

## Article

# Lidar-Based Navigation of Subterranean Environments Using Bio-Inspired Wide-Field Integration of Nearness

Michael T. Ohradzansky <sup>1,\*</sup>  and J. Sean Humbert <sup>2</sup>

<sup>1</sup> Department of Aerospace Engineering Sciences, University of Colorado Boulder, 3775 Discovery Drive, Boulder, CO 80303, USA

<sup>2</sup> Department of Mechanical Engineering, University of Colorado Boulder, 427 UCB, 1111 Engineering Dr, Boulder, CO 80309, USA; sean.humbert@colorado.edu

\* Correspondence: michael.ohradzansky@colorado.edu

**Abstract:** Navigating unknown environments is an ongoing challenge in robotics. Processing large amounts of sensor data to maintain localization, maps of the environment, and sensible paths can result in high compute loads and lower maximum vehicle speeds. This paper presents a bio-inspired algorithm for efficiently processing depth measurements to achieve fast navigation of unknown subterranean environments. Animals developed efficient sensorimotor convergence approaches, allowing for rapid processing of large numbers of spatially distributed measurements into signals relevant for different behavioral responses necessary to their survival. Using a spatial inner-product to model this sensorimotor convergence principle, environmentally relative states critical to navigation are extracted from spatially distributed depth measurements using derived weighting functions. These states are then applied as feedback to control a simulated quadrotor platform, enabling autonomous navigation in subterranean environments. The resulting outer-loop velocity controller is demonstrated in both a generalized subterranean environment, represented by an infinite cylinder, and nongeneralized environments like tunnels and caves.



**Citation:** Ohradzansky, M.; Humbert, J. S. Lidar-Based Navigation of Subterranean Environments Using Bio-Inspired Wide-Field Integration of Nearness. *Sensors* **2022**, *22*, 849. <https://doi.org/10.3390/s22030849>

Academic Editors: Julien Serres, Poramate Manoonpong, Paolo Arena and Luca Patanè

Received: 31 October 2021

Accepted: 18 January 2022

Published: 23 January 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** bio-inspired navigation; sensorimotor convergence; subterranean exploration; quadrotor control

## 1. Introduction

There is significant interest in using small Unmanned Aerial Systems (sUAS) for autonomous tasks, ranging from rapid situational awareness of hazardous environments to search and rescue missions. In each application, the system's ability to sense the environment around itself is critical to autonomous navigation, and generating control commands in a fast and efficient manner is still an ongoing challenge. This challenge is even more difficult in subterranean environments, where degraded sensing, diverse unstructured terrain, and a lack of Global Positioning System (GPS) signals can cause problems with traditional navigation methods. More detail on the challenges and proposed solutions to autonomous subterranean navigation are presented in [1–12]. Generally, proposed methods for subterranean navigation include some mix of the following subsystems: sensing and perception, state-estimation, map generation, planning, and guidance. These subsystems and processes can require multiple sensing modalities and significant computing resources, which ultimately leads to undesired size, weight, and power (SWaP) constraints on the vehicle. Additionally, all of these processes are directly affected by state-estimation accuracy; if the state estimates are noisy or experience significant drift over time, it can affect the quality of the map, which in turn affects the system's ability to navigate. To overcome these challenges and limitations, novel sensors and sensor processing methods need to be developed to continue to push forward robot autonomy.

For inspiration, researchers can look to countless examples in nature of elegant solutions to complex perception and navigation problems. Animals evolved for hundreds of

millions of years. As a result, they developed efficient sensorimotor systems for processing spatially distributed sensor measurements. As engineers, we can learn from these incredible biological systems and use the underlying principals to come up with unique solutions to the different challenges of robot autonomy.

One example is the mechanosensory lateral line system, which allows fish to navigate their surroundings by sensing water flow. The mechanosensory lateral line system is composed of two different types of spatially distributed sensors: the superficial neuromast (SN) and the canal neuromast (CN). The SNs are located on the body of the fish and sense local flow velocity, while the CNs are located in subepidermal canals and sense flow acceleration [13,14]. Specialized neural pathways enable the fish to achieve different behaviors like rheotaxis [15,16], schooling [17], obstacle avoidance [13], and prey detection [18,19]. By mimicking the mechanosensory lateral line system, reflexive obstacle avoidance can be achieved through the spatial decomposition of electric field measurements [20,21]. Another example is the spider mechanosensory system composed of hair and strain sensors that cover the abdomen and legs [22]. By processing many different spatially distributed measurements from the touch sensitive hairs, spiders are more equipped to detect and localize prey [23,24]. These hairs also allow them to sense air flow, allowing them to localize airborne prey and estimate ego-motion [25–27]. A third example of particular interest is the insect visuomotor system, which enables fast flight and obstacle avoidance. The insect compound eye is composed of thousands of ommatidia, or light receptors. Insects rely on optic flow, or the characteristic patterns of visual motion that are excited as they move [28–30]. By processing the changing patterns of light (optic flow) sensed by the ommatidia (the insect compound eye), insects are able to generate reactive motor commands allowing them to make rapid adjustments to their flight and avoid obstacles [31]. Additional analysis of the insect visuomotor system shows how specialized tangential cells process distributed optic flow measurements, inspiring the wide-field integration framework presented in [32–34]. Through observation of insects and other flying animals in different environments, optic flow-based navigation methods were developed [35–42]. Small-obstacle avoidance algorithms based on optic flow measurements are demonstrated onboard a multirotor platform in [43,44].

In each of these examples, hundreds to thousands of spatially distributed sensor measurements are processed simultaneously to produce different behavioral responses. Past studies focused on extracting ego-motion estimates and/or relative proximity to obstacles from optic flow or electric field measurements, whereas the main sensing mode for this study is relative depth observed by a spherical LiDAR sensor model. The proposed controller is reactive, which means that it generates control commands directly from the raw sensor measurements. Other reactive control approaches can be divided into the following subgroups: artificial potential fields, admissible-gap navigation, and laser-based reactive control approaches. In artificial potential field approaches, the known, or sensed, environment is converted to a field of forces [45–48]. The admissible-gap approach uses depth scans to generate a nearness diagram and then identify gaps to traverse [49]. The approach is strictly planar, however improvements to the initial algorithm were made including smooth nearness-diagram navigation [50], closest gap navigation [51], and tangential gap navigation [52]. This approach was recently implemented onboard a small multirotor platform and tested in both indoor and outdoor environments [53]. Other noteworthy LiDAR-based approaches are presented in [54], where depth scans are used to plan a path through the known environment using an improved rapidly-exploring random tree algorithm (RRT\*) and [8], where depth scans are used to generate a centering response for graph navigation.

In this work, a bioinspired wide-field integration (WFI) framework for subterranean navigation is presented where the primary sensing modality is measured depth. Spatially distributed depth measurements are a rich source of information relevant to navigation. Using the spatial inner-product, a WFI method inspired by the Lobula Plate Tangential Cells (LPTCs) of the insect visuomotor system, environment relative states can be extracted from

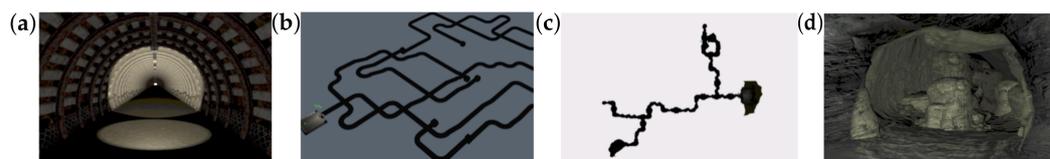
depth scans and applied as feedback to achieve a 3D centering response in subterranean environments. With careful selection of 3D weighting shape functions, or sensitivity shapes, control commands can be computed directly from depth scans in a single operation. The control algorithm is demonstrated onboard a simulated quadrotor platform in both a generalized cylinder environment and other subterranean environments with more diverse topology. The main advantage to this approach is its simplicity: it does not require state-estimates, local or global maps, or planned paths, and control commands are generated directly from each successive depth scan. By integrating across thousands of depth measurements, the algorithm is robust to sensor noise. The algorithm is also robust to changes in the environment. In the development of the algorithm the local environment is approximated as an infinite cylinder, however, navigation in more diverse subterranean environments is demonstrated.

This paper is outlined as follows. In Section 2, the development of the bio-inspired nearness controller is presented, as well as a description of the platform and simulated testing environment. Following the presentation of the algorithm, results from autonomous flight tests are presented in Section 3. The advantages and limitations of the controller are discussed in Section 4

## 2. Materials and Methods

### 2.1. System and Sensing Models

For the purpose of testing the algorithm, the DARPA SubT Simulator is used (Available online: <https://www.subtchallenge.com> (accessed on 17 January 2022)). The DARPA SubT Simulator provides access to a number of different simulated subterranean environments that fall into three main subdomains: tunnel, urban, and cave. Each subdomain provides unique navigation challenges to autonomous platforms, such as the tight and narrow corridors of the tunnel worlds or the nonuniform caverns present in the cave worlds. Examples of these environments are shown in Figure 1. Initial formulation of the controller assumes an infinite cylinder as the generalized environment, which can also be loaded into the DARPA SubT simulator as a custom world. The SubT virtual code repository and resources on how to use it are hosted on GitHub (Available online: <https://github.com/osrf/subt/wiki> (accessed on 17 January 2022)).



**Figure 1.** Screenshots of different virtual environments, each spanning hundreds of meters in length and width. (a) A simple, straight tunnel environment of radius 1.75 m and length 600 m. (b) Overview of a larger, more complex tunnel environment with an average radius of 1.75 m and spanning approximately 200 m width  $\times$  300 m length. (c) Top-down view of a cave environment spanning approximately 600 m length  $\times$  425 m width, with an average cave radius greater than 5 m. (d) Entrance to a cave world environment, spanning approximately 10 m width  $\times$  15 m height  $\times$  20 m length.

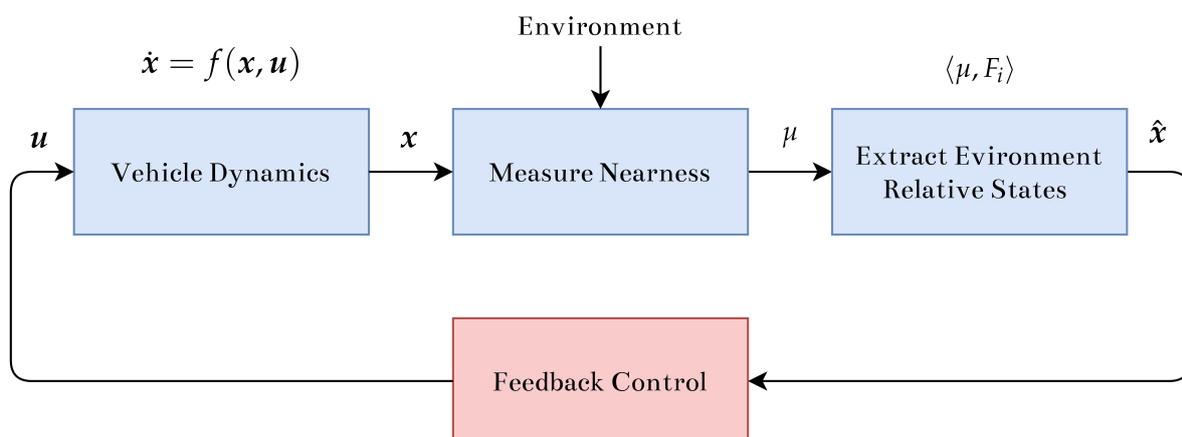
In addition to simulation worlds, the DARPA SubT Simulator provides access to robotic platforms developed by DARPA SubT Challenge Virtual competition teams complete with full sensor suites. Users can customize different platforms, like multirotor aerial systems or quadrupedal ground vehicles, with sensors that suite the needs of their algorithms. For the purpose of this study, the standard quadrotor “X3” model is used with outer-loop velocity control. Navigation of the different environments is achieved by setting a fixed forward speed and generating lateral and vertical velocity commands as well as heading rate commands. The inner-loop attitude controller is handled by the DARPA SubT Simulator quadrotor controller, which converts velocity commands to individual motor

speeds of the simulated vehicle for attitude control and stabilization. A depth sensing model based on existing LiDAR sensors is used to measure depth to discrete points in the environment. Current LiDAR-based sensors have limited field of views, however for the purpose of this study a LiDAR sensor with a full spherical field of view is used. Depth measurements  $d(\theta_v, \phi_v, \mathbf{x})$  are discretized according to the body-referred viewing angles  $\theta_v$  and  $\phi_v$ , where  $\theta_v \in [0, 2\pi]$  is the inclination angle and  $\phi_v \in [-\pi, \pi]$  is the azimuthal angle, and are a function of the environment relative system states  $\mathbf{x}$ , which are described in greater detail in Section 2.2.1. In this case, 64 equally spaced inclination angles and 360 equally spaced azimuthal angles are used, resulting in just over 23,000 depth measurements per scan. In the Robot Operating System (ROS), the incoming depth scans are represented as collections of  $(x, y, z)$  points, and so each depth measurement is made up of 3 floating point numbers. Depth scans are reported at a rate of 20 Hz, and the bandwidth of the sensor data is estimated to be around 5.27 MB/s. The LiDAR model has a max sensing distance  $d_{max} = 100$  m and minimum sensing distance  $d_{min} = 0.05$  m. Additive Gaussian white noise is added to each depth measurement, with a nominal standard deviation of 0.03. Measured depth is hypothetically unbounded, and so depth is converted to nearness, as shown in Equation (1):

$$\mu(\theta_v, \phi_v, \mathbf{x}) = \frac{1}{d(\theta_v, \phi_v)} \in (0, d_{min}^{-1}) \quad (1)$$

## 2.2. Bio-Inspired Nearness Control

The goal is to develop an outer-loop velocity controller that produces a 3D centering response in subterranean environments using spatially distributed nearness measurements, or nearness scans. This is achieved by extracting environment relative information from the nearness scan using the spatial inner-product. First, the local environment is approximated as an infinite cylinder, which appropriately represents a subterranean environment where large obstacles can exist above, below, and to the sides of the vehicle. When navigating unknown environments, it is often desirable to fly as far away from large obstacles, such as walls, floors, and ceilings, as possible. In the case of subterranean environments, this can be achieved by driving the system to the middle of the local environment. By selecting appropriate weighting shapes for the spatial inner-product, these environment relative states can be extracted from the nearness scans of the onboard sensor. Linear feedback control is used to drive the system away from obstacles towards the center of the cylinder and point down the centerline. The general process is shown in Figure 2.



**Figure 2.** Process flow diagram demonstrating interactions between main subsystems. Nearness is measured from environment and environment relative states are extracted and used for control feedback, ultimately affecting vehicle dynamics.

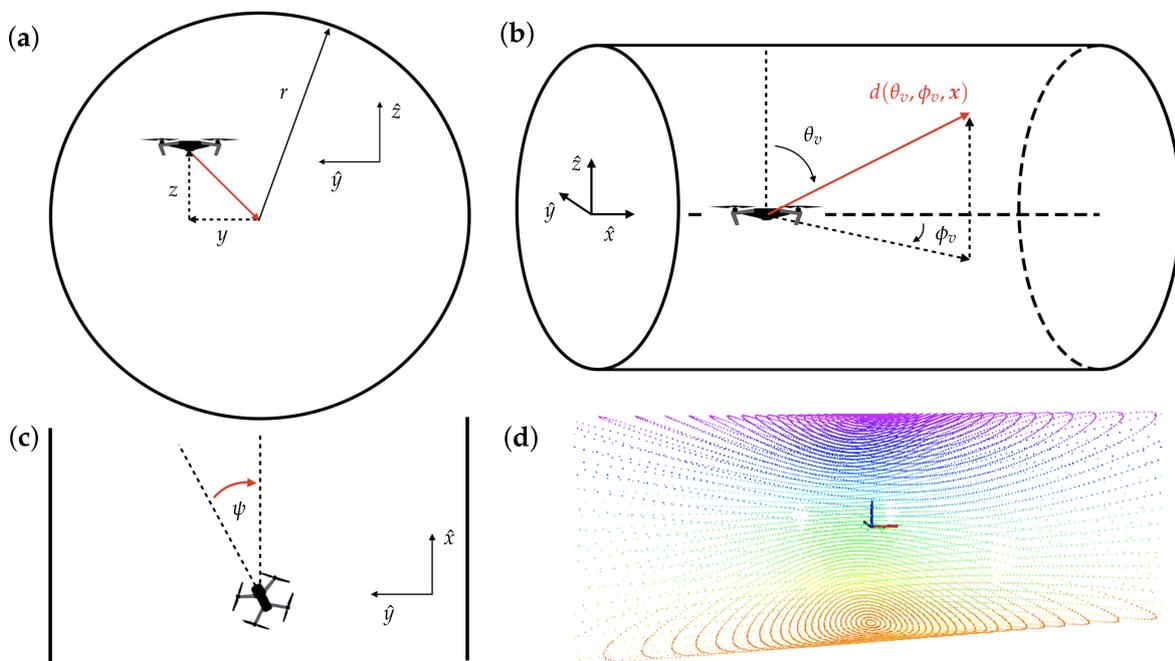
Section 2.2.1 presents the lytic function for nearness in the approximated local environment. In Section 2.2.2, the process of extracting the environment relative states through WFI is presented in detail. Design of the linear feedback controller is presented in Section 2.2.3.

### 2.2.1. Parameterization of the Environment

It is useful to approximate the local environment around the vehicle as some generalized shape with a known analytic representation. This enables the determination of the environment relative states to use for feedback and also allows an initial set of feedback gains to be derived. As stated above, a cylinder is selected as the generalized local environment, as it simulates the presence of obstacles above, below, and to the sides of the vehicle. Figure 3 shows different views of a quadrotor model flying through a simple cylinder environment. The controller goal is to drive the system towards the centerline of the cylinder, and the environment relative states that can be used for feedback to achieve this are the lateral, vertical, and heading displacements from the cylinder centerline defined as  $y$ ,  $z$ , and  $\psi$ . As the system moves through the environment, the expected shape of the measured nearness can be determined by intersecting a line in the body-referred spherical coordinate system with the surface of a cylinder (see Appendix A for a more detailed derivation). One simplifying assumption is made, which is that the roll and pitch angles of the vehicle are assumed to be negligibly small. The analytic representation of nearness as a function of the cylinder radius  $r$ , the body-referred viewing angles  $(\theta_v, \phi_v)$ , and the environment relative states  $x = [y \ z \ \psi]^T$  is shown in the following equation:

$$\mu(\theta_v, \phi_v, x) = \frac{c(\theta_v)^2 + s(\phi_v + \psi)^2 s(\theta_v)^2}{zc(\theta_v) + ys(\phi_v + \psi)s(\theta_v) + \sqrt{ac(\theta_v)^2 + s(\phi_v + \psi)[bs(\phi_v + \psi)s(\theta_v)^2 + yzs(2\theta_v)]}} \quad (2)$$

where  $\sin(x)$  and  $\cos(x)$  were simplified to  $s(x)$  and  $c(x)$ ,  $a = (r - y)(r + y)$ , and  $b = (r - z)(r + z)$ . While this is a complicated nonlinear function, estimates of the environment relative states can be extracted using WFI.



**Figure 3.** Different views of an infinite cylinder environment are shown with corresponding environment relative variables. (a) View looking down cylinder. (b) Full 3D view of cylinder environment. (c) Top-down view of cylinder. (d) Example of a measured depth scan in a cylinder environment using spherical LiDAR sensor. The points are colored based on their  $z$  position, with warmer colors representing points below the  $x$ - $y$  plane and cooler colors representing points above the  $x$ - $y$  plane.

### 2.2.2. Wide-Field Integration

In the visuomotor system of the fly, tangential cells are large motion-sensitive neurons that are sensitive to different flow patterns. Different tangential cells are sensitive to different stimulus patterns, and the integrated output is effectively a comparison between the cell's sensitivity pattern and the measured stimulus. The outputs of these cells are pooled to produce different motor responses. This process can be represented mathematically by the spatial inner-product, shown in Equation (3), which compares two spherical nearness patterns.

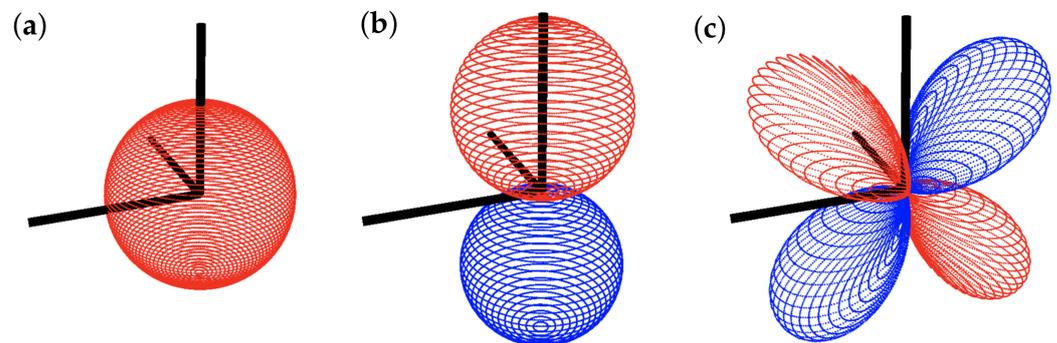
$$\langle \mu, F_i \rangle = \iint_{\Omega} \mu(\theta_v, \phi_v, \mathbf{x}) F_i(\theta_v, \phi_v) d\Omega \quad (3)$$

Here,  $d\Omega = \sin(\theta_v) d\theta_v d\phi_v$  is the solid angle of the sphere. By projecting nearness onto different weighting shape functions  $F_i$ , the function's dependence on the viewing angles  $\theta_v$ , and  $\phi_v$  is removed. Real spherical harmonics, shown in Figure 4, are used as the basis set for the sensitivity shapes, and are parameterized by the viewing angles  $\theta_v$  and  $\phi_v$  according to:

$$Y_l^m(\theta_v, \phi_v) = \sqrt{\frac{(2l+1)(l-m)!}{4\pi(l+m)!}} P_l^m(\cos\theta_v) e^{im\phi_v} \quad (4)$$

where  $P_l^m(\cos\theta)$  is the associated Legendre function  $l \in \mathbb{Z}$ ,  $m \in \mathbb{Z}$ ,  $l \geq 0$ ,  $|m| \leq l$ . With spherical harmonics as the weighting shapes  $F_i$ , Equation (3) can be rewritten as:

$$p_i(\mathbf{x}) = \langle \mu, Y_l^m \rangle = \iint_{\Omega} \mu(\theta_v, \phi_v, \mathbf{x}) Y_l^m(\theta_v, \phi_v) \sin(\theta_v) d\theta_v d\phi_v \quad (5)$$



**Figure 4.** Examples of spherical harmonic shapes, where red points have positive magnitude and blue points have negative magnitude. (a)  $Y_0^0$  Mode (b)  $Y_0^1$  Mode (c)  $Y_{-1}^2$  Mode.

The resulting outputs  $p_i(\mathbf{x})$  from Equation (5) are nonlinear functions of the state, and can be linearized for small motions around  $\mathbf{x}_0$  to produce a set of linear equations of the form  $\mathbf{p} = \mathbf{C}\mathbf{x}$ . This observation model can be inverted to produce a linear relationship between the projections and the environment relative states  $\mathbf{x} = \mathbf{C}^\dagger \mathbf{p}$ , where  $\mathbf{C}^\dagger = (\mathbf{C}^T \mathbf{C})^{-1} \mathbf{C}^T$ . The  $\mathbf{C}^\dagger$  matrix relating the projections to the states for the first nine spherical harmonics is found to be:

$$\mathbf{C}^\dagger = \begin{bmatrix} 0 & 0 & 0 & \frac{r^2}{2\sqrt{\frac{\pi}{3}}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{r^2}{2\sqrt{\frac{\pi}{3}}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3r^2}{8\sqrt{\frac{\pi}{3}}} \end{bmatrix} \quad (6)$$

The spatial inner-product is a linear operator, and so the  $\mathbf{C}^\dagger$  matrix, which relates projections to states, can be moved inside the inner-product resulting in the following:

$$\hat{\mathbf{x}}_i = \langle \mu, \sum_{j=1}^M \mathbf{C}_{ij}^\dagger F_j \rangle \quad (7)$$

where  $M$  is the number of weighting shapes. The second term in the inner-product can be interpreted as the weighting shape for extracting the  $i$ 'th environment relative state:

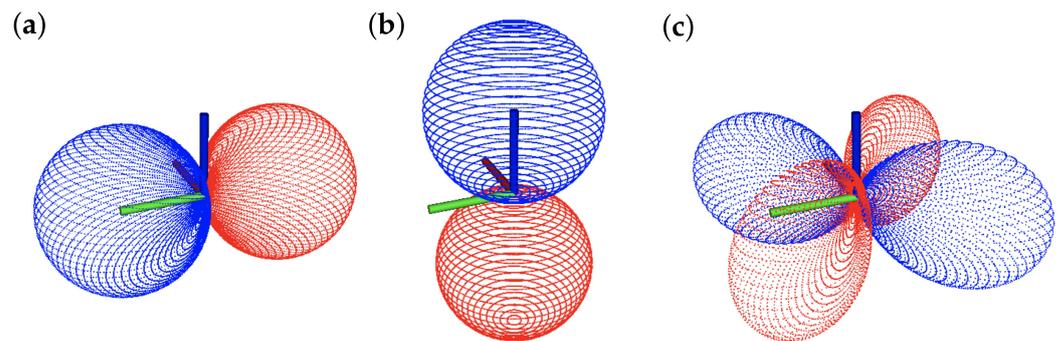
$$F_{\hat{x}_i} = \sum_{j=1}^M C_{ij}^{\dagger} F_j \quad (8)$$

For the first nine spherical harmonics, a single projection correlates with each of the three environment relative states. The closed forms of the state sensitivity shapes are shown in Equations (9) and (10), and a visual representation is shown in Figure 5.

$$F_{\hat{y}} = \frac{r^2}{2\sqrt{\frac{\pi}{3}}} Y_1^{-1}(\theta_v, \phi_v) \quad (9)$$

$$F_{\hat{z}} = \frac{r^2}{2\sqrt{\frac{\pi}{3}}} Y_0^1(\theta_v, \phi_v) \quad (10)$$

$$F_{\hat{\psi}} = \frac{3r^2}{8\sqrt{\frac{\pi}{3}}} Y_2^{-2}(\theta_v, \phi_v) \quad (11)$$



**Figure 5.** State sensitivity shapes where red points have positive magnitude and blue points have negative magnitude. Origin axis markers are of length 0.5 m. In this linearized example, state sensitivity shapes are simply scaled versions of Laplace spherical harmonics. (a) Lateral state  $y$  sensitivity shape  $F_{\hat{y}}$ ; (b) vertical state  $z$  sensitivity shape  $F_{\hat{z}}$ ; (c) heading state  $\psi$  sensitivity shape  $F_{\hat{\psi}}$ .

### 2.2.3. Feedback Control Design

Quadrotor platforms are able to move in any direction, subject to dynamic constraints. At low speeds, the linear dynamics models for the forward, lateral, vertical, and heading states can be decoupled and represented by the following dynamics models:

$$\dot{u} = X_u u + X_{\delta_{Fwd}} \delta_{Fwd} \quad (12)$$

$$\begin{bmatrix} \dot{y} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & Y_v \end{bmatrix} \begin{bmatrix} y \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ Y_{\delta_{Lat}} \end{bmatrix} \delta_{Lat} \quad (13)$$

$$\begin{bmatrix} \dot{z} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & Z_w \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} + \begin{bmatrix} 0 \\ Z_{\delta_{Vert}} \end{bmatrix} \delta_{Vert} \quad (14)$$

$$\begin{bmatrix} \dot{\psi} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & N_r \end{bmatrix} \begin{bmatrix} \psi \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ N_{\delta_{Head}} \end{bmatrix} \delta_{Head} \quad (15)$$

Equation (12) shows the forward velocity dynamics of the vehicle where  $X_u$  is the forward, or longitudinal, stability derivative and  $X_{\delta_{Fwd}}$  is the forward control derivative. A fixed, or regulated forward speed is desired, and no forward position states are required. Equation (13) shows the lateral dynamics model of the system where  $Y_v$ ,  $Y_p$ ,  $L_v$ , and  $L_p$

are the lateral stability derivatives and  $Y_{\delta_{Lat}}$  and  $L_{\delta_{Lat}}$  are the lateral control derivatives. Similarly, Equation (14) shows the vertical dynamics model where  $Z_w$  is the vertical stability derivative and  $Z_{\delta_{Vert}}$  is the vertical control derivative. Last is Equation (15), which shows the heading dynamics of the system where  $N_r$  is the heading stability derivative and  $N_{\delta_{Head}}$  is the heading control derivative. The inputs to the system are the forward, lateral, vertical, and heading control efforts  $\delta_{Fwd}$ ,  $\delta_{Lat}$ ,  $\delta_{Vert}$ , and  $\delta_{Head}$ . The table in Appendix B lists the values used for all the stability derivative constant terms, which were estimated for a similar quadrotor platform in [55].

The control efforts are determined using a simple feedback control scheme where the desired states are compared to the estimated states:

$$\delta_{Lat} = K_v(y_{des} - \hat{y}) \quad (16)$$

$$\delta_{Vert} = K_w(z_{des} - \hat{z}) \quad (17)$$

$$\delta_{Head} = K_r(\psi_{des} - \hat{\psi}) \quad (18)$$

where  $K_v$ ,  $K_w$ , and  $K_r$  are the lateral, vertical, and heading controller gains. A fixed forward speed is used directly as an input to the controller. The closed loop dynamics equations are shown in Equations (19)–(21):

$$\begin{bmatrix} \dot{y} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -Y_{\delta_{Lat}}K_v & Y_v \end{bmatrix} \begin{bmatrix} y \\ v \end{bmatrix} \quad (19)$$

$$\begin{bmatrix} \dot{z} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -Z_{\delta_{Vert}}K_w & Z_w \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} \quad (20)$$

$$\begin{bmatrix} \dot{\psi} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -N_{\delta_{Head}}K_r & N_r \end{bmatrix} \begin{bmatrix} \psi \\ r \end{bmatrix} \quad (21)$$

Stability of the controlled system can be determined using simple linear stability analysis. In each case, with a selection of  $K_i > 0$  the decoupled linearized systems are stable for small deviations from the equilibrium.

Similar to before, the controller gains can also be moved inside the spatial inner-product:

$$u_i = \langle \mu, \sum_{j=1}^M K_i C_{ij}^{\dagger} F_j \rangle \quad (22)$$

Now, the control commands  $u_i$  can be directly computed from the measured nearness scan using the spatial inner-product. In this case, the second term in the inner-product is a spatially distributed weighting shape that is not only sensitive to the different environment relative states, but is also scaled to produce the desired control commands:

$$F_{u_i} = \sum_{j=1}^M K_i C_{ij}^{\dagger} F_j \quad (23)$$

For each of the three environment relative states, a different weighting function  $F_{u_i}$  is used to generate control commands from the measured nearness scans. In this case, the estimated states are linear functions of a single projection, as seen in Equations (9)–(11), and so the control command weighting shapes are just scaled versions of the shapes seen in Figure 5.

While a fixed forward speed can work in most environments, performance can be improved by regulating the forward speed. In this case, the speed response from [33] is used for inspiration for the following control law:

$$u_{des} = u_{max}(1.0 - K_{u_v}|\hat{y}| - K_{u_w}|\hat{z}| - K_{u_\psi}|\hat{\psi}| - K_{u_{fwd}}p(3)) \quad (24)$$

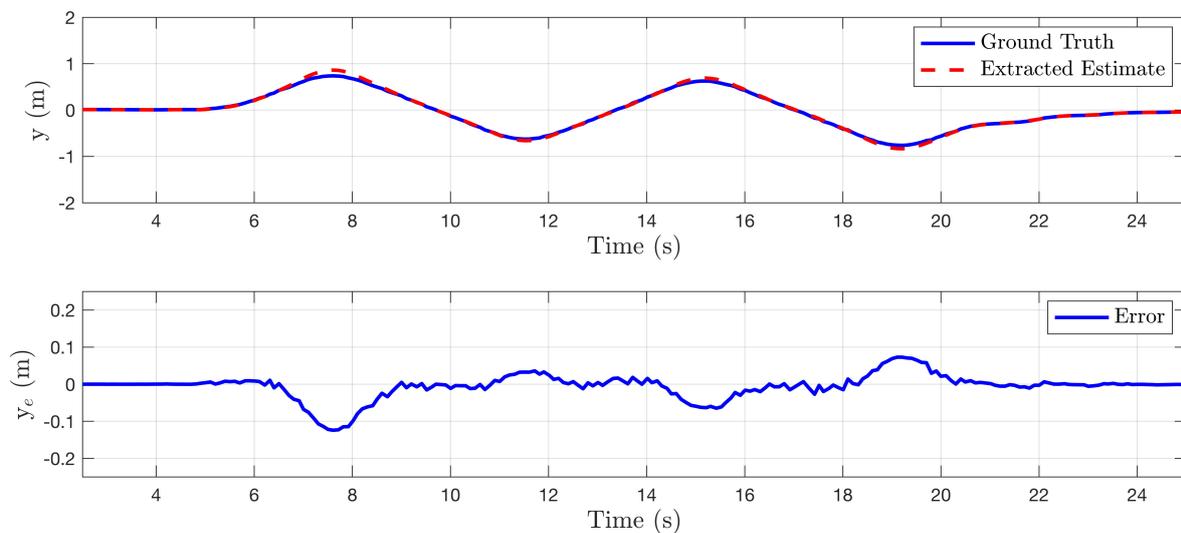
Here, the forward speed control gains  $K_{u_v}$ ,  $K_{u_w}$ ,  $K_{u_\psi}$ ,  $K_{u_{fwd}}$  are used to tune the forward speed regulation response to errors in the lateral, vertical, and heading states as well as approaching obstacles through the  $p(3)$  projection. By regulating the forward speed based on errors in the environment relative states, the system is better equipped to navigate changes in the environment.

### 3. Results

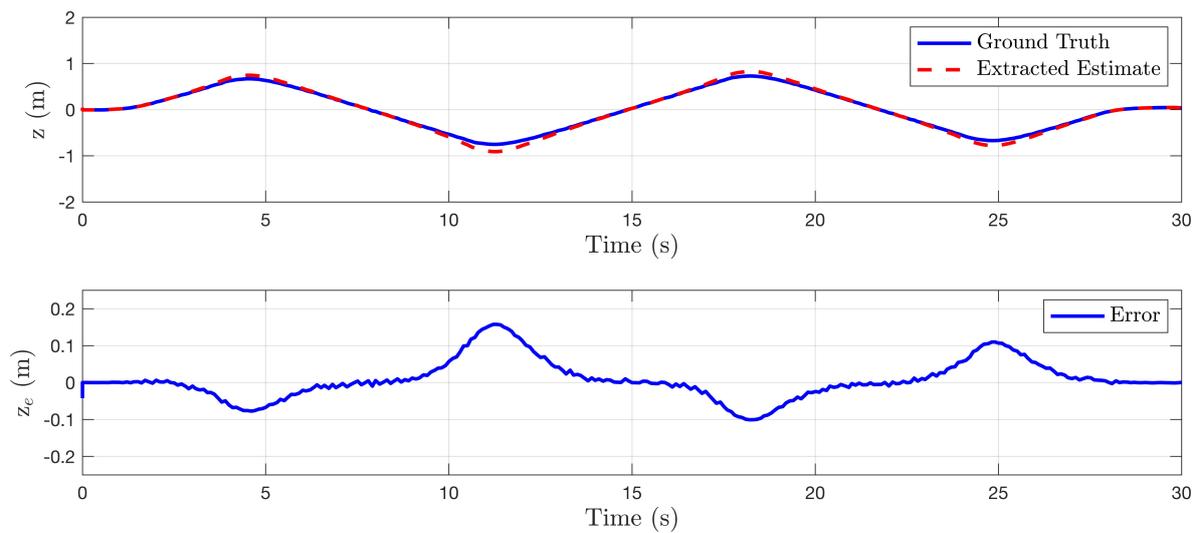
In the following subsections, performance of the algorithm in different simulated environments is presented. In Section 3.1, the performance of the system in a perfect cylinder environment is demonstrated. Zero-mean Gaussian white noise is added to the depth measurements in Section 3.2, demonstrating the algorithm's robustness to sensor noise. To demonstrate the robustness of the algorithm in nongeneralized environments, results from tests conducted in tunnel and cave environments are presented in Section 3.3.

#### 3.1. Performance in Generalized Cylinder Environment

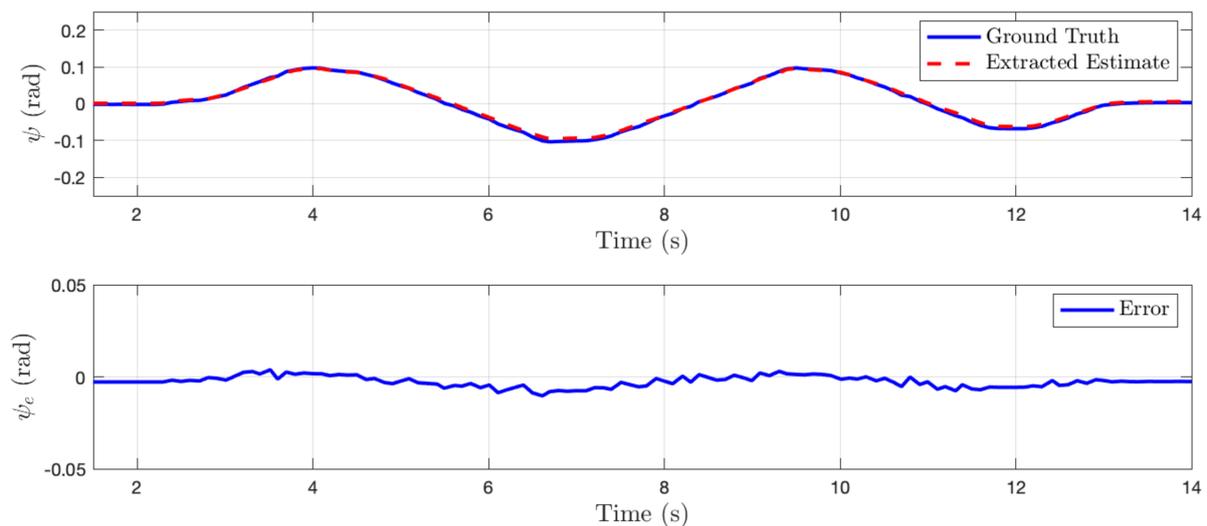
In this section, the algorithm is tested in a simulated perfect cylinder environment to demonstrate the effectiveness of the method. For the first set of tests, the quadrotor system was manually perturbed off the centerline of the cylinder and state estimates generated from the state sensitivity shapes  $F_{x_i}$  were recorded. No sensor noise was added to any of the tests presented in this subsection. Figures 6–8 compare the environment relative state estimates, shown as a red dashed line, to the ground truth, shown in blue. In each of these cases, the estimation error is negligible for small perturbations, however, large perturbations eventually begin to skew the estimated states.



**Figure 6.** (top) Comparison of estimated lateral distance of cylinder centerline to ground truth as system moves back and forth laterally in environment. (bottom) State error plotted as a function of time.

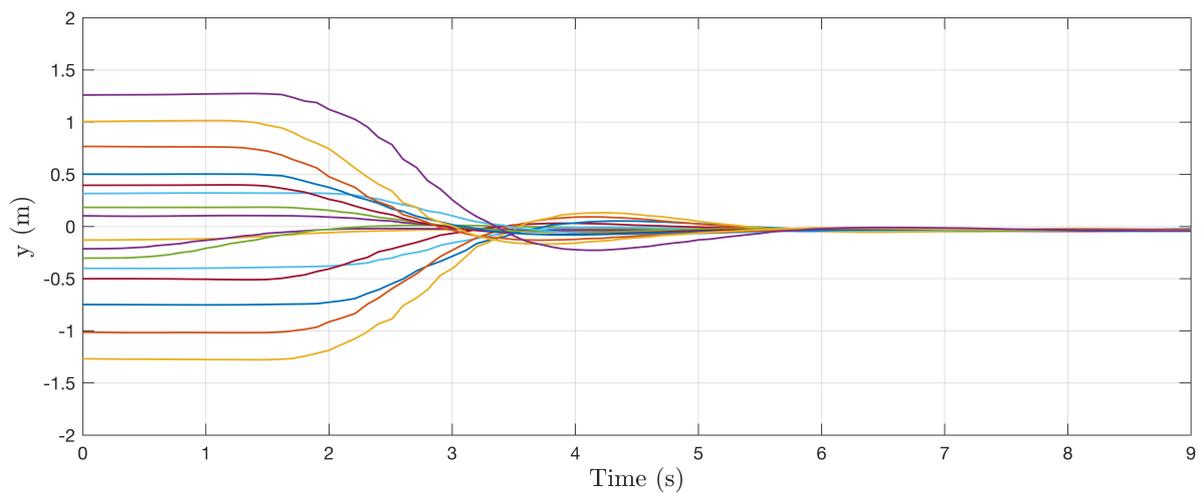


**Figure 7.** (top) Comparison of estimated vertical distance of cylinder centerline to ground truth as system moves up and down in environment. (bottom) State error plotted as a function of time.

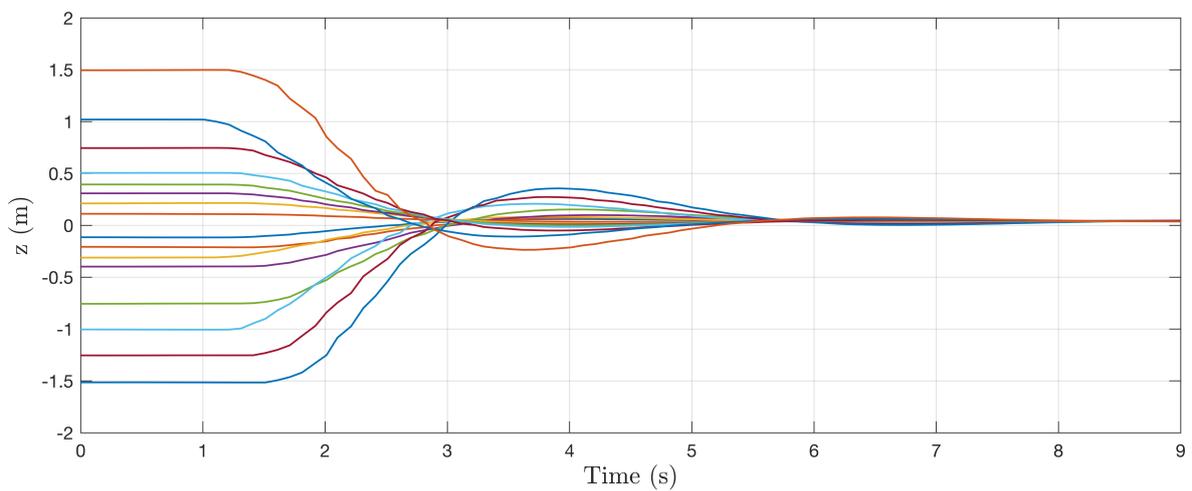


**Figure 8.** (top) Comparison of estimated heading relative to cylinder centerline to ground truth as system yaws back and forth. (bottom) State error plotted as a function of time.

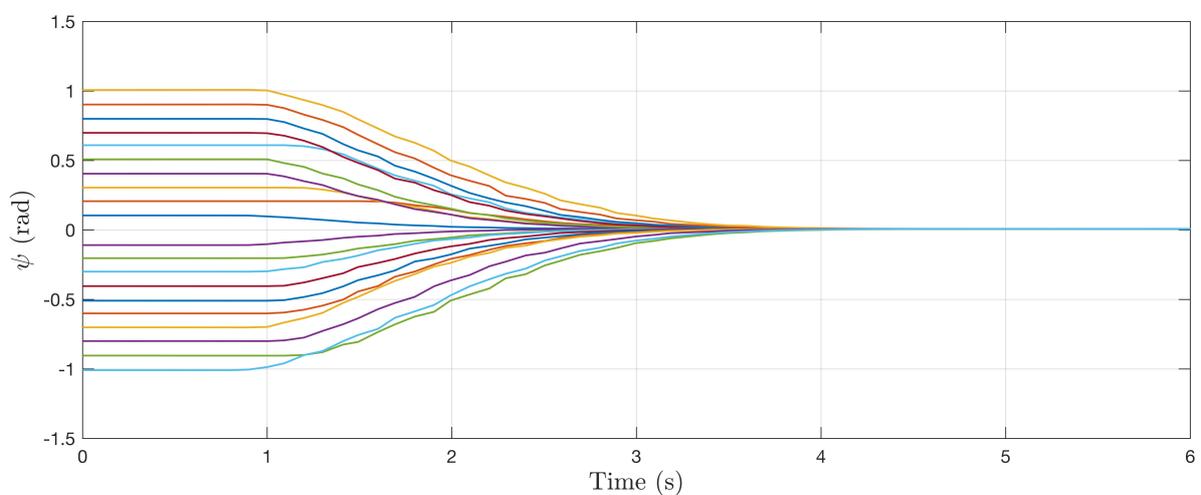
In Figures 9–11, the stability of the controller is demonstrated for different perturbations away from the centerline of the cylinder. Isolated tests were conducted for each state where the system is initialized with some nonzero initial condition. Control commands were generated using the control sensitivity shapes  $F_{u_i}$ , and control gains  $K_v = K_v = K_r = 1$  are used simply to prove stability claims. In each case, the nearness controller drives the system back to the centerline of the cylinder, confirming the stability of the controller. With equal gains, the heading control is seen to have the slowest response, which is to be expected based on the quadrotor model stability derivatives.



**Figure 9.** Collection of system responses to a nonzero initial lateral state. Each line represents a different trajectory with a unique nonzero initial perturbation to lateral position.



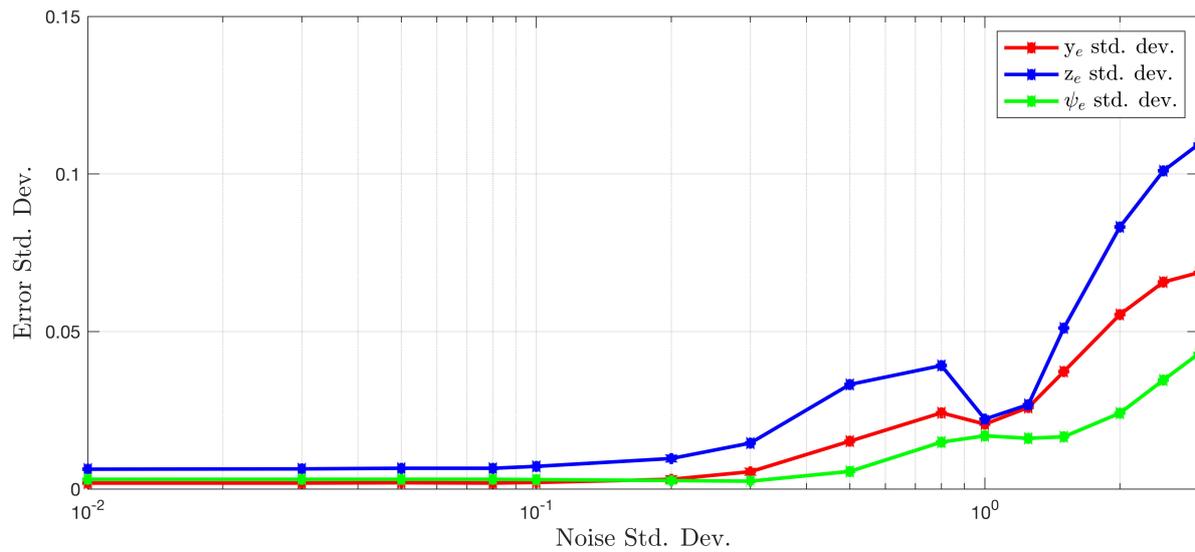
**Figure 10.** Collection of system responses to a nonzero initial vertical state. Each line represents a different trajectory with a unique nonzero initial perturbation to vertical position.



**Figure 11.** Collection of system responses to a nonzero initial heading state. Each line represents a different trajectory with a unique nonzero initial perturbation to heading of system.

### 3.2. Robustness to Noise

To evaluate the algorithm's robustness to additive Gaussian white noise on the depth measurements, the quadrotor platform was piloted through the cylinder environment for 60 s. During the flight, the system was perturbed away from the cylinder centerline along each environment relative state and the sensor data were recorded. Zero-mean Gaussian noise with varying standard deviations was added to the depth measurements offline, and the standard deviation of the error in the state estimates was computed. The results are shown in Figure 12.

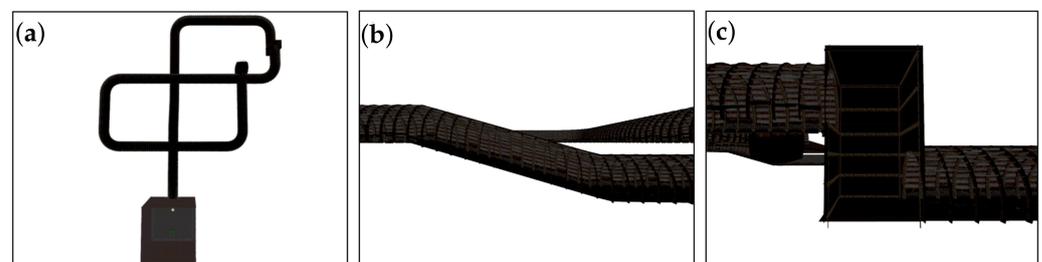


**Figure 12.** Evolution of measurement error distributions as standard deviation of sensor noise increases.

The standard deviation of the state estimates remain extremely small relative to the standard deviation of the added sensor noise. Even with large noise distributions, the algorithm is still able to maintain state estimates that are usable for feedback control.

### 3.3. Performance in Nongeneralized Subterranean Environments

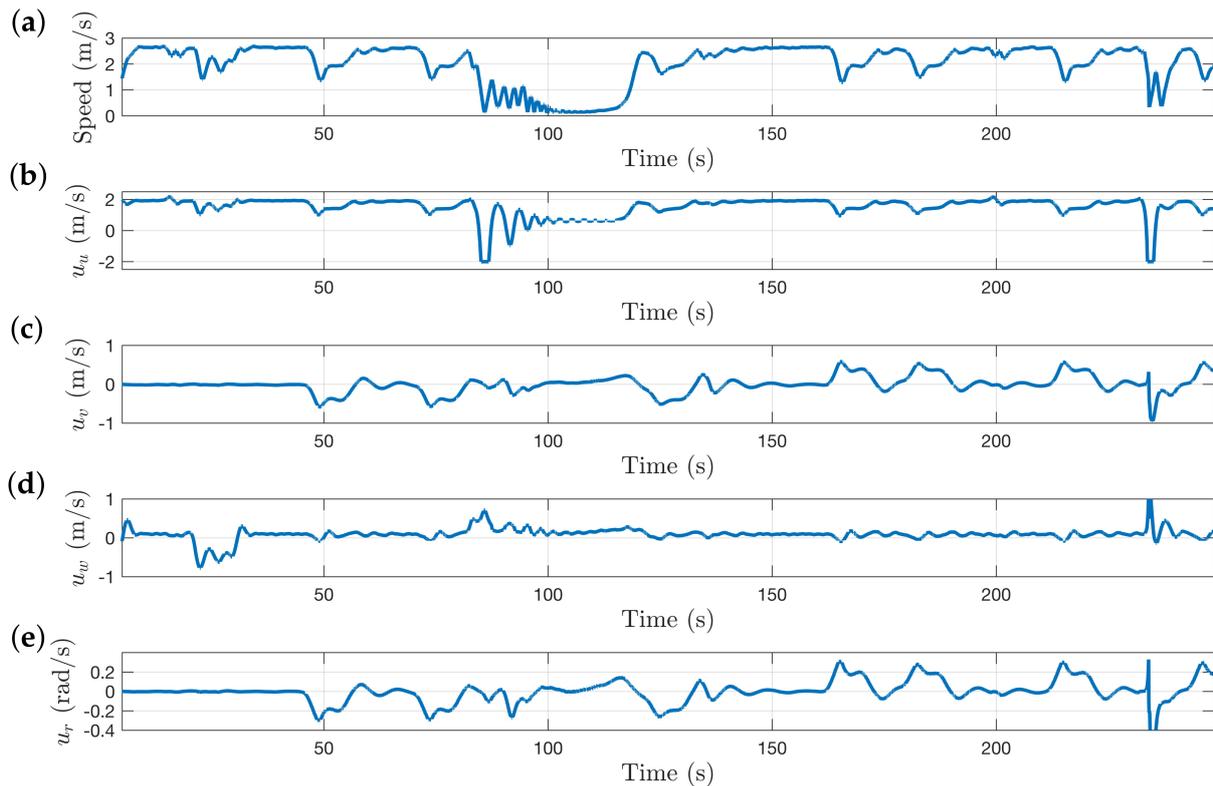
Using different SubT simulation environments, the performance of the algorithm in nongeneralized environments is demonstrated. The control gains listed in the Table in Appendix C were used in both the tunnel and cave tests presented. Figure 13 shows different views of the “Simple Tunnel 03” world, which is composed of sloping ramps, curved bends, vertical shafts, and dead-ends.



**Figure 13.** DARPA SubT simulation world: Simple Tunnel 03, 120 m × 100 m × 10 m. (a) Top-down view of tunnel environment showing banked turns. (b) Side-view of a ramp section where a descent of approximately 7 m occurs over 20 m of forward motion. (c) Side-view of vertical shaft section, which spans about 10 m vertically.

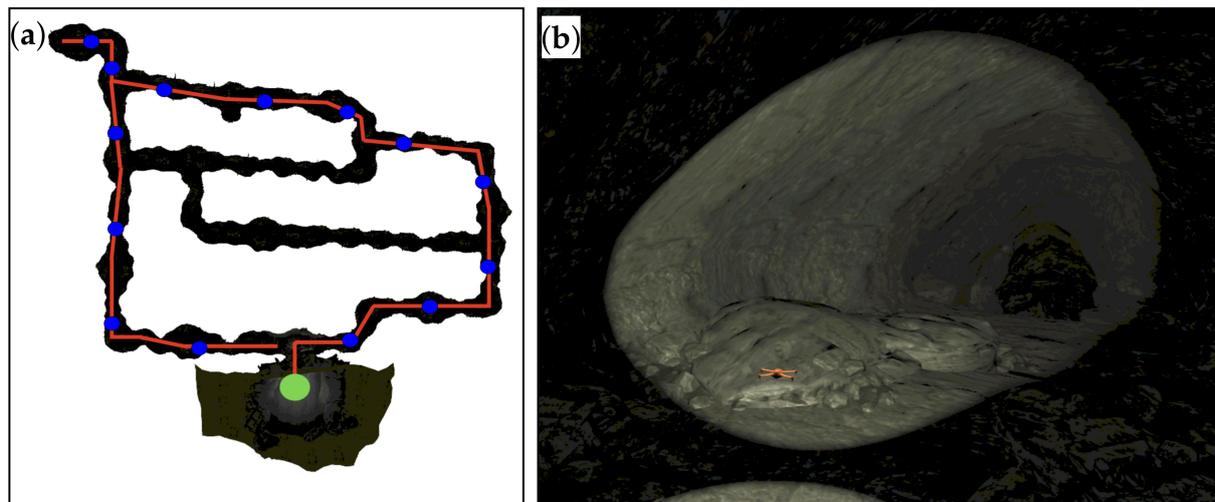
The speed of the system and the control commands are shown in Figure 14. The system is able to quickly navigate through all sections of the course in 4 min, covering 506 m of

tunnel in the process. The average speed of the system was 2.01 m/s. The first section of the environment is the downward sloping ramp shown in Figure 13b. The vertical centering response is demonstrated in the  $u_w$  plot in Figure 14 between the 20 and 35 s mark where the system is commanded to descend. Perturbations in the  $u_v$  commands that correlate with the  $u_r$  commands are indicative of the system navigating a bend in the tunnel. At the 80-s mark, the system approaches the vertical shaft, as seen by the decrease in the forward speed command. Even though the control algorithm is not explicitly designed to navigate vertical shafts, the system is still able to get through the section and continue exploring.

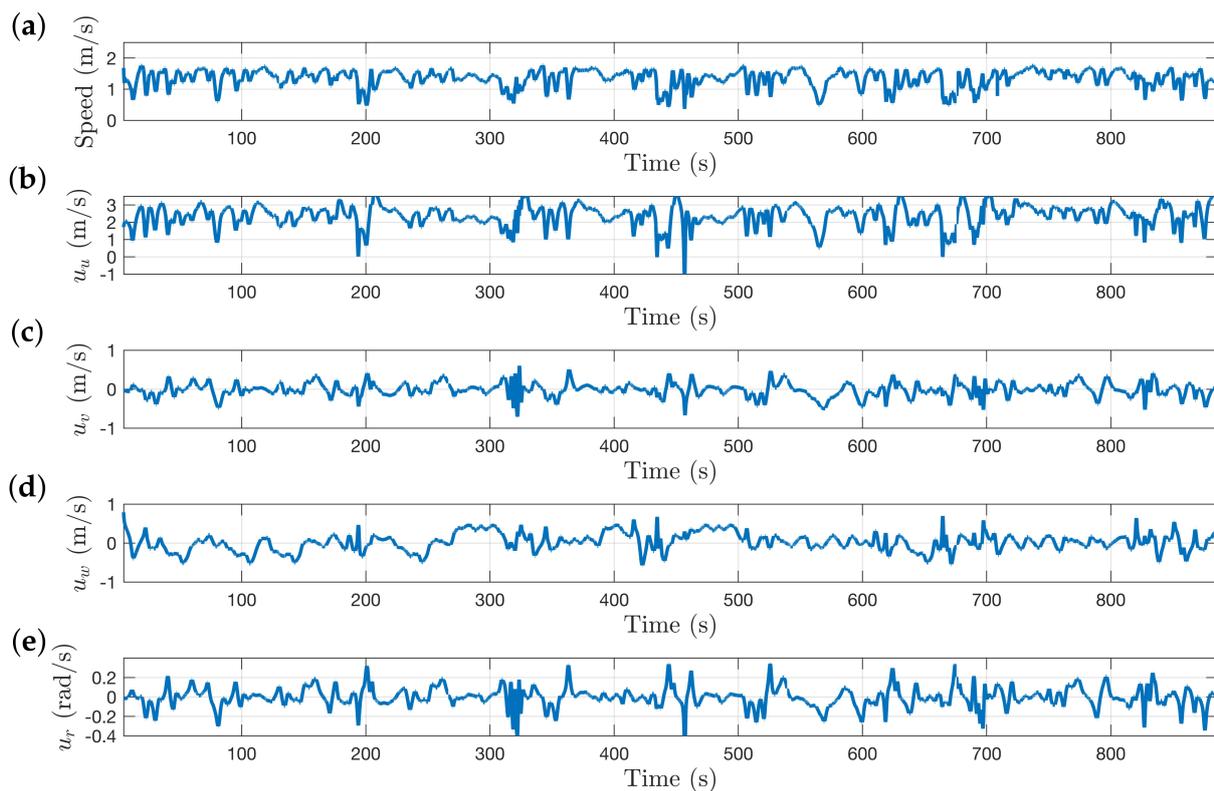


**Figure 14.** Plots showing speed of system and control commands. (a) Speed of system. (b) Forward speed command. (c) Lateral speed command. (d) Vertical speed command. (e) Yaw rate command.

The SubT virtual cave worlds provide even more diverse terrain for systems to navigate, consisting of large caverns connected by tight passageways. Figure 15a shows a top down view of the DARPA SubT sim world “Simple Cave 02”, and a sample of the inside of the environment is shown in Figure 15b. The cave environments deviate massively from the cylinder model used to approximate the local environment, however the algorithm is still able to produce a centering response that keeps the vehicle exploring. The red line in Figure 15 shows the approximate route taken by the system through the cave environment starting at the green dot near the bottom and proceeding counter-clockwise. During this test, the system covered 1.2 km in just under 15 min, with an average speed of 1.35 m/s. The speed of the system and the control commands as it navigates through environment shown in Figure 15 are shown in Figure 16.



**Figure 15.** DARPA SubT simulation world: Simple Cave 02, 230 m  $\times$  170 m  $\times$  75 m (a) Top-down view of cave environment consisting of large unstructured tunnels. The red line in Figure 15 shows the approximate route taken by the system through the cave environment starting at the green dot near the bottom and proceeding counter-clockwise. (b) View of the orange quadrotor system moving through a larger cavern.



**Figure 16.** Plots showing speed of system and control commands as it navigates through environment shown in Figure 15. (a) Speed of the system. (b) Forward speed command. (c) Lateral speed command. (d) Vertical speed command. (e) Yaw rate command.

#### 4. Discussion

In general, the WFI method works well on spatially distributed depth scans, and sensible state sensitivity shapes are presented in Figure 5. The magnitude of the lateral state weighting shape  $F_{\hat{y}}$  is greatest at the sides of the vehicle, which makes sense because these are the measurements that should carry the most weight when estimating  $\hat{y}$ . Similarly,

the vertical state weighting shape  $F_z$  magnitude is largest at the top and bottom poles, and it decreases as they get closer to the  $x - y$  plane. The heading state sensitivity shape  $F_{\hat{\psi}}$  is slightly more interesting. With opposing positive and negative components, this shape becomes sensitive to changes in the heading of the system.

The state estimates from the perfect cylinder environment demonstrated in Section 3.1 were computed using the full state sensitivity shapes. However, different responses can be achieved if subsets of the full shapes are used. For example, if only the front half of the heading state sensitivity shape  $F_{\hat{\psi}}$  is used in the spatial inner-product (i.e., measurements from behind the vehicle are not used), a more reactive steering response is produced. By only processing points that are in front of the vehicle, the system is more responsive to changes in the environment in front of it and can steer accordingly. The results presented in Section 3.3 were achieved using the front half of the  $F_{\hat{\psi}}$ . The same heading state-estimates can be achieved by recomputing the projection coefficient in the  $C^\dagger$  matrix in Equation (6). Other responses, like ground and wall following can be achieved through half shapes as well. With small changes to the feedback controller, a ground-following response can be produced by using only the bottom half of the vertical state sensitivity shape  $F_z$ . Similarly, using only the left- or right-hemisphere of the lateral state sensitivity shape  $F_{\hat{y}}$  can produce a wall-following response.

The algorithm's robustness to additive Gaussian sensor noise is demonstrated in Section 3.2. No appreciable change in the standard deviation of the state estimate errors is excited until relatively large deviations in the noise are added to the measurements. Modern LiDAR depth sensors have noise distributions with standard deviations to the order of millimeters to centimeters, which is on the lower end of the data presented in Figure 12. The main takeaway here is that this algorithm enables the use of lower fidelity (and often cheaper) sensors whose noise levels are unacceptable for other applications.

The algorithm is also robust to occlusions, or discontinuities in the measurements, as long as the occlusions are symmetric. Occlusions in pointclouds could result from using multiple, nonoverlapping depth sensors. In this study, occlusions in the depth scan were present due to the presence of the quadrotor body. The quadrotor model body is detected by the onboard LiDAR sensor in the SubT sim, as it is impossible to place a spherical depth sensing model on a quadrotor and not experience some occlusions. These points are obviously not measurements of the environment, and so they should not be processed by the spatial inner-product. In this case, these measurements can simply be thrown away, as long as the corresponding point is removed from every other hemisphere quadrant. This ensures that the resulting state estimates are not skewed in any one direction.

One of the biggest advantages to this solution is the computational efficiency and simplicity. The entire control algorithm can be condensed down into a single equation, Equation (22), which directly produces controls commands from sensor measurements. Producing a single control command requires  $2N$  floating point operations, where  $N$  is the number of measurements, and so only  $6N$  floating point operations are required to generate a set of control commands. Actual implementation of the algorithm requires some additional data conditioning and processing, however the entire operation takes less than 0.01 s of processing time. This means that scans from equivalent LiDAR sensor models could be processed at rates up to 100 Hz. Faster processing of sensor measurements enables faster reactions to changes in the environment, in which case the limiting factor becomes the vehicle dynamics and control response.

While this algorithm is computationally efficient and enables fast flight, it can be susceptible to local minima in the environment such as corners, split paths, or unique environments that produce trajectory loops. In the case where the vehicle is approaching a symmetric split path or corner, the system will not deviate from its current trajectory because the local environment may already be balanced. In this case, the controller may not generate steering commands to avoid a collision. This situation can be thought of as an unstable equilibrium state. In real environments, these scenarios are rare because the relative environment is rarely perfectly symmetric, and even sensor noise can be enough to

perturb the system off the unstable equilibrium state and back onto a centering trajectory. As mentioned above, trajectory loops can be excited in certain unique environments. For example, if the system were to fly into a large cavern from a small hallway, it may never exit the cavern and get stuck in a continuous loop flying around the cavern.

## 5. Conclusions

Using bio-inspired WFI methods, high-bandwidth sensor data can be efficiently processed to produce a 3D centering response in subterranean environments. The derivation of the sensitivity functions presented in Section 2.2.2 shows how different weighting functions can be used to extract environment relative information. With proper selection of the feedback controller gains, the algorithm is demonstrated to be stable for a range of perturbations from the equilibrium state in the cylinder environment. By integrating across thousands of sensor measurements, the algorithm is robust to additive white noise and occlusions in the depth scans. The 3D centering response is robust to large deviations away from the generalized local environment presented in Section 2.2.1, as demonstrated in the flight tests presented in Section 3.3.

Work on this framework can continue in several different directions. In Section 2.2.1, the local environment was approximated as an infinite cylinder with some known radius  $r$ . In this framework, computing controller gains requires some initial value for the expected radius of the environment. If the actual radius of the local environment is significantly larger than the expected radius, the system performance will be sluggish as the control commands are effectively scaled down. Conversely, if the actual radius is much smaller than the expected radius, the control commands could grow too large and make the system unstable. Using principals of robust control theory, a dynamic controller can be designed which provides performance and stability guarantees for a range of perturbations to the expected radius.

As noted in Section 3.3, the system is able to navigate a vertical shaft using this 3D centering approach; however, it required the vehicle to slow to a near stop. This subterranean exploration framework could be expanded by determining new sensitivity functions for vertically oriented infinite cylinders. By feeding back on a different set of pooled projections, forward- and lateral-centering could be achieved while the system ascends or descends the vertical shaft. This requires additional logic for determining when the system should switch between the different centering modes. Deeper analysis of different projection values may provide insight into signals that could be used to facilitate the mode switching.

This work focuses on a spherical LiDAR sensor model that produces the same distribution of points each scan. In other words, the viewing angles of each scan point are constant and known. Novel depth sensing modalities, such as millimeter wave radar, also produce depth scans, however the incoming, unfiltered points are often spurious and have no known spatial distribution. Future work can be done to apply this algorithm to sensors with unknown spatial distributions of their depth scan points.

**Author Contributions:** Conceptualization, M.T.O. and J.S.H.; methodology, M.T.O. and J.S.H.; software, M.T.O.; validation, M.T.O.; formal analysis, M.T.O.; investigation, M.T.O. and J.S.H.; resources, M.T.O. and J.S.H.; data curation, M.T.O.; writing—original draft preparation, M.T.O.; writing—review and editing, M.T.O. and J.S.H.; visualization, M.T.O.; supervision, J.S.H.; project administration, J.S.H.; funding acquisition, J.S.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported through the DARPA Subterranean Challenge, cooperative agreement number HR0011-18-2-0043.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

### Abbreviations

The following abbreviations are used in this manuscript:

MDPI	Multidisciplinary Digital Publishing Institute
WFI	Wide-Field Integration
DARPA	Defense Advanced Research Projects Agency
SubT	Subterranean (in reference to the DARPA Subterranean Challenge)
LiDAR	Light Detection and Ranging
sUAS	small Unmanned Aerial Systems
UGV	Unmanned Ground Vehicles
SWaP	Size, Weight, and Power
LPTC	Lobula Plate Tangential Cell
ROS	Robot Operating System
RRT	Rapidly-Exploring Random Trees

### Appendix A. Intersection of a Line and a Cylinder Surface

Surface of a cylinder as a function of of the viewing angles  $\theta_v$  and  $\phi_v$ :

$$y_c^2 + z_c^2 = r^2 \quad (\text{A1})$$

Components of a line segment that intersect with the cylinder as a function of the viewing angles  $\theta_v$  and  $\phi_v$ :

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = [[M_3(\phi_v)] [M_2(\theta_v)]]^T \begin{bmatrix} 0 \\ 0 \\ d \end{bmatrix} = d \begin{bmatrix} \cos(\phi_v) \sin(\theta_v) \\ \sin(\phi_v) \sin(\theta_v) \\ \cos(\theta_v) \end{bmatrix} \quad (\text{A2})$$

By plugging in the  $y_c$  and  $z_c$  components of Equation (A2) into Equation (A1):

$$d^2 [\sin^2(\phi_v) \sin^2(\theta_v) + \cos^2(\theta_v)] = r^2 \quad (\text{A3})$$

Solving Equation (A3) for  $d$ :

$$d(\theta_v, \phi_v) = r [\sin^2(\phi_v) \sin^2(\theta_v) + \cos^2(\theta_v)]^{-\frac{1}{2}} \quad (\text{A4})$$

This is the unperturbed case, so now we want to add in the lateral ( $y$ ), vertical ( $z$ ), and heading ( $\psi$ ) states. The new equations for the cylinder and line that encode these values is:

$$(y_c - y)^2 + (z_c - z)^2 = r^2 \quad (\text{A5})$$

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = d \begin{bmatrix} \cos(\phi_v + \psi) \sin(\theta_v) \\ \sin(\phi_v + \psi) \sin(\theta_v) \\ \cos(\theta_v) \end{bmatrix} \quad (\text{A6})$$

Now, if we combine Equations (A5) and (A6) and solve for  $d$  we get:

$$d(\theta_v, \phi_v, y, z, \psi) = \frac{b \pm \sqrt{b^2 - ac}}{a} \quad (\text{A7})$$

$$a = \sin^2(\phi_v + \psi) \sin^2(\theta_v) + \cos^2(\theta_v)$$

$$b = y \sin(\phi_v + \psi) \sin(\theta_v) + z \cos(\theta_v)$$

$$c = y^2 + z^2 - r^2$$

Equation (A7) is simply inverted to get the final closed form solution for nearness presented in Equation (1).

## Appendix B. Quadrotor Model Parameters

**Table A1.** Quadrotor model parameters used for stability analysis and the development of initial controller gains.

Quadrotor Model Parameters		
Parameter	Value	Units
$Y_v$	−0.3022	1/s
$Y_{\delta_{lat}}$	0.0172212	m/(s <sup>2</sup> %)
$Z_w$	−0.1734	1/s
$Z_{\delta_{vert}}$	−14.95504248	m/(s <sup>2</sup> %)
$N_r$	−0.5617	1/s
$N_{\delta_{Head}}$	6.0308	rad/(s <sup>2</sup> %)

## Appendix C. Controller Gains

**Table A2.** Controller gain values used to collect the presented datasets.

Controller Gains	
Gain	Value
$K_v$	1.25
$K_w$	1.5
$K_r$	1.5
$K_{u_v}$	0.4
$K_{u_w}$	0.25
$K_{u_r}$	0.4
$K_{u_{fwd}}$	2.0

## References

- Otsu, K.; Tepsuporn, S.; Thakker, R.; Vaquero, T.S.; Edlund, J.A.; Walsh, W.; Miles, G.; Heywood, T.; Wolf, M.T.; Agha-Mohammadi, A. Supervised Autonomy for Communication-degraded Subterranean Exploration by a Robot Team. In Proceedings of the 2020 IEEE Aerospace Conference, Big Sky, MT, USA, 7–14 March 2020; pp. 1–9. [\[CrossRef\]](#)
- Ebadi, K.; Chang, Y.; Palieri, M.; Stephens, A.; Hatteland, A.; Heiden, E.; Thakur, A.; Funabiki, N.; Morrell, B.; Wood, S.; et al. LAMP: Large-Scale Autonomous Mapping and Positioning for Exploration of Perceptually-Degraded Subterranean Environments. In Proceedings of the IEEE International Conference on Robotics and Automation, Paris, France, 31 May–31 August 2020; pp. 80–86. [\[CrossRef\]](#)
- Miller, I.D.; Cohen, A.; Kulkarni, A.; Laney, J.; Taylor, C.J.; Kumar, V.; Cladera, F.; Cowley, A.; Shivakumar, S.S.; Lee, E.S.; et al. Mine Tunnel Exploration Using Multiple Quadrupedal Robots. *IEEE Robot. Autom. Lett.* **2020**, *5*, 2840–2847. [\[CrossRef\]](#)
- Huang, Y.; Lu, C.; Chen, K.; Ser, P.; Huang, J.; Shen, Y.; Chen, P.; Chang, P.; Lee, S.; Wang, H. Duckiefloat: A Collision-Tolerant Resource-Constrained Blimp for Long-Term Autonomy in Subterranean Environments. *arXiv* **2019**, arXiv:1910.14275.
- Rouček, T.; Pecka, M.; Čížek, P.; Petříček, T.; Bayer, J.; Šalanský, V.; Heřt, D.; Petrlík, M.; Báča, T.; Spurný, V.; et al. DARPA Subterranean Challenge: Multi-robotic Exploration of Underground Environments. In *Modelling and Simulation for Autonomous Systems*; Mazal, J., Fagiolini, A., Vasik, P., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 274–290.
- Khattak, S.; Nguyen, H.; Mascarich, F.; Dang, T.; Alexis, K. Complementary Multi-Modal Sensor Fusion for Resilient Robot Pose Estimation in Subterranean Environments. In Proceedings of the 2020 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 1–4 September 2020; pp. 1024–1029. [\[CrossRef\]](#)
- Santamaria-Navarro, A.; Thakker, R.; Fan, D.D.; Morrell, B.; Agha-mohammadi, A. Towards Resilient Autonomous Navigation of Drones. *arXiv* **2020**. arXiv:2008.09679.
- Ohradzansky, M.T.; Mills, A.B.; Rush, E.R.; Riley, D.G.; Frew, E.W.; Humbert, J.S. Reactive Control and Metric-Topological Planning for Exploration. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4073–4079. [\[CrossRef\]](#)
- Dang, T.; Khattak, S.; Mascarich, F.; Alexis, K. Explore locally, plan globally: A path planning framework for autonomous robotic exploration in subterranean environments. In Proceedings of the 2019 19th International Conference on Advanced Robotics (ICAR), Belo Horizonte, Brazil, 2–6 December 2019; pp. 9–16.

10. Dang, T.; Tranzatto, M.; Khattak, S.; Mascarich, F.; Alexis, K.; Hutter, M. Graph-based subterranean exploration path planning using aerial and legged robots. *J. Field Robot.* **2020**, *37*, 1363–1388. [[CrossRef](#)]
11. Lajoie, P.Y.; Ramtoula, B.; Chang, Y.; Carlone, L.; Beltrame, G. DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams. *IEEE Robot. Autom. Lett.* **2020**, *5*, 1656–1663. [[CrossRef](#)]
12. Papachristos, C.; Khattak, S.; Mascarich, F.; Dang, T.; Alexis, K. Autonomous Aerial Robotic Exploration of Subterranean Environments relying on Morphology-aware Path Planning. In Proceedings of the 2019 International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 11–14 June 2019; pp. 299–305. [[CrossRef](#)]
13. Montgomery, J.C.; Coombs, S.; Baker, C.F. The Mechanosensory Lateral Line System of the Hypogean form of *Astyanax Fasciatus*. *Environ. Biol. Fishes* **2001**, *62*, 87–96. [[CrossRef](#)]
14. Montgomery, J.; Coombs, S.; Halstead, M. Biology of the mechanosensory lateral line in fishes. *Rev. Fish Biol. Fish.* **2005**, *5*, 399–416. [[CrossRef](#)]
15. Montgomery, J.C.; Baker, C.F.; Carton, A.G. The lateral line can mediate rheotaxis in fish. *Nature* **1997**, *389*, 960–963. [[CrossRef](#)]
16. Suli, A.; Watson, G.M.; Rubel, E.W.; Raible, D.W. Rheotaxis in Larval Zebrafish Is Mediated by Lateral Line Mechanosensory Hair Cells. *PLoS ONE* **2012**, *7*, e29727. [[CrossRef](#)]
17. Partridge, B.L.; Pitcher, T.J. The sensory basis of fish schools: Relative roles of lateral line and vision. *J. Comp. Physiol.* **1990**, *135*, 315–325. [[CrossRef](#)]
18. Montgomery, J.; Hamilton, A. Sensory contributions to nocturnal prey capture in the dwarf scorpion fish (*Scorpaena papillosus*). *Mar. Freshw. Behav. Physiol.* **1997**, *30*, 209–223. [[CrossRef](#)]
19. Hoekstra, D.; Janssen, J. Non-visual feeding behavior of the mottled sculpin, *Cottus bairdi*, in Lake Michigan. *Environ. Biol. Fishes* **1985**, *12*, 111–117. [[CrossRef](#)]
20. Dumble, K.D.; Faddy, J.M.; Humbert, J.S. Electrolocation-based underwater obstacle avoidance using wide-field integration methods. *Bioinspir. Biomim.* **2014**, *9*, 016012. [[CrossRef](#)] [[PubMed](#)]
21. Ranganathan, B.; Dumble, K.; Faddy, J.; Humbert, J.S. Underwater navigation behaviors using Wide-Field Integration methods. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 4147–4152. [[CrossRef](#)]
22. Barth, F.G. Spider mechanoreceptors. *Curr. Opin. Neurobiol.* **2004**, *14*, 415–422. [[CrossRef](#)]
23. Bleckmann, H. Prey Identification and Prey Localization in Surface-feeding Fish and Fishing Spiders. In *Sensory Biology of Aquatic Animals*; Atema, J., Fay, R.R., Popper, A.N., Tavolga, W.N., Eds.; Springer: New York, NY, USA, 1988; pp. 619–641.
24. Mhatre, N.; Sivalingham, S.; Mason, A.C. Posture controls mechanical tuning in the black widow spider mechanosensory system. *bioRxiv* **2018**. [[CrossRef](#)]
25. Barth, F. How To Catch the Wind: Spider Hairs Specialized for Sensing the Movement of Air. *Naturwissenschaften* **2000**, *87*, 52–58. [[CrossRef](#)]
26. Guarino, R.; Greco, G.; Mazzolai, B.; Pugno, N. Fluid-structure interaction study of spider’s hair flow-sensing system. *Mater. Today Proc.* **2019**, *7*, 418–425. [[CrossRef](#)]
27. Kant, R.; Humphrey, J.A.C. Response of cricket and spider motion-sensing hairs to airflow pulsations. *J. R. Soc. Interface* **2009**, *6*, 1047–1064. [[CrossRef](#)]
28. Frye, M.A.; Dickinson, M.H. Fly flight: A model for the neural control of complex behavior. *Neuron* **2001**, *32*, 385–388. [[CrossRef](#)]
29. Egelhaaf, M.; Kern, R.; Krapp, H.G.; Kretzberg, J.; Kurtz, R.; Warzecha, A.K. Neural encoding of behaviourally relevant visual-motion information in the fly. *Trends Neurosci.* **2002**, *25*, 96–102. [[CrossRef](#)]
30. Borst, A.; Juergen, H. Neural networks in the cockpit of the fly. *J. Comp. Physiol. Neuroethol. Sens. Neural Behav. Physiol.* **2002**, *188*, 419–437. [[CrossRef](#)]
31. Srinivasan, M.V.; Zhang, S. Visual Motor Computations in Insects. *Annu. Rev. Neurosci.* **2004**, *27*, 679–696. [[CrossRef](#)]
32. Humbert, J.S.; Conroy, J.K.; Neely, C.W.; Barrows, G. Wide-Field Integration Methods for Visuomotor Control. In *FLying Insects Robot*; Springer: Berlin/Heidelberg, Germany, 2009. [[CrossRef](#)]
33. Humbert, J.S.; Hyslop, A.M. Bioinspired Visuomotor Convergence. *IEEE Trans. Robot.* **2010**, *26*, 121–130. [[CrossRef](#)]
34. Keshavan, J.; Gremillion, G.; Escobar-Alvarez, H.; Humbert, J.S. A mu analysis-based, controller-synthesis framework for robust bioinspired visual navigation in less-structured environments. *Bioinspir. Biomim.* **2014**, *9*, 025011. [[CrossRef](#)] [[PubMed](#)]
35. Srinivasan, M.; Chahl, J.S.; Weber, K.; Venkatesh, S.; Nagle, M.G.; Zhang, S. Robot navigation inspired by principles of insect vision. *Robot. Auton. Syst.* **1998**, *26*, 203–216. [[CrossRef](#)]
36. Serres, J.H.; Ruffier, F. *Optic Flow-Based Robotics*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2016; pp. 1–14. [[CrossRef](#)]
37. Serres, J.H.; Ruffier, F. Optic flow based collision-free strategies: From insects to robots. *Arthropod Struct. Dev.* **2017**, *46*, 703–717. [[CrossRef](#)] [[PubMed](#)]
38. Raharijaona, T.; Serres, J.; Vanhoutte, E.; Ruffier, F. Toward an insect-inspired event-based autopilot combining both visual and control events. In Proceedings of the 2017 3rd International Conference on Event-Based Control, Communication and Signal Processing (EBCCSP), Funchal, Portugal, 24–26 May 2017; pp. 1–7. [[CrossRef](#)]
39. Vanhoutte, E.; Ruffier, F.; Serres, J. A quasi-panoramic bio-inspired eye for flying parallel to walls. In Proceedings of the 2017 IEEE SENSORS, Glasgow, UK, 29 October–1 November 2017; pp. 1–3. [[CrossRef](#)]

40. Lecoœur, J.; Baird, E.; Floreano, D. Spatial Encoding of Translational Optic Flow in Planar Scenes by Elementary Motion Detector Arrays. *Sci. Rep.* **2018**, *8*, 5821. [[CrossRef](#)]
41. Serres, J.R.; Morice, A.H.; Blary, C.; Montagne, G.; Ruffier, F. An innovative optical context to make honeybees crash repeatedly. *bioRxiv* **2021**. [[CrossRef](#)]
42. Serres, J.; Evans, T.; Akesson, S.; Duriez, O.; Shamoun-Baranes, J.; Ruffier, F.; Hedenström, A. Optic flow cues help explain altitude control over sea in freely flying gulls. *J. R. Soc. Interface* **2019**, *16*, 20190486. [[CrossRef](#)]
43. Ohradzansky, M.; Alvarez, H.E.; Keshavan, J.; Ranganathan, B.; Humbert, J. Autonomous Bio-Inspired Small-Object Detection and Avoidance. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1–9.
44. Alvarez, H.E.; Ohradzansky, M.T.; Keshavan, J.; Ranganathan, B.; Humbert, J.S. Bio-Inspired Approaches for Small-Object Detection and Avoidance. *IEEE Trans. Robot.* **2019**, *35*, 1220–1232. [[CrossRef](#)]
45. Khatib, O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *Int. J. Robot. Res.* **1986**, *5*, 90–98. [[CrossRef](#)]
46. Montano, L.; Asensio, J.R. Real-Time Robot Navigation in Unstructured Environments Using a 3D Laser Rangefinder. In Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems, Innovative Robotics for Real-World Applications, IROS '97, Grenoble, France, 11 September 1997.
47. Rimon, E.; Koditschek, D. Exact Robot Navigation Using Artificial Potential Functions. *IEEE Trans. Robot. Autom.* **1992**, *8*, 501–518. [[CrossRef](#)]
48. Fan, X.; Guo, Y.; Liu, H.; Wei, B.; Lyu, W. Improved Artificial Potential Field Method Applied for AUV Path Planning. *Math. Probl. Eng.* **2020**, *2020*, 6523158. [[CrossRef](#)]
49. Minguez, J.; Montano, L. Nearness Diagram Navigation (ND): A New Real Time Collision Avoidance Approach. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Takamatsu, Japan, 31 October–5 November 2000.
50. Durham, J.W.; Bullo, F. Smooth Nearness-Diagram Navigation. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008.
51. Mujahad, M.; Fischer, D.; Mertsching, B.; Jaddu, H. Closest Gap Based (CG) Reactive Obstacle Avoidance Navigation for Highly Cluttered Environments. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010.
52. Mujahad, M.; Fischer, D.; Mertsching, B. Tangential Gap Flow (TGF) navigation: A new reactive obstacle avoidance approach for highly cluttered environments. *Robot. Auton. Syst.* **2016**, *84*, 15–30. [[CrossRef](#)]
53. Steiner, J.; He, X.; Bourne, J.; Leang, K.K. Open-sector rapid-reactive collision avoidance: Application in aerial robot navigation through outdoor unstructured environments. *Robot. Auton. Syst.* **2019**, *112*, 211–220. [[CrossRef](#)]
54. Lu, L.; Sampedro, C.; Rodriguez-Vazquez, J.; Campoy, P. Laser-based Collision Avoidance and Reactive Navigation using RRT\* and Signed Distance Field for Multirotor UAVs. In Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 11–14 June 2019.
55. Saetti, U.; Berger, T.; Horn, J.; Lagoa, C.; Lakhmani, S. Design of Dynamic Inversion and Explicit Model Following Control Laws for Quadrotor Inner and Outer Loops. *J. Am. Helicopter Soc.* **2020**, *65*, 1–6.