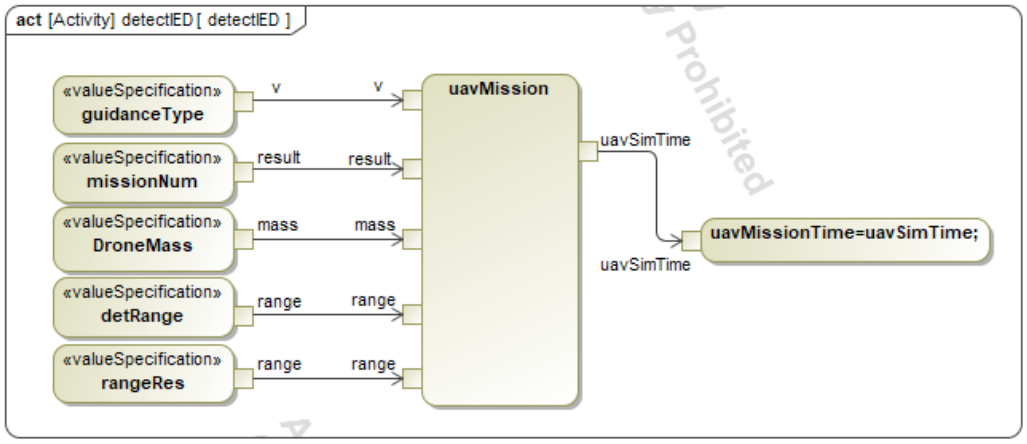


Figure 30. IED Detection Activity Diagram.

The UGV mission block inputs the mission type and the UGV's velocity. It then returns the time taken for the UGV to detonate the IED (see Figure 31).

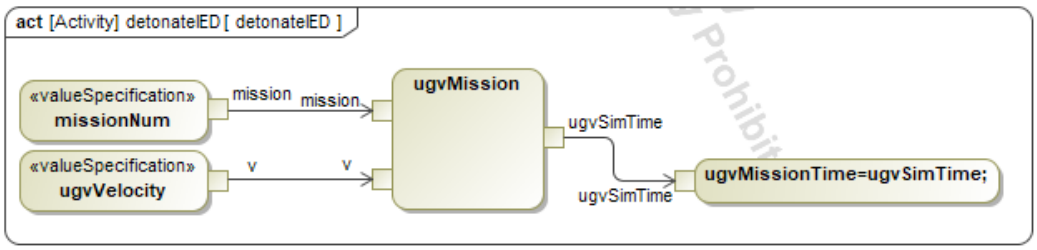


Figure 31. IED Detonation Activity Diagram.

G. MISSION TIME ANALYSIS

The mission time was analyzed by developing a block diagram (see Figure 32) and a parametric diagram (see Figure 33) that specifies the system requirement and constraints.

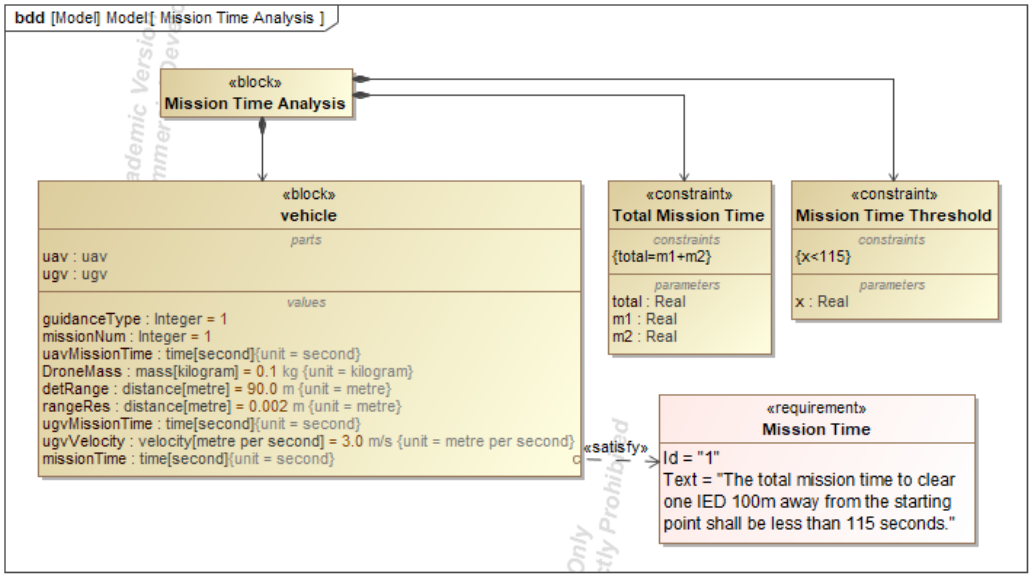


Figure 32. Mission Time Analysis of UAS-UGS Teaming.

As seen in Figure 33, the total mission time was calculated by adding the time that the UAV detects the IED and the time that the UGV detonates the IED. This is the case since the two missions are independently sequential. A threshold of 115 seconds for the total mission time can be included as a constraint in the analysis.

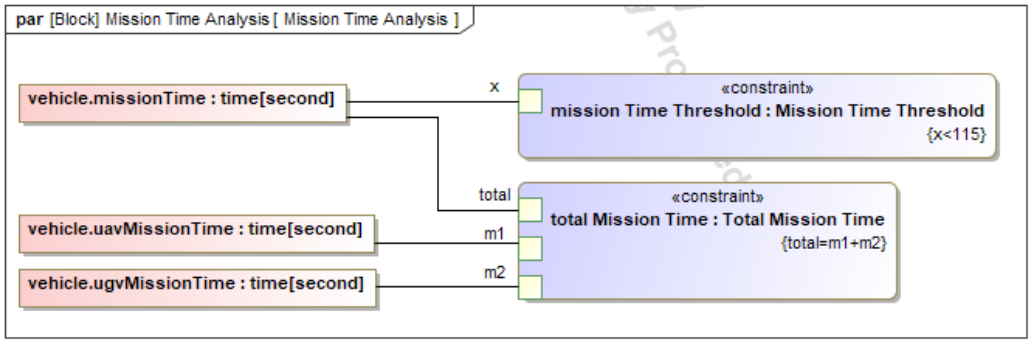


Figure 33. Mission Time Parametric Diagram.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. UAS-UGS MODELING IN SIMULINK

This chapter begins with introducing the external simulation environment used to integrate with Cameo for the analysis of UAS and UGS missions. After that, it presents the Simulink models of the UAS followed by the UGS, which were developed to model the UAS and UGS missions respectively.

A. SIMULINK

Simulink is a software in MATLAB which uses block diagrams while incorporating MATLAB codes and algorithms to develop system-level model-based designs and export simulation results (MathWorks n.d.). Simulink was used to evaluate the system parameters and visualize the results using two-dimensional (2D) plots and three-dimensional (3D) simulation. The input variables from Cameo were passed to Simulink, which runs in the background, and then returns the output variables back to Cameo for further analysis.

B. UAS SIMULINK MODEL

The Simulink model used to simulate the UAS was adopted from the "UAV Package Delivery Example" developed by MathWorks. This example demonstrates how a small quadcopter can take off, navigate, and land to any point within the environment which simulates the delivery of a package. The simulation was modified and reconfigured to meet the test requirements and enable the integration with Cameo. The simulation can be configured to either be fully guided by multiple waypoints or autonomously avoid obstacles until it reaches the destination. As seen in Figure 34, the top-level view of this model includes the quadcopter and the ground control station (GCS) it takes command inputs from. The quadcopter itself has the following subsystems: (1) external sensors; (2) on-board computer; and (3) multirotor.

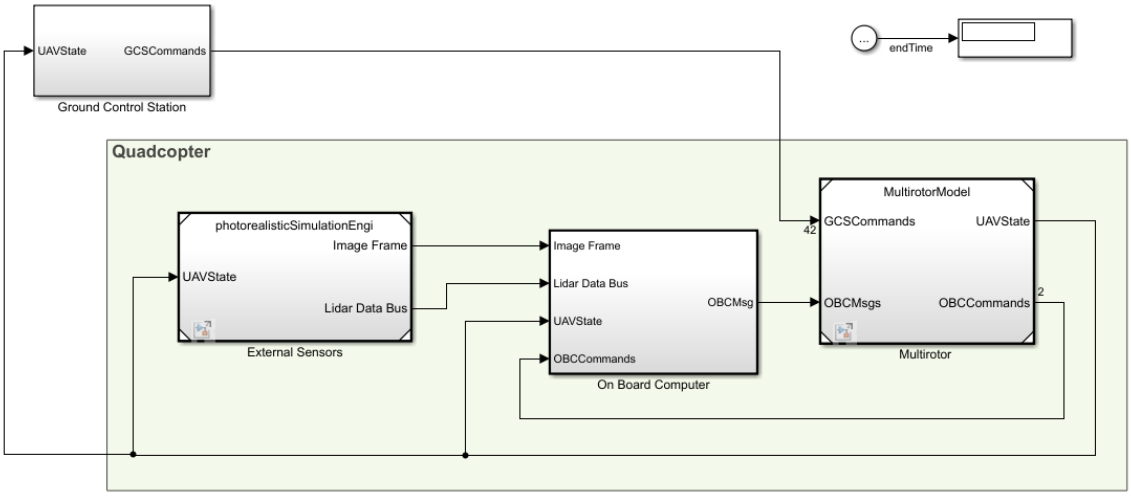


Figure 34. Top-Level View of the UAS Simulink Model. Adapted from MathWorks (2020).

C. GROUND CONTROL STATION MODEL

The GCS in the model simulates the human control of the quadcopter during the operation. The model can be configured to assign waypoints programmatically within MATLAB or dynamically assign waypoints using an external ground station software called QGroundControl (QGC) to simulate an operational ground control console.

1. Using Programmatically Assigned Waypoints

Waypoints can be assigned programmatically to the GCS by passing a structure array with three fields: (1) mode; (2) position; and (3) params. The "mode" is an integer which specifies the quadcopter to take off, navigate or land to the waypoint. The "position" is a 3-by-1 vector which specifies the [x,y,z] parameters of the waypoint. The "params" is a 4-by-1 vector which specifies the [roll, pitch, yaw, time] parameters of the quadcopter.

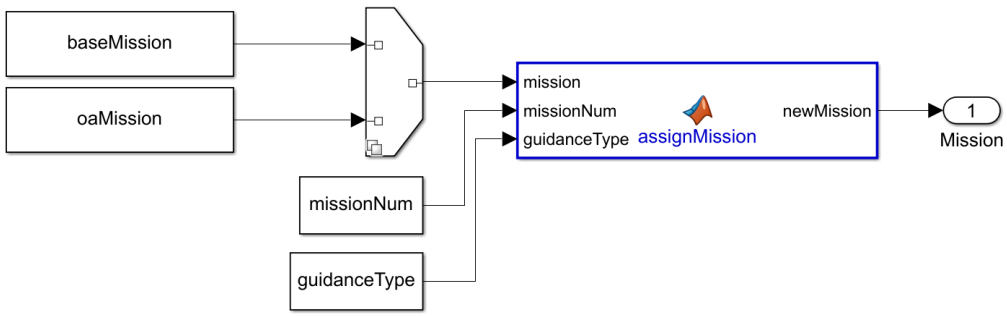


Figure 35. Variation of Missions in GCS Model. Adapted from MathWorks (2020).

As seen in Figure 35, the missions loaded in the GCS can be configured for (1) base or (2) obstacle avoidance. In a base mission, the quadcopter takes off vertically at a high altitude surpassing the highest building and flies directly to the waypoint. In an obstacle avoidance mission, the quadcopter flies at low altitude and autonomously navigate to the waypoint while avoiding any obstacles.

2. Using QGroundControl

QGroundControl (QGC) is an external software which provides the interface to plan mission and control the flight of the quadcopter. The GCS model uses Micro Air Vehicle Link (MAVLink) Interface to enable the communication between QGC and Simulink, as seen in Figure 36. The mission items from the QGC pass through a series of MATLAB algorithms which translates the mission into a waypoint stream that is readable by the Path Manager at the Multirotor subsystem.

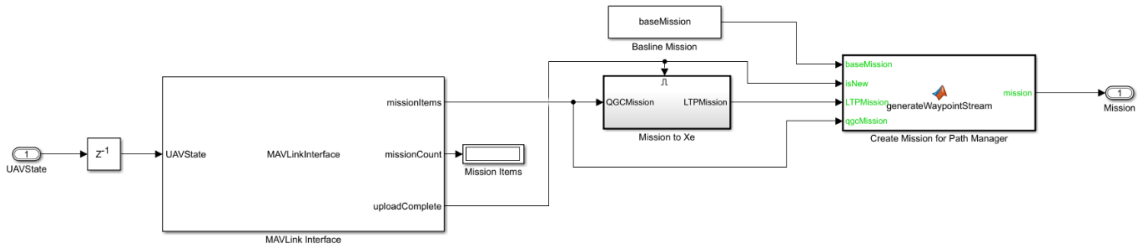


Figure 36. QGC-Enabled GCS Model. Adapted from MathWorks (2020).

D. EXTERNAL SENSORS MODEL

The quadcopter in the model is equipped with external sensors such as (1) Lidar for obstacle detection and (2) camera for visualization during the simulation. The model can be configured to simulate using (1) 3D plot or (2) photorealistic environment.

1. 3D Plot

Simulation using 3D plot runs and generates the results significantly faster than in the photorealistic environment, which makes it more ideal when varying the system parameters in Cameo. As seen in Figure 37, the flight path and lidar sensor of the quadcopter are visualized by feeding its position [x,y,z] and control [roll, pitch, yaw] into the UAV Animation and Scenario blocks respectively.

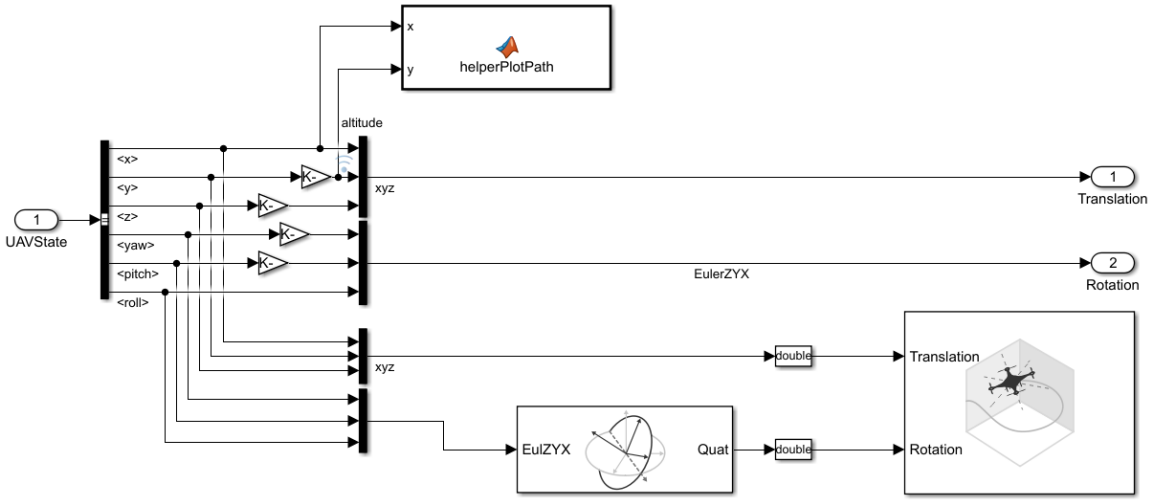
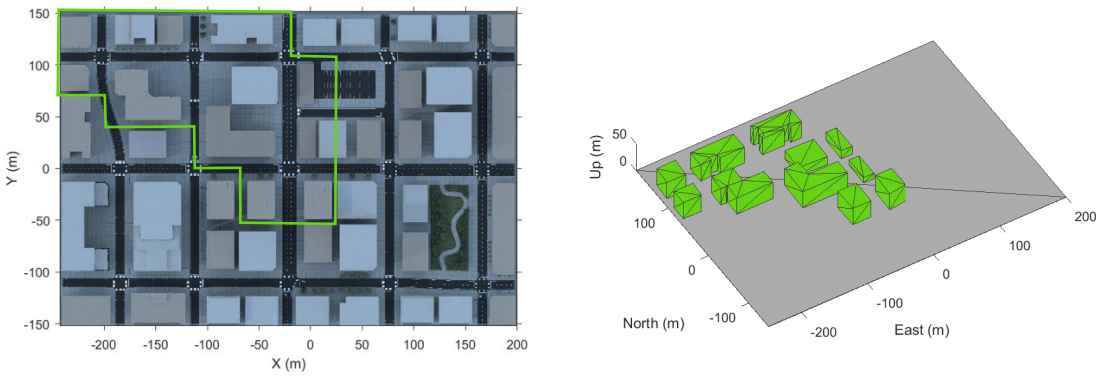


Figure 37. 3D Plot in External Sensors Model. Adapted from MathWorks (2020).

The obstacles in the environment are simulated using polygon meshes in a cuboid scenario (MathWorks n.d.). As seen in Figure 38, a portion of the city block scene (a) in Simulink consisting of 11 buildings were modelled (b).



(a) (b)

Figure 38. Cuboid Scenario Model.

2. Photorealistic Environment

Simulation in photorealistic environment requires more time to run and generate results as it visualizes the quadcopter in a more realistic world. As seen in Figure 39, the model comprises of (1) 3D Simulation Environment; (2) Camera; and (3) Lidar.

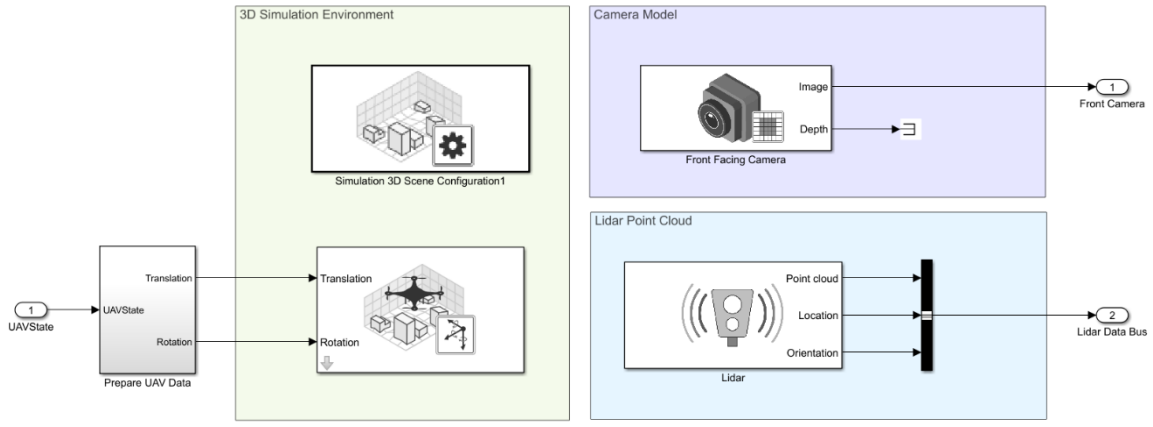


Figure 39. Photorealistic Environment in External Sensors Model. Adapted from MathWorks (2020).

The simulation environment of the model is configured to visualize a pre-built 3D scene called U.S. City Block, which is rendered using Unreal Engine by Epic Games

(MathWorks n.d.). The quadcopter can interact with the 3D simulation environment including the detection of obstacles (e.g., buildings, traffic lights, ground). External sensors such as the camera and lidar allows to obtain a front-facing view and to sense any obstacles respectively.

E. ON-BOARD COMPUTER MODEL

The On-Board Computer (OBC) model implements a series of MATLAB algorithms to process sensor data such as image frame, lidar data, UAV state, and OBC commands as seen in Figure 40. Visualizing the Lidar point cloud, obstacle range, and front-facing camera can be configured to be shown or hidden.

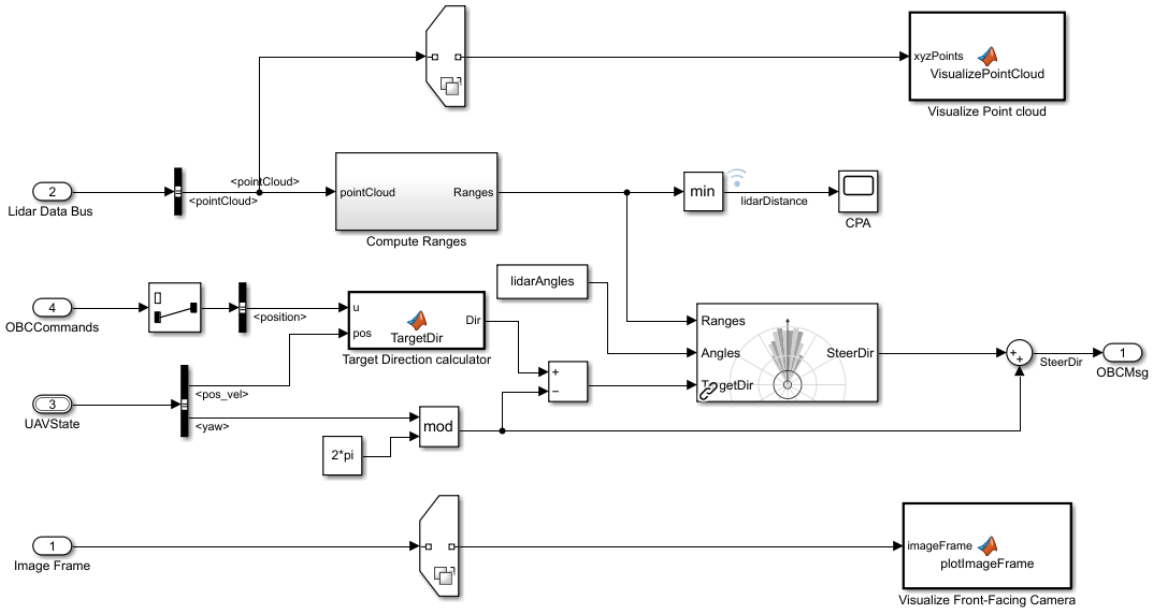


Figure 40. On-Board Computer Model. Adapted from MathWorks (2020).

F. MULTIROTOR MODEL

The Multirotor model controls the behavior of the quadcopter including the guidance logic and the inner loop and plant model (see Figure 41).

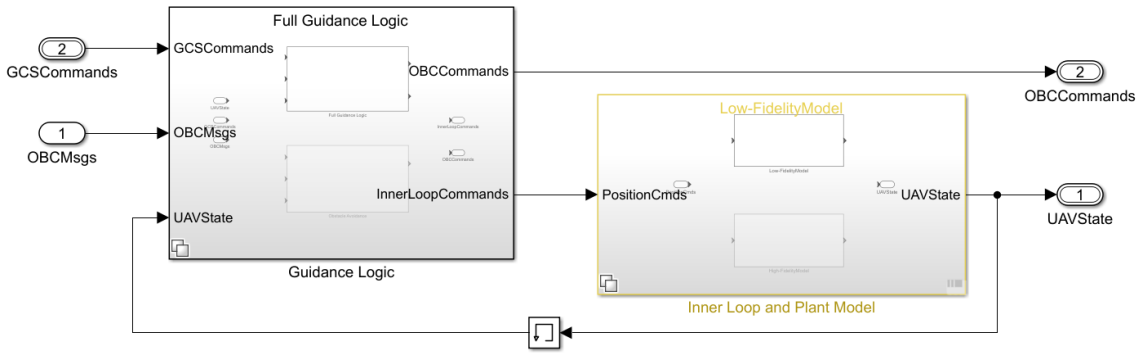


Figure 41. Multirotor Model. Adapted from MathWorks (2020).

1. Guidance Logic

The guidance logic of the model can be configured to be (1) fully guided by multiple waypoints or (2) autonomously avoid obstacles until it reaches the destination.

a. Full Guidance

The model can determine the active flight path using the waypoint stream that passes through the UAV Path Manager and then to the Guidance Mode Selector, which generates the necessary flight controls (i.e., take off, navigate, land), as seen in Figure 42. When the UAS has landed, the simulation stops and the simulation time will be recorded.

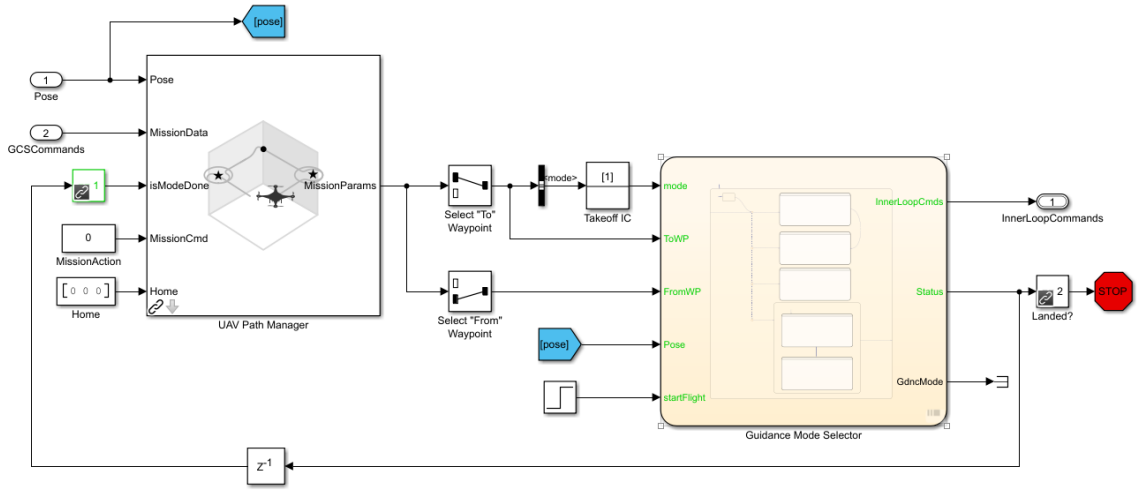


Figure 42. Full Guidance Logic Model. Adapted from MathWorks (2020).

b. Obstacle Avoidance

When obstacle avoidance is enabled, the quadcopter attempts to fly directly towards the waypoint in a straight path but finds and navigates towards an alternative path when the distance to the obstacle gets closer to the quadcopter. As seen in Figure 43, the model uses a Waypoint Follower block which uses Pure Pursuit algorithm to determine the lookahead point based on the current UAS position and lookahead distance. When the UAS has landed, the simulation stops, and the simulation time will be recorded.

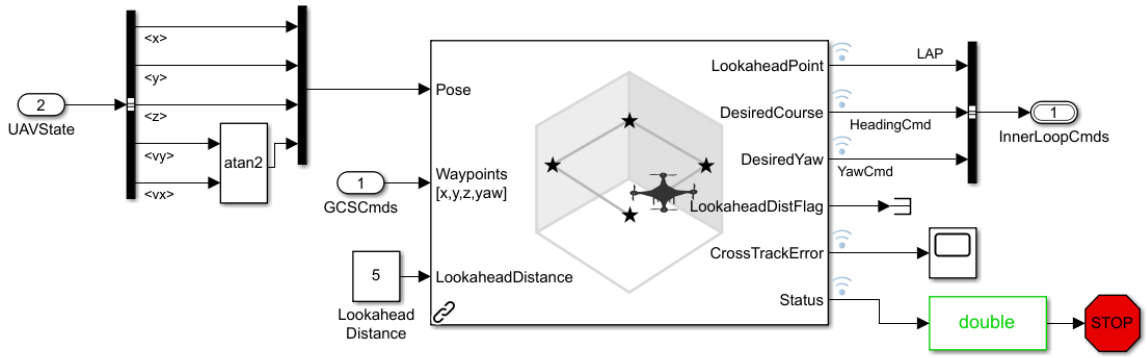


Figure 43. Obstacle Avoidance Guidance Logic Model. Adapted from MathWorks (2020).

2. Inner Loop and Plant Model

The Inner Loop and Plant Model allows the quadcopter to be configured with a low- or high-fidelity.

a. Low-Fidelity Plant Model

The quadcopter with low fidelity uses a reduced-order model which approximates the quadcopter's behavior while in autopilot based on the position, velocity, and attitude stabilization controls as seen in Figure 35.

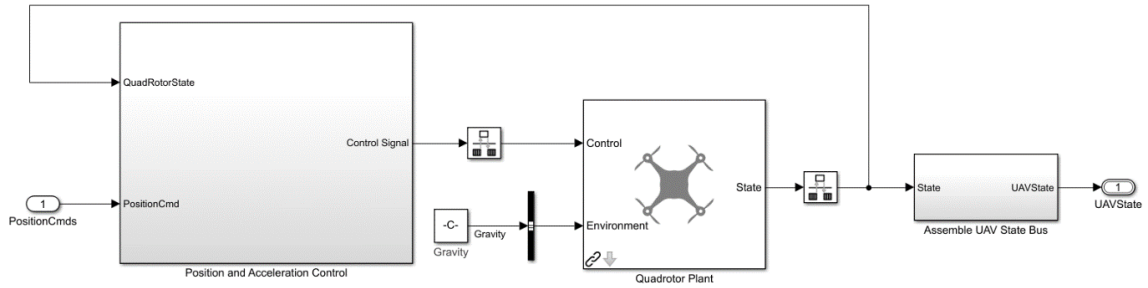


Figure 44. Low Fidelity Plant Model. Adapted from MathWorks (2020).

The Quadrotor Plant represents the guidance model that estimates the UAV state based on the control and environmental inputs, which includes the Euler angles (ϕ,θ,ψ), body angular rates (p,q,r), and thrust (f).

b. High-Fidelity Plant Model

The quadcopter with high fidelity uses a 6-DOF block which integrates 6 degrees-of-freedom (DOF) equations of motion with respect to the quadcopter's axis (MathWorks, n.d.), as seen in Figure 45. The model also takes the weather in consideration by factoring in atmospheric parameters such as air temperature, pressure, and air density.

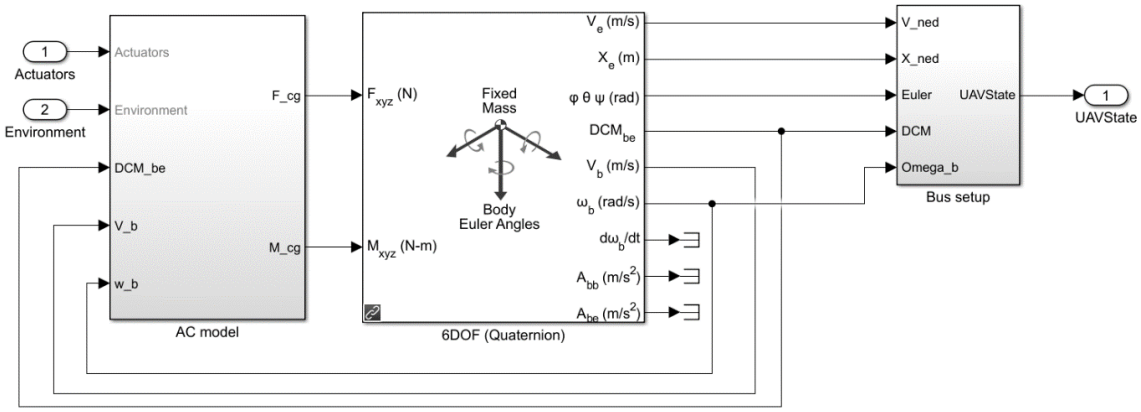


Figure 45. High-Fidelity Plant Model. Source: MathWorks. (2020).

G. UGS SIMULINK MODEL

The Simulink model used to simulate the UGV was adopted from the "Lidar SLAM in 3D Simulation" developed by MathWorks. This example demonstrates how a small UGS can move to any waypoint and record synthetic data from a 3D simulation environment. As seen in Figure 46, the top-level of this model is similar to the Photorealistic Environment of the UAS Simulink model (Figure 25), which includes the following subsystems: (1) 3D Simulation Environment and (2) External Sensors.

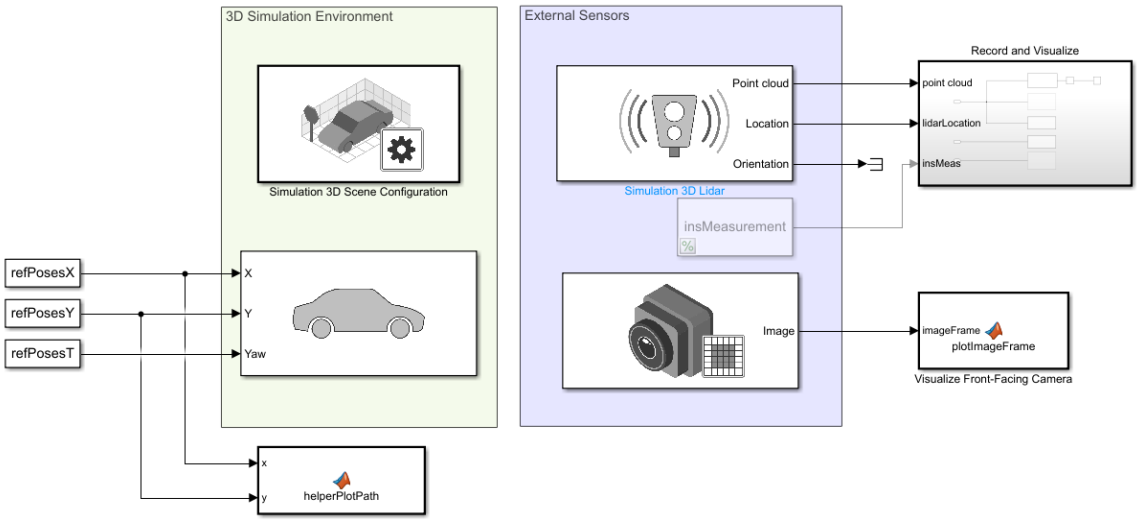


Figure 46. Top-Level View of UGS Simulink Model. Adapted from MathWorks (2019).

The trajectory path for the UGS was pre-determined based on the detected targets of the quadcopter from the UAS Simulink model. The "Select Waypoints for Unreal Engine Simulation" example developed by MathWorks was used to interactively select a series of waypoints and generate the trajectory path which is fed into the UGS Simulink model.

The MATLAB function launches an interactive app for drawing a series of waypoints in the U.S. City Block scene. The selection begins by clicking a point in the interactive app, which serves as the start point. A path is created as a polyline, consisting of multiple points. The function exports two sets of variables in the workspace: (1)

waypoints, an N-by-2 matrix of (x, y) waypoints in world coordinates format; and (2) path poses, an N-by-3 matrix of (x, y, θ) poses for each waypoint.

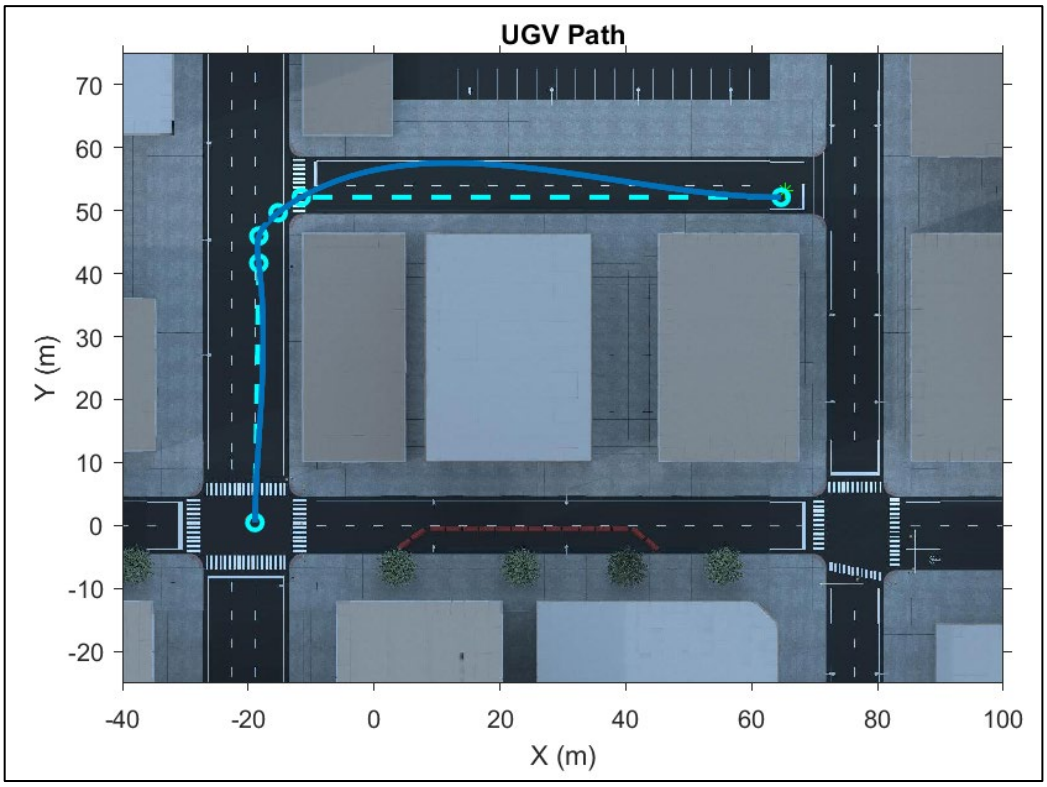


Figure 47. Interactive App for Drawing Waypoints.

The sequence of poses is then transformed from a C^1-continuous vehicle path to a C^2-continuous path, which smoothens the path of the UGS using cubic spline interpolation (see Figure 47). The light blue denotes the piecewise linear path from the generated waypoints while the dark blue denotes the smooth path using cubic spline interpolation. This allows the UGS to move along the trajectory with a constant velocity. The pre-determined waypoints are then loaded into the 3D Vehicle with Ground Following block of the UGS Simulink model.

THIS PAGE INTENTIONALLY LEFT BLANK

V. TEST CASES AND TEST VARIABLES

This chapter provides the scenario and assumptions of the test approach. After that, it proceeds to describe the design of the simulation environment. Finally, this chapter outlines the test cases and input parameters of the UAS followed by the UGS..

A. SCENARIO

To investigate the possibility of UAS-UGS teaming in IED clearance, the experiment was designed to simulate a scenario where the operator launches the UAS to detect and sends the UGS to detonate one IED per mission. The performance measure for this experiment is the time taken for each mission starting from the launch of the UAS until the UGS reaches the target.

The scenario begins when the commander of the allied forces sends the mission to the operator that there is a suspected IED within a complex urban environment. The operator then sends the UAS to confirm the presence of the IED by taking off, navigating across the varying levels of buildings, and landing on the target. Upon landing of the UAS, the operator then sends the UGS to navigate to the detected IED and detonate it.

B. ASSUMPTIONS

The following assumptions were made in the scenario when running the test cases:

- Each mission includes the GPS coordinates of the suspected IED, which is simulated by a waypoint.
- The overall mission time starts when the UAS takes off and ends when the UGS reaches the target.
- Other than the physical obstacles in the 3D scene (i.e., buildings, traffic lights, barricades), the external threats and weather in the operating environment are optimal and have no effect to the UAS and UGS.
- IED detection is simulated by landing the UAS to the target. The effectiveness of the IED detection and detonation are not tested.

- The UGS's trajectory is restricted along the road for better maneuverability.

C. SIMULATION ENVIRONMENT

The experiment was designed to simulate missions of UAS and UGS sequentially in the same 3D environment, namely "US City Block." As seen in Figure 48, there were a total of 15 missions in this experiment with varying complexities: 8 of which were 100 m away from the starting, while 7 were 150 m away. The locations of the IEDs were determined based on the complexity of the flight path and vehicular trajectory of the UAS and UGS respectively. The red dot at the center denotes the starting point of the UAS and UGS. The red and blue circles denote 100m and 150m away from the starting point respectively. The green points denote the locations of the targets (i.e., IED or mine).

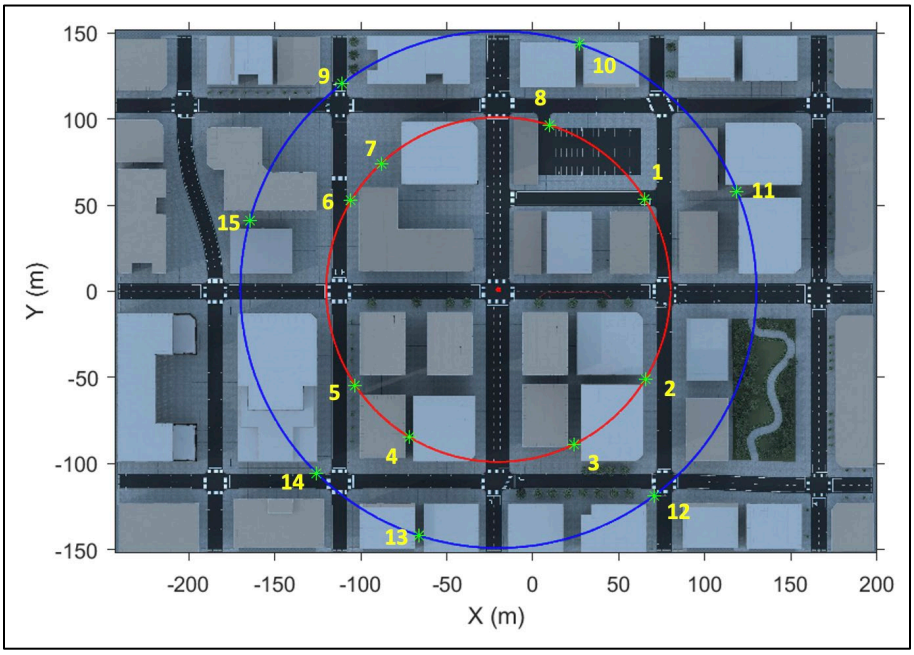


Figure 48. Plot of IEDs in the 3D Simulation Environment.

The starting points of the UAS and UGS is both at [-20, 1] as demarcated with a red dot in Figure 48. The coordinates of the location of each IED are listed in Table 2.

Table 2. Coordinates of IEDs in the 3D Simulation Environment.

Mission (100m)	XY-Coordinates	Mission (150m)	XY-Coordinates
1	[65.2525, -53.2687]	9	[-110.683, -120.485]
2	[65.6067, 50.6865]	10	[27.2984, -120.485]
3	[24.1806, 88.7111]	11	[-118.736, -58.0283]
4	[-71.6184, 84.6478]	12	[70.4924, 118.629]
5	[-103.183, 54.5037]	13	[-66.0999, 141.74]
6	[-105.689, -52.5501]	14	[-125.746, 105.385]
7	[-88.1056, -74.2231]	15	[-164.502, -41.2399]
8	[9.62809, -96.5101]		

D. UAS TEST VARIABLES

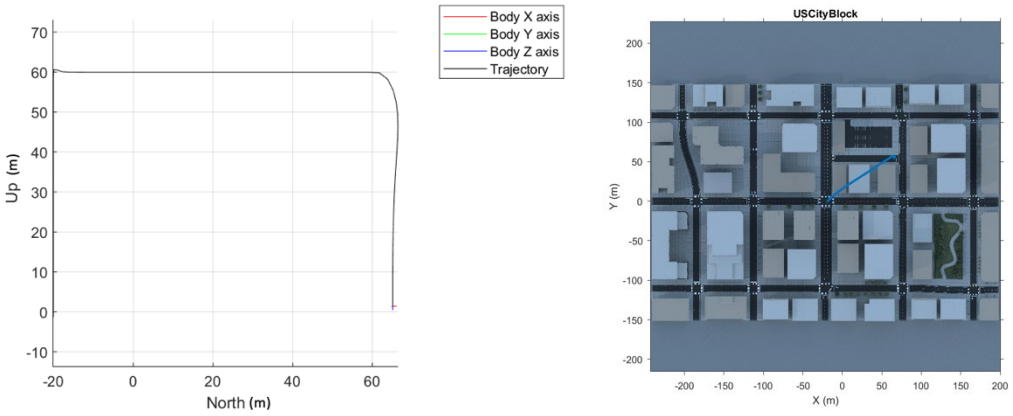
There is a need to determine the test variables which have some effects to the mission time in order to identify the optimal configurations for the UAS. These test variables shall be comparatively observed and statistically assessed for analysis.

1. Guidance Logic

The guidance logic of the UAS shall be assessed to determine if the UAS should be fully guided by the operator or be flown autonomously.

a. Full Guidance

The condition for testing the full guidance logic is that the UAS shall vertically take off and directly fly toward the target at a constant high altitude (see Figure 49a) just above the tallest building along its flight path (see Figure 49b). Flying at a direct flight path towards the target may be an advantage as it should take a shorter time for the UAS complete the mission. However, taking off and landing at a significantly high altitude may add a time penalty into the overall mission time.

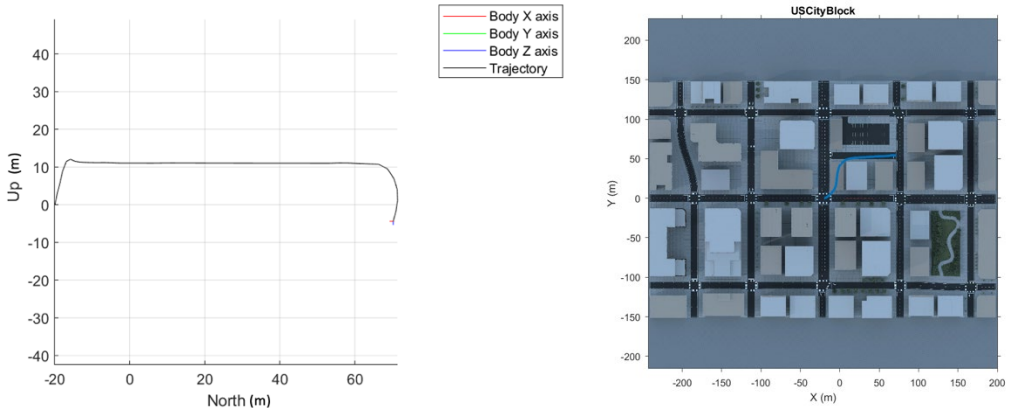


(a) (b)

Figure 49. Trajectory of Fully Guided UAS in Mission 1.

b. Obstacle Avoidance

The condition for testing the obstacle avoidance logic is that the UAS shall autonomously navigate towards the target while avoiding any obstacle at a constant low altitude (see Figure 50a). Flying at a low altitude may be an advantage but avoiding obstacles (see Figure 50b), especially highly complex ones, may require more time to navigate through.



(a) (b)

Figure 50. Trajectory of Autonomous UAS in Mission 1.

2. Detection Range

There are several factors to consider in detecting obstacles, including the altitude, speed, and line spacing, as well as other factors like weather conditions and structures within the operating environment. Man-made structures, such as buildings or traffic lights, tend to absorb great amount of light which can dramatically affect how the laser is bounced back to the sensor. This results in replanning flight paths since collected readings are more accurate when flying at lower altitudes (Halsey 2021). A study has shown that the optimum obstacle detection ranges for a low-flying UAS-mounted lidar are 60–110 m, depending on the speed of the UAS and the type of lidar used (Goodin et al. 2021).

For this simulation, the condition for testing the detection range is that the guidance logic of the UAS shall be obstacle avoidance. The distance of the lidar to the detected obstacle shall be plotted and observed. An example of this plot is shown in Figure 51.

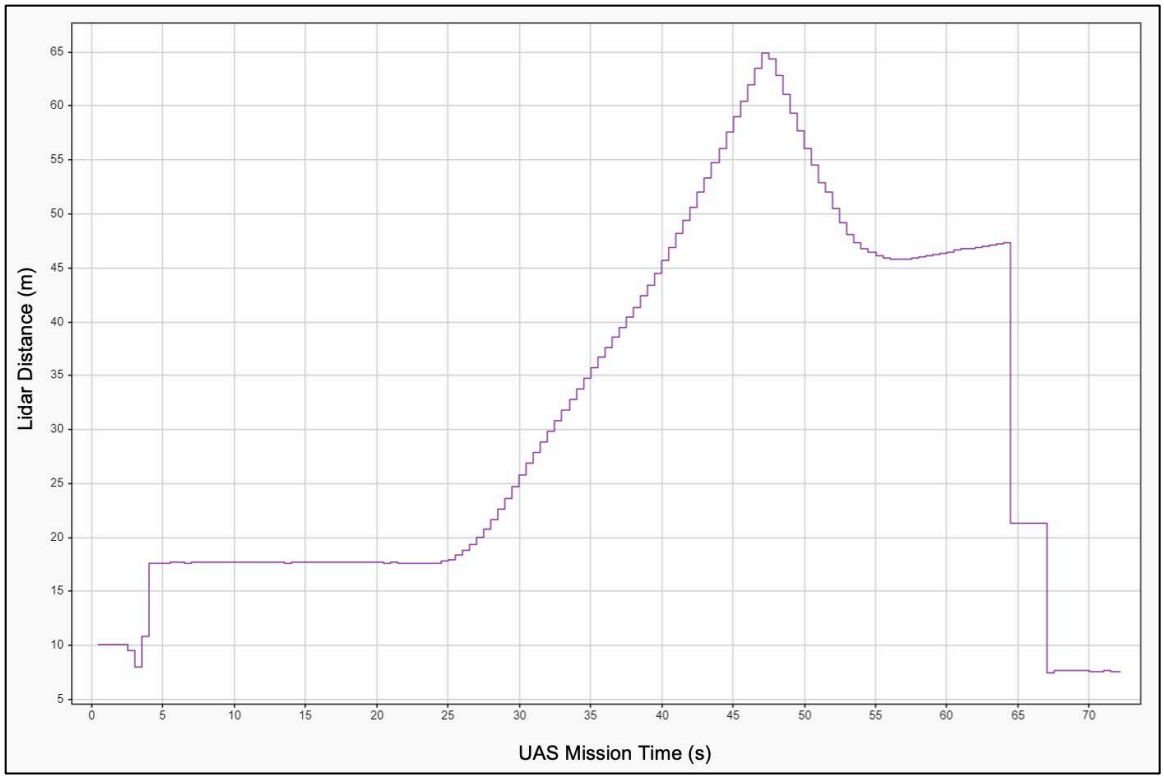


Figure 51. Lidar Distance in Mission 1.

3. Drone Mass

The mass of a small UAS ranges between 0.002 kg and 9 kg depending on its type and purpose (see Table 3). Standard consumer-grade drones have a mean mass of 1.2 kg while the commercial and militarily drones may have a dry weight between 3.3 kg and 6,781 kg (Mario 2019).

Table 3. Comparison of Commercial and Military Drones.

Name	Weight (kg)	Name	Weight (kg)
DJI Mavic 2 ZOOM	0.905	Potensic A20 RC Nano Quadcopter	0.100
DJI Phantom 3 Standard	1.216	Hubsan H111	0.011
DJI Phantom 4 PRO	1.388	Eachine E010	0.022
DJI Phantom 4 Advanced	1.368	Freefly Alta 8	6.200
DJI Mavic Pro	0.734	DJI Matrice 600 PRO	10.000
DJI Mavic 2 PRO	0.907	DJI S900	3.300
DJI Spark	0.300	DJI Agras MG-1	8.800
DJI Mavic Air	0.430	MFD 5000	10.430
DJI Mavic Mini	0.249	Onyxstar Hydra 12	7.000
Tello	0.080	Schieble's CAMCOPTER S-100	110.000

The mass of the drone was found to have some effects the mission time. As seen in Figure 52, the mission time increases as the drone mass increases. There is a significant increment in mission time when the mass of the drone goes higher than 0.121 kg. Therefore, the drone mass shall be used as one of the test variables for this experiment. However, due to the limitation of the model, it was found that the UAS could not take off when the mass of the drone is more than 0.125 kg.

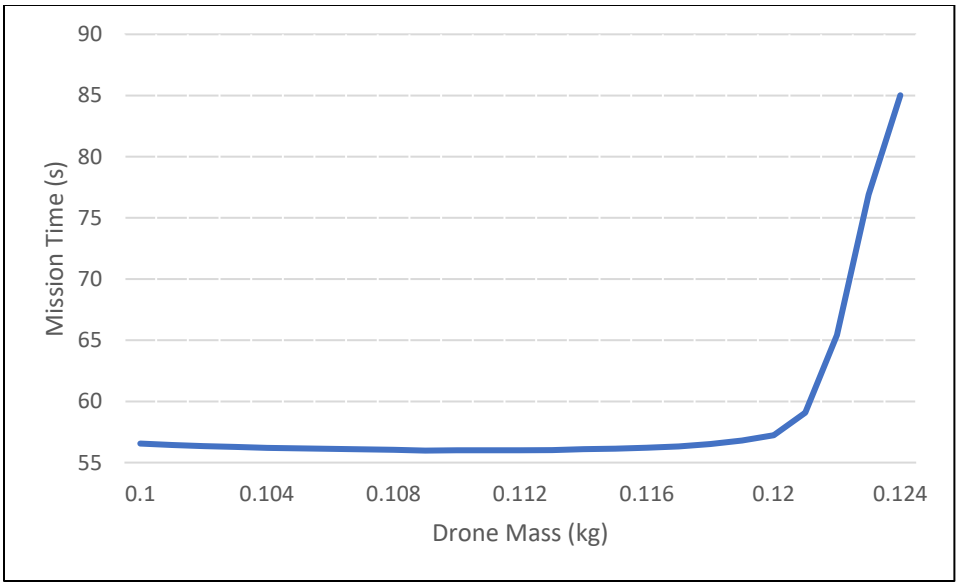


Figure 52. Effects of Drone Mass in Mission 1.

4. Altitude

The altitude of the UAS was found to have some effects the mission time. As seen in Figure 53, the mission time increases as the altitude of the UAS increases.

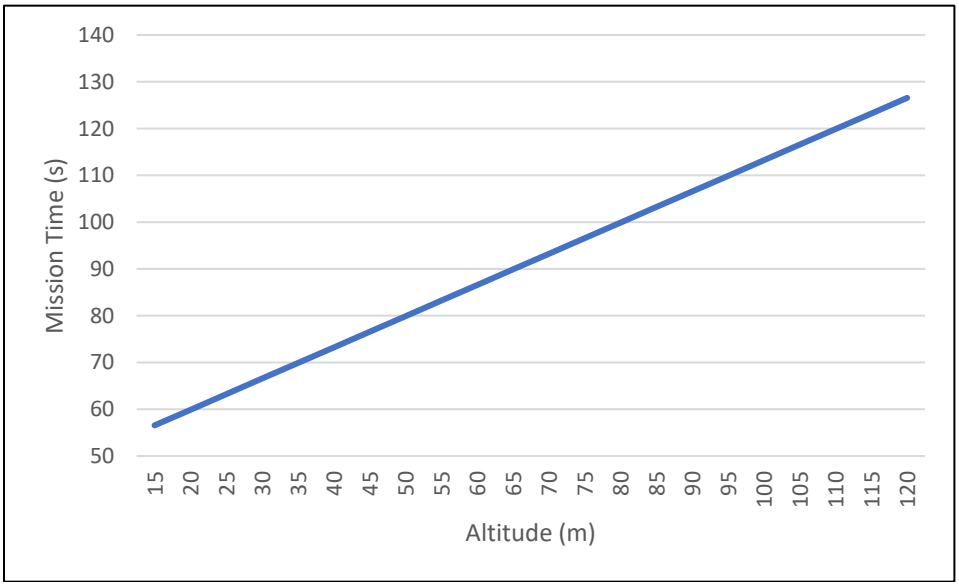


Figure 53. Effects of Altitude in Mission 1.

E. UAS INITIAL PARAMETERS

Tables 4 to 7 provide the initial parameters set for this simulation of the UAS.

Table 4. Constant Parameters.

Parameter	**Value**
Proportional, Pz	6.8
Integral, Iz	0
Derivative, Dz	2.5
Filter Coefficient, Nz	14.4947
Gravity	9.81 m/s^2

Table 5. UAS Controller Parameters.

Parameter	**Value**
PD Roll	[3402.97, 116.67] rad
PD Pitch	[3402.97, 116.67] rad
P Yaw Rate	1950 rad/s
P Thrust	39 N

Table 6. UAS Front-Facing Camera Parameters.

Parameter	**Value**
Focal Length	1109 x 1109 pixels
Optical Center	376 x 240 pixels
Image Size	480 x 752 pixels

Table 7. UAS Lidar Parameters.

Parameter	**Value**
Vehicle Radius	0.5 m
Safety Distance	0.65 m
Minimum Turning Radius	2 m
Field of View (Vertical)	40 deg
Field of View (Horizontal)	360 deg
Resolution (Vertical)	1.25 deg
Resolution (Horizontal)	0.3324 deg

F. UGS TEST VARIABLES

The maximum velocity of UGSs range from 0.05 meters per second (m/s) to 40 m/s. The mean and median maximum velocity for UGSs were found to be 1.5 m/s and 3 m/s respectively. The 25th and 75th percentiles were 0.9 m/s and 2.5 m/s respectively (The Association for Unmanned Vehicle Systems International 2013).

Due to the limitation of the model, the only test variable that have some effects to the mission time of the UGS is its velocity. This test variable shall be comparatively observed and statistically assessed for analysis.

G. UGS INITIAL PARAMETERS

Tables 8 and 9 provide the initial parameters set for this simulation of the UGS.

Table 8. UGS Front-Facing Camera Parameters.

Parameter	**Value**
Focal Length	1109 x 1109 pixels
Optical Center	640 x 360 pixels
Image Size	720 x 1280 pixels

Table 9. UGS Lidar Parameters.

Parameter	**Value**
Detection Range	90 m
Range Resolution	0.002 m
Field of View (Vertical)	40 deg
Field of View (Horizontal)	360 deg
Resolution (Vertical)	1.25 deg
Resolution (Horizontal)	0.3324 deg

VI. SIMULATION RESULTS AND ANALYSIS

This chapter begins by describing the general observations in running the UAS and UGS simulations. After that, it presents the results of the simulated test cases and the corresponding analyses on the UAS-UGS teaming system, starting with the UAS test variables, followed by the UGS test variables, and ends with the mission time constraint.

A. JOINT MISSION SIMULATION

Running the executable system model in Cameo integrated with Simulink models enabled the visualization of the behavior of UAS in IED detection mission (Figures 54a and 54b) and UGS in IED detonation mission (Figures 54c and 54d) in the same 3D environment. It can be observed that the 3D environment offered several types and levels of obstacles to provide variability in the test cases mentioned in Chapter V. It can be observed that when the UAS or UGS crashes on a building, it just passes through and the simulation continues to run, although, in reality, it is supposed to be considered as a mission failure. Therefore, visual observation shall be done on each run to ensure validity of results.

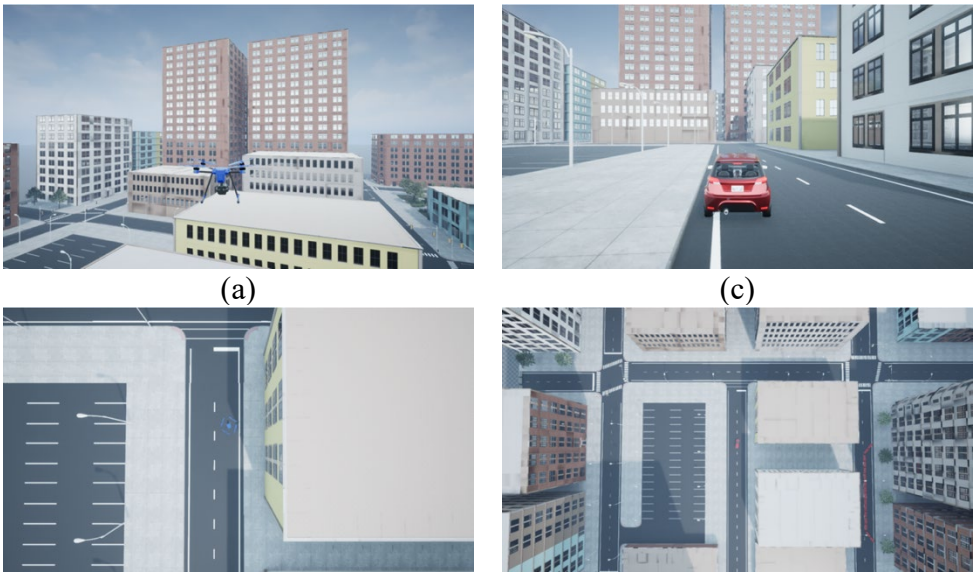


Figure 54. UAS and UGS Missions in 3D Simulation Environment.

B. UAS GUIDANCE LOGIC ANALYSIS

The results of the simulation testing the effects of guidance logic on the UAS mission time are summarized in Figure 55. It can be observed that the UAS with a full guidance logic has lower variance compared to obstacle avoidance.

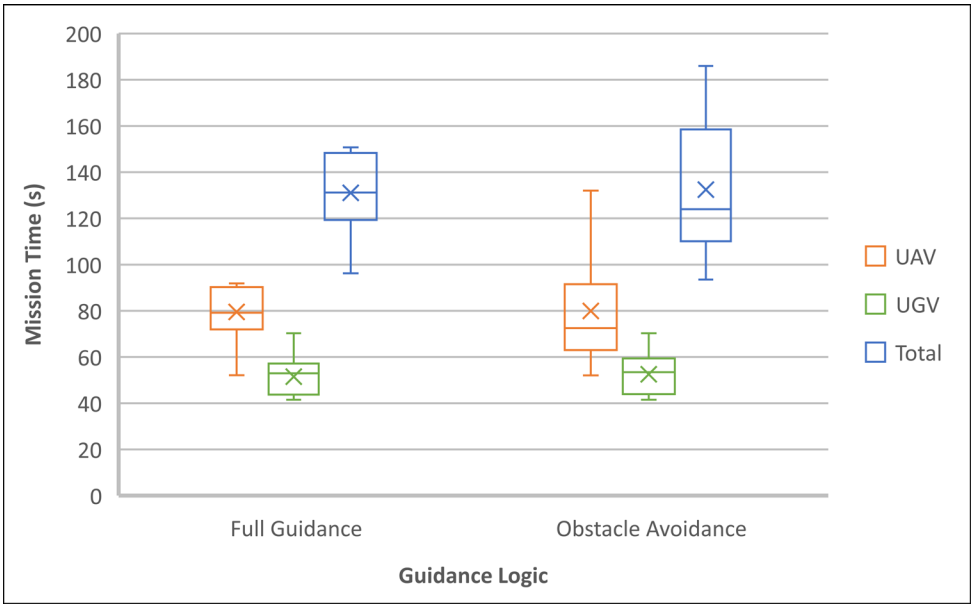


Figure 55. Mission Time vs. Guidance Logic.

C. UAS DRONE MASS ANALYSIS

The results of the simulation testing the effects of drone mass on the UAS mission time are summarized in Figure 56. It can be observed that the effect of the drone mass on the mission time is consistent to both guidance logics, which shows that mission time will increase exponentially as the mass of the drone increases.

It can also be observed that with varying drone mass, the UAS adopting a full guidance logic has consistently shorter mission time compared to obstacle avoidance.

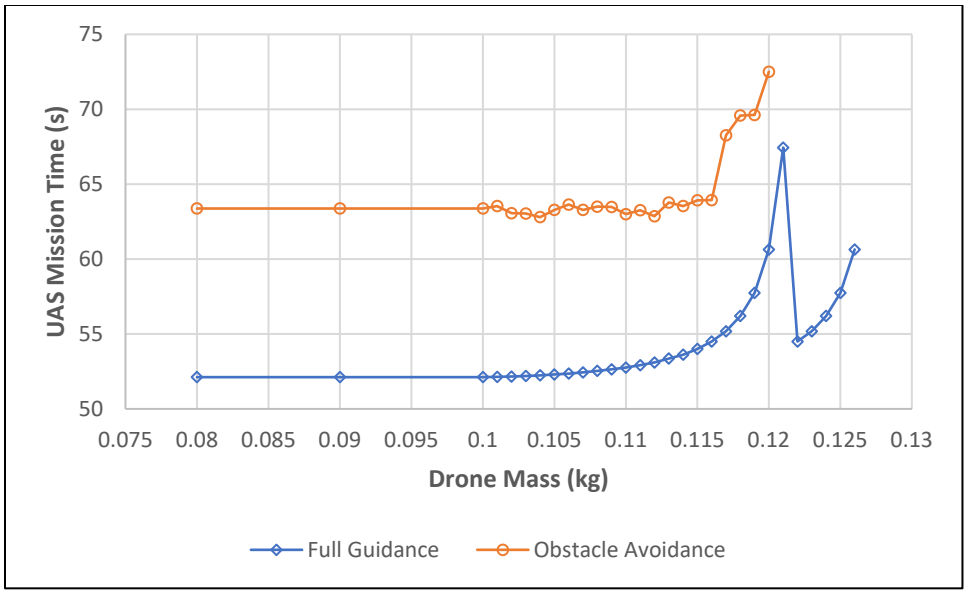


Figure 56. Effects of Drone Mass on UAS Mission Time.

However, when the UAS with a mass of more than 0.121 kg engages into obstacle avoidance guidance logic (and 0.125 kg for full guidance), the UAS did not have sufficient thrust to take off and remained to be on the ground. This could be due to the limitation of the model, but this can be resolved by adjusting other parameters to enable the rotors to obtain more upward thrust.

D. UAS ALTITUDE ANALYSIS

The results of the simulation testing the effects of altitude on the UAS mission time are summarized in Figure 57. It can be observed that higher altitude translates to longer mission time. This is an expected result because the UAS would require additional time to take off and land at a significantly high altitude which increases the overall mission time.

It can also be observed that there is an interim increase of altitude right before it reaches the desired altitude. This is an expected phenomenon when the UAS engages on full throttle as it takes off with maximum net thrust and thus require a short period of time to reach and stabilize at the intended altitude.

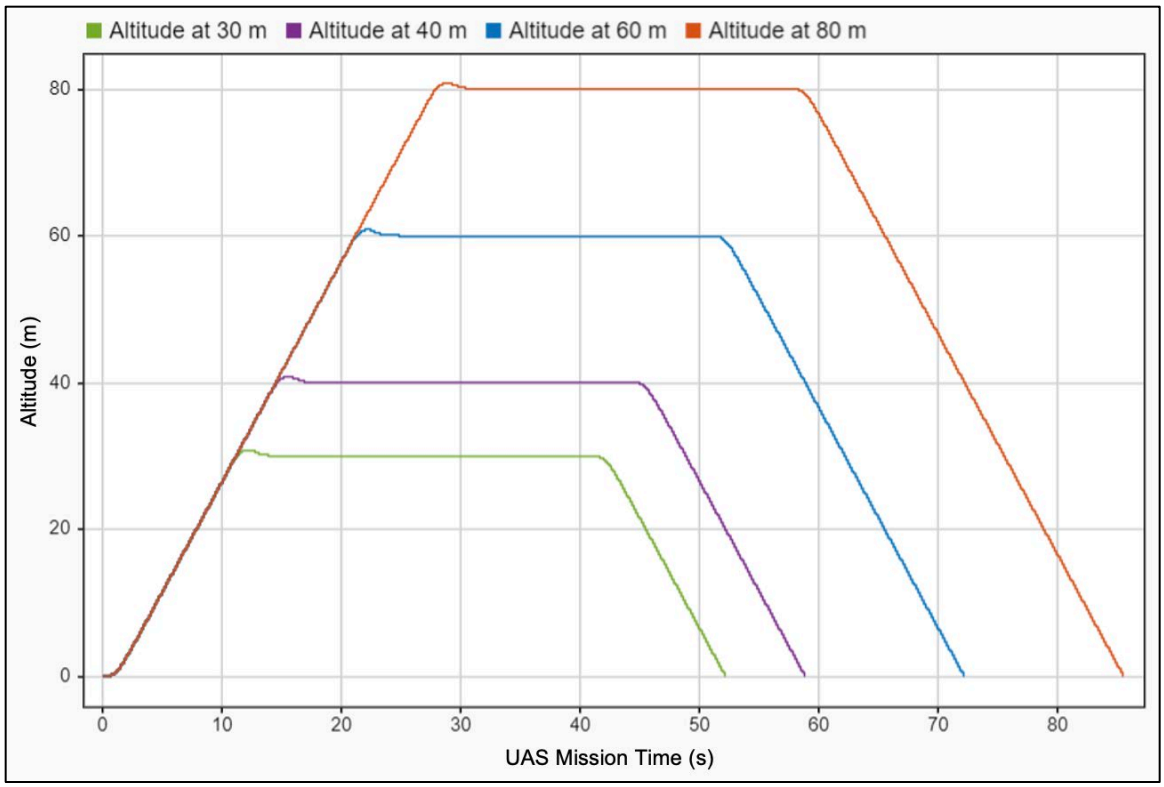


Figure 57. Effects of Altitude on UAS Mission Time.

E. UAS DETECTION RANGE ANALYSIS

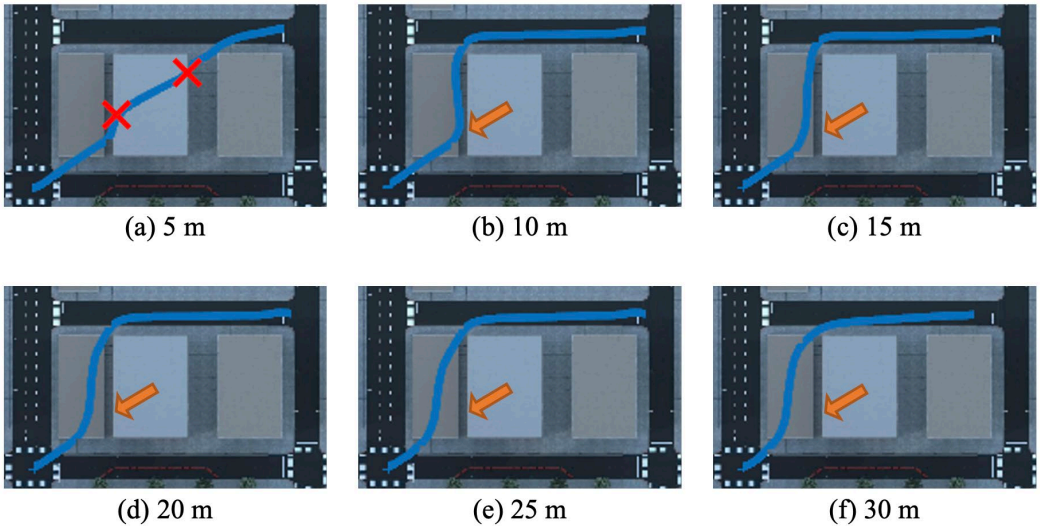
The simulation with varying detection range and the corresponding mission timing for each run was tabulated and summarized in Table 10. The table was color-scaled with green being the shortest mission time (i.e., desired) and yellow being the longest. It should also be noted that the UAS crashed onto the buildings on some runs as highlighted in red.

It was found that the UAS with a detection range of 5 m crashes 87.5% of the time. This can be observed in the flight path of the UAS as shown in Figure 58(a) where the UAS penetrated one side of the building and came out to the other side to continue its path towards the target. Although going through the windows of the buildings could be a tactic to achieve a shorter flight path to reach the target, the conditions of this experiment compel this scenario as a mission failure with the assumptions that the probability of the UAS passing through a small window is low. Similar observations were found for 10 m detection range, which crashed 50% of the time.

Table 10. Effects of Detection Range on UAS Mission Time.

	5 m	10 m	15 m	20 m	25 m	30 m
Mission 1	crashed	67.76 s	66.04 s	64.46 s	63.74 s	63.38 s
Mission 2	crashed	crashed	62.08 s	64.7 s	64.06 s	64.58 s
Mission 3	crashed	63.38 s	62.86 s	64.86 s	66.46 s	68.22 s
Mission 4	crashed	crashed	58.54 s	59.54 s	88.4 s	89.42 s
Mission 5	crashed	66.18 s	65.76 s	68.04 s	67.54 s	68.04 s
Mission 6	crashed	crashed	crashed	63.2 s	62.64 s	61.74 s
Mission 7	crashed	crashed	62.28 s	62.4 s	61.96 s	61.42 s
Mission 8	51.64 s	51.9 s	51.84 s	51.86 s	51.86 s	52.04 s

Other observations found in the experiment include the behavior of the UAS upon early detection of an obstacle. As seen in Figures 58(b) to (f), the UAS reacts and makes the turn earlier as the detection range increases. By having a higher detection range, the UAS is able to make the necessary maneuvers to avoid the obstacle as early as possible.



(a) 5 m (b) 10 m (c) 15 m

(d) 20 m (e) 25 m (f) 30 m

Figure 58. Flight Path of UAS in Mission 1.

However, depending on the complexity of the environment, the preferred detection range (higher or lower) may vary. As seen in Table 10, a lower detection range is preferred for Missions 2, 3, 4, and 5 since the distances between each building are narrower. As for Missions 1, 5, and 8, a higher detection range is preferred since the UAS has more maneuverability in open space.

Furthermore, as seen in Figure 59, it can be observed that the UAS detection ranges 20 m has the least variance with a few outliers. This reassures the operator that the UAS would achieve a more consistent range of mission time if this detection range were utilized.

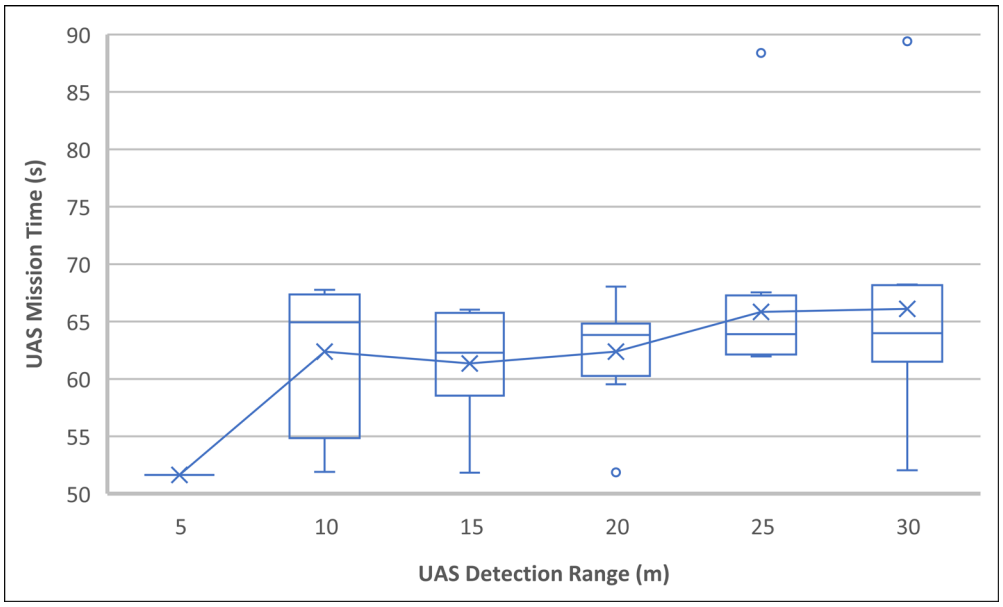


Figure 59. UAS Mission Time vs. Detection Range.

F. UGS VELOCITY ANALYSIS

The results of the simulation testing the effects of velocity on the UGS mission time are summarized in Figure 60. It can be observed that the mission time decreases exponentially as the velocity of the UGS decreases. It is evident that higher velocity would accomplish the mission faster. It can also be observed that higher velocity has lower variance which reassures the operator that the UGS would achieve a more consistent velocity of mission time.

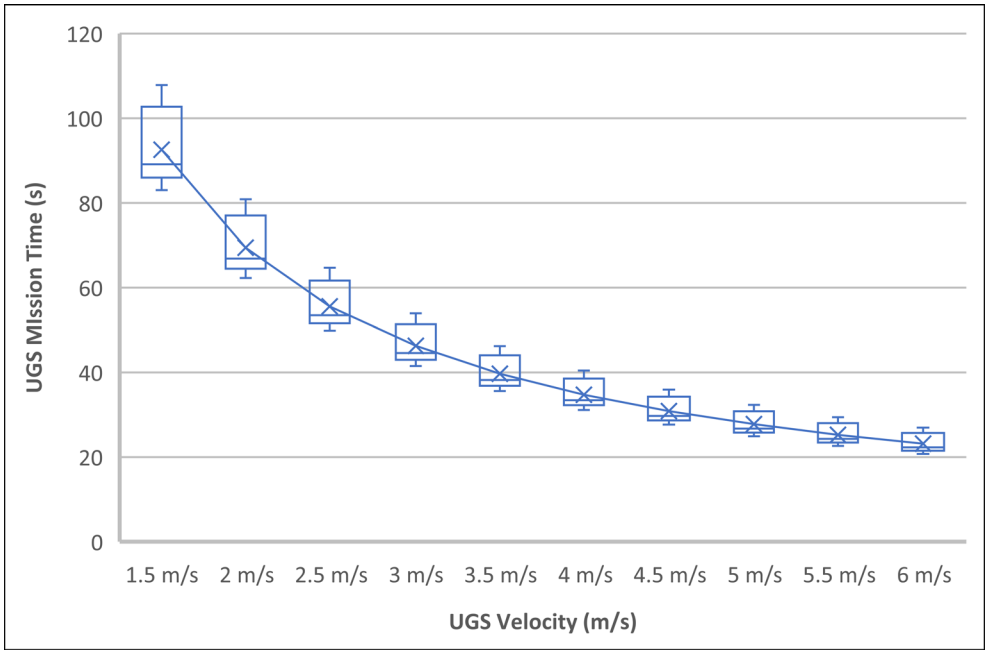


Figure 60. UGS Mission Time vs. UGS Velocity.

G. JOINT MISSION TIME CONSTRAINT ANALYSIS

Based on the mission time constraint that the total mission time to clear one IED 100 m away from the starting point shall be less than 115 s (see Figure 33), it was found that the UAS in obstacle avoidance guidance logic meets this constraint (87.5%) better than in full guidance (12.5%) as seen in Figure 61.

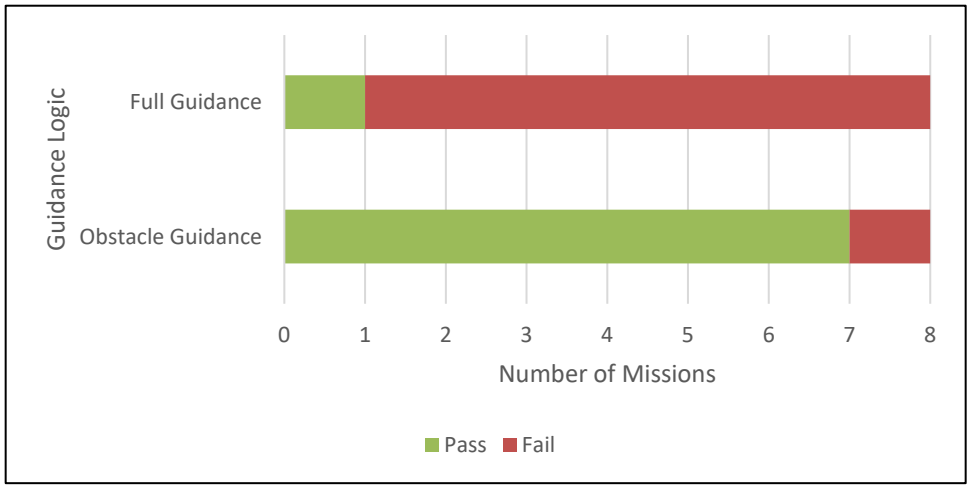


Figure 61. Mission Success Based on Mission Time Constraint (<115 s).

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSION AND RECOMMENDATIONS FOR FUTURE WORK

This chapter summarizes the key findings of the study including the effects of each test variables based on the experiment followed by the recommendations for future work.

A. CONCLUSION

This thesis has explored the possibility of utilizing an autonomous swarm composed of UAS and UGS working in tandem in IED clearance mission in an urban environment. The development of models of UAS and UGS missions using MBSE tools allowed a deeper understanding of the mission context and system architecture of UAS-UGS teaming. The development of UAS and UGS simulation models allowed to analyze the effects of selected system parameters and visualize the missions in a 3D simulation environment.

This thesis answered the following research questions formulated in Chapter I(B):

(1) Can high-fidelity models of UAS and UGS be integrated into the MBSE environment to conduct a study on improving the modern tactics and procedures of IED clearance?

Cameo is a powerful MBSE tool which is capable of developing executable system models, integrating them with external simulation environments such as MATLAB/ Simulink, and executing multiple simulations seamlessly to analyze the function and performance of the system. This thesis demonstrated the integration of high-fidelity models of UAS and UGS, which were developed in MATLAB/Simulink, into the MBSE environment of Cameo using executable system models.

It was established that IED clearance mission in urban environment has several complexities and challenges. Improving modern tactics and procedures rely on testing the system on several test case scenarios. It is evident in the results and illustrations shown in this thesis that the test case design can be modified to simulate different scenarios. It is also evident that external factors (e.g., obstacles and weather conditions) can be included by configuring the 3D simulation environment.

(2) Can the integrated model be used to investigate the optimized configuration of the UAS and UGS in the conduct of IED clearance based on the chosen system parameters?

As shown in the illustrations in this thesis, the integration enabled the visualization of the UAS and UGS performing their respective missions in the same 3D simulation environment. This provides a medium to analyze the behaviors of the UAS and UGS before performing operational testing on actual physical systems. It is evident in the results and illustrations shown in this thesis that the system parameters of the UAS and UGS can be configured to obtain the desired output (e.g., mission time) based on a specific constraint (e.g., less than 115 s).

The integrated model provided the means to investigate and verify the system parameters that have impact to the mission. By expanding the test cases and obtaining sufficient data, it is possible to achieve a set of optimized configurations for the UAS and UGS in the conduct of IED clearance.

With all other parameters remain constant, flying a fully guided UAS at a high altitude generally completes the mission faster than an autonomous UAS that avoids obstacles at a low altitude. However, flying at a higher altitude may incur additional time penalty as more time needs to be catered to take off and land. Obstacle avoidance guidance logic may be suitable for missions with more complex and cluttered environment (i.e., tall buildings and narrow routes). A lower detection range is preferred for such complex environment, although very low detection range may cause the UAS to crash as it would have inadequate safety distance and insufficient time to maneuver away from the obstacle. The mass of the UAS cannot be too light or too heavy such that there is insufficient net thrust for the UAS to take off. It should be noted that the heavier the UAS is, the longer it would take to complete the mission. The higher the velocity of the UGS, the faster it can complete the mission. The UGS would be able to maintain its velocity, even when making sharp turns, through path smoothing.

B. RECOMMENDATIONS FOR FUTURE WORK

This thesis demonstrated the development of executable system models using Cameo and high-fidelity Simulink models. The system parameters were manipulated using instance specifications. Future work could include the development of a GUI in the context of a realistic mockup of an operator's control module (i.e., GCS). This would allow the user to control the UAS and UGS by manipulating the system parameters dynamically and monitor the status of the mission in real-time.

Cameo is a powerful MBSE tool that has the capability, not only of integrating executable system models to external simulation environment, but also controlling external devices. Future work may include implementing the system models into the actual UAS and UGS to perform operational testing on an actual but controlled environment. This would enable to validate the simulation results of this study and test more proof of concepts.

Furthermore, it is noted that the simulation models for both UAS and UGS have their own limitations. This could be further improved by incorporating external factors such as environmental conditions (i.e., weather) and adversaries (i.e., enemy attacks) to simulate an even more realistic warfighting scenario. It is also to believe that there exists a potential to explore implementing the models in actual small-UAS and UGS to further validate the effects of the system parameters in the mission.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Aranda, Miguel, Gonzalo Lopez-Nicolas, Carlos Sagues, and Youcef Mezouar. 2015. "Formation Control of Mobile Robots Using Multiple Aerial Cameras." *IEEE Transactions on Robotics* 31 (4): 1064–71. https://doi.org/10.1109/TRO.2015.2452777.

Armtrac. 2020. "Demining Equipment: How Unmanned Bomb Disposal Saves Lives." June 23, 2020. https://armtrac.net/mechanical-demining-machines/how-unmanned-bomb-disposal-is-saving-lives/.

Beihoff, Bruce, Christopher Oster, Sanford Friedenthal, Christiaan Paredis, Duncan Kemp, Heinz Stoewer, David Nichols, and Jon Wade. 2014. *A World in Motion – Systems Engineering Vision 2025*. San Diego, CA: International Council on Systems Engineering (INCOSE). https://www.incose.org/docs/default-source/aboutse/se-vision-2025.pdf

Bhattacharjee, Angshuman, Arghya Hazra, and Suvam Kumar Sar. 2018. "Quadcopter Control Using Arduino Microcontroller." Thesis, RCC Institute of Information Technology. https://www.rcciit.org/students_projects/projects/aeie/2018/GR2.pdf.

Bock, Conrad, Raphael Barbau, Ion Matei, and Mehdi Dadfarnia. 2017. "An Extension of the Systems Modeling Language for Physical Interaction and Signal Flow Simulation." *Systems Engineering* 20 (5): 395–431. https://doi.org/10.1002/sys.21380.

Chaimowicz, Luiz, and Vijay Kumar. n.d. "Aerial Shepherds: Coordination Among UAVs and Swarms of Robots." In *Distributed Autonomous Robotic Systems* 6, 243–52. Tokyo: Springer Japan. https://doi.org/10.1007/978-4-431-35873-2_24.

Chan, K. W, U Nirmal, and W. G Cheaw. 2018. "Progress on Drone Technology and Their Applications: A Comprehensive Review." In *AIP Conference Proceedings*. Vol. 2030. https://doi.org/10.1063/1.5066949.

Chen, Jie, Xing Zhang, Bin Xin, and Hao Fang. 2016. "Coordination Between Unmanned Aerial and Ground Vehicles: A Taxonomy and Optimization Perspective." *IEEE Transactions on Cybernetics* 46 (4): 959–72. https://doi.org/10.1109/TCYB.2015.2418337.

Ding, Yulong, Bin Xin, and Jie Chen. 2021. "A Review of Recent Advances in Coordination between Unmanned Aerial and Ground Vehicles." *Unmanned Systems* 09 (02): 97–117. doi:10.1142/s2301385021500084.

E. Z. MacArthur, D. MacArthur and C. Crane, Use of cooperative unmanned air and ground vehicles for detection and disposal of simulated mines, Proc. Intelligent Systems in Design and Manufacturing VI, Vol. 5999, International Society for Optics and Photonics (Porto, Portugal, 2005), pp. 909–917.

Faust, Anthony A., C. J. De Ruiter, Anneli Ehlerding, John E. McFee, Eirik Svinsås, and Arthur D. Van Rheenen. "Observations on military exploitation of explosives detection technologies." *In Detection and Sensing of Mines, Explosive Objects, and Obscured Targets* XVI, vol. 8017, p. 801716. International Society for Optics and Photonics, 2011.

Federal Aviation Administration. 2018. "Airspace 101 – Rules of the Sky." Last modified October 30, 2018. https://www.faa.gov/uas/recreational_fliers/where_can_i_fly/airspace_101/.

Friedenthal, Sanford, Alan Moore, and Rick Steiner. 2014. *A Practical Guide to SysML: The Systems Modeling Language*. Third edition. San Francisco: Elsevier Science & Technology.

Giesbrecht, J, D Mackay, J Collier, and S Verret. 2005. "Path Tracking for Unmanned Ground Vehicle Navigation: Implementation and Adaptation of the Pure Pursuit Algorithm."

Goodin, Christopher, Justin Carrillo, J. Gabriel Monroe, Daniel W Carruth, and Christopher R Hudson. 2021. "An Analytic Model for Negative Obstacle Detection with Lidar and Numerical Validation Using Physics-Based Simulation." *Sensors (Basel, Switzerland)* 21 (9): 3211–. https://doi.org/10.3390/s21093211.

Gupte, S, Paul Infant Teenu Mohandas, and James M Conrad. 2012. "A Survey of Quadrotor Unmanned Aerial Vehicles." In *2012 Proceedings of IEEE Southeastcon*, 1–6. IEEE. https://doi.org/10.1109/SECon.2012.6196930.

H. Jing-Lin, S. Xiu-Xia, L. Ri, D. Xiong-Feng, and L. Mao-Long, ``UAV real-time route planning based on multi-optimized RRT algorithm,'' in Proc. 29th Chin. Control Decis. Conf. (CCDC), May 2017, pp. 837–842.

Halsey, Ashley. 2021. "How Drone Based LIDAR is Changing the Game." Geospatial World. January 25, 2021. https://www.geospatialworld.net/blogs/how-drone-based-lidar-is-hcanging-the-game/.

Hu, Cheng, Yixuan Wang, Rui Wang, Tianran Zhang, Jiong Cai, and Meiqin Liu. 2019. "An Improved Radar Detection and Tracking Method for Small UAV Under Clutter Environment." *Science China Information Sciences* 62(2) (December): 29306. https://doi.org/10.1007/s11432-018-9598-x.

International Campaign to Ban Landmines. 2020. “Landmine Monitor 2020.” Accessed March 18, 2021. http://www.the-monitor.org/media/3168934/LM2020.pdf.

International Council on Systems Engineering (INCOSE). 2007. *Systems Engineering Vision 2020*. Report No. TP-2004-004-02. http://www.ccose.org/media/upload/SEVision2020_20071003_v2_03.pdf.

Jiliang Lv, Chenxi Qu, Shaofeng Du, Xinyu Zhao, Peng Yin, Ning Zhao, and Shengguan Qu. 2021. “Research on Obstacle Avoidance Algorithm for Unmanned Ground Vehicle Based on Multi-Sensor Information Fusion.” *Mathematical Biosciences and Engineering: MBE* 18 (2): 1022–39. https://doi.org/10.3934/mbe.2021055.

Käslin, Roman, Peter Fankhauser, Elena Stumm, Zachary Taylor, Elias Müggler, Jeffrey Delmerico, Davide Scaramuzza, Roland Siegwart, and Marco Hutter. 2016. “Collaborative Localization of Aerial and Ground Robots through Elevation Maps.” In *IEEE*. https://doi.org/10.5167/uzh-127901.

Khamis, Alaa. 2013. “Facts About Landmines.” Minesweepers, 2013. https://landminefree.org/facts-about-landmines/.

Khan, Mohd. 2014. “Quadcopter Flight Dynamics.” *International Journal of Scientific & Technology Research* 3(8) (August): 130–135.

Lazna, Tomas, Petr Gabrlik, Tomas Jilek, and Ludek Zalud. 2018. “Cooperation Between an Unmanned Aerial Vehicle and an Unmanned Ground Vehicle in Highly Accurate Localization of Gamma Radiation Hotspots.” *International Journal of Advanced Robotic Systems* 15 (1): 172988141775078–. https://doi.org/10.1177/1729881417750787.

Li, Jiangiang, Genqiang Deng, Chengwen Luo, Qiuzhen Lin, Qiao Yan, and Zhong Ming. 2016. “A Hybrid Path Planning Method in Unmanned Air/Ground Vehicle (UAV/UGV) Cooperative Systems.” IEEE Transactions on Vehicular Technology 65 (12): 9585–96. https://doi.org/10.1109/TVT.2016.2623666.

Luukkonen, Teppo. 2011. “Modelling and Control of Quadcopter.” Thesis, Aalto University. https://sal.aalto.fi/publications/pdf-files/eluu11_public.pdf.

Mario. 2019. “How Much Do Drones Weigh?” Drone Tech Planet. December 3, 2019. https://www.dronetechplanet.com/how-much-do-drones-weigh/.

MathWorks. 2013. “Modeling an Anti-Lock Braking System.” https://www.mathworks.com/help/simulink/slref/modeling-an-anti-lock-braking-system.html.

MathWorks. 2019. “Design Lidar SLAM Algorithm Using Unreal Engine Simulation Environment.” https://www.mathworks.com/help/driving/ug/design-lidar-slam-algorithm-using-3d-simulation-environment.html.

MathWorks. 2019. “Select Waypoints for Unreal Engine Simulation.” https://www.mathworks.com/help/driving/ug/select-waypoints-for-3d-simulation.html.

MathWorks. 2020. “UAV Package Delivery.” https://www.mathworks.com/help/uav/ug/uav-package-delivery.html.

MathWorks. n.d. “Simulink.” Accessed July 28, 2021. https://www.mathworks.com/help/simulink/.

Military Wiki. n.d. “Demining.” Accessed March 18, 2021. https://military.wikia.org/wiki/Demining.

No Magic. 2018. “Cameo Systems Modeler.” February 8, 2018. https://www.nomagic.com/products/cameo-systems-modeler.

No Magic. 2020. “Cameo Simulation Toolkit 19.0 LTR SP4 User Guide”.

Ononiwu, Gordon, Ondoma Onojo, Oliver Ozioko, and Onyebuchi Nosiri. 2016. “Quadcopter Design for Payload Delivery.” *Journal of Computer and Communications* 04 (10): 1–12. doi:10.4236/jcc.2016.410001.

Pavalkis, Saulius. 2018. “Overview of Current SysML/UML and MATLAB/Simulink Integration Use Case and Implementations.” *Modeling Community Blog* (blog), March 29, 2018. https://blog.nomagic.com/overview-of-current-sysml-uml-and-matlab-simulink-integration-use-cases-and-implementations/.

Pessoa, Rodrigo Simões. 2017. “Mathematical Modeling of a Tilt-Rotor Drone.” Thesis, Pontifícia Universidade Católica do Rio de Janeiro. https://doi.org/10.17771/PUCRio.acad.30497.

PivotPoint Technology. n.d. “SysML FAQ: What Are the SysML Diagram Types?” *SysML Forum* (blog). Accessed August 25, 2021. https://sysmlforum.com/sysml-faq/what-are-sysml-diagram-types.html.

Rangel, Diego. 2021. “Executable MBSE Approach with Illustration of a Satellite Engagement Mission Design.” Master’s thesis, Naval Postgraduate School. https://calhoun.nps.edu/handle/10945/67798.

Ravankar, Abhijeet, Ankit A Ravankar, Yukinori Kobayashi, Yohei Hoshino, and Chao-Chung Peng. 2018. “Path Smoothing Techniques in Robot Navigation: State-of-the-Art, Current and Future Challenges.” *Sensors (Basel, Switzerland)* 18 (9): 3170–. https://doi.org/10.3390/s18093170.

Ren, Hongbin, Sizhong Chen, Lin Yang, and Yuzhuang Zhao. 2020. “Optimal Path Planning and Speed Control Integration Strategy for UGVs in Static and Dynamic Environments.” *IEEE Transactions on Vehicular Technology* 69 (10): 10619–29. https://doi.org/10.1109/TVT.2020.3015582.

Ren, Hongbin, Taehyun Shim, Jemyoung Ryu, Sizhong Chen. 2014. “Development of Effective Bicycle Model for Wide Ranges of Vehicle Operations.” SAE Technical Paper 2014–01-0841. https://doi.org/10.4271/2014-01-0841.

S. Zhang, H. Wang, S. He, C. Zhang and J. Liu, An autonomous air-ground cooperative field surveillance system with quadrotor UAV and unmanned ATV robots, 2018 IEEE 8th Annual Int. Conf. CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)(IEEE, 2018), pp. 1527–1532.

Simmons, R. 1996. “The Curvature-Velocity Method for Local Obstacle Avoidance.” In *Proceedings of IEEE International Conference on Robotics and Automation*, 4:3375–3382 vol.4. IEEE. https://doi.org/10.1109/ROBOT.1996.511023.

Stentz, Anthony, Alonzo Kelly, Peter Rander, Herman Herman, Omead Amidi, Robert Mandelbaum, Garbis Salgian, and Jorgen Pedersen. 2018. “Real-time, Multi-perspective Perception for Unmanned Ground Vehicles.” Carnegie Mellon University. doi:10.1184/R1/6560312.v1.

Tan, Chun, Gary Wong Hock Lye, Bryan Soh Chee Wang. n.d. “Introduction to Mine Clearing Technology.” In *DSTA Horizons*, 123–128, www.dsta.gov.sg/docs/default-source/dsta-about/introduction-to-mine-clearing-technology.pdf.

The Association for Unmanned Vehicle Systems International. 2013. *Unmanned Ground Vehicles: Core Capabilities & Market Background*. http://higherlogicdownload.s3.amazonaws.com/AUVSI/958c920a-7f9b-4ad2-9807-f9a4e95d1ef1/UploadedFiles/AUVSIUGVCoreCapabilitiesandMarketBackground08-08-13.pdf

Väljaots, E., & Sell, R. 2019. Energy efficiency profiles for unmanned ground vehicles. *Proceedings of the Estonian Academy of Sciences, 68*(1), 55–. https://doi.org/10.3176/proc.2019.1.04

Wasson, Steven R, Jose Guilberto, Wade Ogg, Kevin Wedeward, Stephen Bruder, and Aly El-Osery. 2004. “An Unmanned Ground Vehicle for Landmine Remediation.” In *Proceedings of SPIE*, 5415:1231–39. SPIE. https://doi.org/10.1117/12.541341.

Weisstein, Eric. n.d. “Spline.” Accessed August 25, 2021. https://mathworld.wolfram.com/Spline.html.

Wilkinson, Mark. 2019. “IEDs and Urban Clearance Variables in Mosul: Defining Complex Environments.” *The Journal of Conventional Weapons Destruction* 23(2): 5.

Yan Peng, Dong Qu, Yuxuan Zhong, Shaorong Xie, Jun Luo, and Jason Gu. 2015. “The Obstacle Detection and Obstacle Avoidance Algorithm Based on 2-D Lidar.” In *2015 IEEE International Conference on Information and Automation*, 1648–53. IEEE. https://doi.org/10.1109/ICInfA.2015.7279550.

Zawodny MacArthur, Erica, Donald MacArthur, and Carl Crane. 2005. “Use of Cooperative Unmanned Air and Ground Vehicles for Detection and Disposal of Mines.” In *Proceedings of SPIE*, 5999:599909–599908. Bellingham WA: SPIE. https://doi.org/10.1117/12.631314.

Zheng, L., Zhang, P., Tan, J., & Li, F. (2019). The Obstacle Detection Method of UAV Based on 2D Lidar. *IEEE Access, 7*, 163437–163448. https://doi.org/10.1109/ACCESS.2019.2952173

zur Muehlen, Michael. 2012. “Integration of M&S (Modeling and Simulation), Software Design and DoDAF (Department of Defense Architecture Framework (RT 24).”

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California