IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2021.Doi Number

Multiple UAV Adaptive Navigation for Three-Dimensional Scalar Fields

Robert K. Lee, Student Member IEEE, Christopher A. Kitts, Senior Member IEEE, Michael A. Neumann, Member IEEE, and Robert T. McDonald, Member IEEE

Santa Clara University, Santa Clara, CA 95053, USA

Corresponding author: Robert K. Lee (e-mail: RLee@scu.edu).

ABSTRACT Adaptive Navigation (AN) control strategies allow an agent to autonomously alter its trajectory based on realtime measurements of the environment. Compared to conventional navigation methods, these techniques can reduce required time and energy to explore scalar characteristics of unknown and dynamic regions of interest (e.g., temperature, concentration level). Multiple Uncrewed Aerial Vehicle (UAV) approaches to AN can improve performance by exploiting synchronized spatially-dispersed measurements to generate realtime information regarding the structure of the local scalar field, which is then used to inform navigation decisions. This article presents initial results of a comprehensive program to develop, verify, and experimentally implement mission-level AN capabilities in three-dimensional (3D) space using our unique multilayer control architecture for groups of vehicles. Using our flexible formation control system, we build upon our prior 2D AN work and provide new contributions to 3D scalar field AN by a) demonstrating a wide range of 3D AN capabilities using a unified, multilayer control architecture, b) extending multivehicle 2D AN control primitives to navigation in 3D scalar fields, and c) introducing state-based sequencing of these primitive AN functions to execute 3D mission-level capabilities such as isosurface mapping and plume following. We verify functionality using high-fidelity simulations of multicopter drone clusters, accounting for vehicle dynamics, outdoor wind gust disturbances, position sensor inaccuracy, and scalar field sensor noise. This paper presents the multilayer architecture for multivehicle formation control, the 3D AN control primitives, the sequencing approaches for specific mission-level capabilities, and simulation results that demonstrate these functions.

INDEX TERMS adaptive navigation, autonomous navigation, cluster-space control, contour mapping, cooperative control, extrema seeking, formation control, level set mapping, modifiable formation, plume following, plume tracing, source seeking.

SUMMARY OF SYMBOLS

Symbol	Description	Symbol	Description
n	Number of vehicles in cluster	P	Number of propellers (per vehicle)
m	Degrees of freedom per vehicle	k	Propeller correction coefficient
${}^{G}x_i \quad {}^{G}y_i \quad {}^{G}z_i \quad {}^{G}\psi_i$	ith vehicle coordinates and heading	ρ	Air density
$D_x \quad D_y \quad D_z$	Vehicle level drag components	A	Propeller disc area (per propeller)
$V_x \quad V_y \quad V_z$	Wind disturbance velocity components	τ	Propeller axial thrust (hovering)
$K_u \quad K_v$	Vehicle gain matrices	$\vec{R}$	Vehicle specific variables vector
M	Vehicle mass	$\vec{C}$	Cluster specific variables vector
m_f	Air mass flow rate	J	Jacobian matrix
σ	Sensor noise standard deviation	$\vec{D}_{diff}$	Differential signals vector
$\vec{g}_{grad}$	Estimated 3D gradient vector	K_p	Proportional gain matrix
$g_x \quad g_y \quad g_z$	Estimated 3D gradient components	$Q \quad Q'$	Complementary selector matrices
$\vec{S}$	Measured scalar field values vector	$\hat{X}_c \quad \hat{Y}_c \quad \hat{Z}_c$	Cluster frame unit vectors
s_i	ith measured scalar field value	Ri	ith vehicle reference frame
S	Cluster translational speed	${}^{G}\hat{n}$	Navigation reference vector
R	Cluster rotational speed	n_{des}	Mapping plane and ${}^{G}\hat{n}$ intersection
$d_{extrema}$	Extrema following direction	$n_{cluster}$	Cluster position (mean of vehicles)
d_{orbit}	Isosurface mapping direction	$n_{threshold}$	Cluster to mapping plane distance
K_{surf}	Isosurface mapping gain	Δn	Isosurface mapping planes delta

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

SUMMARY OF SYMBOLS (CONTINUED)

Symbol	Description	Symbol	Description
${}^{G}X_B \quad {}^{G}Y_B \quad {}^{G}Z_B$	Cluster coordinates for 4-robot cluster	$\vec{P}_B$	Cluster point B position vector
${}^{G}X_C \quad {}^{G}Y_C \quad {}^{G}Z_C$	Cluster coordinates for 9-robot cluster	P_{B_proj}	$\vec{P}_B$ projected onto ${}^{G}\hat{n}$
${}^{G}\Phi_C \quad {}^{G}\Theta_C \quad {}^{G}\Psi_C$	Cluster roll/pitch/yaw	s_{des}	Desired scalar value of isosurface
${}^{G}x_n \quad {}^{G}y_n \quad {}^{G}z_n$	Vehicle *n* coordinates	$s_{cluster}$	Mean of scalar field measurements
${}^{G}\psi_n$	Vehicle *n* heading w.r.t global frame	$s_{threshold}$	Scalar field threshold for isosurface
${}^{C}\psi_n$	Vehicle *n* heading w.r.t cluster frame	$s_{prom\ threshold}$	Plume following start threshold
$L_{12} \quad L_{13} \quad L_{B4}$	4-robot cluster formation lengths	$s_{term\ threshold}$	Plume following end threshold
$L_{C2}\ L_{23}\ L_{C4}\ L_{45}\ L_{C6}\ L_{67}\ L_{C8}\ L_{89}$	9-robot cluster formation lengths	$\delta_{threshold}$	Distance between map start / end
$\alpha \quad \beta \quad \xi$	4-robot cluster shape angles	$\rho_{threshold}$	Angle between map start / end
$\alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \alpha_7\ \alpha_9$	9-robot cluster shape angles (α)	$\gamma_{threshold}$	Angle between $\vec{g}_{grad}$ and ${}^{G}\hat{n}$
$\beta_3\ \beta_4\ \beta_5\ \beta_6\ \beta_7\ \beta_9\ \zeta$	9-robot cluster shape angles (β)	$\Delta_{threshold}$	Differentials magnitude threshold
ζ	9-robot cluster shape angle (ζ)	$\Psi_{C\ threshold}$	Cluster alignment threshold
K_n	Mapping constraint plane gain	$K_{rot\ hold}$	Cluster rotational gain

I. INTRODUCTION

Conventional navigation techniques involve user-provided waypoints or a predefined trajectory for an agent to follow over time. In contrast, Adaptive Navigation (AN) methods[1] allow an agent to autonomously alter its trajectory or direction based on realtime measurements of the environment.

The most basic form of adaptive navigation requires specifying the destination explicitly while allowing the control system to alter the vehicle's path, such as rerouting to avoid obstacles or traffic congestion. More advanced AN systems do not require user specified destinations in any form such as enemy evasion while being pursued or autonomously determining the optimal escape route. A third and more sophisticated form of AN, enables one or more agents to navigate to or along features of interest within a continuous scalar field without prior knowledge of the region, such as finding a local maximum of temperature, light intensity, or pollution concentration level.

Scalar field features represent significant phenomena in a wide range of applications such as environmental monitoring/characterization, disaster response, and exploration. Table I summarizes a number of these features and their applicability in the environmental monitoring domain. A detailed discussion of scalar field features is provided in [1]. For any application, AN can dramatically reduce the time and/or cost of locating and characterizing these features compared to conventional methods requiring exhaustive mapping of an entire region, particularly when the field is time-varying.

TABLE I
APPLICABILITY OF SCALAR FIELD FEATURES

Feature	Physical Application Feature
Maximum	Hot spot, source of pollution, gas or radio emission
Minimum	Quiet/safe spot, anoxic point
Contour	Defining hazard regions, service levels
Ridge	Path to impact area from a source, path of optimal service when leaving a source
Trench	Path of maximum attenuation from a source, path of minimum exposure to a source
Saddle Point	Low energy path between minima, low exposure path between sources
Front	Atmospheric phenomena affecting weather, marine phenomena fueling bio-productivity

[1] Although the term "adaptive navigation" is used routinely in the literature and by the authors, the scope of work discussed in this article focuses primarily on guidance laws that allow one or more robots to move to or along features of interest in a scalar field.

I.A. ADAPTIVE NAVIGATION IN 2D SCALAR FIELDS

Two-dimensional (2D) scalar field navigation techniques have been proposed using a wide variety of control strategies with single and multiple planar vehicles.

Extrema Seeking. The elementary task to autonomously locate a local source (i.e., maxima seeking) in a planar scalar field, without prior knowledge of its location, has attracted an immense amount of interest. Prior research using a single vehicle has successfully simulated this task by assuming simplified vehicle kinematics or point mass models, with control strategies based on the gradient [2], directional derivative [3], sliding mode [4][5], stochastic approximations [6], or hybrid approaches [65][66].

Experimentally, [7] demonstrated scalar field AN with a single planar vehicle in a small indoor testbed (~8 m^2) controlled with a line minimization-based algorithm while [8] used a bioinspired fuzzy inference algorithm to control a wheeled robot equipped with a gas sensor. In [9], a single AirRobot AR-100-B UAV held at fixed altitude, was used to locate a gas source outdoors (~320 m^2) using three different bioinspired spatial dithering algorithms.

In order to improve performance by taking distributed synchronized measurements of the field while also improving resiliency under failure scenarios, multiple vehicle systems have been proposed. The use of multiple, spatially distributed vehicles overcomes the primary drawback to many single vehicle AN control approaches, which is the need to execute time-consuming spatial dithering maneuvers in order to obtain local field information upon which navigation decisions are made [9][10][11][12]. However, these multivehicle systems require increased system sophistication due to the additional complexity associated with group control. Previous research with simulations, assuming negligible

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

agent dynamics and external disturbances, successfully demonstrated this task. Some used gradient-based control strategies for agents in leader-follower roles [13][14], fixed formations [15][71] or bioinspired swarms [17][18][19]. Another team [16] simulated this task by proposing a loosely controlled formation using a probabilistic gradient-estimation control strategy, while [11] dithered vehicle motion with amplitudes proportional to the square of the source location estimation error. [67] performed source seeking with a multivehicle swarm controlled with a Quantum-Leading-Following-Based optimization algorithm which directed robot motion while allowing for obstacle avoidance. Research with simulations using detailed dynamic models successfully demonstrated 2D source seeking [70] using a leader-follower gradient-based control strategy.

Experimentally, source seeking in 2D with multiple planar vehicles has been performed in controlled indoor environments with a limited workspace. In [20], three EPFL E-puck wheeled robots in a fixed formation using a gradient based control strategy, demonstrated light source seeking within a small-scale testbed (~4 m^2). A bioinspired swarm of two K-Team Khepera III wheeled robots in an indoor area (~9 m^2) [21] used a nonexplicit gradient estimation strategy to seek a light source, while [22] implemented a Bayesian-based motion planning algorithm with three iRobot Create wheeled robots seeking a gas plume source in an indoor area (~36 m^2). [72] utilized a particle swarm control strategy using seven TurtleBot3 wheeled robots in an indoor area (~25 m^2) to seek a time-varying and moving source.

Contour Following. The more complex task to track/trace a level set (i.e., contour following) at a pre-specified value in a 2D scalar field without prior knowledge of its location, has also attracted significant research interest and requires a further increase in AN sophistication. Prior research using a single vehicle simulated this task assuming simplified vehicle kinematics with sliding mode [23], variational Bayesian-based [24], and hybrid level set tracking with obstacle avoidance [69] control strategies.

Planar level set tracing/tracking has also been explored experimentally with single vehicle systems. [25] used a wheeled robot to track illumination level in a small-scale indoor testbed (~2 m^2). [26] and [27] completed plume tracing of Rhodamine dye by deploying an Autonomous Underwater Vessel (AUV) at a fixed depth in outdoor field applications (~30k m^2) controlled with a moth-inspired strategy. [28] used sliding-mode control to perform a hybrid experiment using an Activ-Media Pioneer 3-DX wheeled robot navigating with respect to a simulated scalar field (computed, not sensed) in a small indoor testbed (~7 m^2). [29] simulated contour mapping and used a single quadcopter to experimentally demonstrate fixed altitude holding with an anti-windup PI-controller.

Extending level set tracking/tracing in 2D to multiple vehicle systems necessitates a further increase in system sophistication to manage group control. Previous research with simulations using multivehicle systems controlled with a wide array of schemes have successfully demonstrated this task assuming negligible agent dynamics and external disturbances. The method demonstrated in [30] used a single vehicle control approach in parallel within a group of robots. Individual robots already stationed around a time-varying elliptical contour moved independently to maintain their position on the contour; robots then shared their location to collectively generate an estimate of the contour. [31] performs cooperative filtering using Hessian information to move a robot group along noisy level curves with active formation shaping to minimize gradient estimation error. In [32], a moth-inspired control strategy directed a lead robot along a plume using single robot movement strategy with two other robots flanking the leader; as the leader moves out of the plume, a flanking robot would theoretically remain in the plume and become the new leader. Other teams have also demonstrated this task with fixed, circular formations of agents using a gradient-based control strategy [33][34]. A concern for any multivehicle system is ensuring safety through the inclusion of collision avoidance laws; a wide range of such strategies exist in the literature, such as [74].

Experiments have also been performed with UAVs at fixed altitudes in small indoor testbeds. [35] implemented gradient-based control of three DJI Flamewheel 450 UAVs in fixed circular formations to demonstrate moving light source seeking and three Crazyflie 2.0 UAVs to perform level set tracking/tracing about a simulated scalar field (computed, not sensed) in a small enclosed area ~2 m^2. [73] used seven Crazyflie 2.1 UAVs at fixed altitudes to swarm towards simulated scalar field sources (computed, not sensed) in a region ~12 m^2.

Prior 2D AN Work at Santa Clara University. Prior work by this research group has presented AN control strategies for both locating local extreme points and following contours, and has demonstrated these in simulation, lab experiments, and field demonstrations. Furthermore, this prior work has proposed new AN objectives to include moving along ridges/trenches, locating saddle points, and navigating along fronts; all of these control capabilities have been verified in both simulation and experiment [1][47][48].

I.B. ADAPTIVE NAVIGATION IN 3D SCALAR FIELDS

Compared to 2D techniques, research on 3D AN techniques is significantly less developed.

Extrema Seeking. Single vehicle simulations assuming simplified vehicle kinematics or point mass models have demonstrated extrema seeking with strategies utilizing gradient following [36][37], hybrid-sliding mode control [38], and forward and angular velocity regulation as the robot approaches the source [39]. Experimentally, [12]

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

demonstrated source seeking of a dynamic gas plume with a single UAV under fuzzy-logic control-based spatial dithering, in a small indoor testbed (~9 m^2).

3D source seeking with multirobot systems has been demonstrated in simulation studies that assumed negligible agent dynamics and external disturbances. In [10], the authors proposed a formation that adapts to its environment to optimize gradient climbing. In [40] a bioinspired swarm using a Speeding Up and Slowing Down strategy synchronized the direction of motion of the robots by varying the speed of each vehicle based on the measured scalar field value. Others utilized symmetric and fixed formations to navigate based on the collaborative estimation of gradient and Hessian information using noisy measurements [41]. [43] provided a comprehensive survey of "plume tracing and mapping via swarm robots" which we classify as extrema finding since it involves agents moving towards a source.

Contour Following. 2D contour/level set tracking becomes navigation within an isosurface when extended to a 3D implementation. A single vehicle strategy for this was explored in simulation [42] by assuming a simplified kinematic robot, no external disturbances, and minimal system noise. The gradient free strategy utilized the vehicle's pose, the instantaneous scalar field reading, and the time derivative of that signal to generate yaw control for a nonholonomic agent, allowing it to orbit a desired isosurface while cycling vertically between two predefined altitudes at a constant speed.

Multivehicle contour / level set tracking in 3D (e.g., isosurface navigation) has also been demonstrated in simulation typically assuming negligible robot dynamics and limited external disturbances. Using a leader-follower strategy initially published in [15], another research team [44] simulated the task of 2D planar contour mapping on a highly simplified 3D radiation field (i.e., an extruded ellipse). The 3D aspect entailed holding each of the two UAVs at different fixed altitudes and performing concurrent contour mapping in parallel planes. In [45], Jacobi transform based formation control (to decouple the dynamics of the formation's center from its shape) utilized a Hessian information to move a robot group along noisy 3D level set curves to track principal lines of surface curvature. In [46], two vessels in a fixed formation, in which one agent functioned as the sensor and the other as the tracker, simulated dynamic plume front (i.e., time-varying level set) tracking while assuming ideal sensor measurements and an exponentially decaying external disturbance.

I.C. CONTRIBUTIONS TO 3D SCALAR FIELD ADAPTIVE NAVIGATION

This paper presents initial results of a comprehensive program to develop, verify, and experimentally implement mission-level AN capabilities in three-dimensions. We build upon our prior work and contribute to 3D scalar field AN by:

- demonstrating a wide range of diverse 3D AN capabilities using a single, unified, multilayer control architecture;
- defining new four- and nine-robot formation geometries specific to the needs of the new 3D AN primitives, to include the derivation of their associated forward and inverse kinematic transforms;
- extending our 2D multivehicle scalar field AN control primitives to 3D space in order to implement local extrema finding, navigation within isosurfaces, and navigation along plumes;
- introducing the state-based sequencing of different AN control primitives to execute mission-level capabilities such as isosurface mapping and downstream plume following;
- verifying the functionality of these aforementioned capabilities using high-fidelity simulations which account for vehicle dynamics, outdoor wind gust disturbances, position sensor inaccuracy, and scalar field sensor noise.

Because our focus in this article is to present new 3D AN capabilities, we have made a number of simple implementation choices. For example, our proposed control laws are composed of basic proportional control terms. Furthermore, we have used minimally-sized groups of robots that allow instantaneous characteristics of the scalar field to be computed and guidance decisions to be made for the proof of concept capabilities presented. Without question, more sophisticated controller approaches are possible in order to improve performance; such improvements are proposed as future work.

Section II describes the individual vehicle-level dynamic model, external wind disturbance model, and sensor noise models that our high-fidelity simulations are built upon. Section III discusses the multilayered control architecture and provides details of each layer. Section IV presents the control architecture used to implement local extrema seeking (both static and time-varying), the definition of the required multivehicle cluster, the gradient estimation method, the proposed control primitive, and high-fidelity simulation results. Section V presents the implementation of isosurface navigation and mapping within the architecture, a description of the proposed control primitives, the state machine used to perform state-based sequencing for "mission-level" functions, and high-fidelity simulation results. Section VI presents the downstream plume following (away from a source) implementation, the definition of the required multivehicle cluster, the proposed control primitive, the state machine used to perform state-based sequencing for "mission-level" functions, and high-fidelity simulation results. Finally, Section VII discusses broad questions, issues and challenges while Section VIII summarizes this work and discusses ongoing and future research efforts.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

II. VEHICLE DYNAMICS AND CONTROL

The work presented in this paper simulates a cluster of AR.Drone quadcopters, which are consumer-grade UAVs manufactured by Parrot, Inc. Given the use of an onboard autopilot for platform stabilization with limited pitch and roll angles, we consider only the remaining degrees of freedom (DOF) with the relevant position and velocity state of the *i*th quadcopter as defined in (1).

$$\vec{R}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \\ \psi_i \end{bmatrix} \quad , \quad \dot{\vec{R}}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{z}_i \\ \dot{\psi}_i \end{bmatrix} \tag{1}$$

The four DOF dynamic model for an individual quadcopter is shown in (2), where values for the diagonal matrices K_u and K_v have been experimentally determined [51]-[53] and wind disturbance forces have been included. M is the vehicle mass (0.42 kg), $\dot{\vec{R}}_{i\,com}$ is the vector of commanded velocities, and $[D_x, D_y, D_z]^T$ are the drag force components. Additional details of the UAV dynamic model and its validation are presented in Appendix A.

$$\ddot{\vec{R}}_i = K_u \dot{\vec{R}}_{i\,com} - K_v \dot{\vec{R}}_i + \frac{1}{M} \begin{bmatrix} D_x \\ D_y \\ D_z \\ 0 \end{bmatrix} \tag{2}$$

A Von Kármán turbulence model with a mean wind speed of 5 m/s is used to induce normal forces (i.e., ram/momentum drag) on each UAV propeller [54] as a function of mass flow rate m_f and wind velocity $[V_x,V_y,V_z]^T$. Equation (3) illustrates the drag computation with model parameters summarized in Table II and with typical wind gust time histories shown in Fig. 1.

$$\begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} = P \cdot m_f \cdot \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = P \cdot k \cdot \left(\rho A \sqrt{\frac{\tau}{2\rho A}} \right) \cdot \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} \tag{3}$$

TABLE II
VON KÁRMÁN WIND TURBULENCE MODEL PARAMETERS

Number of Propellers	$P = 4$
Correction Coefficient (non-shrouded propeller)	$k = 0.8$
Air Density (sea level @ +15 °C)	$\rho = 1.2260\ {}^{\mathrm{kg}}/_{\mathrm{m}^3}$
Propeller Disc Area	$A = 0.0314\ \mathrm{m}^2$
Propeller Axial Thrust (hovering)	$\tau = 1.03\ \mathrm{N}$

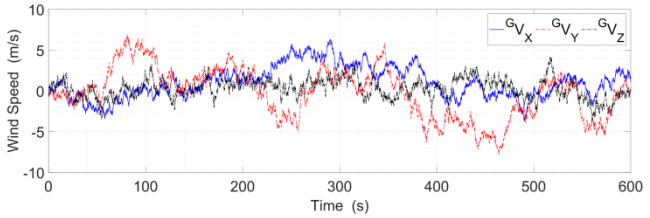


FIGURE 1. Von Kármán turbulence model simulated wind gusts.

Sensor noise is simulated using band-limited white noise to generate normally distributed random numbers which are scaled by a constant factor. For the vehicle position sensors, manufacturer supplied RMS accuracy values can be approximated as one standard deviation σ when there is zero mean error [55][56].

Scaling the simulated noise to attain a standard deviation equal to the RMS value allows us to achieve a 98% Accuracy Distribution (AD) [57] as shown in (4).

$$98\%\ (AD) = 2.8 \cdot (RMS) = 2.8 \cdot \sigma \tag{4}$$

Since typical UAV mounted consumer-grade GPS receivers advertise positional accuracies from 1.0 m to 1.8 m RMS [58][59], the higher end value is used as the simulated position sensor noise standard deviation target. Similarly, electronic noise for vehicle mounted air pollutant sensors (e.g., NO_2, SO_2, CO) use a scaling factor based on the sensor manufacturer's provided standard deviation [60][61]. For comparison, this method generates larger sensor noise amplitudes compared to those used in other high-fidelity simulations [42][46][52][68].

III. MULTIROBOT CONTROL ARCHITECTURE

Our approach to implementing "mission-level" multirobot AN uses a multilayer control architecture, shown in Fig. 2. This architecture is general in the sense that it is applicable to different numbers and types of robots, operating in different domains, and performing different types of adaptive navigation functions. This section discusses the overall control architecture and provides details of the modeling and control implemented in each layer for multiple UAV clusters performing 3D AN.

III.A CLUSTER-SPACE FORMATION CONTROL

Within the architecture, the Cluster-Space Controller layer responds to commands regarding the desired geometry and motion of the cluster by issuing translational and rotational velocity commands to each vehicle in the system. The controller uses a full-DOF operational-space framework, managing the motion and geometry of the group of robots as a virtual articulating mechanism whose shape, size, and orientation can be reconfigured in-flight [49]. The ability to implement a particular geometry allows for precise, spatially dependent scalar field measurements, tunable for (in)sensitivity to certain spatial frequencies.

For a system with an n number of robots, we represent the "robot-space" pose and velocity vectors as a concatenation of the individual robot position and velocity, as shown in (5).

$$\vec{R} = \begin{bmatrix} \vec{R}_1 \\ \vdots \\ \vec{R}_i \\ \vdots \\ \vec{R}_n \end{bmatrix} \quad , \quad \dot{\vec{R}} = \begin{bmatrix} \dot{\vec{R}}_1 \\ \vdots \\ \dot{\vec{R}}_i \\ \vdots \\ \dot{\vec{R}}_n \end{bmatrix} \tag{5}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

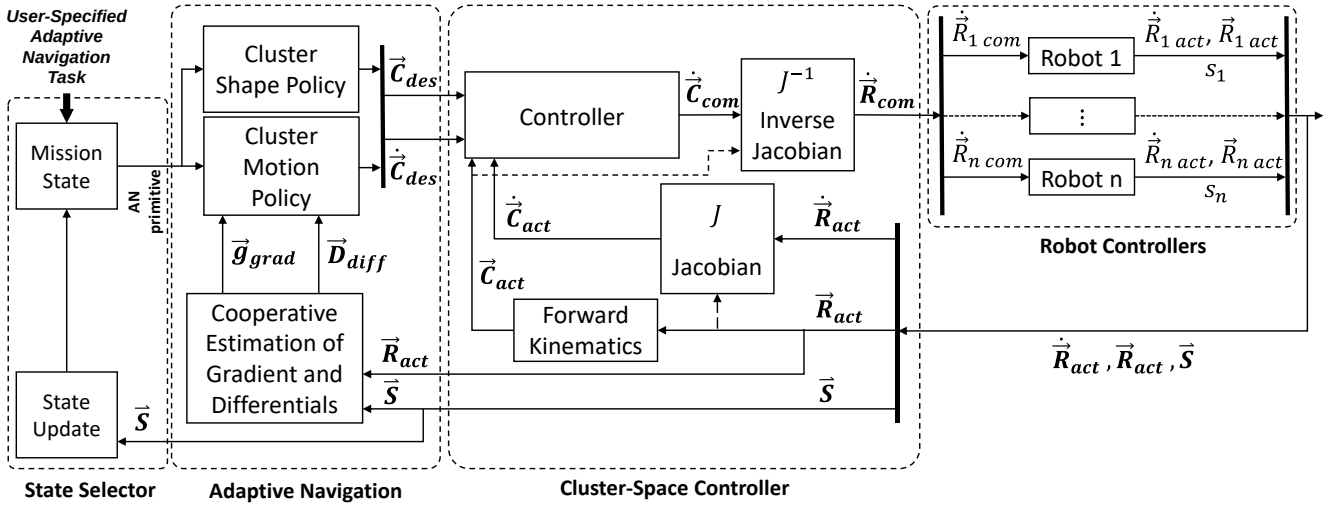


FIGURE 2. **Multilayered Control Architecture Block Diagram**

Within the Cluster-Space Controller layer, however, we use a formation-specific representation of the motion state of the system. The "cluster-space" positions, represented by the pose vector $\vec{c}$, typically include the position and orientation of the overall cluster, variables describing the cluster geometry, and variables that specify the relative orientation of each vehicle. The cluster velocity vector $\dot{c}$ is the time derivative of $\vec{c}$.

A set of kinematic relationships relate the robot-space and cluster-space vectors as presented in (6) and (7), where *n* is the number of robots in the cluster, *m* is the number of DOF per robot, and *J* is the system's Jacobian matrix.

$$\vec{C} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{mn} \end{bmatrix} = KIN(\vec{R}) = \begin{bmatrix} f_1(r_1, r_2, \cdots, r_{mn}) \\ f_2(r_1, r_2, \cdots, r_{mn}) \\ \vdots \\ f_{mn}(r_1, r_2, \cdots, r_{mn}) \end{bmatrix} \quad (6)$$

$$\dot{\vec{C}} = \begin{bmatrix} \dot{c}_1 \\ \dot{c}_2 \\ \vdots \\ \dot{c}_{mn} \end{bmatrix} = J(\vec{R})\dot{\vec{R}} = \begin{bmatrix} \frac{\partial f_1}{\partial r_1} & \frac{\partial f_1}{\partial r_2} & \cdots & \frac{\partial f_1}{\partial r_{mn}} \\ \frac{\partial f_2}{\partial r_1} & \frac{\partial f_2}{\partial r_2} & \cdots & \frac{\partial f_2}{\partial r_{mn}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{mn}}{\partial r_1} & \frac{\partial f_{mn}}{\partial r_2} & \cdots & \frac{\partial f_{mn}}{\partial r_{mn}} \end{bmatrix} \dot{\vec{R}} \quad (7)$$

Cluster-space position variables are generally nonlinear functions of robot position variables, while the cluster velocity vector is an instantaneously linear function of robot velocities as expressed by a Jacobian matrix. The exact nature of these functions depend on the specific cluster definitions used.[2] For the work presented in this article, we use two different cluster definitions which are detailed in Section IV.A and IV.A.

The Cluster-Space Controller layer receives instantaneous cluster mobility and geometry setpoint commands in the form of desired values of $\vec{c}_{des}$ and $\dot{c}_{des}$, which are provided by the Adaptive Navigation control layer. Cluster-space error signals are generated by subtracting estimates of realtime cluster-space values from these setpoints. These estimates are generated by applying the kinematic transforms to robot-space position and velocity commands.[3]

For this study, the control law itself is a simple, error-driven, resolved-rate, proportional controller that computes compensation commands in the cluster-space. This conversion is computed by use of the cluster's inverse Jacobian transform and robot commands consist of instantaneous translational and rotational velocity setpoint commands.[4]

Overall, the set of instantaneous robot-space velocity commands for each vehicle is computed as specified by (8), where K_p is a diagonal matrix of proportional gains appropriate for the variables being controlled. For this study, these gains were empirically developed to achieve the desired performance; in prior work, we demonstrate the use of a control law design process using a partitioned, nonlinear, model-based method.

$$\begin{aligned} \dot{\vec{R}}_{com} &= J^{-1} \cdot \dot{\vec{C}}_{com} \\ &= J^{-1} \cdot \left[K_p \cdot \left[Q \cdot (\vec{C}_{des} - \vec{C}_{act}) + Q' \cdot \left(\dot{\vec{C}}_{des} - \dot{\vec{C}}_{act} \right) \right] \right] \end{aligned} \quad (8)$$

[2] The cluster designer has a great deal of flexibility in defining the "location" of the cluster. Fundamentally, the position and orientation of the cluster frame is defined as a function of robot positions and orientations, and these definitions constitute several of the cluster pose functions in the forward kinematic functions in (6). The frame can be assigned to be coincident with one particular robot, or it could be defined as a function of two or more of the cluster's robots. There are ramifications to this choice in terms of the level of (de)centralization that can be achieved in controlling the cluster.

[3] Alternatively, it is possibly to make direct measurements of some or all of the cluster-space variables, such as the distance between vehicles. Although we do not use that approach in this study, we have implemented such relative sensing in prior work.

[4] In prior work, we have used a variety of other cluster-space control laws, including full PID and nonlinear control laws, dynamic controllers that issue force/torque commands, etc. We have also defined stability criteria and demonstrated the incorporation of robust collision avoidance behaviors and singularity handling strategies.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

III.B ADAPTIVE NAVIGATION CONTROL

The next layer in the control architecture, the Adaptive Navigation layer, issues cluster shape and mobility commands to the Cluster-Space Controller layer. It does this based on the position and measured local scalar field value from each robot as well as the selected "primitive" AN strategy of interest. As shown in Fig. 3, these primitives use robot positions $\vec{R}$ and sensor readings $\vec{S}$ (vector of the measured scalar field values from Robots 1 through n) to compute relevant characteristics of the local scalar field, such as the gradient or differential measurements across various baselines within the cluster. Based on these characteristics, the primitive control law provides instantaneous setpoint commands for the cluster's position $\vec{C}_{des}$ and velocity $\dot{\vec{C}}_{des}$ in order to guide the cluster to/along scalar field features of interest.

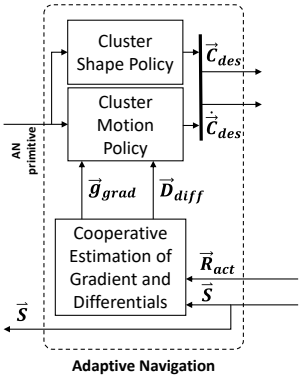


FIGURE 3. The Adaptive Navigation layer.

More specifically, the AN controller provides a setpoint command for the position or velocity (e.g., not both) relating to each cluster-space parameter. In implementing each AN control primitive, complementary diagonal selector matrices, Q and Q', are used to designate which cluster pose variables are position- vs. velocity-controlled. For each cluster-space variable, a value of 1 or 0 on the diagonal of the Q and Q' matrices specify position or velocity control, respectively.

For 2D AN, we have demonstrated "primitive" controllers capable of implementing extrema finding, contour following, saddle point positioning, ridge descent/trench ascent, and front following [1]. For this study, we introduce primitive AN controllers for 3D extrema finding, isosurface navigation, and plume following.

III.C MISSION-SPECIFIC STATE CONTROL

Adaptive navigation primitives are designed to perform a very specific navigation task. In some cases, such as with extrema seeking, the primitive might be all that is required for certain applications, such as finding a local signal source. More complex activities, however, may demand control beyond the execution of a single primitive, requiring a) the combination of an AN primitive with another controller to achieve parallel objectives, b) the cycling of different control parameter values over time for a single AN primitive, and/or c) the sequencing of different AN primitives, based on task and performance. As shown in Fig. 2, our control architecture implements such complex functionality through the use of a state machine.

In this work, two specific 'missions' of interest for 3D scalar fields are described. The missions showcase each of the three aforementioned cases of tailoring the use of the available suite of AN primitives. Simulations of these missions are provided in Sections V.D and VI.D.

IV. LOCAL EXTREMA SEEKING

Local extrema seeking is valuable for a wide range of applications. Examples include tasks such as finding emitters, sources of pollution or leaks, anoxic areas in bodies of water, and so on. In this work, we implement local extrema seeking by using a single AN control primitive. This uses distributed synchronized sensor measurements from each vehicle, in order to compute an estimate of the local field gradient as a reference direction for motion. Commanding the cluster to move in the direction of the gradient results in moving towards a local maximum/source within the field, while traveling in the opposite direction guides a cluster to a local minimum/sink in the field.

For the work presented in this article, we use a four-robot cluster in a tetrahedron-style formation; a shape with the minimum number of robots required to compute an instantaneous 3D field gradient. This allows AN to be implemented without requiring extraneous motions necessary to characterize the nature of the local field. The cluster is sized based on the considerations discussed in Section VII and detailed in [1].

In this section, we present the four-robot cluster definition, the gradient estimation approach, and the gradient-based AN control law; taken together, this provides all information necessary to implement the local extrema finding capability. We demonstrate the functionality of this primitive by providing results of high-fidelity simulations for a static scalar field as well as a time-varying field that is translating.

IV.A FOUR-ROBOT TETRAHEDRON CLUSTER

A four-robot cluster in a nominal tetrahedron shape is shown in Fig. 4, where *R1* through *R4* represent the origins of the individual vehicle frames. Note that this definition is used for the local extrema seeking AN primitive and in Section V, for isosurface navigation and mapping.

The pose of each robot in the figure is represented by its robot-space position and orientation variables listed in Table III. The robot-space pose vector of the group, $\vec{R}_{4R}$, contains all of these variables for each of the four robots. The robot-space velocity vector, $\dot{\vec{R}}_{4R}$, is the time derivative of $\vec{R}_{4R}$.

The cluster definition used for this study placed the origin (Point B) of the cluster frame C_{4R} at the centroid of the triangle formed by Robots 1 to 3, where $\hat{z}_C$ is perpendicular to that plane and $\hat{x}_C$ is pointed towards Robot 1. The six cluster shape variables, which include three distances and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

three angles that define the cluster's geometry, are shown in Fig. 4 and listed in Table IV; this includes a side-angle-side description of the base triangle as well as a distance-azimuth-declination description of Robot 4 with respect to the cluster origin. The cluster-space position vector $\vec{C}_{4R}$ consists of the variables in Table IV while the cluster-space velocity vector $\dot{\vec{C}}_{4R}$ is the time derivative of $\vec{C}_{4R}$. The full listing of the vectors $\vec{R}_{4R}$ and $\vec{C}_{4R}$ are provided in Appendix B.

The forward position kinematic equations of the form described in (6) which transform "robot-space" pose vector $\vec{R}_{4R}$ to the "cluster-space" vector $\vec{C}_{4R}$ are presented in Appendix B.

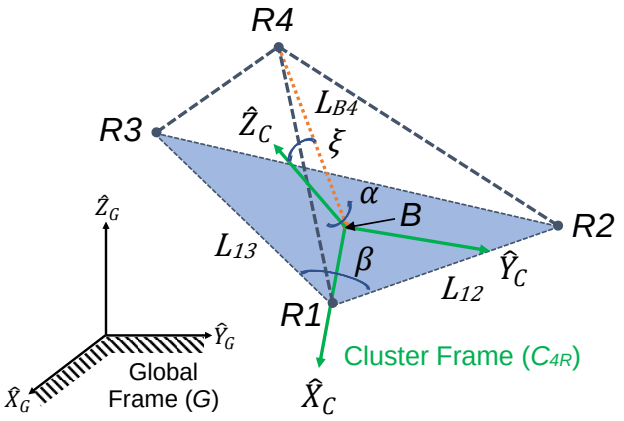


FIGURE 4. Four-robot virtual structure lengths and internal angles.

TABLE III
FOUR-ROBOT FORMATION ROBOT-SPACE VARIABLES

Robot position variables, n = 1,2,3,4	${}^{G}x_n$	${}^{G}y_n$	${}^{G}z_n$
Robot yaw angles, n = 1,2,3,4		${}^{G}\psi_n$	

TABLE IV
FOUR-ROBOT FORMATION CLUSTER-SPACE VARIABLES

Cluster position variables	${}^{G}X_B$	${}^{G}Y_B$	${}^{G}Z_B$
Cluster orientation variables	${}^{G}\Phi_C$	${}^{G}\Theta_C$	${}^{G}\Psi_C$
Cluster shape variables, distances	L_{12}	L_{13}	L_{B4}
Cluster shape variables, angles	α	β	ξ
Robot relative yaw variables, n = 1,2,3,4		${}^{C}\Psi_n$	

These equations are used in the Cluster-Space Controller layer in order to estimate realtime cluster pose, which is an input to the cluster controller. The inverse Jacobian relationship J^{-1} is used to transform cluster control velocity commands to individual robot commands. This 16x16 matrix has not been included given its cumbersome nature.

IV.B GRADIENT ESTIMATION

Equation (9) generates the cooperative estimated gradient [15] at the robot formation's geometric center where ${}^{G}x_i$, ${}^{G}y_i$, ${}^{G}z_i$, are the ith robot position coordinates, s_1 through s_4 are the measured scalar field values from Robots 1 through 4, and g_x, g_y, g_z are the components of the gradient vector as presented in (10).

$$\begin{bmatrix} g_x \\ g_y \\ g_z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{G}x_1 & {}^{G}y_1 & {}^{G}z_1 & 1 \\ {}^{G}x_2 & {}^{G}y_3 & {}^{G}z_2 & 1 \\ {}^{G}x_3 & {}^{G}y_3 & {}^{G}z_3 & 1 \\ {}^{G}x_4 & {}^{G}y_4 & {}^{G}z_4 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} \tag{9}$$

$$\vec{g}_{grad} = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} \tag{10}$$

IV.C ADAPTIVE NAVIGATION CONTROL PRIMITIVE

The gradient-following control primitive generates translational cluster velocity commands to guide the cluster with a speed S in the direction of or opposite to the gradient by setting $d_{extrema}$ to +1 or -1, respectively, as shown in (11).

$$\dot{\vec{C}}_{4R}(1{:}3) = \begin{bmatrix} \dot{X}_{B\,des} \\ \dot{Y}_{B\,des} \\ \dot{Z}_{B\,des} \end{bmatrix} = S \cdot d_{extrema} \cdot \begin{bmatrix} \frac{g_x}{\|\vec{g}_{grad}\|} \\ \frac{g_y}{\|\vec{g}_{grad}\|} \\ \frac{g_z}{\|\vec{g}_{grad}\|} \end{bmatrix} \tag{11}$$

While cluster translational velocity commands are continuously varied based on the estimate of the local field gradient, all other cluster orientation and geometry variables designated by $\dot{\vec{C}}_{4R}(4{:}16)$ are controlled to specified position setpoints through the use of a proportional position controller. This blending of proportional velocity and position control is achieved by the resolved-rate cluster-space controller expressed by (8) given the selector matrix $Q' = diag(1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$ and its complement Q.

Given the empirical tuning used in this study, this fully specifies the control law for the four-robot cluster in order to move along the local gradient of a field in order to navigate to the local minimum or maximum of the field.

IV.D SOURCE SEEKING SIMULATION RESULTS

This section presents high fidelity flight simulation results demonstrating the capabilities of the 3D AN gradient-following primitive, which implements an extrema finding behavior. These simulation results incorporate the verified vehicle dynamics model presented in (2), with wind gust disturbances and system/sensor noise. The velocity command for each vehicle in the four-robot cluster is specified by (8), where $\vec{C}_{4R\,des}$ and $\dot{\vec{C}}_{4R\,des}$ are specified as described in Section IV.C, the gradient estimate is computed as described in Section IV.B, and the four-robot formation and its kinematic transforms used to specify the instantaneous vehicle velocity commands are described in Section IV.A.

Plots of 3D flight paths, time histories of relevant variables, and error metrics serve to illustrate the system performance during mission execution. The simulated scalar fields in this section are documented in Appendix D.

For these simulations, the four-robot cluster is held in a tetrahedron formation with shape variables shown in Table V and with fixed relative cluster attitude angles. Cluster

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

translational speed was 3 m/s, and navigation occurred in a cubic workspace with 800 m length sides.

TABLE V
FOUR-ROBOT FORMATION SHAPE VARIABLES

L_{12}	30 m	α	10°
L_{13}	30 m	β	60°
L_{B4}	24.5 m	ξ	20°

IV.D.1 STATIONARY SOURCE SEEKING IN A STATIC FIELD SIMULATION

First, we demonstrate a four-robot cluster navigating to a stationary maximum point within a time-invariant scalar field simulated as a vertically oriented directional field with a broad beam-width. Note that in [62], we demonstrated time-invariant source seeking without dynamic robot models, wind disturbances, or system noise. For our new work, Fig. 5 illustrates the resulting cluster paths for four different trials, each with a different starting point. As seen in the figure, the cluster successfully navigates to the source in each trial.

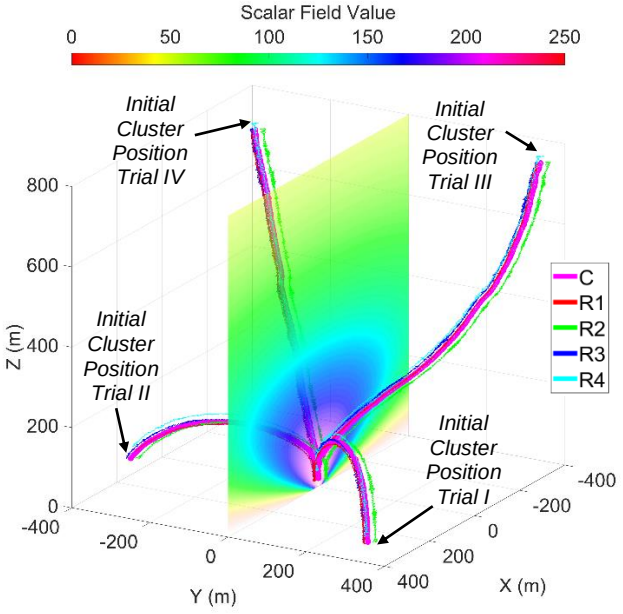


FIGURE 5. Multiple simulation runs showing flight paths of a four-robot cluster seeking a stationary and time-invariant source, starting from different initial cluster locations.

As an example of performance, for trial IV, Fig. 6 shows the average sensed scalar field value of the formation increasing as expected over time, while the cluster navigates to the source. Given that the AN law directs the cluster to move in the direction of the gradient, Fig. 7 shows the angular deviation of these vectors over time; the RMS error for this period was 0.30 radians, until the cluster effectively reached the source. Furthermore, given that the controller is working to hold formation geometry as the cluster navigates, the RMS errors for the cluster size parameters L_{12}, L_{13}, and L_{B4} were 3.5 m, 3.6 m and 0.9 m, respectively; which are within 0.5-2.0 times the standard deviation of the simulated position sensing error, indicating acceptable formation control.

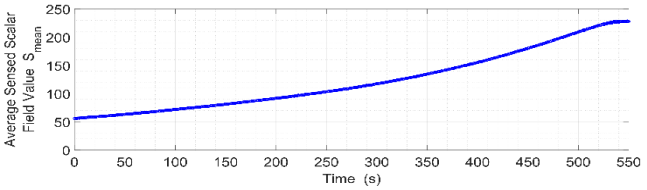


FIGURE 6. Time history of average sensed scalar field value starting from cluster initial position trial IV.

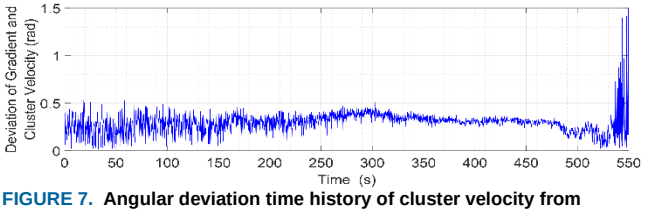


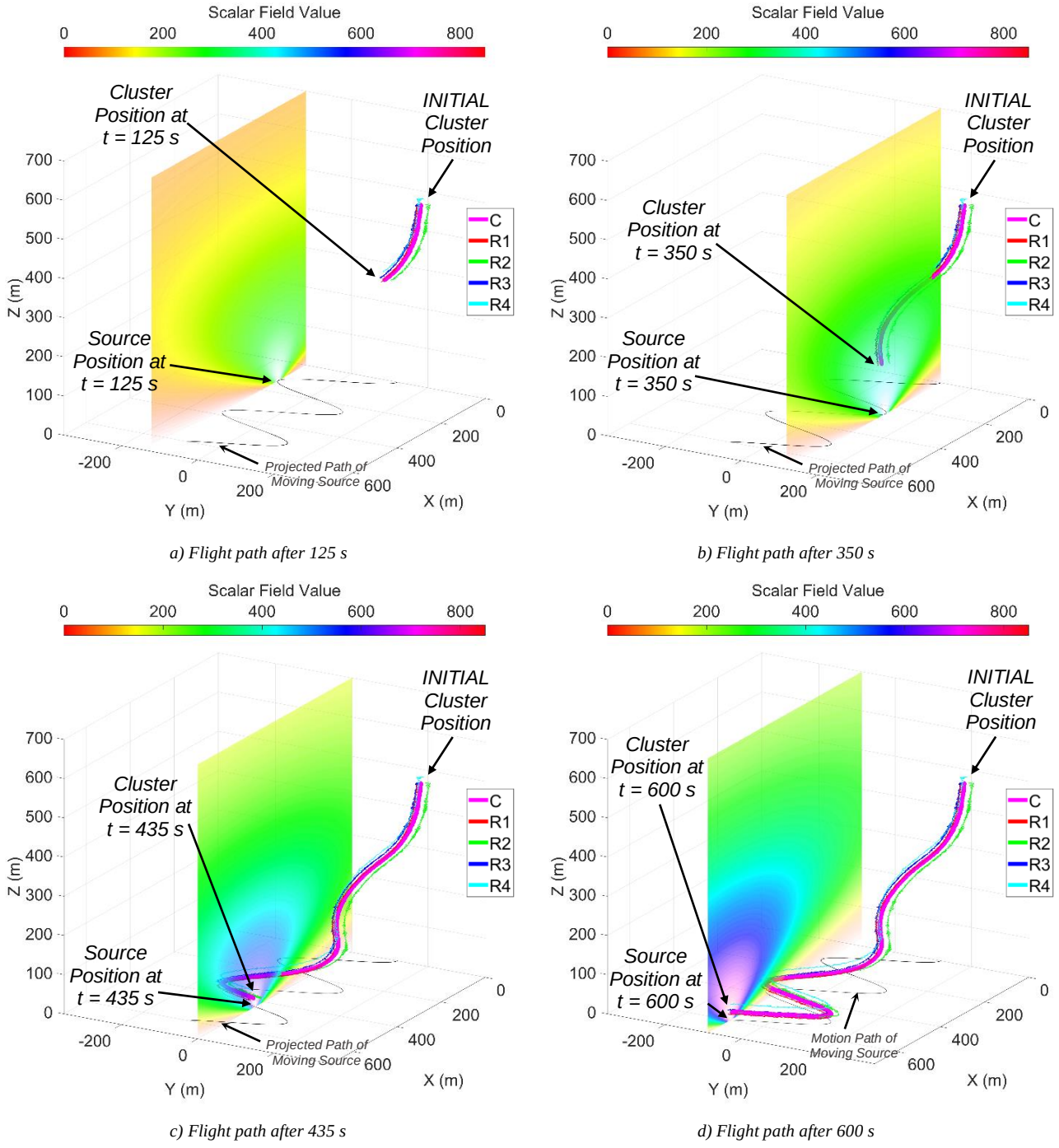
FIGURE 7. Angular deviation time history of cluster velocity from estimated gradient starting from initial position trial IV. Cluster reaches the local maximum at approximately 530 s. The cluster dithers in the vicinity of the maximum without a motion termination condition.

IV.D.2 MOVING SOURCE SEEKING IN A TIME-VARYING FIELD SIMULATION

The extrema seeking AN primitive also enables a cluster to track a moving source in a dynamic field, provided that the cluster speed is greater than the source's speed and the evolution of the field is well-behaved. Without fully characterizing the limitations of this capability, we offer here a simple scenario of navigation in which the cluster locates a moving source within a time-varying field. Multipart Fig. 8 shows a cluster performing this function. The multipart figure shows how the source (which lies in the z = 0 plane) moves over time in a sinusoidal pattern, while the intensity of the field grows over time. Cluster shape and speed as well as the size of the navigation region were the same as used in Section IV.D.1. For this scenario, the speed of the source is approximately 75% that of the cluster. The multimedia file associated with this article provides a video of this maneuver.

To characterize performance, Fig. 9 shows the average sensed scalar value for the formation increasing as expected over time, as the cluster navigates toward the moving source. Fig. 10 shows the angular deviation of the cluster motion and gradient vectors over time, given that they should ideally be aligned; the RMS error for the shown maneuver was 0.22 radians. Note that at approximately t = 390 s, the cluster "catches up" with the moving source so that small changes in tracking position result in large angular deviations between the estimated gradient and cluster velocity vector. As for formation control performance as the cluster navigates, the RMS errors for the cluster size parameters L_{12}, L_{13}, and L_{B4} were 3.5 m, 3.5 m and 0.9 m, respectively; these are nearly identical to the results shown in Section IV.D.1 and are in a range of 0.5-2.0 times the standard deviation of the simulated position sensor error, indicating acceptable formation control.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access



a) Flight path after 125 s

b) Flight path after 350 s

c) Flight path after 435 s

d) Flight path after 600 s

FIGURE 8. **Flight path of a four-robot cluster seeking/tracking a moving and time-varying source.**

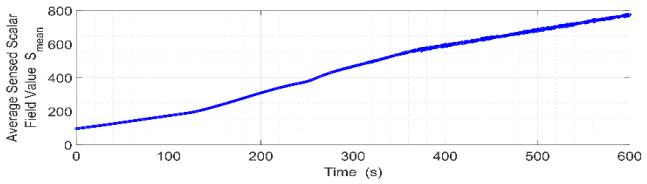


FIGURE 9. **Time history of average sensed scalar field values while seeking a moving and time-varying field.**

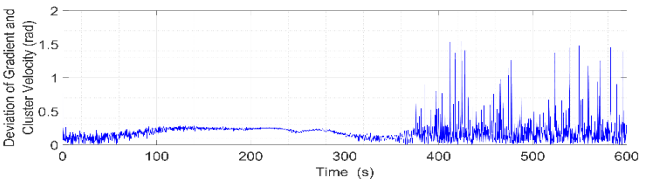


FIGURE 10. **Angular deviation time history of cluster velocity from estimated gradient. Note that at approximately 390 s, the cluster "catches up" with the moving source so that small changes in tracking position result in large angular deviations between the estimated gradient and cluster velocity vector.**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

V. ISOSURFACE NAVIGATION AND MAPPING

The extension of 2D contour following to 3D scalar fields is movement along a scalar isosurface. Accordingly, this AN primitive controller moves a cluster to and maintains a cluster's position on an isosurface; however, motion within the isosurface is unconstrained, allowing other control objectives to be met. One such control objective is structured isosurface mapping which will be discussed in Sections V.C and V.D.

To move a cluster to and then maintain its position on an isosurface, we utilize the same four-robot cluster definition defined in Section IV.A. Sensed scalar readings from these robots are used to compute the local field gradient as provided in (9). In subsections V.A and V.B, we explain the isosurface AN primitive and show simulations of its performance. Then in subsections V.C and V.D, we propose and then show the performance of a mission-level state machine to implement structure isosurface mapping.

V.A ISOSURFACE NAVIGATION CONTROL PRIMITIVE

The proposed isosurface AN control strategy establishes a default direction of travel that is perpendicular to the gradient vector and therefore roughly parallel to the local surface tangent plane; a corrective vector term directed towards the desired surface and proportional to the scalar error is added to this default vector. The net vector is used to specify the direction of travel, and a constant speed is used to set the magnitude of the vector. This strategy is a 3D extension of the contour following strategy used in [1], which in turn was originally based on a field-implemented planar path-following controller presented in [63].

Equations (12) and (13) specify the cluster translational velocity commands to implement this motion control strategy. In (12), the first term directs the cluster in a direction that is tangential to the desired surface given the cross product with the gradient vector $\vec{g}_{grad}$; the specific direction is dictated by the choice of the navigation reference vector ${}^{G}\hat{n}$ and d_{orbit} which is set to +1 and -1 for counterclockwise and clockwise travel relative to ${}^{G}\hat{n}$, respectively. The second term in (12) guides the cluster along the gradient towards the desired surface in the event that it is not already on the surface. As part of this term, K_{surf} is a proportional corrective gain, s_{des} is the scalar value of the desired surface, and $s_{cluster}$ is the mean value of the cluster's scalar measurement s_1 through s_4.

$$v_{b-isosurface} = d_{orbit} \cdot \frac{\left(\vec{g}_{grad} \times {}^{G}\hat{n}\right)}{\left\|\vec{g}_{grad} \times {}^{G}\hat{n}\right\|} + \cdots$$
$$K_{surf} \cdot (s_{des} - s_{cluster}) \cdot \vec{g}_{grad} \qquad (12)$$

$$\dot{\bar{C}}_{4R}(1{:}3) = \begin{bmatrix} \dot{X}_{B\,des} \\ \dot{Y}_{B\,des} \\ \dot{Z}_{B\,des} \end{bmatrix} = S \cdot \frac{v_{b-isosurface}}{\left\|v_{b-isosurface}\right\|} \qquad (13)$$

While cluster translational velocity commands are continuously varied based on the estimate of the local field gradient, all other cluster orientation and geometry variables, designated by $\dot{\bar{C}}_{4R}(4{:}16)$, are controlled to specified position setpoints through the use of a proportional position controller. This blending of proportional velocity and position control is achieved by the resolved-rate cluster-space controller expressed by (8) given the selector matrix Q' = *diag(1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0)* and its complement Q.

It is noted that the navigation reference vector ${}^{G}\hat{n}$ is a user specified travel reference vector, given the constraint that it may not be collinear with the local gradient vector. Section V.D discusses how this vector may be selected in order to support specific mission-level objectives.

V.B BASIC ISOSURFACE NAVIGATION SIMULATION

This section demonstrates high fidelity simulations of isosurface navigation for two different navigation reference vector orientations as shown in Fig. 11. For each trial, the UAV cluster starts in the same location and travels to and then along the isosurface defined by s = 7 scalar units, a speed of S = 3 m/s, and a direction setting of d_{orbit} = 1. In trial A, a vertical navigation reference vector ${}^{G}\hat{n} = {}^{G}\hat{Z} = [0 \quad 0 \quad 1]^{T}$ is used, leading to a navigation path that orbits this vector in a counter-clockwise fashion once the isosurface is reached. In trial B, a similar result is achieved for a horizontal navigation reference vector ${}^{G}\hat{n} = {}^{G}\hat{Y} = [0 \quad 1 \quad 0]^{T}$, leading to a navigation path that orbits this vector in a counter-clockwise fashion once the isosurface is reached.

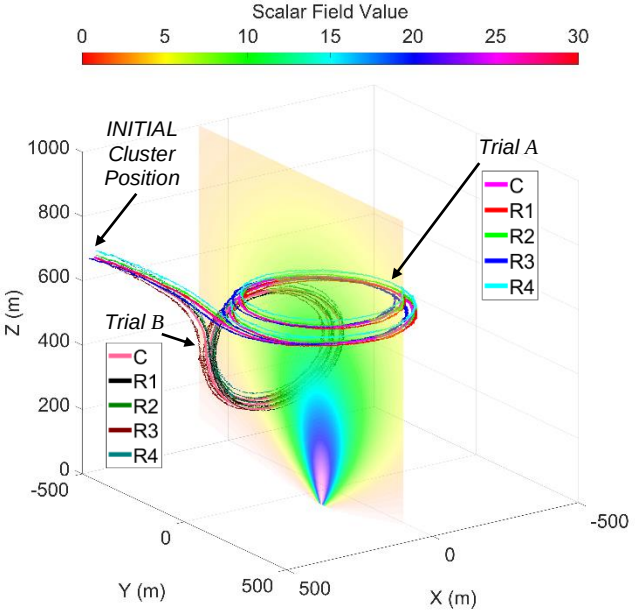


FIGURE 11. Basic isosurface AN with navigation reference vector set to ${}^{G}\hat{n} = {}^{G}\hat{Z} = [0 \quad 0 \quad 1]^{T}$ and ${}^{G}\hat{n} = {}^{G}\hat{Y} = [0 \quad 1 \quad 0]^{T}$ for trials A and B, respectively. The primitive controller executes closed loop control to move the cluster to and hold it along the desired scalar surface; however motion within the surface is not controlled.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

It is critical to note that this primitive controller executes closed loop control to move the cluster to and hold it along the desired scalar surface; however, motion within that surface is open loop. Hence, for both of these cases, motion of the cluster once it arrives at the isosurface wanders in the direction of navigation reference vector ${}^{G}\hat{n}$ while orbiting that vector.

To demonstrate performance for trial B in Fig. 11, Fig. 12 shows the average sensed scalar value of the formation increasing over time and settling at the desired isosurface scalar value as the cluster navigates to and then travels along the surface. Fig. 13 shows the alignment of the cluster's direction of motion with the gradient vector, indicating how navigation transitioned from moving toward to moving along the surface as it neared the surface.

Formation control performance is indicated by noting that the RMS errors for the cluster size parameters L_{12}, L_{13}, and L_{B4} were 3.4 m, 3.5 m and 1.0 m, respectively; these are all in a range of 0.5-2.0 times the standard deviation of simulated position sensor error, indicating acceptable formation control.

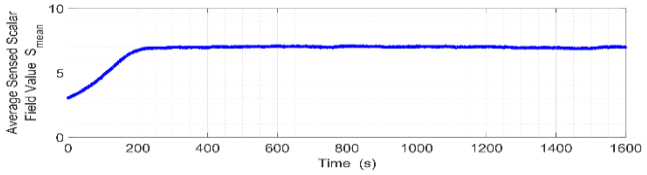


FIGURE 12. Time history of average sensed scalar field values during trial B as the formation navigates to and then travels along the surface.

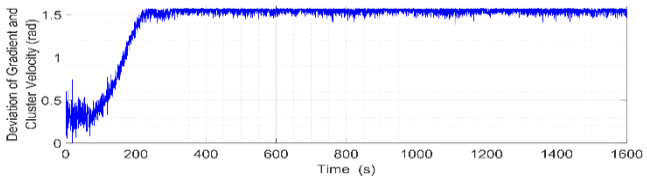


FIGURE 13. Angular deviation time history of cluster velocity from estimated gradient. As the cluster reaches the surface, the two vectors are perpendicular to each other, as is seen at approximately t = 200 s.

V.C ISOSURFACE MAPPING STATE MACHINE

Isosurface mapping consists of navigating along the surface of interest in order to characterize its structure through a systematic 'rendering' of the surface made by contour traces in periodically spaced planar slices of the surface; this concept is notionally depicted in Fig. 14. Navigating along a contour in a given plane consists of simultaneously performing isosurface navigation while constrained to a plane that intersects the surface. Once that contour has been circumnavigated, the cluster moves to another parallel plane and repeats the process.

To achieve this, isosurface mapping requires the specification of a set of mission parameters. This is done by defining the desired isosurface scalar value s_{des} a navigation reference vector perpendicular to the desired planar slices ${}^{G}\hat{n}$, and the distance between planes Δn is along the axis defined by the ${}^{G}\hat{n}$ vector. The parameter d_{orbit} is used to specify the direction of travel about the navigation vector when contour following; this typically does not matter for this particular mission type, and for our purposes it will be +1, indicating a counterclockwise direction of travel. Finally, the desired speed S must be specified.

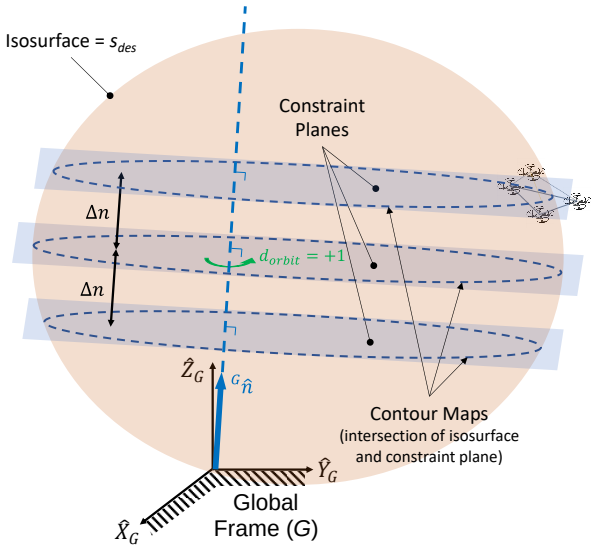


FIGURE 14. Isosurface mapping while constrained to a plane which intersects the surface.

The control states that constitute the state machine for this mission make use of these mission parameters. State transitions indicate transitions between states as well as the change of parameter values for a given state.

Control States. Although isosurface mapping can be implemented in a variety of ways, the state diagram shown in Fig. 15 illustrates how it has been implemented for the results reported in this article.

The state machine sequences the cluster through two control states. The first control state, "State 1: Locate Surface", uses only the basic isosurface navigation primitive discussed in Section V.A to guide the cluster to the surface with a scalar value of interest s_{des}. The specific controller in this state is the isosurface controller provided in (12) and (13).

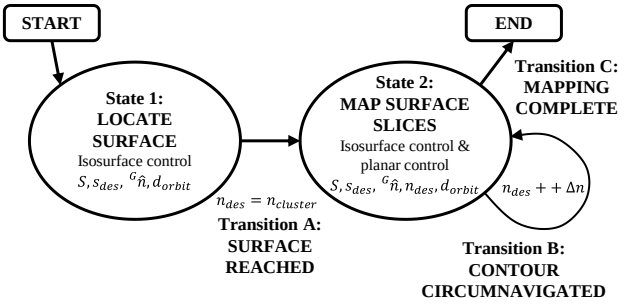


FIGURE 15. State machine for structured isosurface mapping. Each of the two states consists of one or more AN-level controllers with specific control parameters; the n_{des} control parameter is updated at each state transition.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

Once the surface is reached, "State 2: Map Surface Slices" enables navigation along the contour at the intersection of the desired surface and a specified plane; this is accomplished by a control equation that couples isosurface navigation as per (12) with a control term that guides the cluster to the intersection of the desired surface with a desired plane that intersects the surface. This multi-objective control equation for cluster translational velocity is shown in (14). As with (12), the first two terms in (14) direct the cluster to move within the desired scalar surface. A new third term adds a corrective velocity component that exploits a degree of freedom of motion within the isosurface in order to guide the cluster to the contour line at the intersection of the isosurface and the desired plane. Within the third term, K_n is a proportional corrective gain, and n_{des} is the planar location along the navigation reference vector ${}^{G}\hat{n}$. P_{B_proj} is the projection of the cluster point B position vector onto ${}^{G}\hat{n}$ as defined in (15).

$$v_{b-iso-mapping} = d_{orbit} \cdot \frac{\left(\vec{g}_{grad} \times {}^{G}\hat{n}\right)}{\left\|\vec{g}_{grad} \times {}^{G}\hat{n}\right\|} + \cdots$$
$$K_{surf} \cdot (s_{des} - s_{cluster}) \cdot \vec{g}_{grad} + \cdots$$
$$K_n \cdot \left(n_{des} - P_{B\_proj}\right) \cdot {}^{G}\hat{n} \quad (14)$$

where $$P_{B\_proj} = dot\left(\vec{P}_{B\ act},\ {}^{G}\hat{n}\right) \quad (15)$$

The result is then used in (16) to generate desired velocity commands for cluster frame translational control, where S is the user specified translational speed.

$$\dot{\vec{C}}_{4R}(1{:}3) = \begin{bmatrix} \dot{X}_{B\ des} \\ \dot{Y}_{B\ des} \\ \dot{Z}_{B\ des} \end{bmatrix} = S \cdot \frac{v_{b-iso-mapping}}{\left\|v_{b-iso-mapping}\right\|} \quad (16)$$

While cluster translational velocity commands are continuously varied based on the estimate of the local field gradient, all other cluster orientation and geometry variables, designated by $\dot{\vec{C}}_{4R}(4{:}16)$, are controlled to specified position setpoints through the use of a proportional position controller. This blending of proportional velocity and position control is achieved by the resolved-rate cluster-space controller expressed by (8) given the selector matrix Q' = *diag(1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0)* and its complement Q.

For the initial execution of "State 2: Map Surface Slices", n_{des} is set to be the value of P_{B_proj} upon arrival at the surface. Once a contour is completely circumnavigated within the desired plane, the state machine resets control in the same state with a new value of n_{des} that has been incremented by the specified value of Δn.

State Transition Criteria. Formal definitions of the state machine transition criteria consist of the following. The "Transition A: Surface Reached" criteria that transitions between the first and second states is based on having $s_{cluster}$, the cluster's average scalar value, settle to within a prescribed threshold to the value s_{des}, as shown in (17).

$$\left\|s_{des} - s_{cluster}\right\| < s_{threshold} \quad (17)$$

Once in "State 2: Map Surface Slices", the "Transition B: Contour Circumnavigated" criteria is used to change the planar slice setpoint n_{des}. To implement this, the criteria first requires the cluster to settle on the desired plane to within a prescribed threshold to n_{des} as shown in (18).

$$\left\|n_{des} - P_{B-proj}\right\| < n_{threshold} \quad (18)$$

Typically, during the first execution of the "State 2: Map Surface Slices" state, this criteria is immediately true given that n_{des} is set to the value of $n_{cluster}$ at the time of the state transition. However, for subsequent executions of the "State 2: Map Surface Slices" state, there is typically a transient period since the cluster must navigate from one plane to the next through a distance of Δn. Once (17) and (18) are satisfied, the state machine checks for a full revolution of travel about the ${}^{G}\hat{n}$ vector for each contour map slice. When the magnitude of the differences between the contour map start and end positions (δ for distance and ρ for angle) are within the prescribed thresholds shown in (19) and (20), the state machine increments the planar slice setpoint n_{des} per (21).

$$\left\|\delta_{start} - \delta_{end}\right\| < \delta_{threshold} \quad (19)$$

$$\left\|\rho_{start} - \rho_{end}\right\| < \rho_{threshold} \quad (20)$$

$$n_{des\ next} = n_{des\ current} + \Delta n \quad (21)$$

For this paper, isosurface mapping continues until the cluster reaches the "end" of the surface in the given direction at which point, the estimated gradient and navigation reference vectors align with each other (other termination criteria are certainly possible). The "Transition C: Mapping Complete" termination condition is defined as the angle between the gradient $\vec{g}_{grad}$ and the navigation reference vector ${}^{G}\hat{n}$ falling below a prescribed threshold as shown in (22).

$$cos^{-1}\left(\frac{\vec{g}_{grad} \cdot {}^{G}\hat{n}}{\left\|\vec{g}_{grad}\right\| \cdot \left\|{}^{G}\hat{n}\right\|}\right) < \gamma_{threshold} \quad (22)$$

V.D ISOSURFACE MAPPING MISSION SIMULATION

In this section, execution of isosurface mapping missions using a four-robot cluster is demonstrated using the state-based strategy described in Section V.C. The technique is demonstrated in three scenarios, each "slicing" the isosurface in a different planar orientation, with those

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

planes perpendicular to the ${}^{G}\hat{X}$, ${}^{G}\hat{Y}$, and ${}^{G}\hat{Z}$ axes. The four-robot cluster was held in the shape of a tetrahedron with sides on the order of 25 to 35 m in length, cluster translational speeds of 3 to 5 m/s, and both clockwise and counterclockwise circumnavigation. Navigation occurred in workspaces with sides 1,000 m in length. The simulated scalar fields in this section are documented in Appendix D.

SCENARIO A

TABLE VI
STATE MACHINE CONTROL PARAMETERS: SCENARIO A

$S = 4\ {}^{m}/_{s}$
$s_{des} = 8$ units
${}^{G}\hat{n} = [0 \quad 0 \quad 1]^{T}$
$\Delta n = -110$ m
$d_{orbit} = +1$
$s_{threshold} = 0.25$ units
$n_{threshold} = 10$ m
$\delta_{threshold} = 50$ m
$\rho_{threshold} = 10°$
$\gamma_{threshold} = 10°$

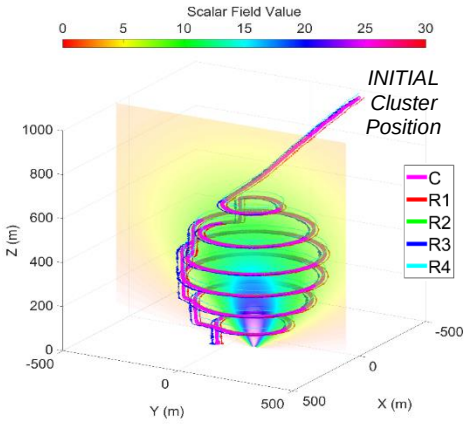


FIGURE 16. Isosurface mapping of symmetric plume with constraint planes 100 m apart and parallel to the ground.

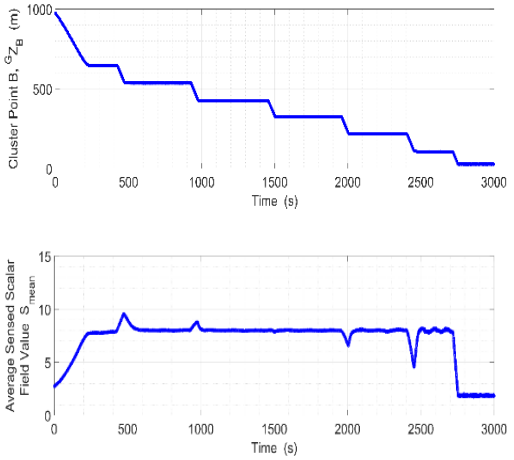


FIGURE 17. Time history of the cluster Z-coordinate and average measured scalar field level.

SCENARIO B

TABLE VII
STATE MACHINE CONTROL PARAMETERS: SCENARIO B

$S = 5\ {}^{m}/_{s}$
$s_{des} = 7.0$ units
${}^{G}\hat{n} = [1 \quad 0 \quad 0]^{T}$
$\Delta n = -100$ m
$d_{orbit} = +1$
$s_{threshold} = 0.5$ units
$n_{threshold} = 10$ m
$\delta_{threshold} = 50$ m
$\rho_{threshold} = 10°$
$\gamma_{threshold} = 10°$

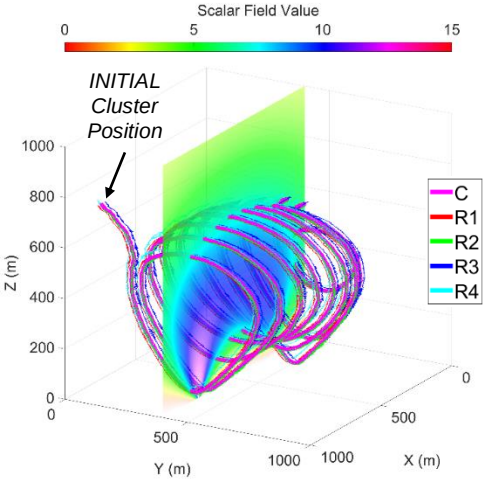


FIGURE 18. Isosurface mapping of overlapping plumes with 100m-spaced constraint along the ${}^{G}\hat{X}$ axis.

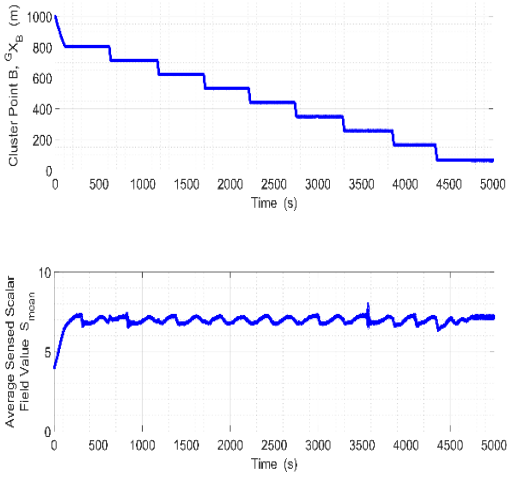


FIGURE 19. Time history of the cluster X-coordinate and average measured scalar field level.

Scenario A. Scenario A involves mapping a symmetric vertical plume using horizontal contour slices. State machine control parameters used for this scenario are summarized in Table VI. Fig. 16 shows cluster motion, which starts using the basic State 1 navigation primitive to locate the s = 8 scalar units isosurface. Upon arriving at the isosurface, n_{des} is initialized at the current value of 650 m, defining the first constraint plane perpendicular to the vertical navigation reference vector, ${}^{G}\hat{n}$. The cluster then navigates using the State 2 control law in order to circumnavigate the contour defined by the intersection of the isosurface and the constraint plane. The cluster then repeats this circumnavigation process with incremental planar offsets of Δn = -110 m until the termination condition is met.

Performance is characterized by the time histories in Fig. 17 which show a) the incremental change in altitude as isosurface slices are mapped, and b) the ability of the cluster to navigate on the desired surface, with transient deviations each time the cluster moves to a new plane. Formation control performance, as the cluster navigates, can be evaluated based on the RMS errors of the cluster

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

size parameters. RMS errors for L_{12}, L_{13}, and L_{B4} were 3.5 m, 3.5 m and 0.9 m, respectively. These results are in a range of 0.5-2.0 times the standard deviation of position error, indicating acceptable formation control given the levels of sensor noise and the size of the cluster.

Scenario B. Scenario B consists of an even more complex scalar field consisting of overlapping scalar signals. In this case, Fig. 18 shows planar slices that are spaced every 100 m and are perpendicular to the ${}^{G}\hat{X}$ axis, with a desired isosurface scalar field value of s = 7 scalar units. Table VII summarizes the state machine control parameters for this scenario. As before, time histories of the cluster's position along the ${}^{G}\hat{X}$ axis and the average measured scalar value evolve as expected, as shown in Fig. 19. RMS errors for L_{12}, L_{13}, and L_{B4} were 3.3 m, 3.0 m and 1.4 m, respectively. These results are in a range of 0.5-2.0 times the standard deviation of position error, indicating acceptable formation control given the levels of sensor noise and the size of the cluster.

VI. DOWNSTREAM PLUME FOLLOWING

Being able to follow a plume outward, away from its source, has real-world applications such as determining the impact zone from a pollution source. In this work, we implement plume following outward with a differential-based control strategy that uses synchronized and spatially relevant sensor measurements from each vehicle in the cluster. It is important to note that following a plume inwards/towards a source can be accomplished via a gradient-ascent extrema seeking primitive, such as the one presented in Section IV.C; however, following a plume outward, away from the source, is not possible using a gradient-descent algorithm.

The strategy for 3D plume following is an extension of 2D ridge descent [1], a depiction of which is shown in Fig. 20. For the 2D primitive, a ridge straddling strategy is used. In particular, assuming that the cluster is already straddling the ridge, differential scalar signals are generated by Robots 2 through 5 to position and orient the cluster with respect to the ridge.

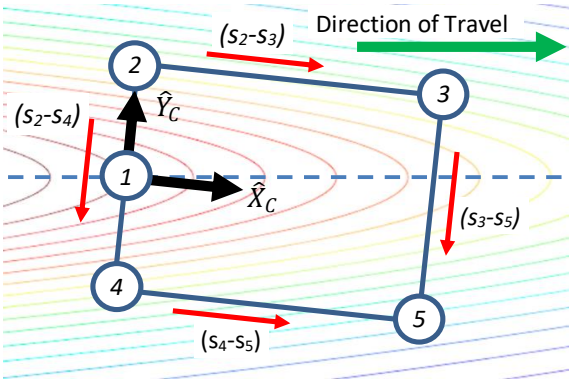


FIGURE 20. Differential drive compensation signals for ridge following in 2D with a five-robot cluster.

The differential between Robots 2 and 4 and Robots 3 and 5 are used for lateral positioning and orientation control, while the differentials between Robots 2 and 3 and Robots 4 and 5 set the direction of travel. Robot 1 is used to ensure that the cluster is, in fact, well-positioned on the ridge with the expectation that its sensed scalar value is greater than those of Robots 2 and 4.

A ridge in 2D space becomes a plume in a 3D space; a simple example of this may be visualized by rotating the contours of Fig. 20 about the dotted line in that figure. Accordingly, the planar ridge descent strategy is extended by complementing the five-robot formation in Fig. 20 with an additional formation that is oriented by rotating the formation by 90° about its $\hat{X}_C$ unit vector. Since Robot 1 in each portion of the formation is coincident, only one robot is used in this position. This approach gives rise to a nine-robot formation which, when properly positioned on a plume, "straddles" the plume in perpendicular planes. This allows AN to be implemented without requiring extraneous motions necessary to characterize the nature of the local field. The cluster is sized based on the considerations discussed in Section VII and detailed in [1]. Fig. 21 shows a simple representation of a formation properly straddling a plume.

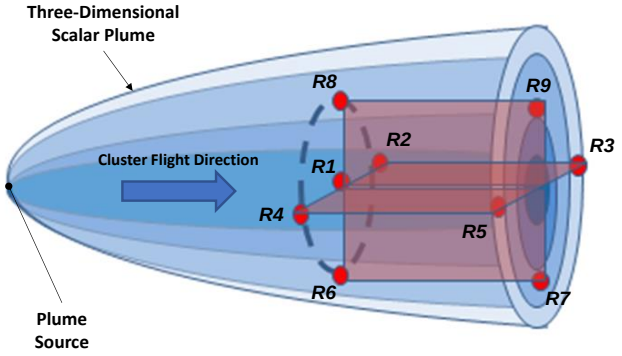


FIGURE 21. Plume following outward (i.e., ridge descent in 3D) using a nine-robot cluster in a four-sided prism formation.

Given this plume following approach, this section presents the nine-robot cluster definition and a differential-based AN control law assuming the cluster is straddling the plume. A mission-level state machine is then described to first lead the cluster to the plume and then compel the cluster to follow the plume. Finally, high fidelity simulations are used to show a nine-UAV cluster successfully navigating several plumes.

VI.A NINE-ROBOT PRISMATIC CLUSTER

The nine-robot cluster can be thought of as two intersecting, planar five-robot formations that share one of the robots, as shown in Fig. 22.

The pose of each robot in the figure is represented by its robot-space position and orientation variables, listed in Table VIII. The robot-space pose vector of the group $\vec{R}_{9R}$ contains these variables for each of the nine robots. The robot-space velocity vector $\dot{\vec{R}}_{9R}$ is the time derivative of $\vec{R}_{9R}$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

For this study, the origins of cluster frame C_{9R} and Robot 1 frame R1 are coincident and 21 shape variables are used to define the geometries of the four, two-robot serial chains that kinematically extend from Robot 1, as shown in Fig. 22 and listed in Table IX. The cluster-space position vector $\vec{C}_{9R}$ consists of the variables in Table IX and the cluster-space velocity vector $\dot{\vec{C}}_{9R}$ is the time derivative of $\vec{C}_{9R}$. A full listing of vectors $\vec{R}_{9R}$ and $\vec{C}_{9R}$ are provided in Appendix C.

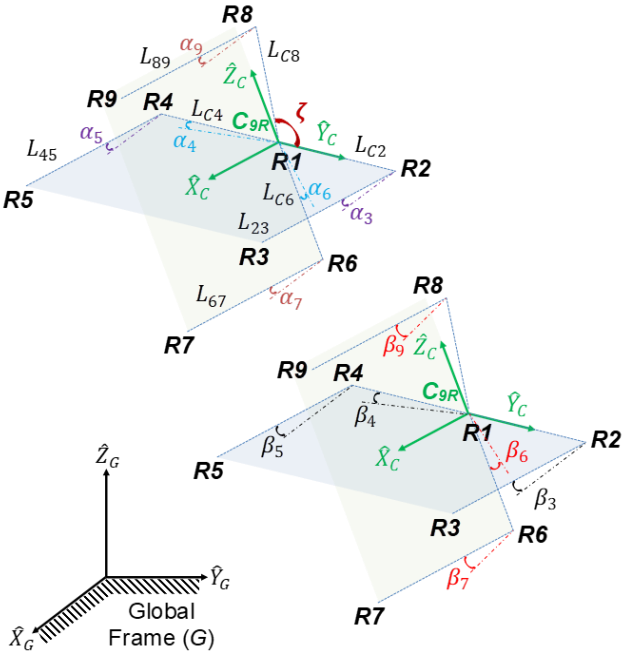


FIGURE 22. Nine-robot virtual structure lengths and internal angles, α is perpendicular to $\hat{X}_C\hat{Y}_C$ or $\hat{X}_C\hat{Z}_C$ planes and β is parallel to $\hat{X}_C\hat{Y}_C$ or $\hat{X}_C\hat{Z}_C$ planes. For clarity, the variables are shown in two separate formations.

TABLE VIII
NINE-ROBOT FORMATION ROBOT-SPACE VARIABLES

Robot positions, *n* = 1-9	${}^{G}x_n$ ${}^{G}y_n$ ${}^{G}z_n$
Robot yaw angles, *n* = 1-9	${}^{G}\psi_n$

TABLE IX
NINE-ROBOT FORMATION CLUSTER-SPACE VARIABLES

Cluster position variables	${}^{G}X_C$ ${}^{G}Y_C$ ${}^{G}Z_C$
Cluster orientation variables	${}^{G}\Phi_C$ ${}^{G}\Theta_C$ ${}^{G}\Psi_C$
Robot relative headings, *n* = 1-9	${}^{C}\Psi_n$
Cluster shape variables, distances	L_{C2} L_{23} L_{C4} L_{45} L_{C6} L_{67} L_{C8} L_{89}
Cluster shape variables, angles	α_3 α_4 α_5 α_6 α_7 α_9 β_3 β_4 β_5 β_6 β_7 β_9
Cluster shape, angle btw $\hat{Y}_C$ & L_{C8}	ζ

The forward position kinematic equations of the form described in (6) which transform "robot-space" pose vector $\vec{R}_{9R}$ to the "cluster-space" vector $\vec{C}_{9R}$ are presented in Appendix C. These equations are used in the Cluster-Space Controller layer in order to estimate realtime cluster pose, which is an input to the cluster controller. The inverse Jacobian J^{-1} is used to transform cluster control velocity commands to individual robot commands. This 36x36 matrix has not been included given its cumbersome nature.

VI.B DOWNSTREAM PLUME FOLLOWING CONTROL PRIMITIVE

Assuming the cluster is properly positioned, the plume following AN primitive is implemented by issuing corrective cluster translational and rotational control commands as shown in (23) to (26), where *S* and *R* represent user-specified translational and rotational speeds, respectively.

$$\vec{D}_{tran} = \begin{bmatrix} sign\{(s_2 - s_3) + (s_4 - s_5)\} \\ sign\{(s_2 - s_4) + (s_3 - s_5)\} \\ sign\{(s_8 - s_6) + (s_9 - s_7)\} \end{bmatrix} \quad (23)$$

$$\vec{D}_{rot} = \begin{bmatrix} 0 \\ sign\{(s_7 - s_9) - (s_6 - s_8)\} \\ sign\{(s_3 - s_5) - (s_2 - s_4)\} \end{bmatrix} \quad (24)$$

$$\dot{\vec{C}}_{9R}(1{:}3) = \begin{bmatrix} \dot{X}_{C\,des} \\ \dot{Y}_{C\,des} \\ \dot{Z}_{C\,des} \end{bmatrix} = S \cdot \frac{\vec{D}_{tran}}{\|\vec{D}_{tran}\|} \quad (25)$$

$$\dot{\vec{C}}_{9R}(4{:}6) = \begin{bmatrix} \dot{\varphi}_{C\,des} \\ \dot{\theta}_{C\,des} \\ \dot{\Psi}_{C\,des} \end{bmatrix} = R \cdot \frac{\vec{D}_{rot}}{\|\vec{D}_{rot}\|} \quad (26)$$

A proportional velocity controller is used to implement the computed cluster translational and rotational velocity commands in (25) and (26). All other cluster orientation and geometry variables are controlled to designated position setpoints. This is done using a proportional position controller. Given the nine-robot definitions of $\vec{C}_{9R}$ and $\dot{\vec{C}}_{9R}$, this blending of proportional velocity and position control is achieved within the resolved-rate cluster-space controller, as defined in (8), through the use of the selector matrix Q' = *diag(1,1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)*, with Q defined as its complement.

Because this "straddling descent" strategy is only valid if the formation is indeed, already straddling the plume, its use is only valid when Robot 1's sensed scalar value is greater than any of the sensed values from Robots 2, 4, 6, and 8. Furthermore, we typically require that Robot 1's value exceeds that of all the others by a certain threshold to ensure not only proper positioning of the cluster but also that the plume feature is "prominent" enough to be worth following. This prominence threshold criteria is formally expressed in (27).

$$(s_1 - max[s_2, s_4, s_6, s_8]) > s_{prom\ threshold} \quad (27)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

VI.C PLUME LOCATION AND DESCENT STATE MACHINE

Autonomous plume location and descent consists of finding the source of a plume and following it out to the impact area while being able to reacquire the plume should it inadvertently maneuver "off" the feature.

Execution of this mission requires the specification of a set of mission parameters, such as the minimum plume prominence value. As before, the user specifies desired constant cluster translational and rotational speeds, *S* and *R*, respectively. In general, a four-sided prismatic formation is used to obtain distributed data in all three dimensions in order to attain accurate differential signals between robots as illustrated in Fig. 23. The size of the prism is selected to appropriately straddle a plume with a particular size of interest.

Control States. The state diagram shown in Fig. 23 illustrates the plume location and descent approach that has been implemented for the results reported in this article.

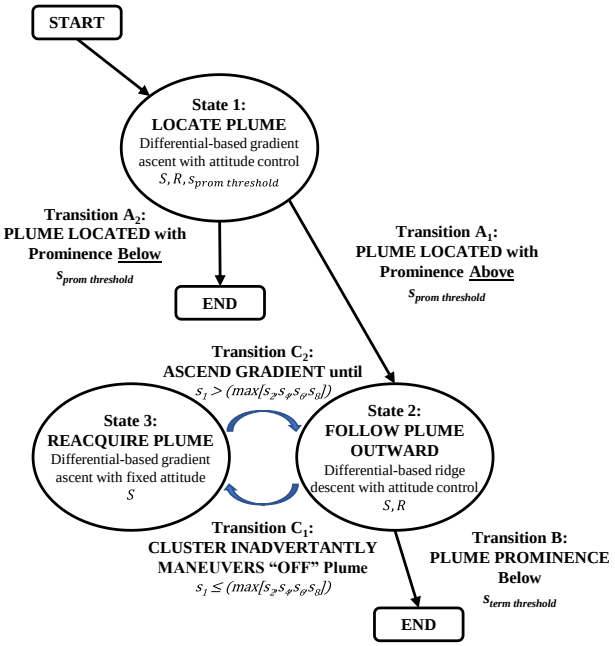


FIGURE 23. State machine for plume location and descent. Each of the three states consist of one or more AN-level controllers with specific control parameters.

The primary state in this strategy is "State 2: Follow Plume Outward", in which the cluster is properly straddling the plume and implementing the plume following AN primitive described in Section VI.B.

The first state, "State 1: Locate Plume", is used to initially move to and straddle the plume. This state uses the plume following AN primitive with the exception that the X-component in (23) is negative, which enables gradient ascent in that dimension. This strategy moves the cluster towards a local maximum while aligning the cluster with a plume stemming from that source, if one exists. If the cluster successfully locates a plume that meets the prespecified minimum prominence threshold, the state machine proceeds to "State 2: Follow Plume Outward".

The third state, "State 3: Reacquire Plume", is used if the cluster violates its prominence threshold requirement while descending the plume. This is when the cluster is still straddling and aligned with the plume, but not centered in it. The strategy we have adopted in this case is to reverse the longitudinal direction of motion, as done in "State 1: Locate Plume", by making the X-component in (23) negative and holding the cluster orientation constant. As shown in (28), cluster rotation is computed using a simple proportional controller, where K_{rot_hold} is the rotational gain, to hold the current orientation of the cluster in lieu of (24).

$$\vec{D}_{rot\_hold} = \begin{bmatrix} 0 \\ K_{rot\_hold} \cdot \left({}^{G}\Theta_{C\_hold} - {}^{G}\Theta_{C\_act} \right) \\ K_{rot\_hold} \cdot \left({}^{G}\Psi_{C\_hold} - {}^{G}\Psi_{C\_act} \right) \end{bmatrix} \quad (28)$$

State Transition Criteria. Formal definitions of the state machine transition criteria consist of the following. The criteria for "Transition A_1" that must be satisfied to move between "State 1: Locate Plume" and "State 2: Follow Plume Outward" are based on the magnitude of the differentials being below a prescribed threshold as shown in (29), the difference in the measured scalar field value from Robot 1 and the maximum from Robots 2, 4, 6, and 8 by a prescribed minimum threshold previously shown in (27), and the cluster heading alignment with the downstream direction of the plume to within a prescribed threshold as shown in (30) to (33). Note that (27) defines the minimum plume "prominence" which functions as a "go" or "no go" condition (i.e., if the plume contains a feature of interest) as to whether the cluster should proceed with following the plume outward. For compactness, (33) uses the notation C(#) and S(#) for cosine and sine functions, respectively.

$$\left\| \begin{bmatrix} \left(\frac{s_3+s_5}{2}\right) - \left(\frac{s_2+s_4}{2}\right) \\ \left(\frac{s_2+s_3}{2}\right) - \left(\frac{s_4+s_5}{2}\right) \\ \left(\frac{s_8+s_9}{2}\right) - \left(\frac{s_6+s_7}{2}\right) \end{bmatrix} \right\| < \Delta_{threshold} \quad (29)$$

$$\left\| {}^{G}\Psi_{C\,des} - {}^{G}\Psi_{C\,act} \right\| < \Psi_{C\,threshold} \quad (30)$$

where $${}^{G}\Psi_{C\,des} = ATAN2\left(\frac{-D_{y\,diff}}{-D_{x\,diff}}\right) \quad (31)$$

$$\vec{D}_{diff} = \begin{bmatrix} D_{x\,diff} \\ D_{y\,diff} \\ D_{z\,diff} \end{bmatrix} = {}^{G}_{C}R \cdot \begin{bmatrix} (s_3 - s_2) + (s_5 - s_4) \\ (s_2 - s_4) + (s_3 - s_5) \\ (s_8 - s_6) + (s_9 - s_7) \end{bmatrix} \quad (32)$$

$${}^{G}_{C}R = \begin{bmatrix} C\Psi_C C\Theta_C & C\Psi_C S\Theta_C S\Phi_C - C\Phi_C S\Psi_C & C\Phi_C S\Theta_C C\Psi_{C_C} + S\Psi_C S\Phi_C \\ S\Psi_C C\Theta_C & S\Phi_C S\Theta_C S\Psi_C + C\Psi_C C\Phi_C & S\Psi_C S\Theta_C C\Phi_C - S\Phi_C C\Psi_C \\ -S\Theta_C & C\Theta_C S\Phi_C & C\Theta_C C\Phi_C \end{bmatrix} \quad (33)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

While in "State 2: Follow Plume Outward", in the event that the cluster inadvertently maneuvers "off" the feature, the state machine "Transition C_1" criteria to move to "State 3: Reacquire Plume" is shown in (34).

$$s_1 \leq (max[s_2, s_4, s_6, s_8]) \tag{34}$$

While in "State 3: Reacquire Plume", state machine "Transition C_2" criteria to move back to "State 2: Follow Plume Outward" is shown in (35).

$$s_1 > (max[s_2, s_4, s_6, s_8]) \tag{35}$$

Following the plume outward continues until the cluster reaches a point where the plume prominence falls below a prescribed threshold with "Transition B" criteria shown in (36).

$$s_{term\ threshold} \leq (max[s_2, s_4, s_6, s_8] - s_1) \tag{36}$$

VI.D DOWNSTREAM PLUME FOLLOWING MISSION SIMULATION

In this section, we demonstrate execution of two mission scenarios to locate and descend a plume using a nine-robot cluster using the state-based strategy described in Section VI.C. As previously discussed, the objective is to have the cluster locate an unknown plume with a minimum prominence value, move along it in the direction away from the source, reacquire the plume if it inadvertently maneuvers "off" the feature, and to hold its position and attitude when it reaches a point where the prominence is below a prescribed threshold.

In the first scenario, we utilize a simulated plume with a "tail" of several hundred meters, a constant commanded cluster geometry held in the shape of a rectangular prism with shape variables shown in Table X, and constant cluster translational and rotational speeds. State machine control parameters for this simulation are summarized in Table XI.

TABLE X
NINE-ROBOT FORMATION SHAPE VARIABLES

$L_{C2},\ L_{23},\ L_{C4},\ L_{45}$	35 m
$L_{C6},\ L_{67},\ L_{C8},\ L_{89}$	35 m
ζ	90°
$\alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_9$	0°
$\beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_9$	0°

TABLE XI
STATE MACHINE CONTROL PARAMETERS

Cluster Translational Speed	$S = 5\ {}^{m}/_{s}$
Cluster Rotational Speed	$R = 5\ {}^{deg}/_{s}$
Plume Prominence Threshold (Start)	$s_{prom\ threshold} = 20$ units
Differentials Threshold (plume location)	$\Delta_{threshold} = 10$ units
Cluster Heading Alignment Threshold (prior to following plume outward)	$\Psi_{C\ threshold} = 15°$
Plume Prominence Threshold (End)	$s_{term\ threshold} = 2$ units

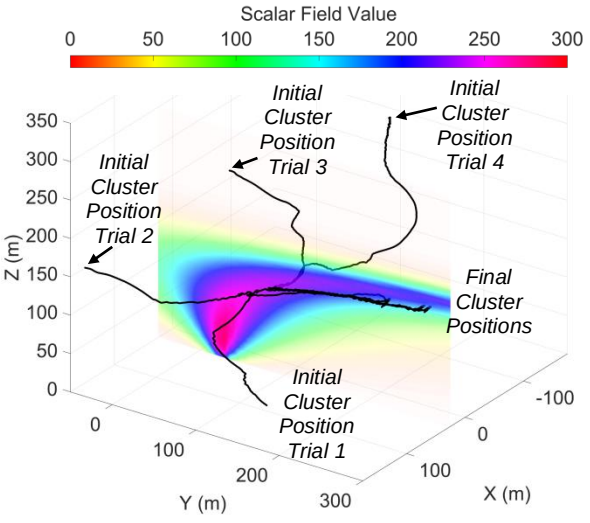
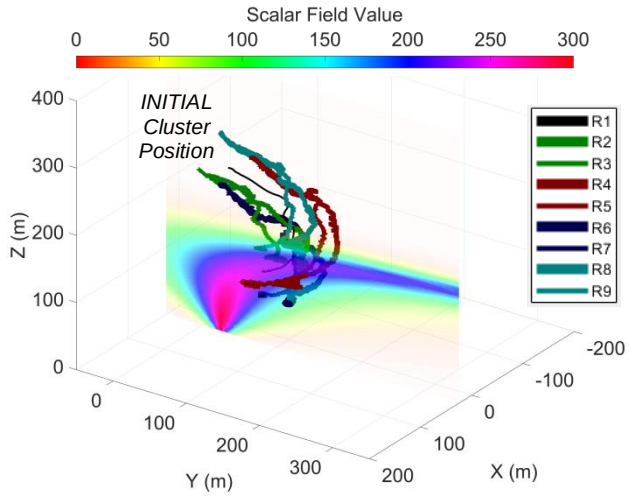


FIGURE 24. Multiple simulation trials showing flight paths of nine-robot clusters that locate and then follow a scalar plume.

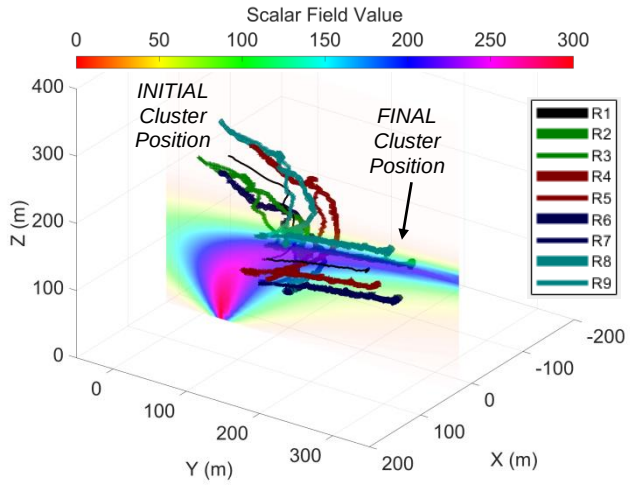
Fig. 24 illustrates the cluster flight paths for four different trials, where each has a different starting point. For clarity, only the cluster origin positions are shown for each case. As shown in the figure, in each trial, the cluster successfully navigates to and then moves along the plume in the direction away from the source.

To provide a more detailed description of functionality, the motion of trial 3 is detailed in the multipart Fig. 25, which shows the motion of each individual UAV in the cluster. In part (a), the cluster operates in the "Locate Plume" control state, moving to and straddling the plume, a process that takes the first 99 s. The controller then switches to the "Follow Plume Outward" state, shown in part (b), where the cluster successfully moves down the plume until t = 163 s, at which point the plume loses its definition per the prominence threshold. Fig. 26 contains time histories of the measured scalar field values from the robots located in the cluster's "back" plane which includes Robots 1, 2, 4, 6, and 8. These time histories show several indications that the cluster is properly aligned once the cluster locates and straddles the plume at t = 99 s. First, Robot 1's scalar reading is higher than all other robots in that plane. Second, the scalar values of the robot pairs 2-4 and 6-8 converge, indicating that the controller has successfully nulled the differential values in each of the cluster's dimensions. Although not shown, similar results exist for the lateral positioning of the robots in the "front" plane as well as for the rotational differentials used to align the cluster with the plume.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access



a) Flight path after 99 s



b) Flight path after 180 s

FIGURE 25. Flight path of a nine-robot cluster as it a) locates a plume, and then b) descends/follows the plume away from the source.

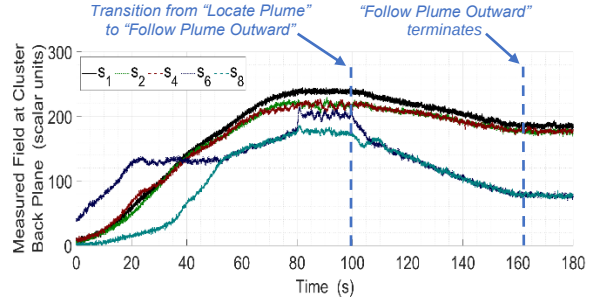


FIGURE 26. Time histories of measured scalar field values from robots in the cluster's back plane show that s_1 (Robot 1) remains higher, than the values measured by the surrounding robots, while following the plume outward and away from the source.

Furthermore, given that the controller is working to hold formation geometry as the cluster navigates, control performance is indicated by the RMS errors for the cluster size parameters L_{C2}, L_{23}, L_{C4}, L_{45}, L_{C6}, L_{67}, L_{C8}, L_{89}, which were 3.2 m, 3.0 m, 3.2 m, 2.9 m, 2.7 m, 3.1 m, 2.6 m, 3.2 m, respectively; these are all in a range of 1.5-2.0 times the standard deviation of position error, indicating acceptable formation control.

For a second simulation scenario, a plume with a different downwind direction and higher scalar field magnitude is used.

TABLE XII
NINE-ROBOT FORMATION SHAPE VARIABLES

L_{C2}, L_{23}, L_{C4}, L_{45}	25 m
L_{C6}, L_{67}, L_{C8}, L_{89}	25 m
ζ	90°
$\alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7, \alpha_9$	0°
$\beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_9$	0°

TABLE XIII
STATE MACHINE CONTROL PARAMETERS

Cluster Translational Speed	$S = 5\ ^{\mathrm{m}}/_{\mathrm{s}}$
Cluster Rotational Speed	$R = 5\ ^{\mathrm{deg}}/_{\mathrm{s}}$
Plume Prominence Threshold (Start)	$s_{prom\ threshold} = 20$ units
Differentials Threshold (plume location)	$\Delta_{threshold} = 15$ units
Cluster Heading Alignment Threshold	$\Psi_{C\ threshold} = 15°$
Plume Prominence Threshold (End)	$s_{term\ threshold} = 3$ units

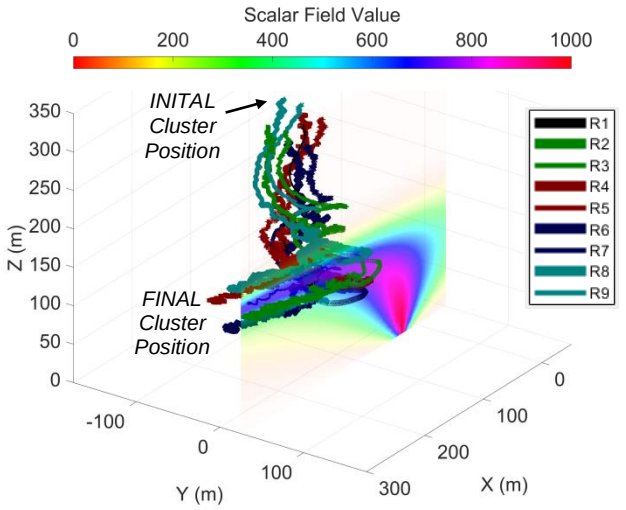


FIGURE 27. Flight path of nine-robot cluster completing the mission to first a) locate a plume and then b) descend/follow the plume outward and away from the source.

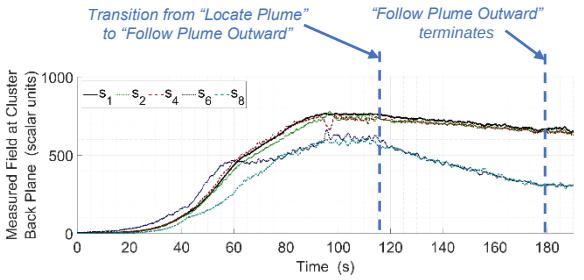


FIGURE 28. Time histories of measured scalar field values from robots in the cluster's back plane show that s_1 (Robot 1) remains higher, than the values measured by the surrounding robots, while following the plume outward and away from the source.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

A constant rectangular prism cluster geometry is used once again with shape variables shown in Table XII. State machine control parameters used in this simulation are summarized in Table XIII. Fig. 27 shows the cluster as it first operates in the "Locate Plume" state, moving toward and aligning itself with the plume, which occurs at about t = 114 s. The cluster then switches to the "Follow Plume Outward" state until the plume loses prominence at t = 179 s. As with the previous scenario, time histories of the scalar values of the back plane robots are shown in Fig. 28.

When following the plume, the values are as expected, with Robot 1 having the highest scalar value and with values converging for robot pairs 2-4 and 6-8. RMS errors for the cluster size parameters L_{C2}, L_{23}, L_{C4}, L_{45}, L_{C6}, L_{67}, L_{C8}, L_{89} were 3.2 m, 3.8 m, 3.3 m, 4.1 m, 2.8 m, 3.1 m, 2.7 m, 3.1 m, respectively, are all in a range of 1.5-2.0 times the standard deviation of position error, indicating acceptable formation control. The simulated scalar fields in this section are documented in Appendix D.

VII. DISCUSSION

Several implementation issues pertaining to the use of cluster-space and AN control methodologies have been explored in detail in other publications, but are worthy of mention here.

Within the cluster-space formation control methodology, there is significant flexibility in how a cluster may be defined. These choices influence the degree of interdependency among the cluster pose state variables, which in turn, affects issues such as the amount of computation in the servo loop, the existence of singularities, and the level of (de)centralization [49]. For singularities, as an example, several approaches have been successfully demonstrated, such as dynamically switching to new cluster pose definitions and the use of dual quaternions [50]. Regarding (de)centralization, while the examples used for this study were implemented in a centralized manner, decentralized cluster definitions at varying levels are possible, to include the ability to implement swarm-like capabilities. However, the technique includes managerial overhead (like the explicit enumeration of vehicles) that is unnecessary if there is no desire to take advantage of cluster-space control benefits (such as explicit formation control, well-behaved motion in the cluster-space, etc.). In separate work, we are exploring formalized swarm approaches that support the comprehensive AN capabilities being explored in this paper.

With respect to AN, a significant issue is selecting the size of the cluster given that it drives AN performance by trading aperture size with the spatial resolution of the cluster. Given the field characterization techniques (local gradient, differential over a baseline, etc.) used by the AN primitives, spatial filtering occurs [1]. For example, the gradient approach used in isosurface mapping assumes that the size of the cluster is "small" compared to the spatial variation of interest for the field (e.g., perhaps at most ¼ of the spatial wavelength of the scalar features of interest). Scalar features with smaller wavelengths are spatially smoothed, thus decreasing the responsiveness of AN motion. Because the cluster size is a specifiable mission attribute, the operator may select a size suitable for the application at hand, typically trading spatial resolution, gradient/differential signal amplification, noise suppression, and the breadth of field exploration for a given number of robots. As a practical example, previous work in using a cluster of automated boats to find large scale bathymetric scalar features used cluster sizes in the range of 10-20 m [63]. This allowed for navigation with respect to geologic features on the order of 50+ meters, while effectively filtering out "noise" created by small rocks and boulders.

Regarding edge cases, two cases are particularly noteworthy. The first is encountering singular formations within the cluster-space controller. As previously mentioned, however, we have developed a variety of ways to address this issue. A second issue arises for "flat" or uniform scalar fields such that there is not enough variation in the field for the AN primitives to react in a meaningful way. In separate work, we use an additional state at the mission layer that reacts to this condition by using a more conventional navigation strategy, such as "mowing the lawn," until a significant signal is observed. Identifying and addressing additional edge cases is the subject of future work.

VIII. CONCLUSIONS AND FUTURE WORK

This paper presents a multilayer control architecture for implementing multirobot AN capabilities for 3D scalar fields using UAVs. Verification of these capabilities is provided through high fidelity simulations that incorporate real vehicle flight dynamics, wind gust disturbances, vehicle position sensor inaccuracy, and scalar field sensor noise. Specific scenarios demonstrating these capabilities include:

- seeking/tracking a local source, both stationary as well as moving within a time-varying field;
- acquisition of a desired isosurface within a field and then performing structured mapping of single and overlapping signals, with an option for the orientation and spacing of planar slices;
- acquisition and following of a plume outward (i.e., away from a source).

Building on our group's prior work in multirobot AN, the contributions presented in this article significantly extend the current state of the art in terms of AN functionality for 3D scalar fields. Distinct aspects of the presented work include:

- the use of a unified control architecture to demonstrate a wide range of AN missions and control primitives for clusters of varying sizes and shapes;
- the kinematic definition of new four- and nine-robot clusters appropriate for the AN tasks of interest;
- the extension of our existing 2D multirobot AN primitives to 3D fields in order to implement local extrema finding, isosurface navigation, and downstream plume following;

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

- the use of state-based sequencing of AN primitives and control parameters to implement mission-level capabilities such as characterizing the structure of isosurfaces and acquiring and following plumes;
- the verification of the proposed AN techniques via high-fidelity simulations that include vehicle-level dynamics, wind gusts, and sensor noise.

This work lays the foundation for experimental demonstration and verification, which will begin with a cluster of four octocopter vehicles. This system uses commercial autopilots on each UAV in combination with a flight control station that allows pilots to a) fly individual vehicles, b) gracefully constitute and decompose clusters, and c) implement the full AN control architecture which is implemented via Matlab/Simulink [64]. Initial scalar fields based on RF signal strength are being implemented using Xbee 2.4 GHz transmitters. Longer term, we will apply the new 3D AN capabilities presented in this paper to larger clusters as well as to our fleet of underwater robots.

Beyond experimental verification, our ongoing and future work in multirobot AN has several objectives. We are certainly interested in exploring the benefits and costs of larger-sized clusters and more sophisticated control laws. We are also interested in formally characterizing how cluster shape and size affect navigation performance given the nature of a scalar field, exploring adaptive cluster sizing to tune cluster responsiveness to features of various sizes, and extending techniques for periodic sampling, navigation in turbulent and discontinuous fields, etc. Each of these is motivated by the needs of clients with specific AN missions in mind. We have also initiated the development of AN controllers based on decentralized swarm controllers, currently with a focus on 2D fields.

APPENDIX A
UAV DYNAMIC MODEL

[53] initially proposed a simplified dynamic model for a consumer-grade UAV, with an onboard autopilot for platform stabilization with limited pitch and roll angles, in lieu of a full quadrotor dynamic model. Using Parrot Inc's AR.Drone, model parameters were determined by measuring responses for various input signals. They used the simplified model and experimentally demonstrated precise positioning and trajectory tracking in 3D. [52] extended the work to experimentally demonstrate leader-follower positioning and trajectory tracking with two vehicles in 3D. [51] validated the dynamic model through comprehensive experiments for parameter identification and performed precise trajectory tracking with obstacle avoidance in 3D using a single vehicle. Equations (A1) and (A2) contain the K_u and K_v matrices from [51] which are used in this paper.

$$K_u = \begin{bmatrix} 12.63 & 0 & 0 & 0 \\ 0 & 7.61 & 0 & 0 \\ 0 & 0 & 6.63 & 0 \\ 0 & 0 & 0 & 1.89 \end{bmatrix} \quad \text{(A1)}$$

$$K_v = \begin{bmatrix} 1.43 & 0 & 0 & 0 \\ 0 & 0.84 & 0 & 0 \\ 0 & 0 & 7.56 & 0 \\ 0 & 0 & 0 & 0.54 \end{bmatrix} \quad \text{(A2)}$$

APPENDIX B
FORWARD KINEMATICS FOR FOUR-ROBOT CLUSTER

Equations (B1) and (B2) define the cluster-space vector $\vec{C}_{4R}$ and robot-space vector $\vec{R}_{4R}$, respectively, for the four-robot cluster.

$$\vec{C}_{4R} = \begin{bmatrix} {}^{G}X_B & cluster\ X\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}Y_B & cluster\ Y\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}Z_B & cluster\ Z\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}\Phi_C & cluster\ Roll\ angle\ w.r.t.\ global\ frame \\ {}^{G}\Theta_C & cluster\ Pitch\ angle\ w.r.t.\ global\ frame \\ {}^{G}\Psi_C & cluster\ Yaw\ angle\ w.r.t.\ global\ frame \\ L_{12} & cluster\ length\ from\ Robot\ 1\ and\ 2 \\ L_{13} & cluster\ length\ from\ Robot\ 1\ and\ 3 \\ L_{B4} & cluster\ length\ from\ Point\ B\ and\ Robot\ 4 \\ \alpha & cluster\ shape\ angle\ for\ L_{B4} \\ \beta & cluster\ shape\ angle\ between\ L_{12}\ and\ L_{13} \\ \xi & cluster\ shape\ angle\ for\ L_{B4} \\ {}^{C}\Psi_1 & Robot\ 1\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ {}^{C}\Psi_2 & Robot\ 2\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ {}^{C}\Psi_3 & Robot\ 3\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ {}^{C}\Psi_4 & Robot\ 4\ Yaw\ angle\ w.r.t.\ cluster\ frame \end{bmatrix} \quad \text{(B1)}$$

$$\vec{R}_{4R} = \begin{bmatrix} {}^{G}x_1 & Robot\ 1\ X\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}y_1 & Robot\ 1\ Y\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}z_1 & Robot\ 1\ Z\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}\psi_1 & Robot\ 1\ Yaw\ angle\ w.r.t.\ global\ frame \\ {}^{G}x_2 & Robot\ 2\ X\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}y_2 & Robot\ 2\ Y\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}z_2 & Robot\ 2\ Z\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}\psi_2 & Robot\ 2\ Yaw\ angle\ w.r.t.\ global\ frame \\ {}^{G}x_3 & Robot\ 3\ X\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}y_3 & Robot\ 3\ Y\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}z_3 & Robot\ 3\ Z\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}\psi_3 & Robot\ 3\ Yaw\ angle\ w.r.t.\ global\ frame \\ {}^{G}x_4 & Robot\ 4\ X\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}y_4 & Robot\ 4\ Y\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}z_4 & Robot\ 4\ Z\ coordinate\ w.r.t.\ global\ frame \\ {}^{G}\psi_4 & Robot\ 4\ Yaw\ angle\ w.r.t.\ global\ frame \end{bmatrix} \quad \text{(B2)}$$

Equations (B3) through (B12) are used to compute the cluster frame coordinates and attitude angles.

$${}^{G}X_B = \left({}^{G}x_1 + {}^{G}x_2 + {}^{G}x_3\right)/3 \quad \text{(B3)}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

$$ {}^{G}Y_B = \left({}^{G}y_1 + {}^{G}y_2 + {}^{G}y_3 \right)/3 \quad \text{(B4)} $$

$$ {}^{G}Z_B = \left({}^{G}z_1 + {}^{G}z_2 + {}^{G}z_3 \right)/3 \quad \text{(B5)} $$

$$ {}^{G}\Psi_C = ATAN2\left(\frac{{}^{G}y_1 - {}^{G}Y_B}{{}^{G}x_1 - {}^{G}X_B} \right) \quad \text{(B6)} $$

$$ {}^{G}\Theta_C = sin^{-1}\left(- \frac{\left({}^{G}z_1 - {}^{G}Z_B \right)}{\sqrt{\left({}^{G}x_1 - {}^{G}X_B \right)^2 + \left({}^{G}y_1 - {}^{G}Y_B \right)^2 + \left({}^{G}z_1 - {}^{G}Z_B \right)^2}} \right) \quad \text{(B7)} $$

$$ {}^{G}\Phi_C = ATAN2\left(\frac{{}^{G}_{y}X_C \cdot a_1 - {}^{G}_{x}X_C \cdot a_2}{a_3} \right) \quad \text{(B8)} $$

where

$$ \begin{bmatrix} {}^{G}_{x}X_C \\ {}^{G}_{y}X_C \\ {}^{G}_{z}X_C \end{bmatrix} = \begin{bmatrix} \frac{\left({}^{G}x_1 - {}^{G}X_B \right)}{\sqrt{\left({}^{G}x_1 - {}^{G}X_B \right)^2 + \left({}^{G}y_1 - {}^{G}Y_B \right)^2 + \left({}^{G}z_1 - {}^{G}Z_B \right)^2}} \\ \frac{\left({}^{G}y_1 - {}^{G}Y_B \right)}{\sqrt{\left({}^{G}x_1 - {}^{G}X_B \right)^2 + \left({}^{G}y_1 - {}^{G}Y_B \right)^2 + \left({}^{G}z_1 - {}^{G}Z_B \right)^2}} \\ \frac{\left({}^{G}z_1 - {}^{G}Z_B \right)}{\sqrt{\left({}^{G}x_1 - {}^{G}X_B \right)^2 + \left({}^{G}y_1 - {}^{G}Y_B \right)^2 + \left({}^{G}z_1 - {}^{G}Z_B \right)^2}} \end{bmatrix} \quad \text{(B9)} $$

$$ a_1 = \left[\left({}^{G}y_2 - {}^{G}y_1 \right)\left({}^{G}z_3 - {}^{G}z_1 \right) - \left({}^{G}y_3 - {}^{G}y_1 \right)\left({}^{G}z_2 - {}^{G}z_1 \right) \right] \quad \text{(B10)} $$

$$ a_2 = \left[\left({}^{G}x_3 - {}^{G}x_1 \right)\left({}^{G}z_2 - {}^{G}z_1 \right) - \left({}^{G}x_2 - {}^{G}x_1 \right)\left({}^{G}z_3 - {}^{G}z_1 \right) \right] \quad \text{(B11)} $$

$$ a_3 = \left[\left({}^{G}x_2 - {}^{G}x_1 \right)\left({}^{G}y_3 - {}^{G}y_1 \right) - \left({}^{G}x_3 - {}^{G}x_1 \right)\left({}^{G}y_2 - {}^{G}y_1 \right) \right] \quad \text{(B12)} $$

Equations (B13) through (B15) are used to compute the cluster formation lengths.

$$ L_{12} = \sqrt{\left({}^{G}x_2 - {}^{G}x_1 \right)^2 + \left({}^{G}y_2 - {}^{G}y_1 \right)^2 + \left({}^{G}z_2 - {}^{G}z_1 \right)^2} \quad \text{(B13)} $$

$$ L_{13} = \sqrt{\left({}^{G}x_3 - {}^{G}x_1 \right)^2 + \left({}^{G}y_3 - {}^{G}y_1 \right)^2 + \left({}^{G}z_3 - {}^{G}z_1 \right)^2} \quad \text{(B14)} $$

$$ L_{B4} = \sqrt{\left({}^{G}x_4 - {}^{G}X_B \right)^2 + \left({}^{G}y_4 - {}^{G}Y_B \right)^2 + \left({}^{G}z_4 - {}^{G}Z_B \right)^2} \quad \text{(B15)} $$

Equations (B16) through (B20) are used to compute the internal angles of the cluster formation.

$$ \alpha = ATAN2\left({}^{C}_{y}L_{B4} / {}^{C}_{x}L_{B4} \right) \quad \text{(B16)} $$

$$ \beta = cos^{-1}\left(\frac{\left({}^{G}x_2 - {}^{G}x_1 \right)\left({}^{G}x_3 - {}^{G}x_1 \right) + \left({}^{G}y_2 - {}^{G}y_1 \right)\left({}^{G}y_3 - {}^{G}y_1 \right) + \left({}^{G}z_2 - {}^{G}z_1 \right)\left({}^{G}z_3 - {}^{G}z_1 \right)}{L_{12} * L_{13}} \right) \quad \text{(B17)} $$

$$ \xi = ATAN2\left(\sqrt{{}^{C}_{x}L_{B4}{}^2 + {}^{C}_{y}L_{B4}{}^2} \Big/ {}^{C}_{z}L_{B4} \right) \quad \text{(B18)} $$

where

$$ \begin{bmatrix} {}^{C}_{x}L_{B4} \\ {}^{C}_{y}L_{B4} \\ {}^{C}_{z}L_{B4} \end{bmatrix} = {}^{C}_{G}R \cdot \begin{bmatrix} \left({}^{G}x_4 - {}^{G}X_B \right) \\ \left({}^{G}y_4 - {}^{G}Y_B \right) \\ \left({}^{G}z_4 - {}^{G}Z_B \right) \end{bmatrix} \quad \text{(B19)} $$

$$ {}^{C}_{G}R = \begin{bmatrix} C\Psi_C C\Theta_C & S\Psi_C C\Theta_C & -S\Theta_C \\ C\Psi_C S\Theta_C S\Phi_C - C\Phi_C S\Psi_C & S\Phi_C S\Theta_C S\Psi_C + C\Psi_C C\Phi_C & C\Theta_C S\Phi_C \\ C\Phi_C S\Theta_C C\Psi_C + S\Psi_C S\Phi_C & S\Psi_C S\Theta_C C\Phi_C - S\Phi_C C\Psi_C & C\Theta_C C\Phi_C \end{bmatrix} \quad \text{(B20)} $$

For compactness, (B20) uses the notation C(#) and S(#) for cosine and sine functions, respectively. Equations (B21) through (B23) are used to compute individual robot yaw angles relative to frame C, for Robots 1 to 4 where n = 1,…,4.

$$ {}^{C}\Psi_n = ATAN2\left[{}^{C}_{y}X_n / {}^{C}_{x}X_n \right] \quad \text{(B21)} $$

where

$$ {}^{G}_{Rn}R = \begin{bmatrix} {}^{C}_{x}X_n & {}^{C}_{x}Y_n & {}^{C}_{x}Z_n \\ {}^{C}_{y}X_n & {}^{C}_{y}Y_n & {}^{C}_{y}Z_n \\ {}^{C}_{z}X_n & {}^{C}_{z}Y_n & {}^{C}_{z}Z_n \end{bmatrix} = {}^{C}_{G}R \cdot {}^{G}_{Rn}R \quad \text{(B22)} $$

$$ {}^{G}_{Rn}R = \begin{bmatrix} C\psi_n C\theta_n & C\psi_n S\theta_n S\varphi_n - C\varphi_n S\psi_n & C\varphi_n S\theta_n C\psi_n + S\psi_n S\varphi_n \\ S\psi_n C\theta_n & S\varphi_n S\theta_n S\psi_n + C\psi_n C\varphi_n & S\psi_n S\theta_n C\varphi_n - S\varphi_n C\psi_n \\ -S\theta_n & C\theta_n S\varphi_n & C\theta_n C\varphi_n \end{bmatrix} \quad \text{(B23)} $$

APPENDIX C
FORWARD KINEMATICS FOR NINE-ROBOT CLUSTER

Equations (C1) and (C2) define the cluster-space vector $\vec{C}_{9R}$ and robot-space vector $\vec{R}_{9R}$, respectively, for the nine-robot cluster. Equations (C3) through (C10) are used to compute the cluster frame coordinates and attitude angles.

$$ \vec{C}_{9R} = \begin{bmatrix} {}^{G}X_C \\ {}^{G}Y_C \\ {}^{G}Z_C \\ {}^{G}\Phi_C \\ {}^{G}\Theta_C \\ {}^{G}\Psi_C \\ L_{C2} \\ L_{23} \\ L_{C4} \\ L_{45} \\ L_{C6} \\ L_{67} \\ L_{C8} \\ L_{89} \\ \zeta \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_9 \\ \beta_3 \\ \beta_4 \\ \beta_5 \\ \beta_6 \\ \beta_7 \\ \beta_9 \\ {}^{C}\Psi_1 \\ {}^{C}\Psi_2 \\ {}^{C}\Psi_3 \\ {}^{C}\Psi_4 \\ {}^{C}\Psi_5 \\ {}^{C}\Psi_6 \\ {}^{C}\Psi_7 \\ {}^{C}\Psi_8 \\ {}^{C}\Psi_9 \end{bmatrix} \begin{array}{l} cluster\ X\ coordinate\ w.r.t.\ global\ frame \\ cluster\ Y\ coordinate\ w.r.t.\ global\ frame \\ cluster\ Z\ coordinate\ w.r.t.\ global\ frame \\ cluster\ Roll\ angle\ w.r.t.\ global\ frame \\ cluster\ Pitch\ angle\ w.r.t.\ global\ frame \\ cluster\ Yaw\ angle\ w.r.t.\ global\ frame \\ cluster\ length\ from\ Point\ C\ and\ Robot\ 2 \\ cluster\ length\ from\ Robot\ 2\ and\ Robot\ 3 \\ cluster\ length\ from\ Point\ C\ and\ Robot\ 4 \\ cluster\ length\ from\ Robot\ 4\ and\ Robot\ 5 \\ cluster\ length\ from\ Point\ C\ and\ Robot\ 6 \\ cluster\ length\ from\ Robot\ 6\ and\ Robot\ 7 \\ cluster\ length\ from\ Point\ C\ and\ Robot\ 8 \\ cluster\ length\ from\ Robot\ 8\ and\ Robot\ 9 \\ cluster\ shape\ angle\ between\ \hat{Y}_C\ and\ L_{C8} \\ cluster\ shape\ angle\ \perp\ to\ \hat{X}_C\hat{Y}_C\ plane \\ cluster\ shape\ angle\ \perp\ to\ \hat{X}_C\hat{Y}_C\ plane \\ cluster\ shape\ angle\ \perp\ to\ \hat{X}_C\hat{Y}_C\ plane \\ cluster\ shape\ angle\ \perp\ to\ \hat{X}_C\hat{Z}_C\ plane \\ cluster\ shape\ angle\ \perp\ to\ \hat{X}_C\hat{Z}_C\ plane \\ cluster\ shape\ angle\ \perp\ to\ \hat{X}_C\hat{Z}_C\ plane \\ cluster\ shape\ angle\ \parallel\ to\ \hat{X}_C\hat{Y}_C\ plane \\ cluster\ shape\ angle\ \parallel\ to\ \hat{X}_C\hat{Y}_C\ plane \\ cluster\ shape\ angle\ \parallel\ to\ \hat{X}_C\hat{Y}_C\ plane \\ cluster\ shape\ angle\ \parallel\ to\ \hat{X}_C\hat{Z}_C\ plane \\ cluster\ shape\ angle\ \parallel\ to\ \hat{X}_C\hat{Z}_C\ plane \\ cluster\ shape\ angle\ \parallel\ to\ \hat{X}_C\hat{Z}_C\ plane \\ Robot\ 1\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 2\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 3\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 4\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 5\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 6\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 7\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 8\ Yaw\ angle\ w.r.t.\ cluster\ frame \\ Robot\ 9\ Yaw\ angle\ w.r.t.\ cluster\ frame \end{array} \quad \text{(C1)} $$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

$$\vec{R}_{9R} = \begin{bmatrix} {}^{G}x_1 \\ {}^{G}y_1 \\ {}^{G}z_1 \\ {}^{G}\psi_1 \\ {}^{G}x_2 \\ {}^{G}y_2 \\ {}^{G}z_2 \\ {}^{G}\psi_2 \\ {}^{G}x_3 \\ {}^{G}y_3 \\ {}^{G}z_3 \\ {}^{G}\psi_3 \\ {}^{G}x_4 \\ {}^{G}y_4 \\ {}^{G}z_4 \\ {}^{G}\psi_4 \\ {}^{G}x_5 \\ {}^{G}y_5 \\ {}^{G}z_5 \\ {}^{G}\psi_5 \\ {}^{G}x_6 \\ {}^{G}y_6 \\ {}^{G}z_6 \\ {}^{G}\psi_6 \\ {}^{G}x_7 \\ {}^{G}y_7 \\ {}^{G}z_7 \\ {}^{G}\psi_7 \\ {}^{G}x_8 \\ {}^{G}y_8 \\ {}^{G}z_8 \\ {}^{G}\psi_8 \\ {}^{G}x_9 \\ {}^{G}y_9 \\ {}^{G}z_9 \\ {}^{G}\psi_9 \end{bmatrix} \begin{matrix} Robot\ 1\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 1\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 1\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 1\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 2\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 2\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 2\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 2\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 3\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 3\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 3\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 3\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 4\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 4\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 4\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 4\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 5\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 5\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 5\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 5\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 6\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 6\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 6\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 6\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 7\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 7\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 7\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 7\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 8\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 8\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 8\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 8\ Yaw\ angle\ w.r.t.\ global\ frame \\ Robot\ 9\ X\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 9\ Y\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 9\ Z\ coordinate\ w.r.t.\ global\ frame \\ Robot\ 9\ Yaw\ angle\ w.r.t.\ global\ frame \end{matrix} \quad \text{(C2)}$$

$$\begin{bmatrix} {}^{G}X_C & {}^{G}Y_C & {}^{G}Z_C \end{bmatrix}^T = \begin{bmatrix} {}^{G}x_1 & {}^{G}y_1 & {}^{G}z_1 \end{bmatrix}^T \quad \text{(C3)}$$

$${}^{G}\Psi_C = \cdots$$

$$ATAN2\left(\frac{-{}^{G}x_1\,{}^{G}z_2 + {}^{G}x_2\,{}^{G}z_1 + {}^{G}x_1\,{}^{G}z_8 - {}^{G}x_8\,{}^{G}z_1 - {}^{G}x_2\,{}^{G}z_8 + {}^{G}x_8\,{}^{G}z_2}{{}^{G}y_1\,{}^{G}z_2 - {}^{G}y_2\,{}^{G}z_1 - {}^{G}y_1\,{}^{G}z_8 + {}^{G}y_8\,{}^{G}z_1 + {}^{G}y_2\,{}^{G}z_8 - {}^{G}y_8\,{}^{G}z_2}\right) \quad \text{(C4)}$$

$${}^{G}\Theta_C = \cdots$$

$$\sin^{-1}\left(\frac{-{}^{G}x_1\,{}^{G}y_2 + {}^{G}x_2\,{}^{G}y_1 + {}^{G}x_1\,{}^{G}y_8 - {}^{G}x_8\,{}^{G}y_1 - {}^{G}x_2\,{}^{G}y_8 + {}^{G}x_8\,{}^{G}y_2}{b_1 * b_2}\right) \quad \text{(C5)}$$

$${}^{G}\Phi_C = ATAN2\left(\frac{({}^{G}z_2 - {}^{G}z_1)(b_1 * b_2)}{({}^{G}x_2 - {}^{G}x_1)b_3 + ({}^{G}y_2 - {}^{G}y_1)b_4}\right) \quad \text{(C6)}$$

where $b_1 = \sqrt{({}^{G}x_2 - {}^{G}x_1)^2 + ({}^{G}y_2 - {}^{G}y_1)^2 + ({}^{G}z_2 - {}^{G}z_1)^2}$ (C7)

$$b_2 = \sqrt{({}^{G}x_8 - {}^{G}x_1)^2 + ({}^{G}y_8 - {}^{G}y_1)^2 + ({}^{G}z_8 - {}^{G}z_1)^2} \quad \text{(C8)}$$

$$b_3 = ({}^{G}x_1\,{}^{G}z_2 - {}^{G}x_2\,{}^{G}z_1 - {}^{G}x_1\,{}^{G}z_8 + {}^{G}x_8\,{}^{G}z_1 + {}^{G}x_2\,{}^{G}z_8 - {}^{G}x_8\,{}^{G}z_2) \quad \text{(C9)}$$

$$b_4 = ({}^{G}y_1\,{}^{G}z_2 - {}^{G}y_2\,{}^{G}z_1 - {}^{G}y_1\,{}^{G}z_8 + {}^{G}y_8\,{}^{G}z_1 + {}^{G}y_2\,{}^{G}z_8 - {}^{G}y_8\,{}^{G}z_2) \quad \text{(C10)}$$

Equations (C11) through (C18) are used to compute the cluster formation lengths.

$$L_{C2} = \sqrt{({}^{G}x_2 - {}^{G}X_C)^2 + ({}^{G}y_2 - {}^{G}Y_C)^2 + ({}^{G}z_2 - {}^{G}Z_C)^2} \quad \text{(C11)}$$

$$L_{23} = \sqrt{({}^{G}x_3 - {}^{G}x_2)^2 + ({}^{G}y_3 - {}^{G}y_2)^2 + ({}^{G}z_3 - {}^{G}z_2)^2} \quad \text{(C12)}$$

$$L_{C4} = \sqrt{({}^{G}x_4 - {}^{G}X_C)^2 + ({}^{G}y_4 - {}^{G}Y_C)^2 + ({}^{G}z_4 - {}^{G}Z_C)^2} \quad \text{(C13)}$$

$$L_{45} = \sqrt{({}^{G}x_5 - {}^{G}x_4)^2 + ({}^{G}y_5 - {}^{G}y_4)^2 + ({}^{G}z_5 - {}^{G}z_4)^2} \quad \text{(C14)}$$

$$L_{C6} = \sqrt{({}^{G}x_6 - {}^{G}X_C)^2 + ({}^{G}y_6 - {}^{G}Y_C)^2 + ({}^{G}z_6 - {}^{G}Z_C)^2} \quad \text{(C15)}$$

$$L_{67} = \sqrt{({}^{G}x_7 - {}^{G}x_6)^2 + ({}^{G}y_7 - {}^{G}y_6)^2 + ({}^{G}z_7 - {}^{G}z_6)^2} \quad \text{(C16)}$$

$$L_{C8} = \sqrt{({}^{G}x_8 - {}^{G}X_C)^2 + ({}^{G}y_8 - {}^{G}Y_C)^2 + ({}^{G}z_8 - {}^{G}Z_C)^2} \quad \text{(C17)}$$

$$L_{89} = \sqrt{({}^{G}x_9 - {}^{G}x_8)^2 + ({}^{G}y_9 - {}^{G}y_8)^2 + ({}^{G}z_9 - {}^{G}z_8)^2} \quad \text{(C18)}$$

Equations (C19) through (C37) compute the internal angles of the cluster formation.

$$\zeta = cos^{-1}\left(\frac{\begin{bmatrix} ({}^{G}x_2 - {}^{G}X_C) \\ ({}^{G}y_2 - {}^{G}Y_C) \\ ({}^{G}z_2 - {}^{G}Z_C) \end{bmatrix}}{L_{C2}} \bullet \frac{\begin{bmatrix} ({}^{G}x_8 - {}^{G}X_C) \\ ({}^{G}y_8 - {}^{G}Y_C) \\ ({}^{G}z_8 - {}^{G}Z_C) \end{bmatrix}}{L_{C8}}\right) \quad \text{(C19)}$$

$$\alpha_4 = -sin^{-1}[\frac{({}^{G}y_1 - {}^{G}y_4)(C\,{}^{G}\Psi_C S\,{}^{G}\Phi_C - C\,{}^{G}\Phi_C S\,{}^{G}\Psi_C S\,{}^{G}\Theta_C)}{L_{C4}} + \cdots \frac{({}^{G}x_4 - {}^{G}x_1)(S\,{}^{G}\Psi_C S\,{}^{G}\Phi_C + C\,{}^{G}\Phi_C C\,{}^{G}\Psi_C S\,{}^{G}\Theta_C)}{L_{C4}} + \cdots \frac{({}^{G}z_4 - {}^{G}z_1)(C\,{}^{G}\Phi_C C\,{}^{G}\Theta_C)}{L_{C4}} \quad \text{(C20)}$$

$$\alpha_6 = sin^{-1}[\frac{({}^{G}x_1 - {}^{G}x_6)(S\,{}^{G}\Psi_C C\,{}^{G}\Phi_C - S\,{}^{G}\Phi_C C\,{}^{G}\Psi_C S\,{}^{G}\Theta_C)}{L_{C6}} + \cdots \frac{({}^{G}y_6 - {}^{G}y_1)(C\,{}^{G}\Psi_C C\,{}^{G}\Phi_C + S\,{}^{G}\Phi_C S\,{}^{G}\Psi_C S\,{}^{G}\Theta_C)}{L_{C6}} + \cdots \frac{({}^{G}z_6 - {}^{G}z_1)(S\,{}^{G}\Phi_C C\,{}^{G}\Theta_C)}{L_{C6}} \quad \text{(C21)}$$

For h = 3, 5:

$$\alpha_h = -sin^{-1}[\frac{({}^{G}y_{h-1} - {}^{G}y_h)(C\,{}^{G}\Psi_C S\,{}^{G}\Phi_C - C\,{}^{G}\Phi_C S\,{}^{G}\Psi_C S\,{}^{G}\Theta_C)}{L_{(h-1)(h)}} + \cdots \frac{({}^{G}x_h - {}^{G}x_{h-1})(S\,{}^{G}\Psi_C S\,{}^{G}\Phi_C + C\,{}^{G}\Phi_C C\,{}^{G}\Psi_C S\,{}^{G}\Theta_C)}{L_{(h-1)(h)}} + \cdots \frac{({}^{G}z_h - {}^{G}z_{h-1})(C\,{}^{G}\Phi_C C\,{}^{G}\Theta_C)}{L_{(h-1)(h)}} \quad \text{(C22)}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3107854, IEEE Access

For j = 7, 9:

$$\alpha_j = sin^{-1}[\frac{\left({}^G x_{j-1} - {}^G x_j\right)\left(S\,{}^G\Psi_C C\,{}^G\Phi_C - S\,{}^G\Phi_C C\,{}^G\Psi_C S\,{}^G\Theta_C\right)}{L_{(j-1)(j)}} + \cdots$$
$$\frac{\left({}^G y_j - {}^G y_{j-1}\right)\left(C\,{}^G\Psi_C C\,{}^G\Phi_C + S\,{}^G\Phi_C S\,{}^G\Psi_C S\,{}^G\Theta_C\right)}{L_{(j-1)(j)}} + \cdots$$
$$\frac{\left({}^G z_j - {}^G z_{j-1}\right)\left(S\,{}^G\Phi_C C\,{}^G\Theta_C\right)}{L_{(j-1)(j)}} \quad \text{(C23)}$$

$$\beta_4 = ATAN2[q_1/(b_5 + b_6 + b_7)] \quad \text{(C24)}$$

where
$$q_1 = C\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G x_4 - {}^G x_1\right) + S\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G y_4 - {}^G y_1\right) + \cdots S\,{}^G\Theta_C\left({}^G z_1 - {}^G z_4\right) \quad \text{(C25)}$$

$$b_5 = \left({}^G x_4 - {}^G x_1\right)\left(S\,{}^G\Psi_C C\,{}^G\Phi_C - S\,{}^G\Phi_C C\,{}^G\Psi_C S\,{}^G\Theta_C\right) \quad \text{(C26)}$$

$$b_6 = \left({}^G y_4 - {}^G y_1\right)\left(C\,{}^G\Psi_C C\,{}^G\Phi_C + S\,{}^G\Phi_C S\,{}^G\Psi_C S\,{}^G\Theta_C\right) \quad \text{(C27)}$$

$$b_7 = \left({}^G z_1 - {}^G z_4\right)\left(S\,{}^G\Phi_C C\,{}^G\Theta_C\right) \quad \text{(C28)}$$

$$\beta_6 = ATAN2[q_2/(b_8 + b_9 + b_{10})] \quad \text{(C29)}$$

where
$$q_2 = C\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G x_1 - {}^G x_6\right) + S\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G y_1 - {}^G y_6\right) + \cdots S\,{}^G\Theta_C\left({}^G z_6 - {}^G z_1\right) \quad \text{(C30)}$$

$$b_8 = \left({}^G x_1 - {}^G x_6\right)\left(S\,{}^G\Psi_C S\,{}^G\Phi_C + C\,{}^G\Phi_C C\,{}^G\Psi_C S\,{}^G\Theta_C\right) \quad \text{(C31)}$$

$$b_9 = \left({}^G y_6 - {}^G y_1\right)\left(C\,{}^G\Psi_C S\,{}^G\Phi_C - C\,{}^G\Phi_C S\,{}^G\Psi_C S\,{}^G\Theta_C\right) \quad \text{(C32)}$$

$$b_{10} = \left({}^G z_1 - {}^G z_6\right)\left(C\,{}^G\Phi_C C\,{}^G\Theta_C\right) \quad \text{(C33)}$$

For p = 3, 5:

$$\beta_p = ATAN2[\frac{\left({}^G x_{p-1} - {}^G x_p\right)\left(S\,{}^G\Psi_C C\,{}^G\Phi_C - S\,{}^G\Phi_C C\,{}^G\Psi_C S\,{}^G\Theta_C\right)}{b_{11}} + \cdots$$
$$\frac{\left({}^G y_p - {}^G y_{p-1}\right)\left(C\,{}^G\Psi_C C\,{}^G\Phi_C + S\,{}^G\Phi_C S\,{}^G\Psi_C S\,{}^G\Theta_C\right)}{b_{11}} + \cdots$$
$$\frac{\left({}^G z_p - {}^G z_{p-1}\right)\left(S\,{}^G\Phi_C C\,{}^G\Theta_C\right)}{b_{11}} \quad \text{(C34)}$$

where
$$b_{11} = C\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G x_p - {}^G x_{p-1}\right) + S\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G y_p - {}^G y_{p-1}\right) + \cdots S\,{}^G\Theta_C\left({}^G z_{p-1} - {}^G z_p\right) \quad \text{(C35)}$$

For r = 7, 9:

$$\beta_r = ATAN2[\frac{\left({}^G x_{r-1} - {}^G x_r\right)\left(S\,{}^G\Psi_C S\,{}^G\Phi_C + C\,{}^G\Phi_C C\,{}^G\Psi_C S\,{}^G\Theta_C\right)}{b_{12}} + \cdots$$
$$\frac{\left({}^G y_r - {}^G y_{r-1}\right)\left(C\,{}^G\Psi_C S\,{}^G\Phi_C - C\,{}^G\Phi_C S\,{}^G\Psi_C S\,{}^G\Theta_C\right)}{b_{12}} + \cdots$$
$$\frac{\left({}^G z_{r-1} - {}^G z_r\right)\left(C\,{}^G\Phi_C C\,{}^G\Theta_C\right)}{b_{12}} \quad \text{(C36)}$$

where
$$b_{12} = C\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G x_r - {}^G x_{r-1}\right) + S\,{}^G\Psi_C C\,{}^G\Theta_C\left({}^G y_r - {}^G y_{r-1}\right) + \cdots S\,{}^G\Theta_C\left({}^G z_{r-1} - {}^G z_r\right) \quad \text{(C37)}$$

Equations (B21) through (B23) are used to compute individual robot yaw angles relative to frame C, for Robots 1 to 9 where $n = 1,\ldots,9$.

APPENDIX D
SCALAR FIELD PLUMES

Plume source coordinates x_p and y_p with respect to G, are constant for stationary plumes. Variables x, y, z are coordinates of Robot n with respect to G. Equation (D1) generates:

$$s_n = p_1 * \frac{e^{-[|0.001\cdot z|*(sign(z)+1)]}}{\left(\frac{\sqrt{\left(x-(x_p+p_2)\right)^2+\left(y-(y_p+p_3)\right)^2}}{p_4*(0.1*z+1)}\right)^2+1} \quad \text{(D1)}$$

Section IV.D.1: symmetric, stationary, and time- invariant plume where $p_1 = 250$, $p_2 = 0$, $p_3 = 0$, and $p_4 = 25$
Section IV.D.2: symmetric, moving, and time-varying plume where $p_1 = (250 + t)$, $p_2 = 0$, $p_3 = 0$, and $p_4 = 25$
Section V.B and V.D: symmetric, stationary, and time-invariant plume where $p_1 = 30$, $p_2 = 0$, $p_3 = 0$, and $p_4 = 8$
Section V.D: asymmetric, stationary, and time-invariant plume where $p_1 = 300$, $p_2 = 0$, $p_3 = e^{0.03\cdot z}$, and $p_4 = 30$
Section V.D: overlapping, symmetric, stationary, and time-invariant plumes where $p_1 = 15$, $p_2 = 0$, $p_3 = 0$, and $p_4 = 10$
Section VI.D: asymmetric, stationary, and time-invariant plume where $p_1 = 300$, $p_2 = 0$, $p_3 = e^{0.035\cdot z}$, and $p_4 = 10$
Section VI.D: high scalar magnitude, asymmetric, stationary, and time-invariant plume where $p_1 = 1000$, $p_2 = e^{0.035\cdot z}$, $p_3 = 0$, and $p_4 = 10$.

REFERENCES

[1] C.A. Kitts, R.T. McDonald, and M.A. Neumann, "Adaptive Navigation Control Primitives for Multirobot Clusters: Extrema Finding, Contour Following, Ridge/Trench Following, and Saddle Point Station Keeping," *IEEE Access*, Vol 6, pp.17625-17642, 2018.
[2] C. Zhang, D. Arnold, N. Ghods, A. Siranosian, and M. Krstic, "Source Seeking with Non-Holonomic Unicycle without Position Measurement and with Tuning of Forward Velocity," in *Systems & Control Letters*, Vol.56, No.3, pp.245-252, 2007.
[3] C.G. Mayhew, R.G. Sanfelice, and A.R. Teel, "Robust Source-Seeking Hybrid Controllers for Nonholonomic Vehicles," in *Proceedings of Annual American Control Conference*, pp.2722-2727, Jun. 2008.
[4] A.S. Matveev, H. Teimoori and A.V. Savkin, "Navigation of a Non-Holonomic Vehicle for Gradient Climbing and Source Seeking without Gradient Estimation," in *Proceedings of Annual American Control Conference*, pp.219-223, Jun. 2010.
[5] A.S. Matveev, H. Teimoori and A.V. Savkin, "Navigation of a Unicycle-Like Mobile Robot for Environmental Extremum Seeking," *Automatica*, Vol.47, pp.85-91, 2011.
[6] S. Azuma, M.S. Sakar, and G.J. Pappas, "Stochastic Source Seeking by Mobile Robots," *IEEE Transactions on Automatic Control*, Vol.57,No.9, pp.2308-2321, Sep. 2012.

[7] C.G. Mayhew *et al.*, "Robust Hybrid Source-Seeking Algorithms Based on Directional Derivatives and Their Approximations," in *Proceedings of IEEE Conference on Decision and Control*, pp.1735-1740, Dec. 2008.

[8] S. Shigaki, Y. Shiota, D. Kurabayashi, and R. Kanzaki, "Modeling of the Adaptive Chemical Plume Tracing Algorithm of an Insect Using Fuzzy Inference," *IEEE Transactions on Fuzzy Systems*, Vol.28, No.1, pp.72-84, Jan. 2020.

[9] P.P. Neumann, V.H. Bennetts, A.J. Lilienthal, M. Bartholmai, and J.H. Schiller, "Gas Source Localization with a Micro-Drone Using Bio-inspired and Particle Filter-Based Algorithms," Advanced Robotics, Vol.27, No.9, pp.725-738, Apr. 2013.

[10] P. Ogren *et al.*, "Cooperative Control of Mobile Sensor Networks: Adaptive Gradient Climbing in a Distributed Environment," *IEEE Transactions on Automatic Control*, Vol.49, No.8, pp.1292–1302, 2004.

[11] S. Zhuo, "Source Seeking of Multi-UAV Based on Extremum Seeking Algorithm," in *Proceedings of 17th International Conference on Control, Automation and Systems (ICCAS)*, pp.1062-1067, Oct. 2017.

[12] S. Eu and K.M. Yap, "Chemical Plume Tracing: A Three-Dimensional Technique for Quadrotors by Considering the Altitude Control of the Robot in the Casting Stage," *International Journal of Advanced Robotic Systems*, Vol.15, No.1, 2018.

[13] E. Biyik and M. Arcak, "Gradient Climbing in Formation via Extremum Seeking and Passivity-Based Coordination Rules," in *Proceedings of IEEE Conference on Decision and Control*, pp.3133-3138, Dec. 2007.

[14] S. Zhu, D. Wang, and C.B. Low, "Cooperative Control of Multiple UAVs for Moving Source Seeking," in *Proceedings of International Conference on Unmanned Aircraft Systems*, pp.193-202, May 2013.

[15] B.J. Moore and C. Canudas-de-Wit, "Source Seeking via Collaborative Measurements by a Circular Formation of Agents," in *Proceedings of Annual American Control Conference*, pp.6417-6422, Jun. 2010.

[16] N. A. Atanasov, J.L. Ny, and G. J. Pappas, "Distributed Algorithms for Stochastic Source Seeking with Mobile Robot Networks," *Journal on Dynamic Systems, Measurement & Control*, Vol.173, No.3, pp.1-9, 2015.

[17] V. Gazi *et al.*, "Robot Swarms: Dynamics and Control," *Mobile Robots for Dynamic Environments*, E. F. Kececi and M. Ceccarelli, Eds. New York, NY, USA: ASME, Ch.4, pp.79-107, 2015.

[18] A. Turgeman and H. Werner, "Multiple Source Seeking using Glowworm Swarm Optimization and Distributed Gradient Estimation," in *Proceedings of American Control Conference*, pp.3558-3563, Jun. 2018.

[19] Z. Li, K. You, and S. Song, "Cooperative Source Seeking via Networked Multi-vehicle Systems," *Automatica*, May, 2020.

[20] S. Li, R. Kong, and Y. Guo, "Cooperative Distributed Source Seeking by Multiple Robots: Algorithms and Experiments," *IEEE Transactions on Mechatronics*, Vol.19, No.6, pp.1810-1820, 2014.

[21] W. Wu *et al.*, "Bio-inspired Source Seeking with No Explicit Gradient Estimation," in *Proceedings of IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pp.240-245, Sep. 2012.

[22] J.R. Bourne, E.R. Pardyjak, and K.K. Leang, "Coordinated Bayesian-Based Bioinspired Plume Source Term Estimation and Source Seeking for Mobile Robots," *IEEE Transactions on Robotics*, Vol. 35, No.4, pp.967-986, Aug. 2019.

[23] A.S. Matveev *et al.*, "Tracking of Deforming Environmental Level Sets of Dynamic Fields by a Nonholonomic Robot without Gradient Estimation," in *Proceedings of IEEE ICCA*, pp.1754-1759, Jun. 2013.

[24] A. Newaz, S. Jeong, H. Lee, and H. Ryu, "UAV-Based Multiple Source Localization and Contour Mapping of Radiation Fields," *Robotics and Autonomous Systems*, Vol.85, pp.12-25, 2016.

[25] S. Srinivasan *et al.*, "ACE in the Hole: Adaptive Contour Estimation using Collaborating Mobile Sensors," in *Proceedings of Int. Conference on Information Processing in Sensor Networks*, pp.147-158, Apr. 2008.

[26] W. Li, J. A. Farrell, S. Pang, and R. M. Arrieta, "Moth-Inspired Chemical Plume Tracing on an Autonomous Underwater Vehicle," *IEEE Transactions on Robotics*, Vol.22, No.2, pp.292-307, Apr. 2006.

[27] Y. Tian, W. Li, A. Zhang, J. Yu, Q. Zhang, and Y. Li, "From Simulation to Validation - Moth-Inspired Chemical Plume Tracing with an Autonomous Underwater Vehicle," in *Proceedings of IEEE Oceans Conference*, St. John's, pp.1-10, Sep. 2014.

[28] A.S. Matveev *et al.*, "Robot Navigation for Monitoring Unsteady Environmental Boundaries without Field Gradient Estimation," *Automatica*, Vol.62, pp.227-235, Dec. 2015.

[29] Z. Shen, Z. He, S. Li, Q. Wang, and Z. Shao, "A Multi-Quadcopter Cooperative Cyber-Physical System for Timely Air Pollution Localization," *ACM Transactions on Embedded Computing Systems*, Vol. 16, No. 3, Art. 70, Apr. 2017.

[30] Z. Jin and A. L. Bertozzi, "Environmental Boundary Tracking and Estimation Using Multiple Autonomous Vehicles," in *Proceedings of IEEE Conference on Decision and Control*, pp.4918-4923, Dec. 2007.

[31] F. Zhang and N. E. Leonard, "Cooperative Filters and Control for Cooperative Exploration," *IEEE Transactions on Automatic Control*, Vol.55, No.3, pp.650-663, Mar. 2010.

[32] X. Kang and W. Li, "Moth-Inspired Plume Tracing via Multiple Autonomous Vehicles Under Formation Control," *Adaptive Behavior*, Vol.20, No.2, pp.131-142, Apr. 2012.

[33] J. Han and Y. Chen, "Cooperative Source Seeking and Contour Mapping of a Diffusive Signal Field by Formations of Multiple UAVs," in *Proc of International Conf on Unmanned Aircraft Systems*, pp.35-40, May 2013.

[34] J. Han and Y. Chen, "Multiple UAV Formations for Cooperative Source Seeking and Contour Mapping of a Radiative Signal Field," *Journal of Intelligent and Robotic Systems*, Vol.74, pp.323-332, 2014.

[35] Z. Cook, M. Kazemeini, A. Barzilov, and W. Yim, "Low Altitude Contour Mapping of Radiation Fields Using UAS Swarm," *Intelligent Service Robotics*, Vol. 12, pp.219-230, Apr. 2019.

[36] J. Cochran, N. Ghods, and M. Krstic, "3D Nonholonomic Source Seeking Without Position Measurement," in *Proceedings of Annual American Control Conference*, pp.3518-3523, Jun. 2008.

[37] J. Cochran, A. Siranosian, N. Ghods, and M. Krstic, "3-D Source Seeking for Underactuated Vehicles Without Position Measurement," *IEEE Transactions on Robotics*, Vol.25, No.1, pp.117-129, Feb. 2009.

[38] A.S. Matveev, M.C. Hoy, and A.A. Semakova, "3D Environmental Extremum Seeking Navigation of a Nonholonomic Mobile Robot," *Automatica*, Vol.50, pp.1802-1815, Jun. 2014.

[39] J. Lin *et al.*, "3-D Velocity Regulation for Nonholonomic Source Seeking Without Position Measurement," *IEEE Transactions on Control Systems Technology*, Vol.24, No.2, pp.711-718, Mar. 2016.

[40] S. Al-Abri, W. Wu, and F. Zhang, "A Gradient-Free 3-dimensional Source Seeking Strategy with Robustness Analysis," *IEEE Transactions on Automatic Control*, Vol.66, No.8, pp.3439-3446, Aug. 2019.

[41] L. Briñón-Arranz *et al.*, "Multirobot Symmetric Formations for Gradient and Hessian Estimation with Application to Source Seeking," *IEEE Transactions on Robotics*, Vol.35, No.3, pp.782-789, Jun. 2019.

[42] A.S. Matveev and A.A. Semakova, "Gradient-Free Navigation of a Nonholonomic Robot for Tracking Unsteady Environmental Boundaries in 3D," *in Proceedings of 22nd International Conference on System Theory, Control, and Computing*, pp.670-676, 2018.

[43] Y. Tan, "Chemical Plume Tracing and Mapping via Swarm Robots," *Handbook of Research on Design, Control, and Modeling of Swarm Robotics*, Ch.16, pp.421-455, 2016.

[44] J. Han, Y. Xu, L. Di, and Y. Chen, "Low-cost Multi-UAV Technologies for Contour Mapping of Nuclear Radiation Field," *Journal of Intelligent and Robotic Systems*, Vol.70, pp.401-410, Apr. 2013.

[45] W. Wu and F. Zhang, "Cooperative Exploration of Level Surfaces of Three Dimensional Scalar Fields," *Automatica*, Vol.47, No.9, pp.2044-2051, 2011.

[46] J.W. Wang, Y. Guo, M. Fahad, and B. Bingham, "Dynamic Plume Tracking by Cooperative Robots," *IEEE/ASME Transactions on Mechatronics*, Vol.24, No.2, pp.609-620, Apr. 2019.

[47] R.T. McDonald, C.A. Kitts, and M.A. Neumann,, "Experimental Implementation and Verification of Scalar Field Ridge, Trench, and Saddle Point Maneuvers Using Multirobot Adaptive Navigation," *IEEE Access*, Vol. 7, pp.62950-62961, 2019.
[48] R.T. McDonald, M. Condino, M.A. Neumann, and C.A. Kitts, "Navigation of Scalar Fronts with Multirobot Clusters in Simulation and Experiment," *IEEE Systems Journal*, Vol. 14, No. 3, pp.3755-3766, Sep. 2020.
[49] C.A. Kitts and I. Mas, "Cluster space specification and control of mobile multirobot systems," *IEEE/ASME Transactions on Mechatronics*, Vol.14, No.2, pp.207-218, Apr. 2009.
[50] I. Mas and C.A. Kitts, "Dynamic Control of Mobile Multirobot Systems - The Cluster Space Formation," *IEEE Access*, Vol.2, pp.558-570, May. 2014.
[51] M. Santos *et al.*, "A Novel Null-Space-Based UAV Trajectory Tracking Controller with Collision Avoidance," *IEEE/ASME Transactions on Mechatronics*, Vol.22, No. 6, pp.2543-2553, Dec. 2017.
[52] L.V. Santana *et al.*, "Navigation and Cooperative Control Using the AR Drone Quadrotor," *Journal of Intelligent and Robotic Systems*, Vol.84, pp.327-350, Dec. 2016.
[53] L.V. Santana *et al.*, "A Trajectory Tracking and 3D Positioning Controller for AR.Drone Quadrotor," in *Proceedings of International Conference on Unmanned Aircraft Systems*, pp.756-767, May. 2014.
[54] M.S. Selig, "Modeling Propeller Aerodynamics and Slipstream Effects on Small UAVs in Realtime", in *Proceedings of AIAA Atmospheric Flight Mechanics Conference*, Toronto, Canada, pp.7938-7961, Aug. 2010.
[55] GPS World Staff, "GPS Accuracy: Lies, Damn Lies, and Statistics," *GPS World*, Jan. 1998.
[56] National Research Council. *The Global Positioning System: A Shared National Asset*. DC: The National Academies Press, 1995, Append D.
[57] GPS World Staff, "Update: GNSS Accuracy: Lies, Damn Lies, and Statistics," *GPS World*, Jan. 2007.
[58] Trimble Navigation GPS Receiver data sheets, http://trl.trimble.com/docushare/dsweb/Get/Document-873444/
[59] NovAtel GPS Receiver data sheets, https://www.novatel.com/support/info/documents/1825
[60] G.M. Bolla *et al.*, "ARIA: Air Pollutants Monitoring Using UAVs," in *Proceedings of 5th IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pp.225-229, 2018.
[61] Alphasense Air Quality Sensor data sheets, http://www.alphasense.com/index.php/air/downloads/
[62] R.K. Lee, C. Kitts, M. Neumann, and R. McDonald, "3-D Adaptive Navigation: Multirobot Formation Control for Seeking and Tracking of a Moving Source," in *Proceedings of 41st IEEE Aerospace Conference*, Mar. 2020.
[63] C. Kitts *et al.*, "Field Operation of a Robotic Small Waterplane Area Twin Hull Boat for Shallow-Water Bathymetric Characterization," *Journal of Field Robotics*, Vol.29, No.6, pp.924-938, Nov. 2012.
[64] Z. Cameron, B. Engh, A. Kambe, and A. Krishnan, "Adaptive Navigation Using a Drone Cluster," *Santa Clara University Senior Capstone Report*, Jun. 2020, unpublished.
[65] A. Kashyap and D. Ghose, "Pursuing a Time Varying and Moving Source Signal Using a Sensor Equipped UAV," *2017 International Conference on Unmanned Aircraft Systems*, pp.506-515, Jun. 2017.
[66] A. Kamthe and S. Ghosh, "Gradient-based Augmentation to Maxima Turn Switching Strategy for Source-Seeking using Sensor-Equipped UAVs," *2020 International Conference on Unmanned Aircraft Systems*, pp.1525-1532, Sep. 2020.
[67] R.G. Li and H.N. Wu, "Multi-Robot Source Location of Scalar Fields by a Novel Swarm Search Mechanism With Collision/Obstacle Avoidance," *IEEE Transactions on Intelligent Transportation Systems*, pp.1-16, Jul. 2020.
[68] L. Yang and S. Liu, "Distributed Stochastic Source Seeking for Multiple Vehicles over Fixed Topology," *Journal of Systems Science and Complexity*, Vol.33, pp.652-671, Jun. 2020.
[69] A.S. Matveev and M.S. Nikolaev, "Hybrid Control for Tracking Environmental Level Sets by Nonholonomic Robots in Maze-Like Environments," *Nonlinear Analysis: Hybrid Systems*, Vol.39, pp.1-20, Feb. 2021.
[70] Z. Yuan *et al.*, "Leader–Follower Formation Source Seeking Control of Multiple Ships Using Sliding Mode Active Disturbance Rejection Observer," *Measurement and Control*, pp.1-11, May. 2020.
[71] A. Renzaglia and L. Briñón-Arranz. "Search and Localization of a Weak Source with a Multi-Robot Formation," *Journal of Intelligent & Robotic Systems*, Vol.97, pp.623-634, Apr. 2019.
[72] W. Fu *et al.*, "Resilient Cooperative Source Seeking of Double-Integrator Multi-robot Systems Under Deception Attacks," *IEEE Transactions on Industrial Electronics*, Apr. 2020.
[73] A. Datar, P. Paulsen, and H. Werner. "Flocking Towards the Source: Indoor Experiments with Quadrotors," *European Control Conference (ECC)*, May. 2020.
[74] T. Paul *et al.*, "UAV Formation Flight Using 3D Potential Field," *2008 16th Mediterranean Conference on Control and Automation*, pp.1240-1245, Jun. 2008.