ARL-TR-9533 ● AUG 2022

DEVCOM ARMY RESEARCH LABORATORY

# An Adaptable Nonlinear Control for Quadcopters in Heavy Winds

by Carl Lederman and Brent Kraczek

Approved for public release: distribution unlimited.

## NOTICES

### Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-9533 ● AUG 2022


# An Adaptable Nonlinear Control for Quadcopters in Heavy Winds

**Carl Lederman**
*Lufburrow & Company, Inc*

**Brent Kraczek**
*DEVCOM Army Research Laboratory*

**Approved for public release: distribution unlimited.**

| REPORT DOCUMENTATION PAGE | Form Approved<br>OMB No. 0704-0188 |
|---|---|

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
**PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED (From - To) |
|---|---|---|
| August 2022 | Technical Report | 1 January 2021–1 April 2022 |

| 4. TITLE AND SUBTITLE | 5a. CONTRACT NUMBER |
|---|---|
| An Adaptable Nonlinear Control for Quadcopters in Heavy Winds | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S)<br>Carl Lederman and Brent Kraczek | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>DEVCOM Army Research Laboratory<br>ATTN: FCDD-RLC-ED<br>Aberdeen Proving Ground, MD 21005 | 8. PERFORMING ORGANIZATION REPORT NUMBER<br>ARL-TR-9533 |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
| | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

**12. DISTRIBUTION/AVAILABILITY STATEMENT**

Approved for public release: distribution unlimited.

**13. SUPPLEMENTARY NOTES**

ORCID IDs: Carl Lederman, 0000-0002-0544-7181

**14. ABSTRACT**

The safe and effective use of quadcopters within urban areas with environmental factors is of tremendous importance to the US military and civilian sectors. This technical report explores a highly adaptable simulation setup, with a nonlinear controller containing learning elements. Other model factors—such as the drone geometry, weighting, and wind forces—are easily modifiable within the presented framework. The simulation, performed with Unreal Engine, can incorporate real-world city data, realistic winds, and existing open-source software.

**15. SUBJECT TERMS**

Military Information Sciences; Network, Cyber, and Computational Sciences; UAS; nonlinear controls; wind effects

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Brent Kraczek |
| Unclassified | Unclassified | Unclassified | UU | 28 | 19b. TELEPHONE NUMBER (Include area code)<br>(410) 278-8881 |

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

## Contents

## List of Figures

# 1. Introduction and Relevance to the US Army

The use of unmanned systems and unmanned aerial systems (UASs) is proliferating throughout the world's militaries, with applications in communications, surveillance, reconnaissance, and combat (Nacouzi et al. 2018). In hostile regions, UASs will be subject to several varieties of threats, including cyber and physical threats, as well as environmental hazards. Survival and mission success will often depend on the ability to operate with minimal communications or reliance on Global Navigation Satellite Systems (GNSS), such as GPS (Guvenc et al. 2018; Sathyamoorthy et al. 2020; Fan et al. 2022). For example, communications from the UAS can be used to detect and obtain the UAS position, while satellite-based navigation is easily spoofed or jammed, as the signals are very weak. Other sensors are also often used to augment GNSS location analysis and could be used to take its place, such as optical systems—including cameras, radar, Light Detection and Ranging (LiDAR) systems, and inertial measurement units (IMUs) (Angelino et al. 2012). These present their own challenges. IMUs are standard equipment but are only able to detect linear and angular accelerations while determining orientation by detecting the local magnetic field of the earth (a total of 9 degrees-of-freedom). Thus, position errors, the second integral in time of the measured accelerations, accumulate over time. Other sources of concern when using IMUs for UAS navigation include environmental effects (i.e., wind or precipitation). Physical changes to UAS construction, such as the addition of a sensor or weapons package, including changes once a weapon has been discharged, further complicate efforts. Such changes in mass and mass distribution alter the center-of-mass and inertia tensor of the UAS. Optical sensors, radar, and LiDAR systems add weight and often emit RF or light, making them more detectable and/or requiring processing resources. The added weight and/or processing can have adverse effects on battery life, and, thus, run time and overall reliability.

To address these issues, we are investigating the use of control algorithms in windy environments to understand how IMU signals may be used within control to account for (and/or alter calculations of) UAS position. Coupled with uncertainty measures, these could ultimately be used to detect changes to UAS flight performance, or the spoofing of GNSS signals.

Urban environments are a second area of concern for safe and reliable UAS operations (Watkins 2020). They are recognized as a challenging domain for DOD operations, as well as a huge area for technological growth in the government and commercial services. In this report, we demonstrate a simulation space that we are building specifically for modeling UASs in urban environments to address issues of autonomous and semiautonomous control, with a focus on environmental

interactions, including wind and static collision threats. Key parts of the physics and controls are directly implemented in C++. Beyond that, where possible, we are leveraging current free- and open-source resources (i.e., software, software frameworks, and data), with the caveat that we include the use of some tools that require a fee after a product is commercially successful. We have taken a modular approach that will enable a flexible transition to other software frameworks and systems as they mature. Our current system has based this on the PX4 controller library for small UASs and real-time-publish-scribe (RTPS) data transfer protocols. RTPS should enable our developments to transition to other tools as they mature, and to other tools and data, such as computed wind data, using common application programming interfaces (i.e., APIs). For graphics and user interface, we use Unreal Engine (UE) (Matej 2016), a game engine that provides state-of-the-art graphics capabilities and some of the physics used in our models—most importantly collision detection between the UAS and its environment.

Sections 2–4 detail the major computing components of the overall simulation: the incorporation of real-world city data, the generation of realistic wind models, drone geometry and physics modeling, and linear and nonlinear controls. Our reliance on open-source software for each of these major pieces of the overall simulation, such as UE, OpenStreetMap (OSM) (Anderson et al. 2019), Mapbox, and AirSim (Shah et al. 2017) is detailed as needed (see Fig. 1; e.g., models of real cities imported into gaming engines). Sections 5 and 6 provide sample results and concluding remarks.

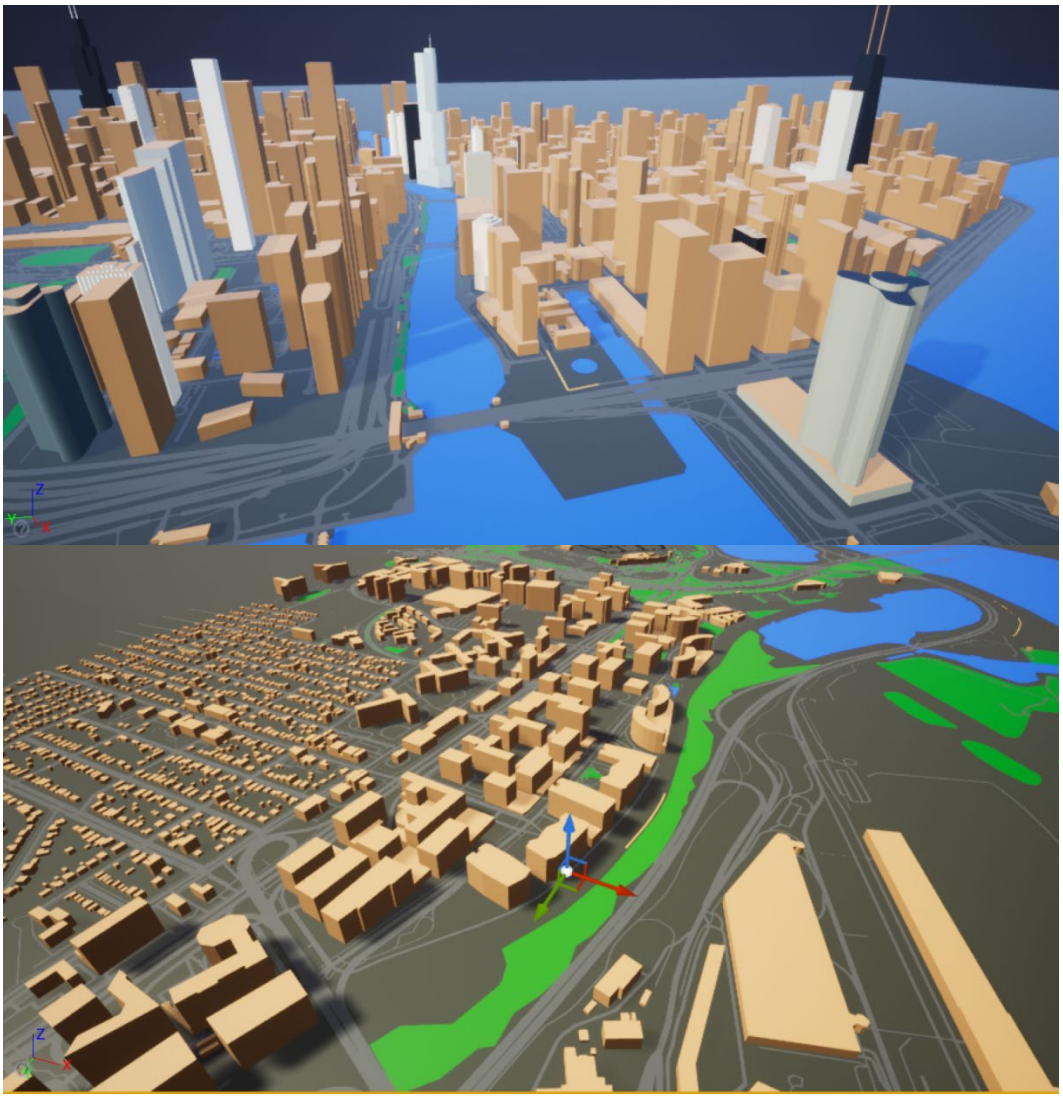

**Fig. 1 Two examples of city data incorporated into UE for large-scale simulations: Chicago, Illinois (top); Crystal City, Virginia (bottom). Both images were created using open-source tools to import open-source Mapbox city data into UE.**

## 2. Quadcopter Mesh and Physical Models

This section provides the equations relating to the state variables of the quadcopter system, quadcopter kinematics, and quadcopter modeling on computer meshes. These equations are general to most computer models of 3-D objects, with an emphasis on equations best suited to UAS control theory.

The position of the quadcopter is given by

$$q = \left[q_x, q_y, q_z\right]^T \in \mathbb{R}^3 , \tag{1}$$

with its velocity given by the time derivative of $q$. The rotational orientation is defined by the following set of three angles, referring to the pitch, roll, and yaw:

$$\sigma = [\phi, \theta, \psi]^T \in \mathbb{R}^3 \,, \tag{2}$$

and rotational velocities are derived from differentiating in time. In sum, the state vector characterizing the quadcopter, $x$, is given by

$$x(t) = [x_1^T(t), x_2^T(t)]^T = [q_1^T(t), \sigma_1^T(t) x_2^T(t), \sigma_2^T(t)]^T \in \mathbb{R}^{12} \quad . \tag{3}$$

An alternative characterization of the thrust, pitch, roll, and yaw is given by

$$\nu(t) = [v_1^T(t), \nu_2^T(t)]^T = \\ [\eta_1(t), \phi_1(t), \theta_1(t), \psi_1(t), \eta_2(t), \phi_2(t), \theta_2(t), \psi_2(t)]^T \in \mathbb{R}^8 \,. \tag{4}$$

This vector naturally presents intuitive values closely related to the motor inputs (Khan 2014), but it does not fully quantify the state of the system. For example, two quadcopters with the same $v$ vectors could be at different locations in space. Forces on the quadcopter from the rotations of its propellers only allow for specific types of forces. More general forcing on the drone is also possible from environmental factors (i.e., wind) and hence two different functions are defined:

$$f(x)\colon \mathbb{R}^{12} \to \mathbb{R}^6, g(x, \nu_2)\colon \mathbb{R}^{12} \times \mathbb{R}^4 \to \mathbb{R}^6 \quad . \tag{5}$$

The function $g$ takes in thrust, pitch, roll, and yaw values to produce force, while $f$ is the more general one. With these two functions defined, the ordinary differential equation (Strikwerda 2004) governing the trajectory of the drone is given by

$$\begin{gathered} \frac{dx_1}{dt}(t) = x_2(t) \\ \frac{dx_2}{dt}(t) = f\big(x(t)\big) + g(x(t), \nu_2(t)) \end{gathered} \,. \tag{6}$$

Quadcopter analysis often speaks of an internal frame and a body frame (Nelson 1998; Khan 2014). The three basis vectors for the inertial frame are simply

$$n_x \stackrel{\text{def}}{=} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, n_y \stackrel{\text{def}}{=} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, n_z \stackrel{\text{def}}{=} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad . \tag{7}$$

A unit vector $n \in \mathbb{R}_u^3$, where $\mathbb{R}_u^3 \stackrel{\text{def}}{=} \{n \in \mathbb{R}^3 \colon \|n\| = 1\}$ can be rotated to the body frame under the action of a rotation matrix, $R$, that is constructed from the three angles of $\sigma$:

$$R(\sigma, n)\colon \mathbb{R}^3 \times \mathbb{R}_u^3 \to \mathbb{R}_u^3 \,. \tag{8}$$

Thus, for example, one component of $g$, arising from the thrust can then be given as

$$g_\eta(\eta, \sigma): \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^3 = F_\eta R(n_z) \ . \tag{9}$$

Details of the equations governing the quadcopter are presented elsewhere (Nelson 1998). Often, these analyses will make simple assumptions about the exact shape and mass distribution (e.g., that it might be composed of two thin uniform rods and a point mass) (Khan 2014). A more general approach is to model the quadcopter as a tetrahedral mesh defined by node locations, velocities, and connections to each other:

$$M = [M_1^T(t), M_2^T(t)]^T \in \mathbb{R}^{6m} \ . \tag{10}$$

Such a mesh makes no assumptions about the geometry of the drone. The physical mesh employed was small and simple to establish baseline results, as given in Section 5; however, incorporating a more complex geometry in the future should be trivial. Figure 2 shows the physical and visualization meshes used in this report. The visualization mesh is more detailed.

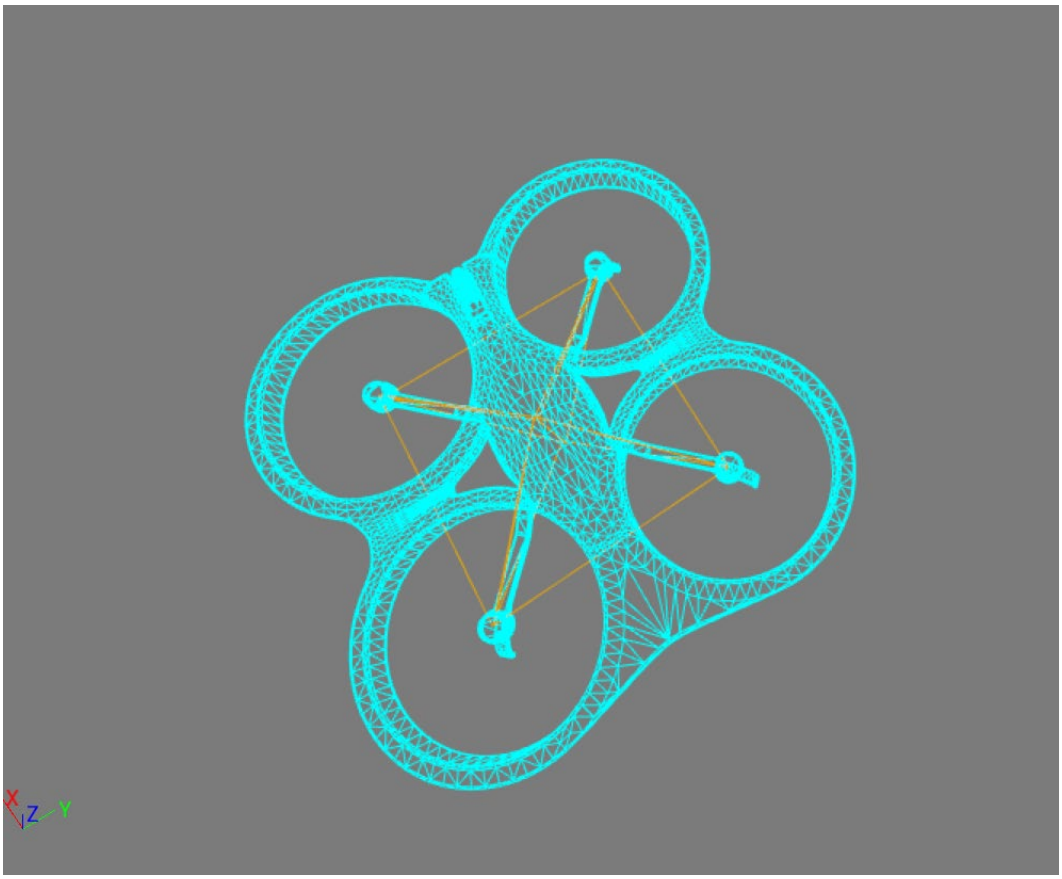


**Fig. 2 Visualization mesh (blue) and simpler physics mesh (yellow)**

The finite-element-based, partial differential equation (Johnson 2012) governing the evolution of the mesh is

$$\begin{matrix} \frac{dM_1}{dt}(t) = M_2(t) \\ \frac{dM_2}{dt}(t) = f_M\big(M(t)\big) + g_M(M(t), v_2(t)) \end{matrix} . \tag{11}$$

Detailed forces from the wind, limited only by the scale of the mesh are defined by the fundamental wind equation:

$$F_w = \frac{1}{2}\rho v^2 a \, , \tag{12}$$

where $a$ is a surface area, $\rho$ is the density of air, and $v$ is the relative wind velocity. More detailed wind equations and turbulence (Tennekes and Lumley 1972; Stull 1997) can also be implemented. For a wind field, interpolated from $j$ locations, $W \in \mathbb{R}^{6j}$, the aggregate force effect from the wind, applied to every surface node on the mesh, is

$$f_w(M, W): \mathbb{R}^{6m} \times \mathbb{R}^{6j} \to \mathbb{R}^{3m} \, . \tag{13}$$

Another component of $f_M$ is simply gravity:

$$f_G(M_1): \mathbb{R}^{3m} \to \mathbb{R}^{3m} \, . \tag{14}$$

The material forces maintaining the structure of the quadcopter are defined as

$$f_E(M): \mathbb{R}^{6m} \to \mathbb{R}^{3m} \, . \tag{15}$$

Two specific models were implemented that prevented any significant deformation to the drone and produced similar results. One was an elastic model; however, it contained stiffness parameters so large that it closely approximated a rigid body. The other model was an exact rigid body (Featherstone 2014). For the presented purposes, a pure rigid body was sufficient, but modeling modestly deformable materials may warrant further investigation. It may be a worthwhile trade-off to build a drone with a low-cost, ultralight, and fracture-resistant material that may be modestly deformable. Furthermore, a deformable drone could be desirable because it could change shape to allow for optimal performance under various operating conditions. For example, the V-22 Osprey has two configurations—one for fast efficient flight and one for landing—and birds have all sorts of configurations.

Lastly, forces are applied to the quadcopter from the four thrusters as given by

$$g_m(M, v): \mathbb{R}^{6m} \times \mathbb{R}^4 \to \mathbb{R}^{3m} \, , \tag{16}$$

and the other forces are summed as

$$f_M(M,W) = f_w(M,W) + f_g(M_1) + f_E(M)\ . \tag{17}$$

The reconstruction operator $\mathcal{R}$ and the projection operator $\mathcal{P}$ can be used to go between the model of the quadcopter, where the state is defined by 12 numbers and the model with the arbitrary mesh:

$$\mathcal{P}(M)\colon \mathbb{R}^{6m} \to \mathbb{R}^{12}, \mathcal{R}(x)\colon \mathbb{R}^{12} \to \mathbb{R}^{3m}\ . \tag{18}$$

With these operators, $f$, can be defined simply as

$$f\big(x(t)\big) = \mathcal{P}(f_M(\mathcal{R}(x(t)))\ . \tag{19}$$

This $f$ is used in the examples in Section 5. It allows for arbitrary quadcopter geometry and arbitrary wind effects but keeps the computations faster and the controls described in Section 3 more manageable. Preliminary tests reveal realistic quadcopter behavior that compares favorably to either AirSim or a naïve implementation of built-in UE physics capabilities.

## 3. Quadcopter Controls

Controls are everywhere. Almost ubiquitous in modern machinery, controls act as a go-between for the user inputs and physical operation of a machine. Existing algorithms work well in ideal or mild weather conditions. Severe weather may be rarer but can cause catastrophic results. This section details two control algorithms applied to quadcopters. Demonstrations with both algorithms presented are described in Section 5.

A known reference solution $r(t) \in \mathbb{R}^{12}$ is defined and the error between the system state $x(t)$ and $r(t)$ is:

$$e(t) = r(t) - x(t)\ . \tag{20}$$

The system is considered controlled by making $e(t)$ small (Åström and Murray 2021). The reference solution satisfies

$$\begin{matrix} \frac{dr_1}{dt} \\ \frac{dr_2}{dt} \end{matrix} = \begin{matrix} r_2 \\ f_r\big(r(t)\big) + g_r\big(r(t), v_r(t)\big) \end{matrix}\ . \tag{21}$$

The function $g_r$ , which takes the motor inputs of the reference is considered fully known. However, $f_r$ is considered fully unknown. This function may move the quadcopter in an unphysical way (e.g., by a simple command motion from one point

to another without a conversion to motor inputs). Additionally, wind terms, whether in $f$ or $f_r$ are not directly known by the controls.

A starting point for the controls is to simply add a force to the governing ordinary differential equation (ODE):

$$\begin{matrix} \frac{dx_1}{t} \\ \frac{dx_2}{dt} \end{matrix} = \begin{matrix} x_2 \\ f\big(x(t)\big) + g\big(x(t), v_2(t)\big) + u \end{matrix} . \tag{22}$$

But, as noted in Section 2, $u$ cannot be arbitrary. There must be aan$v_2$ such that $u = g(x, v_2)$.

The linear controller will be a proportional–integral–derivative (PID) controller, of the form (Hang et al. 1991; Åström and Murray 2021):

$$v_{pid} = v_{crm} + v_{pd} + v_i \, . \tag{23}$$

PID controllers push an observed variable to a reference by applying a proportional force (P), a force relative to the integral difference of the observed and control variable (I), and a force based on the difference of the derivative of observed and reference variable (D). This simple control is the most common type of control used in real-world engineering applications.

The value of $v_{crm}$ is the direct input from a remote control converted to physical forces that can be added to the model:

$$v_{crm} = v_r \quad . \tag{24}$$

The other two terms of $v_{pid}$ require an inversion given as

$$g^{-1}: \mathbb{R}^{12} \to \mathbb{R}^8 \quad . \tag{25}$$

The 12 variables defining the state vector are converted to coordinates that can be directly affected by the quadcopter motors:

$$g^{-1}(x) = \chi = [\chi_1^T, \chi_2^T]^T \quad . \tag{26}$$

The reference solution is similarly inverted, but with an awareness of $x$, and slightly modified coordinates to allow differences between the reference and solution to appear. For example, if the reference and quadcopter were shifted in space and stationary, thrust and rotation values would be the same. The inversion of the reference solution leaves a “trail of breadcrumbs” to allow a PID to work. The reference is inverted with

$$\widetilde{g^{-1}}(x,r) = \varrho = [\varrho_1^T, \varrho_2^T]^T \ . \tag{27}$$

The error is given by

$$[\varepsilon_1^T, \varepsilon_2^T]^T = \varepsilon(t) = \varrho(t) - \chi(t) = \widetilde{g^{-1}}(x(t), r(t)) - g^{-1}(x(t)) \ . \tag{28}$$

In these converted coordinates $v_{pd}$ can now be defined as

$$v_{pd} = k_p \varepsilon_1 + k_d \frac{d\varepsilon_1}{dt} = k_p \varepsilon + k_d \varepsilon_2 \ , \tag{29}$$

and similarly, the $v_i$ term is

$$v_i = k_i \int_o^t \varepsilon_1(\tau) d\tau \ , \tag{30}$$

where $k_p$, $k_d$, and $k_i$ are appropriately chosen constants (Hang et al. 1991). Once $v_{pid}$ is found, it can be converted to update $x$ by applying $g$ and replacing the generic $u$ in the ODE:

$$u_{pid} = g\big(x, v_{pid}\big) \ . \tag{31}$$

PID controls like this are relatively easy to understand and validate. It is harder to definitively establish proper functioning and guarantee stability for nonlinear controls. However, as in other mathematical areas where a nonlinear model replaces a linear one, there is the potential for superior computing capabilities. The low cost and unmanned nature of quadcopters and similar drones (e.g., in comparison to a manned airplane) may allow for more experimentation while at the same time allowing for the development of excellent control for high-precision military applications.

The easiest way to understand the nonlinear control described in this section is through a close consideration of the "I" term in PID. The integral can be thought of, in this context, as a simple type of learning. If the quadcopter is continually missing its target over a long period of time, it will, in the simplest use of the term, "learn," that the simpler model using only a proportional and damping term is inadequate and an adjustment needs to be made. However, context goes hand-in-hand with learning. If the state of the system changes, for whatever reason, such a simple model can potentially use what it has learned in a context that does not apply.

The nonlinear control method does not just learn accumulated error; it learns error in the context of specific system states and their small variations. It has a broader and more robust ability to learn and automatically adjust because of past inadequate

responses. The general equation for the nonlinear controller is (Chowdhary and Johnson 2010; Chowdhary et al. 2014)

$$\nu_{non} = \nu_{crm} + \nu_{pd} - \nu_{ad} \ . \tag{32}$$

To begin the construction of $\upsilon_{ad}$, the following vector of size $Q$ is defined as

$$\Phi(x) = \left[\varphi_1(x), \varphi_2(x), \dots, \varphi_q(x), \dots, \varphi_Q(x)\right]^T \ , \tag{33}$$

where each term in the vector is a Gaussian radial basis function:

$$\varphi_q(x) = \exp\left(\frac{-\|x - c_q\|^2}{2\mu_q^2}\right). \tag{34}$$

The center of each of the basis functions is located at $c_q$. These basis functions require careful placement selection and calibration. If there are no basis functions near $x$, the $\upsilon_{ad}$ term will be 0 and the control will resort to a proportional–derivative (PD) control. Conversely, if there are too many widely spread basis functions, $\nu_{ad}$ will just resemble an $\upsilon_i$ term and the control will be similar to a PID. Simulation results in Section 5 were performed in real time with evenly spaced basis functions with known minimum and maximum ranges for the velocity and position in the main direction of motion, pitch, roll, pitch velocity, and roll velocity. In each of the six directions, there were either four basis functions in cartesian coordinate directions, or eight basis functions in polar coordinate directions. Better results may have been achieved by gridding up all 12 dimensions, but the computational cost was too high. There are many established mathematical means of dealing with high-dimensional spaces and quickly excluding large numbers of irrelevant basis functions (Donoho 2000), but this has not been implemented. In a simulation, as a steppingstone, many basis functions could be used, likely slowing the simulation well below real time, to assess the worthwhileness of the controls before implementing an algorithm that approximates this by carefully selecting only the most computationally important basis functions.

The nonlinear model incorporates knowledge of what the linear model is doing; therefore, the basic behavior of the linear model does not need to be re-learned. A matrix associated with the PD controller is

$$A_{pd} = \begin{bmatrix} 0 & 1 \\ -k_p & -k_d \end{bmatrix} \ . \tag{35}$$

A Lyapunov matrix, $P$, satisfies, for any positive definite matrix $Q$,

$$0 = A^T P + PA + Q \ . \tag{36}$$

It is this matrix $P$ that is used in the nonlinear controller. It carries over information from how the linear controller works in a way that provides control stability. With the entries of $P$ set as

$$P = \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{00} \end{bmatrix} . \tag{37}$$

Values $W_q$ that update in time to essentially tell the control how to adjust in certain configurations satisfy

$$\frac{dW_q}{dt} = -L(\varepsilon_1 P_{01} + \varepsilon_2 P_{11})\varphi_q(x) \ , \tag{38}$$

where $L$ is the learning rate. Finally, $\nu_{ad}$ can be given as

$$\nu_{ad} = \sum_{q=1}^{q=Q} W_q \, \varphi_q \ . \tag{39}$$

This term is nonlinear in the sense that $x$ is used to determine the relevance of a learning term and basis function, $\varphi_q(x)$, and then used again when multiplying by an error term, $\varepsilon$, which also includes $x$.

As with the PID controller, $\nu_{non}$ is appropriately transformed:

$$u_{non} = g(x, \nu_{non}) \ . \tag{40}$$

More technical details of this approach are given in the literature (Chowdhary and Johnson 2010; Chowdhary et al. 2014). This nonlinear controls process can also exist within the Gaussian process framework (Murray-Smith 2003), allowing for rigorous uncertainty analysis, though this is beyond the scope of the present work.

## 4. Virtual Worlds, Real-World Cities, Localized Wind, and an Adaptable Computing Pipeline

The drone physics and control codes were put into a modular form to make it easier to update in the future and to be more easily integrated into AirSim and potentially other software platforms. AirSim works within UE; its capabilities with PX4 RPCs (i.e., remote procedure calls), realistic sensor modeling, and video recording are utilized. UE Blueprints are employed to present the major software pieces (basic controls, nonlinear controls, physics, etc.) and their connections (see Fig. 3). UE seamlessly interacts with C++ code, where drone physics and control processes are calculated.

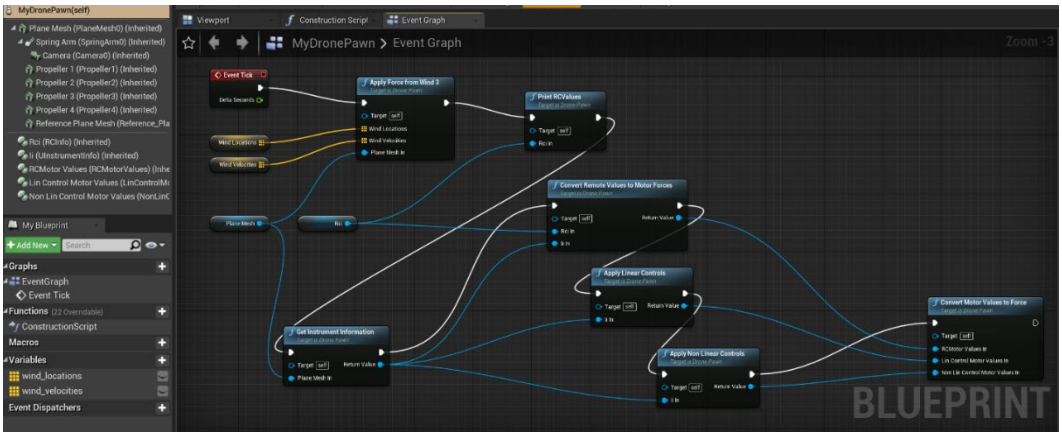


**Fig. 3 Working flowchart using UE Blueprints and showing the major modules of the overall simulation**

Our data formatting pipeline allows for the efficient and automatic conversion of raw city data provided to objects that can be fully manipulated with UE. In addition to the capacity to easily modify lighting, textures, and other aspects of the data, UE has potentially useful built-in physics capabilities and a wide array of plug-ins built by the community. Options for importing real-world city data from Mapbox and OSM as well computational fluid dynamic (CFD) wind simulation results (Wang 2020) have been explored and implemented.

This work has been done as part of a broader effort to integrate UAS control with the operational environment, with a focus on virtual representations of real-world cities and wind within those cities. We seek to leverage emerging industry and Army technologies in virtual reality (VR), augmented reality (AR), and virtual worlds. We have discussed these efforts in greater detail elsewhere; therefore, we briefly summarize here (Klipp et al. 2021; Lederman et al. 2021; Kirk et al. 2022).

On the industry side, advances in computer graphics, VR and AR headsets, "big data," and artificial intelligence (AI) are inspiring discussion of the development of the "Metaverse" or "Omniverse." Some businesses and analysts predict that these virtual worlds will extend the current Internet to be more immersive, online, and digital experiences. Similarly, current efforts in simulation are promising that "digital twins" of machines, factories, cities, and/or biological systems will increase efficiency and accelerate design, engineering, and science. For example, IBM states: "A digital twin is a virtual representation of an object or system that spans its life cycle, is updated from real-time data, and uses simulation, machine learning, and reasoning to help decision-making" (IBM Corporation n.d.).

The US Army is funding the development of its own virtual worlds, for training, mission planning, and operations. These include three interrelated projects: 1) the

Common Synthetic Environment (CSE), providing a basis for the Army's virtual worlds; 2) the Synthetic Training Environment (STE), for training; 3) and One World Terrain (OWT), for mapping and geographic information management.

The CSE/STE/OWT systems are currently under development; therefore, we have chosen to work with the free-to-use commercial gaming engine, UE, which has state-of-the-art graphics and physics capabilities, and OSM, an online, free, and open-source mapping tool. We intend for our work to remain flexible, so that our developments can be transferred over to CSE/STE/OWT, as appropriate. In addition to the work on controls described in this report, our work includes the use of city information and the calculation of localized wind fields using a lattice-Boltzmann model (Kirk et al. 2022). We are working toward combining these within the virtual environment.

## 5. Sample Results

Direct use of a remote control becomes more challenging in the presence of wind, as the user must carefully offset the weather effects while trying to move the drone along the desired trajectory. This multitasking can be difficult in practice and the relation between the degree of accuracy in the user control and the severity of the environmental factors can be difficult to establish. Appropriate control algorithms potentially allow for the quadcopter to accurately match the reference solution despite undesirable environmental factors. Like a human pilot, the control algorithm has no direct knowledge of the local weather and must respond based on the observed effect on the drone.

Two examples are presented to demonstrate the effectiveness of the two control algorithms described in Section 3. In the first example, with results given in Figs. 4 and 5, the reference moves with a sinusoidal trajectory while the quadcopter attempts to match. A spatially varying wind is present, and without continual adjustments, the drone will begin to drift away. The PID controller decently matches $r(t)$ with a bit of overshoot and undershoot. The nonlinear controller matches very well and does better on the second oscillation. The terms $W_q$ continue to build up and the controller learns to adjust better based on experience from the previous oscillation.

In the second example, the quadcopter moves to a fixed reference location in the presence of an even greater spatially varying wind velocity. Pictures and data are given in Figs. 6 and 7. The quadcopter with the linear control rolls to move to the right and then pitches to offset the force from the wind moving toward the camera. It slightly overshoots the location it is aiming for and then slowly moves to the fixed reference location because of the build-up of the "I" term in the PID controller. In

contrast, the other controller can more directly approach the reference location and reduce the positional error at a reasonable pace until reaching numerical error limitations.

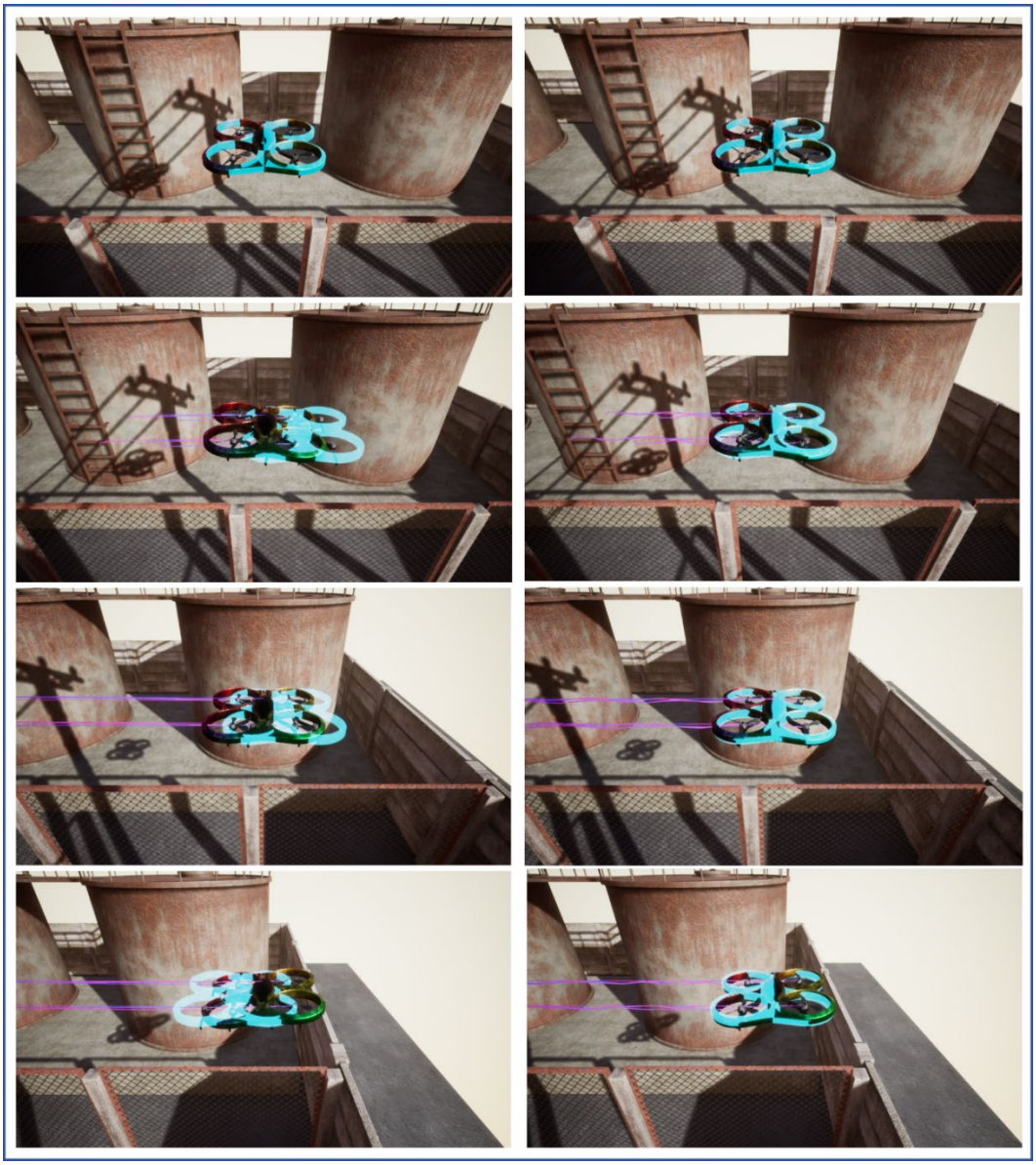

**Fig. 4 Example 1 setup that shows a reference solution (light-blue quadcopter) moving buildings in the presence of wind. The reference quadcopter moves sinusoidally in time. The left column shows the linear controller at times 0, 2, 4, and 6 s, while the right column shows the results from the nonlinear controller.**

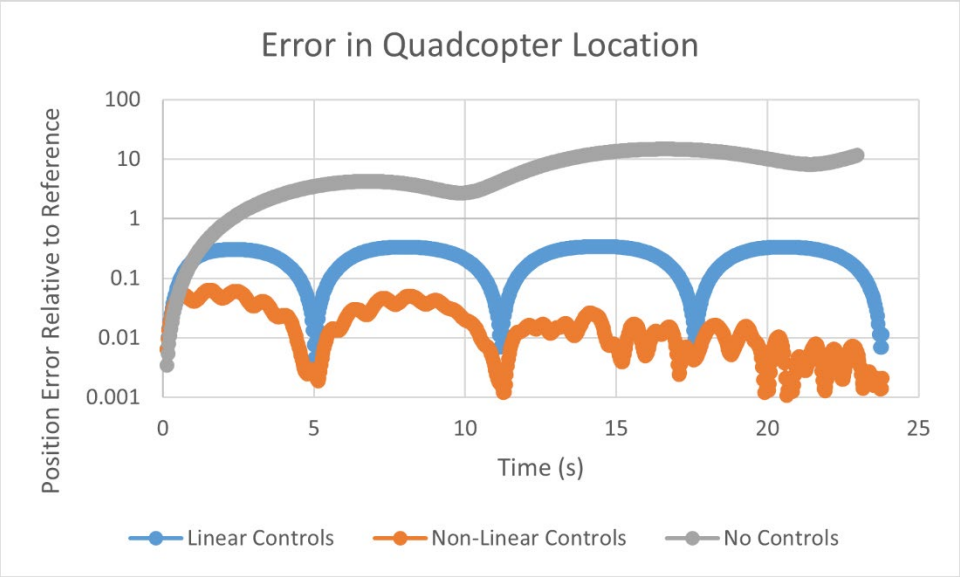


**Fig. 5 Position error for the linear and nonlinear controls. The linear controller performs decently but is subject to under- and overshooting the reference target. The nonlinear controls allow the quadcopter to match up very well with the reference solution. Furthermore, the reference solution pattern is repeated twice, and the nonlinear control shows an ability to learn and further reduce error on the second try.**

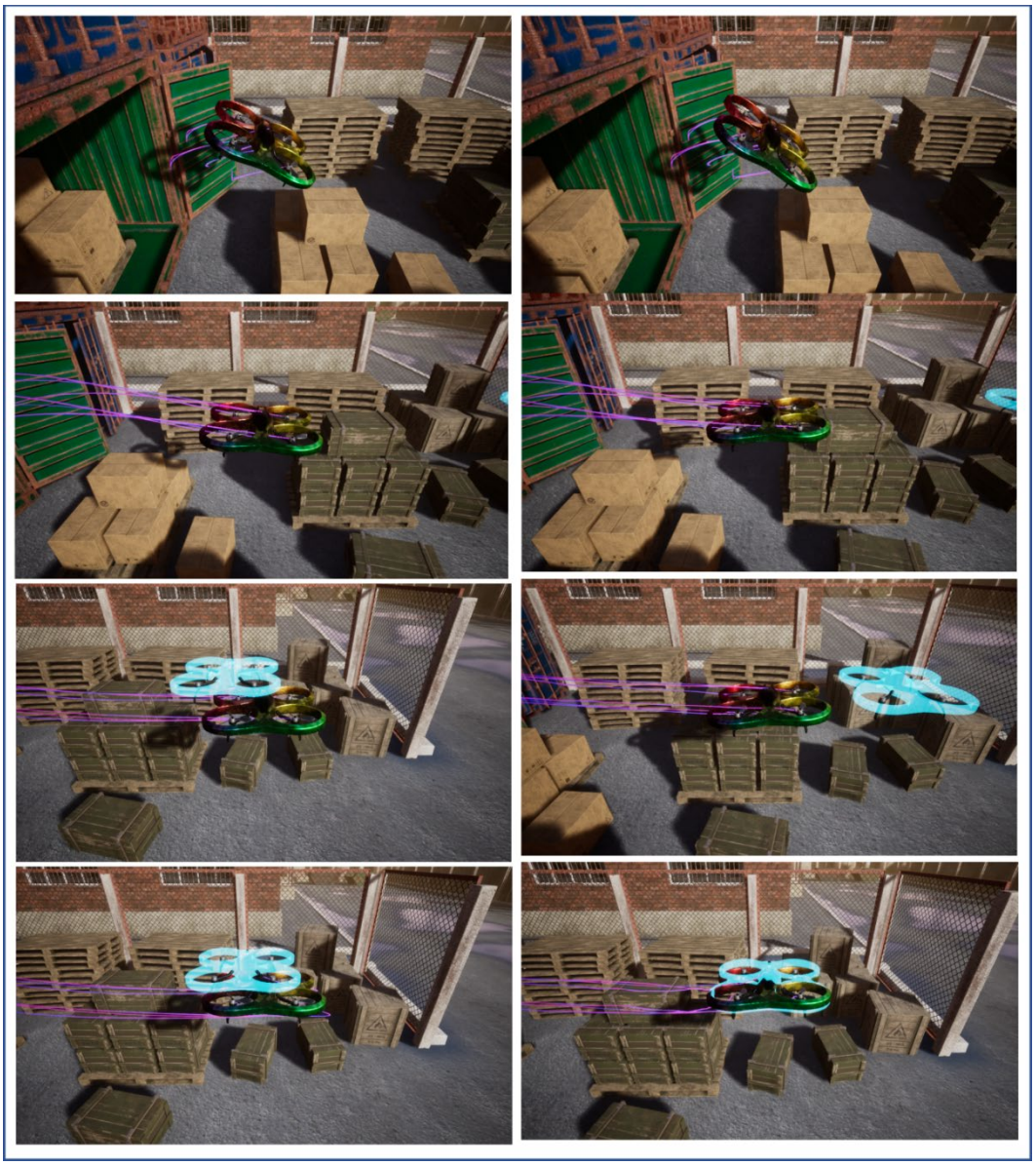

**Fig. 6 The left column shows the drone at 0.5 s (top left), 1.5 s (upper-middle left), 2.5 s (lower-middle left), and 25 s (bottom left) as it tries to move to a fixed location in the presence of heavy wind with linear controls. It quickly rolls to the right and then pitches forward to adjust for wind pushing it toward the camera. In the third image down on the left, it can be seen overshooting the target and taking some time to move closer. The drone with the nonlinear controls, shown on the right, is better able to approach the target head-on and reach the desired location faster. Images are taken at 0.5 s (top right), 1.5 s (upper-middle right), 2.5 s (lower-middle right), and 7.5 s (bottom right).**

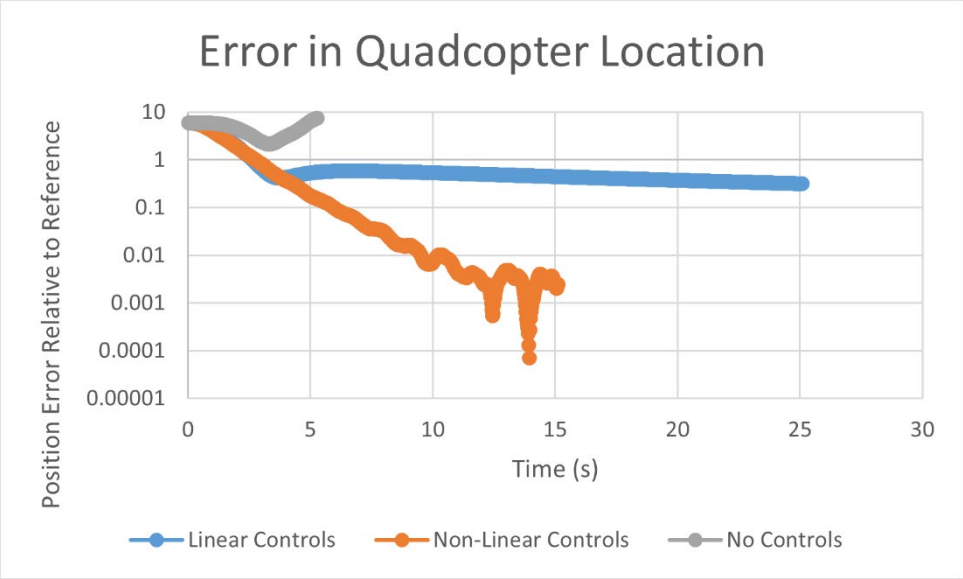


**Fig. 7 The position error vs. time is shown for the linear and nonlinear controls. After an initial burst, the linear control is only able to reach the reference point very slowly.**

## 6. Conclusion

A major benefit of the proposed approach lies in its adaptability. If set on a specific drone geometry and operational capacity, with limited variability in mission and environmental factors, it would be possible to fine-tune a better linear or simpler nonlinear control along with tailored physics. However, the ability for this method to work on a quality mesh of any geometry, to easily add any desired forces, and the method's control approach that can learn and work well without close system analysis make it a solid choice to quickly employ on a wide range of simulation scenarios.

Coinciding with adaptability is the advantage from modularized construction and the ability to incorporate open-source software. In contrast to a simulation capacity built from scratch, this method is designed to easily leverage well-designed existing and future software and algorithms. Furthermore, the software design may allow for modules or pieces to be used elsewhere in potentially larger simulation efforts.

Lastly, while more computationally expensive than some more tailored models and simulations, it is not so expensive as to prevent it from running real-time or near-real-time experiments on a single average computer. This still relatively low computational cost allows for the testing and uncertainty analysis of drone physics, environmental effects, control algorithms, and their interactions with a non-inhibitory amount of time and effort.

## 7. References


Anderson J, Sarkar D, Palen L. Corporate editors in the evolving landscape of Open Street Map. ISPRS Inter J Geo-Infor. 2019;8(5).

Angelino CV, Baraniello VR, Cicala L. UAV position and attitude estimation using IMU, GNSS and camera. Proc IEEE 15th Int Conference on Information Fusion; 2012.

Åström KJ, Murray RM. Feedback systems: an introduction for scientists and engineers, 2nd ed. Princeton University Press; 2021.

Chowdhary G, Johnson E. Concurrent learning for convergence in adaptive control without persistency of excitation. In: Proceedings of 49th IEEE Conference on Decision and Control (CDC). 2010;2674–3679.

Chowdhary G, Kingravi HA, How JP, Vela PA. Bayesian nonparametric adaptive control using Gaussian Processes. IEEE Trans. Neural Networks and Learning Systems. 2014;26:537–550.

Donoho DL. High-dimensional data analysis: the curses and blessings of dimensionality. AMS Math Challenges Lecture. 2000;1–32.

Fan Y, Lou H, Yu S. Review of the development status of UAV countermeasures. Proc. SPIE. 2022;12166.

Featherstone R. Rigid body dynamics algorithms. Springer; 2014.

Guvenc I, Koohifar F, Singh S, Sichitiu ML, Matolak D. Detection, tracking and interdiction for amateur drones. IEEE Communications Magazine. 2018:0163–6804.

Hang CC, Åström KJ, Ho WK. Refinements of the Ziegler–Nichols tuning formula. IEEE Proceedings D (Control Theory and Applications). 1991;138(2):111–118.

IBM Corporation. What is a digital twin? IBM Corporation; n.d. [accessed 2022 Mar]. https://www.ibm.com/topics/what-is-a-digital-twin.

Johnson C. Numerical solution of partial differential equations by the finite element method. Cambridge University Press; 1987.

Khan M. Quadcopter flight dynamics. Int J Sci Tech Res. 2014;3:130–135.

Kirk K, Lederman C, Wang Y, Kraczek B. The inclusion of realistic winds in a simulated environment for the study of wind-unmanned aircraft system (UAS) interactions. DEVCOM Army Research Laboratory (US); 2022 May. Report No.: ARL-TR-9465.

Klipp C, Kirk K, Lederman C, Kraczek B. Update on including turbulent winds in simulated environments for unmanned aircraft systems. DEVCOM Army Research Laboratory (US); 2021 Oct. Report No.: ARL-TR-9335.

Lederman C, Kirk K, Perry V, Kraczek B. Simulation environment for development of quad-copter controls incorporating physical environment in urban setting. Proceedings of the SPIE 11758, Unmanned Systems Technology XXIII; 2021. https://www.spiedigitallibrary.org/conference-proceedings-of-spie/11758/117580G/Simulation-environment-for-development-of-quad-copter-controls-incorporating-physical/10.1117/12.2586218.full.

Matej J. Virtual reality and vehicle dynamics in unreal engine environment. MM Science J. 2016;09:1141–1144.

Murray-Smith R, Sbarbaro D, Rasmussen CE, Girard A. Adaptive, cautious, predictive control with Gaussian process priors. IFAC Proceedings Volumes. 2003;36(16):1155–1160.

Nacouzi G, Williams JD, Dolan B, Stickells A, Luckey D, Ludwig C, Xu J, Shokh Y, Gerstein DM, Decker MH. Assessment of the proliferation of certain remotely piloted aircraft systems. RAND Corporation; 2018. Report No.: Research Report 2369.

Nelson RC. Flight stability and automatic control. WCB/McGraw Hill; 1998.

Sathyamoorthy D, Amin ZFM, Selamat E, Hassan SA, Firdaus A, Kazmar A, Zaimy Z. Evaluation of the vulnerabilities of unmanned aerial vehicles (UAVs) to global positioning system (GPS) jamming and spoofing. Defense S&T Technical Bulletin. 2020;13(2):333–343.

Shah S, Dey D, Lovett C, Kapoor A. AirSim: high-fidelity visual and physical simulation for autonomous vehicles. Field and Service Robotics; 2017.

Strikwerda J. Finite difference schemes and partial differential equations, 2nd ed. SIAM; 2004.

Stull RB. An introduction to boundary layer meteorology. Kluwer Academic Press; 1997.

Tennekes H, Lumley JL. A first course in turbulence. MIT Press; 1972.

Wang Y, Decker J, Permyak ER. Large-eddy simulations of turbulent flows around buildings using the atmospheric boundary layer environment-lattice Boltzmann model (ABLE-LBM). J Appl Meteorol Climatol. 2020:59;2020.

Watkins S, Burry J, Mohamed A, Marino M, Prudden S, Fisher A, Kloet N, Jakobi T, Clothier R. Ten questions concerning the use of drones in urban environments. Building and Environment. 2020;167:106458.

## List of Symbols, Abbreviations, and Acronyms

| | |
|---|---|
| 3-D | three-dimensional |
| AI | artificial intelligence |
| API | application programming interface |
| AR | augmented reality |
| CFD | computational fluid dynamic |
| CSE | Common Synthetic Environment |
| DOD | Department of Defense |
| GNSS | Global Navigation Satellite Systems |
| GPS | global positioning system |
| IMU | inertial measurement unit |
| LiDAR | Light Detection and Ranging |
| ODE | ordinary differential equation |
| OSM | OpenStreetMap |
| OWT | One World Terrain |
| PD | proportional–derivative |
| PID | proportional–integral–derivative |
| RF | radio frequency |
| RTPS | real-time-publish-scribe |
| STE | Synthetic Training Environment |
| UAS | unmanned aerial system |
| UE | Unreal Engine |
| VR | virtual reality |

| | |
|---|---|
| 1 (PDF) | DEFENSE TECHNICAL INFORMATION CTR DTIC OCA |
| 1 (PDF) | DEVCOM ARL FCDD RLD DCI TECH LIB |
| 1 (PDF) | DEVCOM ARL FCDD RLC ED B KRACZEK |
| 1 (PDF) | LUFBURROW & COMPANY, INC C LEDERMAN |