drones

MDPI

Communication

Entropy-Based Distributed Behavior Modeling for Multi-Agent UAVs

Luke Fina [1], Douglas Shane Smith Jr. [2], Jason Carnahan [3] and Hakki Erhan Sevil [2,*]

[1] Mechanical Engineering, University of West Florida, Pensacola, FL 32514, USA; laf37@students.uwf.edu
[2] Intelligent Systems & Robotics, University of West Florida, Pensacola, FL 32514, USA; ds117@students.uwf.edu
[3] I3, Huntsville, AL 35806, USA; jac0081@uah.edu
* Correspondence: hsevil@uwf.edu

Abstract: This study presents a novel distributed behavior model for multi-agent unmanned aerial vehicles (UAVs) based on the entropy of the system. In the developed distributed behavior model, when the entropy of the system is high, the UAVs get closer to reduce the overall entropy; this is called the grouping phase. If the entropy is less than the predefined threshold, then the UAVs switch to the mission phase and proceed to a global goal. Computer simulations are performed in AirSim, an open-source, cross-platform simulator. Comprehensive parameter analysis is performed, and parameters with the best results are implemented in multiple-waypoint navigation experiments. The results show the feasibility of the concept and the effectiveness of the distributed behavior model for multi-agent UAVs.

Keywords: multi-agent system; distributed behavior modeling; unmanned aerial vehicles (UAVs); Tsallis entropy

Citation: Fina, L.; Smith, D.S., Jr.; Carnahan, J.; Sevil, H.E. Entropy-Based Distributed Behavior Modeling for Multi-Agent UAVs. *Drones* **2022**, *6*, 164. https://doi.org/10.3390/drones6070164

Academic Editor: Sergio Salazar

Received: 12 May 2022
Accepted: 28 June 2022
Published: 29 June 2022

1. Introduction

Recent advances in unmanned aerial vehicles (UAVs) make their use in various applications feasible, including but not limited to precision agriculture, infrastructure monitoring, security, telecommunication, and entertainment [1]. UAVs have also been designed and produced in many different forms and sizes, both custom-built and off-the-shelf, to offer solutions for general remote sensing applications [2]. There are still key research challenges for UAVs; one of them is to find efficient techniques for applications with multi-agent UAVs [3]. Deploying large numbers of UAVs can be advantageous in particular situations, such as search and rescue, disaster response, and surveillance, and various approaches and applications have been presented in the literature with multi-agent UAVs: autonomous navigation by a group of unmanned vehicles using potential fields [4], cooperative task assignment of multi-agent heterogeneous unmanned aerial vehicles using a modified genetic algorithm (GA) [5], manned and unmanned team approaches for situational awareness [6,7], dynamic data-driven applications systems (DDDAS) for command/control and mission planning/re-planning for unmanned aerial vehicles (UAVs) [8], market-economy-based cost function estimator for collaborative tasking of tightly constrained multi-robot missions [9], 3D modeling using a UAV swarm [10], an algorithm to control and coordinate the actions of heterogeneous unmanned air and ground systems [11], automatic target recognition (ATR) using UAVs based on an insect-colony-inspired multi-agent technique [12], and obstacle avoidance for small autonomous aircraft using self-directed collaborative navigation [13].

The distributed systems for multi-agent UAV applications have also been well adopted in recent years [14]. Most of the studies presented are based on maximizing global gain while having relatively simple individual members in the team [14]. Most of the presented methods are inspired from nature, which provides good solutions to manage groups,

i.e., fish schools, ant swarms, animal packs, and bird flocks, and with the growing desire of humans to create similar distributed behavior, mathematical models of swarms have arisen within the last two and a half decades. These mathematical models have lead to controlling groups of multi-agents. Swarm robotics is the application of these swarm algorithms to robots. These systems are often referred to as multi-agent or multi-robot systems. There are numerous focus areas within multi-agent systems. The main ones are the following: task allocation, communication, uncertainty modeling, multi-agent planning, and control [15,16]. In swarm robotics, there is great interest in decentralized control due to scalability plus robustness when losing single agents. Some multi-agent formation-control algorithms lend themselves better for implementation due to limited complex computations. Behavior-based formation control is one of those methods.

Several examples of behavior-based methods are worth mentioning here. Balch presented formation testing with reactive formation control that was divided into avoiding obstacles, avoiding robots, moving to the goal, and maintaining formation [17]. One method involved modeling behaviors by defining the environment with attractors and repellers to maintain a triangle formation and avoid obstacles [18]. Lawton et al. introduced three behavior control strategies: coupled dynamics formation control, coupled dynamics formation control with passivity-based inter-robot damping, and saturated control [19]. A vision-based method where general formation control with local sensing has also been presented in the literature [20]. Furthermore, formation control to achieve minimum entropy has been presented by Caglioti et al. [21]. Monteiro et al. use nonlinear attractor dynamics to create a framework for controlling a team of robots [22]. Xu et al. present a variance of initial formation and a subsequent formation controller [23]. Other studies on formation control strategies have been presented by Olfati-Saber [24], mixing potential field and graph-based, and, more recently, by Vásárhelyi [25].

Particle swarm optimization, PSO, is another common approach introduced in swarm intelligence. Updated swarm intelligence strategies are constantly under development, such as a relatively new dragonfly algorithm, which is an alternative to PSO for optimization [26]. Some methods have implemented PSO or a derivative of PSO for UAV formations with additional measures including bounding boxes for localization [27] and a GA–PSO hybrid [28].

Leader–follower and virtual leader–follower methods have also been presented in the literature [29,30]. Null-space-based formation control with a UAV and a ground unit is an example of this type [31]. More similar to our developed method is an approach presented by Lee and Chwa that is decentralized behavior-based [32] and only uses relative distance information between neighbors and obstacles. Another similar method is consensus formation control [33]. Moreover, a method that uses an artificial potential field for obstacle avoidance and a consensus algorithm for maintenance and reconstruction of the formation has been introduced in the literature [34]. Consensus formation control was recently applied to four parrot bebop drones in a real-time experiment with obstacle avoidance [35] as well. Three-dimensional formations based only on local relative position measurements is another relative-position method that was performed on Crazyflie [36]. Some methods modify flocking algorithms; for example, in the literature there is a swarm method of object-focused learning with modular state–action–reward–state–action (SARSA) applied to Reynolds flocking [37]. Another consensus algorithm is a nonsmooth consensus method to control the formation of multi-agent quadrotors using a theoretical analysis in which the quadrotors converge to the desired formation [38]. Another distributed method is one where a distributed nonlinear formation controller uses the leader–follower structure to control the formation with a focus on investigation of unknown bounded disturbances [39]. Few multi-robot platform implementations have been introduced in the literature for swarm testing. One noteworthy innovation is the Robotarium, which provides open access swarm algorithm testing on hardware [40]. Limited work on entropy control or analysis of swarms exists in the literature, aside from a recent paper on potentially using cross-entropy to determine robustness of a swarm [41]. Areas of active research for appli-

cation in formation control include, but are not limited to, the following: improvements towards autonomous farming with swarm robotics for agricultural applications (SAGA) [42], satellites in space [43], and natural disaster relief [44].

Although significant research has been done on behavior-based methods that work better than the original Reynolds algorithm, it is still an open area of research within multi-agent systems. Little attention has been imparted for development of a generic cost function suitable for heterogeneous platforms/sensors. In this paper, we introduce a behavior-based modeling approach for distributed control, relying on a combination of probability distribution of the agents and the Tsallis entropy of the system. This method has advantages over other methods due to low computation times, robustness to unknowns, and dependence only on communication between agents instead of mathematical convergence. This method seeks to address major challenges outlined in the literature where there is a need for formation control to be simple, robust, reliable, and fault-tolerant [45]. Original contributions of our study are: (i) development of a distributed behavior model with decentralized rules over the group of agents (UAVs), (ii) development of entropy of the system-based algorithm to switch between different behaviors of multi-agent UAVs, (iii) evaluation of entropy threshold values to allow multi-agent UAVs to navigate effectively without interfering with each other or failing to navigate along the desired course.

The remainder of the paper is organized as follows. The dynamic model of the UAVs will be discussed in Section 2, followed by the distributed behavior model in Section 3. Parameter analysis will be presented in Section 4, followed by the results and discussion in Section 5. Conclusion will be presented in Section 6.

2. Dynamic Model of UAVs

The quadrotor model used in this study is adopted from AirSim [46] and is briefly outlined below. Figure 1 depicts a model of the quadrotor used in our study, and, considering the whole vehicle model, the equations of motion from Newton's law and Euler's equation can be written as [47],

$$\mathbf{F} = m \cdot \mathbf{a} \tag{1}$$

$$\tau = \boldsymbol{J} \cdot \dot{\omega} + \omega \times \boldsymbol{J} \cdot \omega \tag{2}$$

where $\boldsymbol{J}$ is the inertia matrix of the quadrotor, $\mathbf{a}$ is the acceleration, ω is the angular velocity, and m is the mass of the quadrotor. In Equation (1), the linear part is expressed in the inertial frame (world frame); on the other hand, the rotational part in Equation (2) is represented in the vehicle body frame. Considering an n-motor vehicle, Equations (1) and (2) can be generalized as [47]

$$\sum_{i=0}^{n-1} (\boldsymbol{R}_{WB}(\boldsymbol{F}_{T,i} + \boldsymbol{F}_{D,i})) + \boldsymbol{F}_G = m \cdot \mathbf{a} \tag{3}$$

$$\sum_{i=0}^{n-1} (\boldsymbol{M}_{R,i} + \boldsymbol{M}_{D,i} + \boldsymbol{F}_i \times \boldsymbol{r}_i) = \boldsymbol{J} \cdot \dot{\omega} + \omega \times \boldsymbol{J} \cdot \omega \tag{4}$$

where $\boldsymbol{F}_T$ is thrust force, $\boldsymbol{F}_D$ is drag force, $\boldsymbol{M}_R$ is the rolling moment, $\boldsymbol{M}_D$ is the moment from rotor blade drag, $\boldsymbol{R}_{WB}$ is the rotation matrix from the body frame B to the world frame W, and $\boldsymbol{r}_i$ is the vector from the center of gravity of the quadrotor to the center of the i^{th} motor. The sum of $(\boldsymbol{F}_{T,i} + \boldsymbol{F}_{D,i})$ can be simply represented as the total force, $\mathbf{F}_i$. In our study, the adopted quadrotor model [46] is defined as four connected vertices with thrust forces $\mathbf{F}_i$, torques from the propellers τ_i, and control inputs u_i (Figure 1).

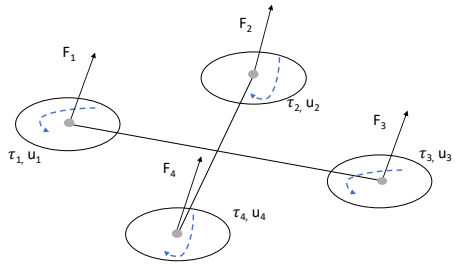


Figure 1. Representation of quadrotor model.

The forces and torques [46] are defined as

$$\mathbf{F}_i = C_T \rho \omega_{max}^2 D^4 u_i \tag{5}$$

$$\tau_i = \frac{1}{2\pi} C_{pow} \rho \omega_{max}^2 D^5 u_i \tag{6}$$

where ρ is the air density, C_T is the thrust coefficient, ω_{max} is the max angular velocity in revolutions per minute, C_{pow} is the power coefficients, and D is the propeller's diameter. In the simulator, the magnitude ($|\cdot|$) of the linear drag force on the body is used, defined as [46]:

$$|\mathbf{F}_d| = \frac{1}{2} \rho |\mathbf{v}^2| C_{lin} A \tag{7}$$

where $\mathbf{v}$ is the velocity vector and drag force acts in the opposite direction to the velocity vector, C_{lin} is the linear air drag coefficient, and A is the vehicle cross-section. Consider an infinitesimal surface area ds in the UAV body with angular velocity ω. The linear velocity $\mathbf{dv}$ experienced by ds equals $\mathbf{r}_{ds} \times \omega$; thus, the linear drag equation for ds becomes [46]

$$|\mathbf{dF}| = \frac{1}{2} \rho |\mathbf{r}_{ds} \times \omega|^2 C_{lin} ds \tag{8}$$

where the direction of $\mathbf{dF}$ is $-\mathbf{r}_{ds} \times \omega$. The drag torque can now be computed by integrating over the entire surface as: $\tau_d = \int_S \mathbf{r}_{ds} \times \mathbf{dF}$, and the body for the drag force is approximated as a set of connected faces and then approximated as a rectangular box. Further, the net force is calculated as [46]

$$\mathbf{F}_{net} = \sum_i \mathbf{F}_i + \mathbf{F}_d, \tag{9}$$

and the torque is calculated as

$$\tau_{net} = \sum_i |\tau_i + \mathbf{r}_i \times \mathbf{F}_i| + \tau_d. \tag{10}$$

The velocity is calculated with the Velocity Verlet algorithm [46]. Since the simulation is performed on AirSim, it enables a straightforward transition to hardware in the loop. Three UAVs are used for the parameter trials, and three and six UAVs are used for multiple waypoint testing. The state variables are obtained through the ground truth API, which can be IMU-based when implemented on real hardware. A screenshot of the UAVs in the grouping and mission phases in AirSim is depicted in Figure 2.

The built-in flight controller in AirSim is used for the UAVs; it is a cascaded PID controller [48]. The "simple_flight" code is utilized, which controls the UAVs by taking in the desired input as heading and velocity. In "simple_flight" code, the cascade of PID controllers generates actuator signals [48]. Basically, in the code, the position PID drives the velocity PID, the velocity PID drives the angle level PID, and then the angle level PID drives the angle rate PID [48]. This internally nested controller structure enables us to just focus on the commanded velocity (V_{com}) and commanded heading (θ_{com}) calculations in our entropy-based distributed behavior algorithm. Our algorithm uses current position

and heading of the corresponding UAV agent, calculates V_{com} and θ_{com} considering the distributed behavior model, and sends V_{com} and θ_{com} to the "simple_flight" controller, which generates actuator signals for that UAV agent. The controller equations for heading and velocity control are given as [49]

$$u_\theta = I_y\left(\frac{\alpha_\theta}{\mu_\psi}\dot{e}_\theta - \frac{\beta_\theta}{\mu_\theta^2}e_\theta - \frac{e_\psi e_\phi}{\mu_\psi \mu_\phi}\left(\frac{I_z - I_x}{I_y}\right)\right) \tag{11}$$

$$u_v = U_v\sigma\left(\frac{K_{v1}}{U_v}\dot{e}_v + \frac{1}{2}\sigma\left(\frac{K_{v2}}{U_v}\dot{e}_v + \frac{K_{v1}K_{v2}}{U_v}e_v\right)\right) \tag{12}$$

where β, α, μ are control parameters, e is the error in the corresponding variable, $K_{v1}, K_{v2} \geq 0$ are controller gains, σ is the saturation function, and U is the input bound limit constant. More details on dynamic modeling and controller design of the adopted approach and all of the equations can be found in [46–49].

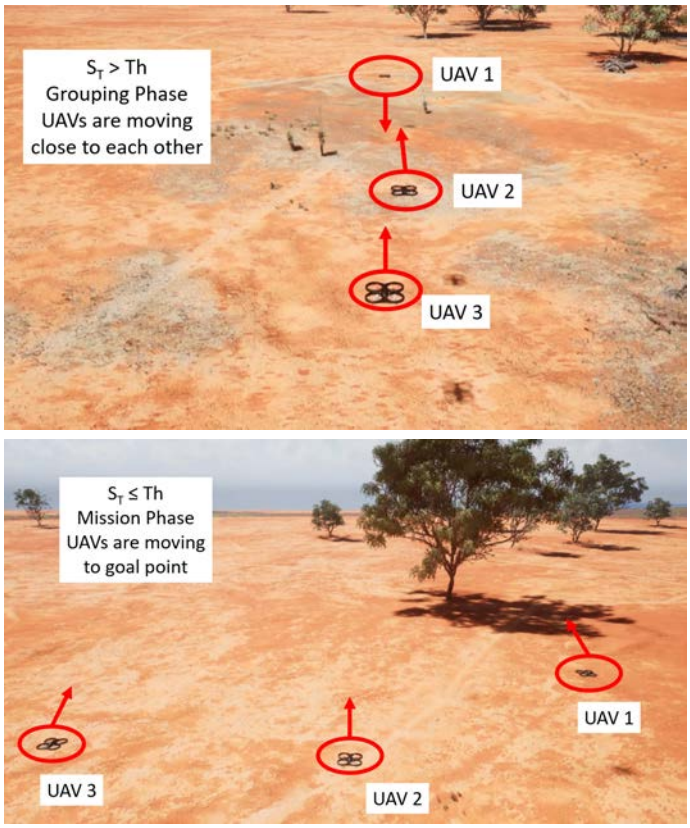


Figure 2. Screenshot of UAVs in AirSim software.

3. Distributed Behavior Model

Entropy originates from information theory. It is characterized as the disorder of a system. Entropy can be generalized to a group of particles, so with this in mind the objective is to use entropy for formation control to improve scalability, reliability, and robustness of the multi-agent group. In this study, a generalized case for entropy, Tsallis entropy, is used to control the formation.

Tsallis entropy is defined as [50]

$$S_T = \frac{1 - \sum_i p_i^q}{q - 1} \tag{13}$$

where S_T is the calculated entropy, p_i is a discrete set of probabilities, and q is the Tsallis entropy parameter, which is a real number and sometimes called the "entropic-index". En-

tropy fluctuates between its highest and lowest values as the UAVs perform the given task. Higher entropy means, for instance, the distance between agents is rather far, and system stability is low, i.e., the probability of remaining a coordinated system may decrease. In this study, our focus for UAVs is to form a group for a global mission; hence, system stability is related to distributed behavior of the UAVs. Consequently, the entropy of the system should be as low as possible, and the developed distributed model switches between different phases (i.e., grouping phase and mission phase) based on the overall system entropy.

The Tsallis entropy approach works for homogeneous and heterogeneous agents because it is generalized for any system of particles. Every agent has case-dependent variable X. The probability distribution function is calculated with the following equation [51]

$$p_i(X) = \frac{1}{\Gamma(k)\theta^k} X^{k-1} e^{\frac{-X}{\theta}} \tag{14}$$

where Γ is a general gamma distribution to calculate the probability, and k ($k > 0$) and θ ($\theta > 0$) are shape and scale parameters, respectively. The calculated probability for an agent in Equation (14), p_i, is used in Equation (13), and this is performed for each and every UAV agent in the system. The Tsallis entropy is calculated for each agent, where an agent-specific case-dependent variable is selected; e.g., operational time, liability (completion of tasks), computational workload, or capabilities (sensors onboard) can be the case-dependent variable. In this study, relative distance is used as the case-dependent variable. As the Tsallis entropy for each agent is calculated, it is shared amongst its neighbors. The primary goal is to minimize Tsallis entropy for the entire team of UAVs.

The broad overview is that there are two main actions of the developed algorithm. There is a grouping phase and a global objective (mission) phase. In our study, the mission phase is used as all the agents go to a waypoint, but it could be changed for another application. The Tsallis entropy is calculated for each agent for the selected specific case-dependent variable. Then, the value of each agent is shared with its neighbors. The goal is to minimize the entropy of the system while remaining stable.

Algorithm 1 is the developed algorithm. In our study, each agent is controlled in a distributed fashion depending on the entropy algorithms and its controller; details of the UAV controller are given in the last paragraph of Section 2. Algorithm 1 takes the current (x, y) and θ for the corresponding UAV, the desired (x, y) coordinate for that UAV, and the maximum velocity as inputs. Then, it outputs the commanded velocity and commanded angle for the corresponding UAV agent. First, in each agent separately, distances and angles between neighbor UAVs and the desired waypoint are calculated. Next, the distances between neighbor UAVs are compared, and the closest and farthest UAVs are identified. Then, the error is bounded between a maximum and minimum distance. Following that, entropy is calculated by summing e_i / d_{max}^q for the total number of neighbor UAVs and then dividing the sum by $q - 1$. The subsequent 'if' case proceeds as follows if entropy exceeds the threshold: First, the close UAVs are grouped. Then, the farther agents are grouped. Finally, if the UAVs are too close to each other, they move apart so as to not collide. The threshold allows the system to be in a grouping phase, in which UAVs move closer to each other, or in a mission phase where they move towards the global waypoint. Another variant of the developed algorithm is presented in Algorithm 2. This variation prioritizes aggressive grouping over smooth waypoint maneuvers, which can be seen by comparing commanded velocity (V_{com}) values between the two algorithms.

Algorithm 1: Pseudocode Representation of Algorithm 1

Input : $[X_{cur}, Y_{cur}, \theta_{cur}]$ for each UAV, (X_{des}, Y_{des}) global waypoint, V_{max}
Output: V_{com}, θ_{com}
Distances and angles are calculated between each UAV and then to the desired waypoint
Distances between UAVs are compared and the closest and farthest UAVs are identified
if $d_{betweenUAVs} > d_{max}$ **then**
  $e = d_{max}$
else if $d_{betweenUAVs} < d_{max}$ & $d_{betweenUAVs} > d_{min}$ **then**
  $e = d_{betweenUAVs}$
else if $d_{betweenUAVs} \leq d_{min}$ **then**
  $e = d_{min}$
$S_T = \frac{1-\sum_i^{numUAVs}(\frac{e_i}{d_{max}})^q}{q-1}$
$Threshold \longleftarrow S_T$
if $S_T < Threshold$ **then**
  if $d_{closestUAV} < d_{min}$ **then**
    $\theta_{com} = \theta_{waypoint} - \theta_{cur}$
    $V_{com} = V_{max}$
  else
    $\theta_{com} = (\theta_{index} - \theta_{cur}) + \pi$
    $V_{com} = V_{max}$
else if $d_{closestUAV} > d_{min}$ **then**
  $\theta_{com} = \theta_{index} - \theta_{cur}$
  $V_{com} = 2 * V_{max}$
else if $d_{furthestUAV} > d_{min}$ **then**
  $\theta_{com} = \theta_{index2} - \theta_{cur}$
  $V_{com} = 2 * V_{max}$
else if $d_{closestUAV}$ *or* $d_{furthestUAV} < d_{min}$ **then**
  $\theta_{com} = (\theta_{index} - \theta_{cur}) + \pi$
  $V_{com} = 1/2 * V_{max}$
Map all angles between $-\pi$ to π

Algorithm 2: Pseudocode Representation of Algorithm 2

Input : $[X_{cur}, Y_{cur}, \theta_{cur}]$ for each UAV, (X_{des}, Y_{des}) global waypoint, V_{max}
Output: V_{com}, θ_{com}
Distances and angles are calculated between each UAV and then to the desired waypoint
Distances between UAVs are compared and the closest and furthest UAVs are identified
if $d_{betweenUAVs} > d_{max}$ **then**
  $e = d_{max}$
else if $d_{betweenUAVs} < d_{max}$ & $d_{betweenUAVs} > d_{min}$ **then**
  $e = d_{betweenUAVs}$
else if $d_{betweenUAVs} \leq d_{min}$ **then**
  $e = d_{min}$
$S_T = \frac{1-\sum_i^{numUAVs}(\frac{e_i}{d_{max}})^q}{q-1}$
$Threshold \longleftarrow S_T$
if $S_T < Threshold$ **then**
  if $d_{closestUAV} < d_{min}$ **then**
    $\theta_{com} = \theta_{waypoint} - \theta_{cur}$
    $V_{com} = V_{max}$
  else
    $\theta_{com} = (\theta_{index} - \theta_{cur}) + \pi$
    $V_{com} = 1/2 * V_{max}$
else if $d_{closestUAV} > d_{min}$ **then**
  $\theta_{com} = \theta_{index} - \theta_{cur}$
  $V_{com} = 1/2 * V_{max}$
else if $d_{furthestUAV} > d_{min}$ **then**
  $\theta_{com} = \theta_{index2} - \theta_{cur}$
  $V_{com} = 1/2 * V_{max}$
else if $d_{closestUAV}$ *or* $d_{furthestUAV} < d_{min}$ **then**
  $\theta_{com} = (\theta_{index} - \theta_{cur}) + \pi$
  $V_{com} = 2 * V_{max}$
Map all angles between $-\pi$ to π

4. Parameter Analysis

Four parameters were tested to characterize the behavior of the developed algorithm. The trials ran until all three UAVs were within 20 m of the waypoint. The following are the cost functions used to compare different results.

$$f_1 = \sum_i^{numUAVs} \sqrt{V_X^2 + V_Y^2} \tag{15}$$

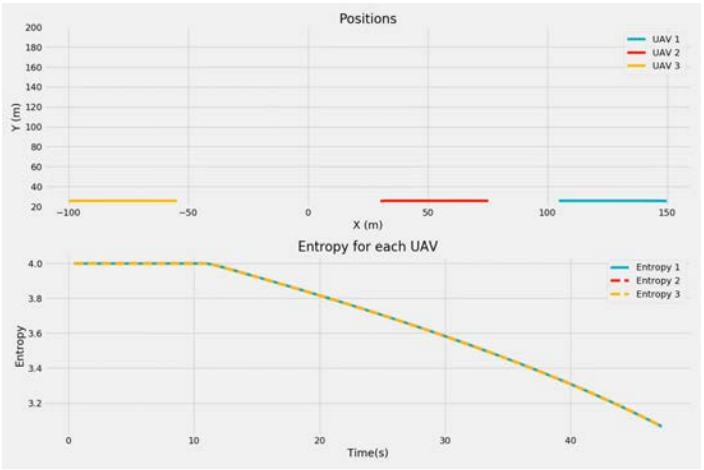
$$f_2 = \sum_i^{numUAVs} \sqrt{\Delta X_{Total}^2 + \Delta Y_{Total}^2} \tag{16}$$

The UAVs were controlled via velocity control, so the cost function in Equation (15) characterizes the summation of the total control inputs per UAV. The cost function in Equation (16) characterizes the summation of total distance traveled per UAV. No exact parameter set is the best in all cases for parameter analysis. The exact application dictates which parameter set is preferred. The resulting cost function values and the parameters used for each trial are given in Tables 1 and 2, respectively.

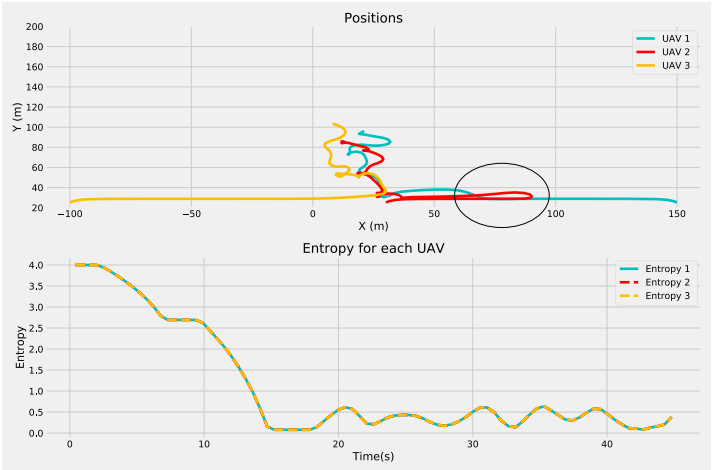
The parameters tested were threshold, maximum velocity (V_{max}), maximum distance (d_{max}), and q (Table 2). Minimum distance (d_{min}) is manipulated in the latter half of the

trials to prevent the entropy of the system from becoming negative. Minimum distance serves to prevent UAVs from getting too close, which in turn also prevents negative overall entropy calculations. The q is the Tsallis entropy constant given in Equation (13). Maximum distance is the divisor for the summation of entropy of the system. Maximum velocity is the highest allowed velocity per UAV. Threshold is the transition point in system entropy to switch from grouping to moving toward the global waypoint.

In this paper, the resulting plots have two parts; the top plots show the (x, y) position of UAVs, and bottom plots show the entropy variations of UAVs (Figure 3). All of the UAVs have similar beginning paths in trials, as depicted in Figure 3a; all trials except Trial 3 output some backtracking motion, as shown in Figure 3b. The best graphic visualization of this backtracking is in Trial 6, where the UAV backtracking section is circled for clarity. Trial 9 and 12 only had the UAVs within 30 m of the waypoint compared to the 20 m for all other trials due to the minimum distance of 30 m. General trends are not significantly affected by this, aside from some influence on f_2.



(**a**)

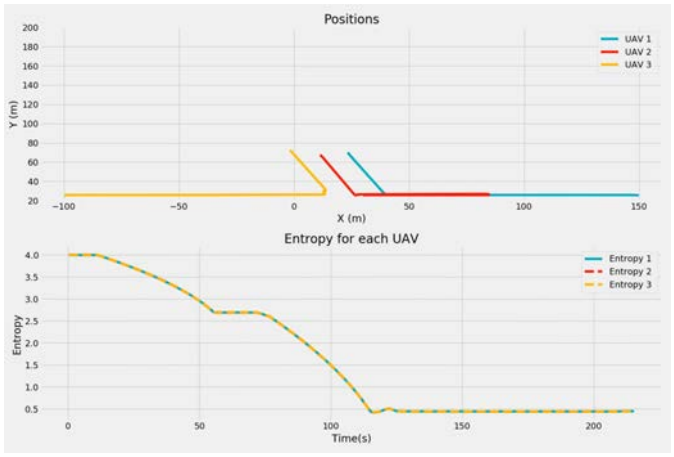


(**b**)

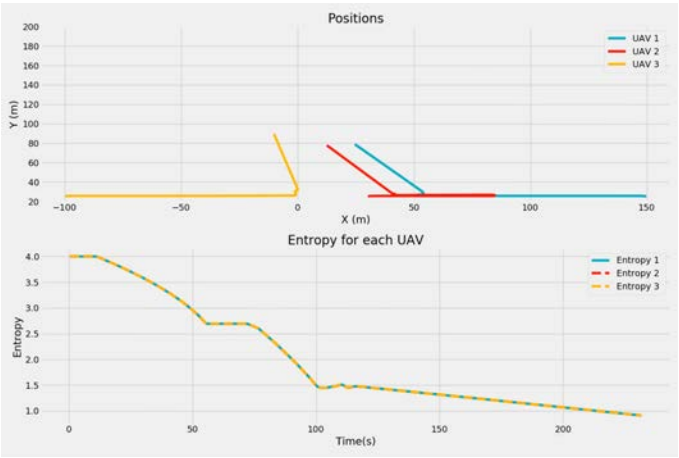
Figure 3. Examples of different beginning paths of UAVs during parameter trials: (**a**) usual beginning; (**b**) UAV2 backtracking in Trial 6.

Trials 1–3, shown in Figure 4, were performed by varying the threshold parameter. Depending on the threshold value, the entropy of the system kept decreasing as UAVs got closer (grouping phase); then, they started going to the waypoint (mission phase). UAVs ended up closest in Trial 1 (Figure 4a) before going to the waypoint. This resulted in the highest cost for velocities, f_1, and distance, f_2, in comparison to Trial 2 and 3.

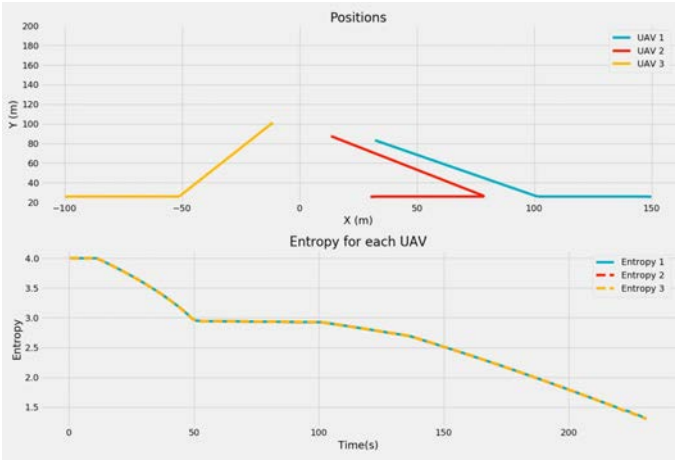
As the threshold increased, the cost functions f_1 and f_2 decreased because less grouping resulted in lower velocities and lower overall distances traveled to the waypoint. Trial 2, in Figure 4b, had UAV 1 and 2 group closely together then backtrack before prioritizing the global waypoint.



(**a**)



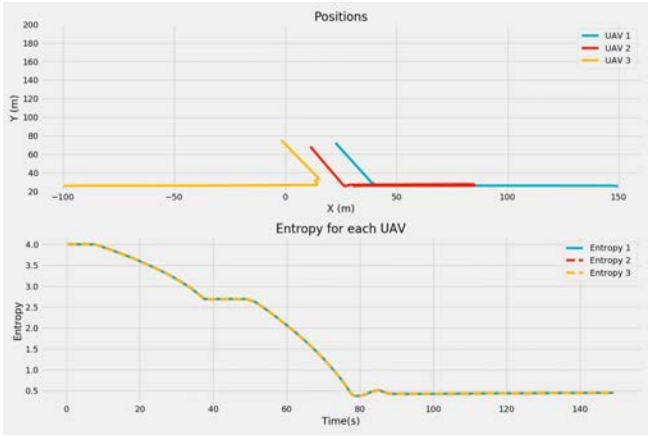
(**b**)



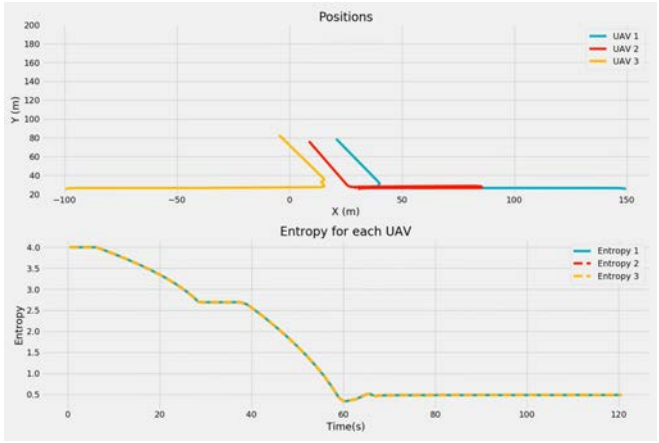
(**c**)

Figure 4. Results of parameter analysis—Trials 1–3: (**a**) Trial 1; (**b**) Trial 2; (**c**) Trial 3.

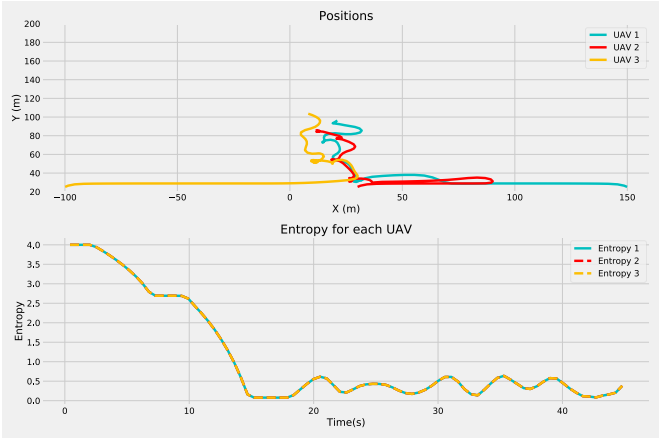
Trials 4–6 shown in Figure 5 varied the maximum velocity parameter. Higher maximum velocity resulted in less stable individual agent trajectories, but the system of agents still effectively reached the waypoint without colliding with the highest tested maximum velocity of 5 m/s. Trial 4 in Figure 5a showed a similar trajectory to Trial 1, as evident in the parameter costs and graphically. Trial 6 in Figure 5c had the highest cost of f_1 of any trial and the highest average f_2. Thus, the general trend is: as the maximum velocity increases, so do the cost functions f_1 and f_2.



(**a**)



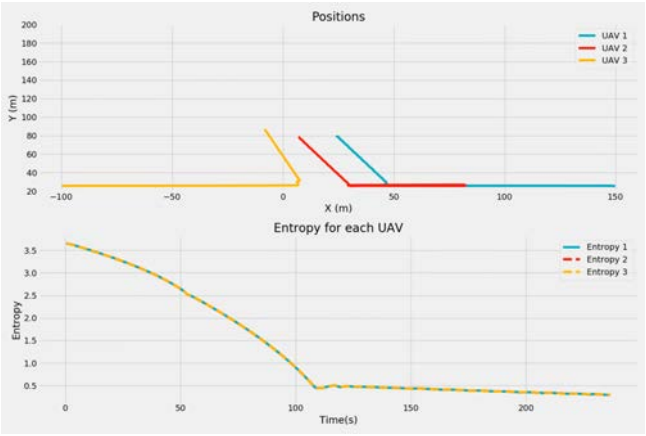
(**b**)



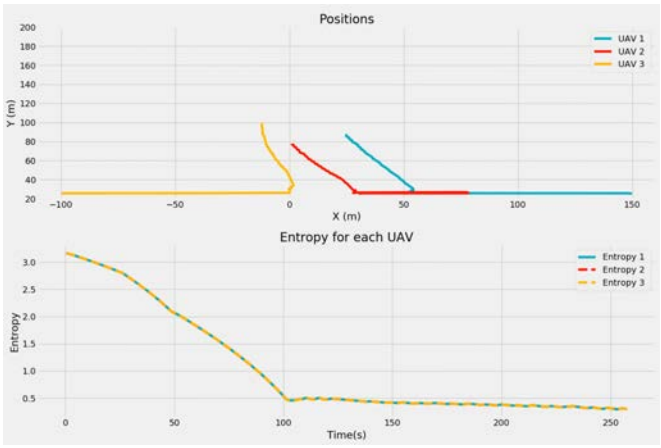
(**c**)

Figure 5. Results of parameter analysis—Trials 4–6: (**a**) Trial 4; (**b**) Trial 5; (**c**) Trial 6.

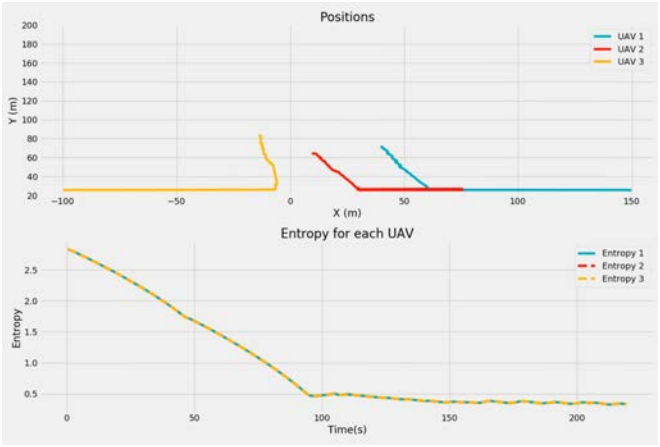
Trials 7–9 shown in Figure 6 varied the maximum distance parameter. Changing the maximum distance had minor effects on the overall behavior because minimum distance must be increased to prevent negative entropy in the system. Due to the minimum distance of Trial 9 being 30 m, the proximity to the waypoint was increased to 30 m. Trials 8 and 9 exhibited minor fluctuations in their trajectories, but the largest distinction is that the calculated overall entropy of the system is smaller than other parameter runs.



(**a**)



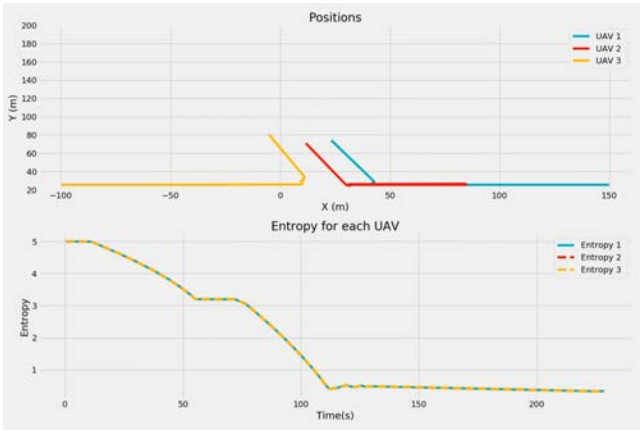
(**b**)



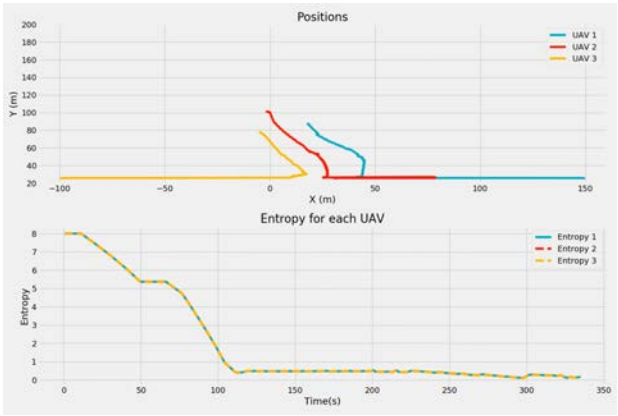
(**c**)

Figure 6. Results of parameter analysis—Trials 7–9: (**a**) Trial 7; (**b**) Trial 8; (**c**) Trial 9.

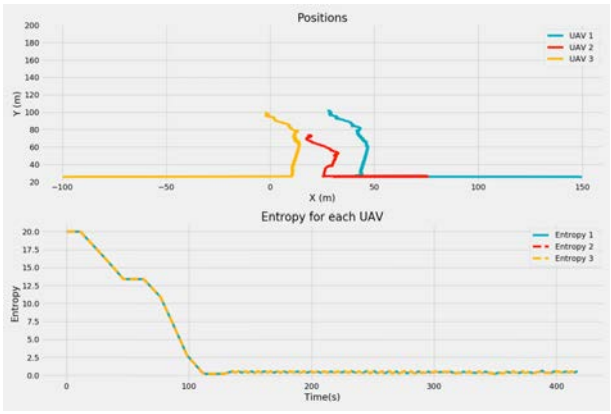
Trials 10–12, shown in Figure 7, varied the q parameter, which is given in Equation (13). The minimum distance also had to be changed for Trials 11 and 12 to ensure that negative entropy would not be mathematically possible. Additionally, in Trial 12, the proximity to the waypoint was increased to 30 m because of the minimum distance being 30 m, as in Trial 9. The q parameter greatly affects the scale of the entropy of the system. With a q of 0.6, 0.75, and 0.9, the highest entropy becomes 5, 8 and 20, respectively. With a higher q parameter, the trajectories become less smooth, which is undesirable in most applications. As the parameter q increases, the cost functions f_1 and f_2 also increase.



(**a**)



(**b**)



(**c**)

Figure 7. Results of parameter analysis—Trials 10–12: (**a**) Trial 10; (**b**) Trial 11; (**c**) Trial 12.

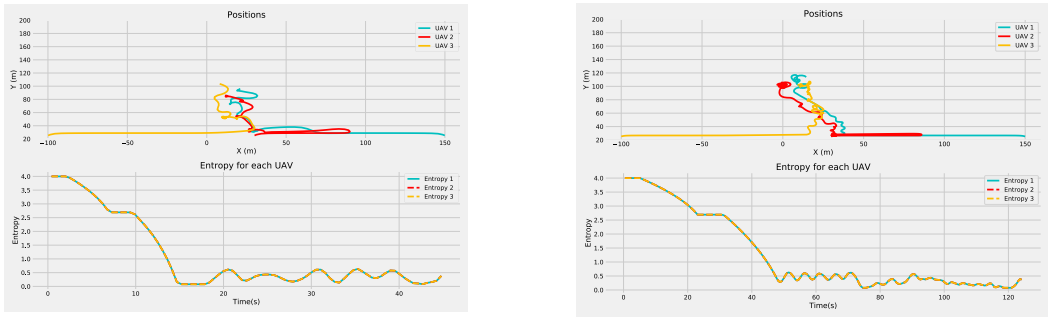
Table 1. Parameter Analysis Results—Cost Function Values, f_1 and f_2, in 12 Trials

	f_1UAV1	f_1UAV2	f_1UAV3	f_2UAV1	f_2UAV2	f_2UAV3
Trial 1	1706	1706	1706	177	174	177
Trial 2	1632	1632	1632	168	166	170
Trial 3	1387	1387	1387	145	144	140
Trial 4	1747	1748	1748	180	176	181
Trial 5	1755	1755	1755	179	175	181
Trial 6	3105	3105	3105	237	219	249
Trial 7	1672	1672	1672	172	169	174
Trial 8	1778	1778	1778	180	168	181
Trial 9	1593	1593	1593	161	151	165
Trial 10	1730	1730	1730	178	175	180
Trial 11	2244	2179	2244	199	206	195
Trial 12	2856	2560	2865	248	188	245

Table 2. Parameter Analysis Results—Parameter values used in 12 Trials.

	Threshold	V_{max}	d_{max}	q	d_{min}
Trial 1	0.5	0.5	100	0.5	12
Trial 2	1.5	0.5	100	0.5	12
Trial 3	3	0.5	100	0.5	12
Trial 4	0.5	0.75	100	0.5	12
Trial 5	0.5	1	100	0.5	12
Trial 6	0.5	5	100	0.5	12
Trial 7	0.5	0.5	150	0.5	17
Trial 8	0.5	0.5	200	0.5	25
Trial 9	0.5	0.5	250	0.5	30
Trial 10	0.5	0.5	100	0.6	12
Trial 11	0.5	0.5	100	0.75	24
Trial 12	0.5	0.5	100	0.9	30

Algorithm 1 was used for parameter analysis and multiple waypoint testing. Algorithm 2 resulted in better control with higher velocities. This is most apparent when maximum velocity was 5 m/s. The scalars for the commanded velocities are the only distinction between the two algorithms. Algorithm 2 exhibits unstable characteristics within the interaction between UAV agents, while Algorithm 1 exhibits higher mission smoothness. Algorithm 2 more aggressively avoids each agent, which might make it better for higher speed maneuvers. Algorithm 1 also reaches the desired waypoint in roughly half the time of Algorithm 2, which is desirable when time is a mission constraint. This is evident in Figure 8a,b. Further, in those figures it can be seen that system entropy has larger oscillations with Algorithm 1 than Algorithm 2. Since the oscillations are more closely bound in Algorithm 2, it can be concluded that Algorithm 2 is more robust to larger disturbances than Algorithm 1. This could be crucial for controlling the UAV system at high speed or with large disturbances, such as strong winds.



(**a**) (**b**)

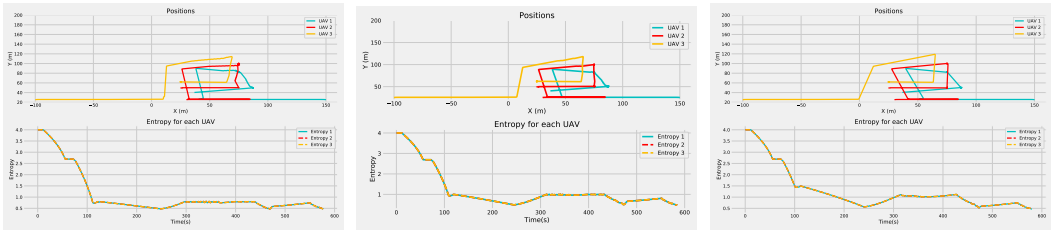
Figure 8. Comparison of (**a**) Algorithm 1 and (**b**) Algorithm 2 results—both Algorithms with V_{max} = 5 m/s.

5. Multiple Waypoint Simulation Results and Discussion

The developed entropy-based distributed behavior model could theoretically be applied to any multi-agent system. The developed algorithm has a handful of adjustable parameters to improve performance based on application. In this section, multiple waypoint navigation and an increased number of UAVs are presented to demonstrate a multiple-objective application and scalability. One limitation of the algorithm is that platform-specific tuning is needed to minimize the potential risk for the update rate to cause the UAV(s) to turn in circles during simulation experiments. An update rate of 0.1 s was used in simulation trials in AirSim, but it might be different if another platform is selected in the simulation or if the algorithm is applied to real hardware.

5.1. Multiple Waypoint Navigation with Three UAVs

Multiple waypoint navigation results are depicted in Figure 9. The UAVs navigate to waypoints in the following order: (25,100) to (75,100) to (75,50) to (25,50). As the selected entropy threshold increases, the smoothness of the UAV trajectories increased. Increasing the entropy threshold is another means by which to configure system grouping and waypoint priorities. Solely based on the smoothest trajectory, the threshold of 1.5 is the best, but that smoothness was at the cost of UAV formation tightness in the beginning. Multiple waypoints with a threshold of 1.0, as shown in Figure 9b, is an adequate mix of smooth trajectories and early formation cohesion. Reasonable waypoint placement also influences final trajectory smoothness, so lower thresholds would work better with unplanned or random multiple waypoints.



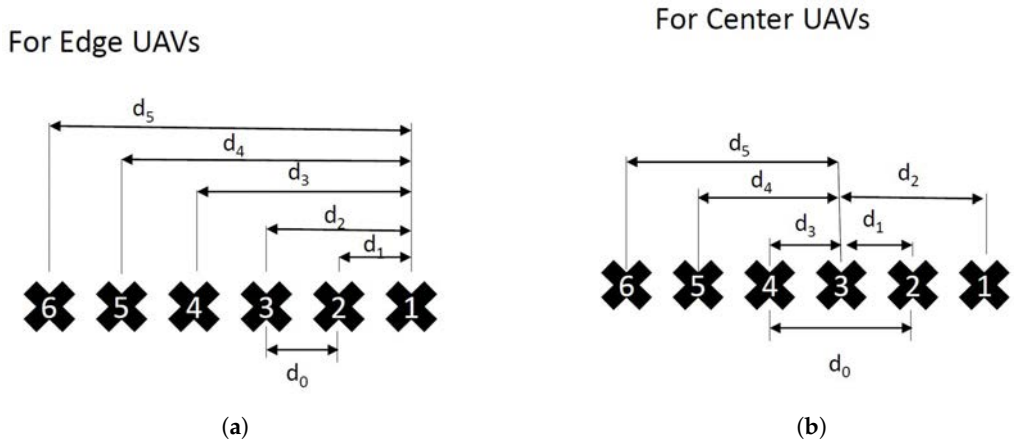
(**a**) (**b**) (**c**)

Figure 9. Multiple waypoint navigation with three UAVs: (**a**) Threshold = 0.8; (**b**) Threshold = 1.0; (**c**) Threshold = 1.5.

5.2. Multiple Waypoint Navigation with Six UAVs

An increased number of UAVs was also tested. In the simulations with six UAVs, UAVs first form a group, then navigate to the multiple waypoints. One main difference between application of the developed algorithm with three UAVs and with six UAVs is how distances are calculated. The distance calculation approach is presented in Figure 10a for edge UAVs and in Figure 10b for center UAVs. The distance between two UAVs closest to a UAV of interest is represented by d_0 for all cases. The other distances calculated are

relative to the UAV of interest; for instance, in Figure 10b, the distances are calculated relative to UAV 3. In UAV cases with three UAVs, distances between all UAVs (between UAV 1 and UAV 2, between UAV 2 and UAV 3, and between UAV 1 and UAV 3) are used in calculations for each UAV, which leads to all UAVs having the same entropy values. This is not same in the case with six UAVs. This results in the case with six UAVs having different entropy values for each UAV. Even though the case with six UAVs differs in this sense, the UAVs that are closest have similar entropy trends (Figure 11).



(**a**) (**b**)

Figure 10. Distance calculation approach for 6 UAVs: (**a**) edge UAVs; (**b**) center UAVs.

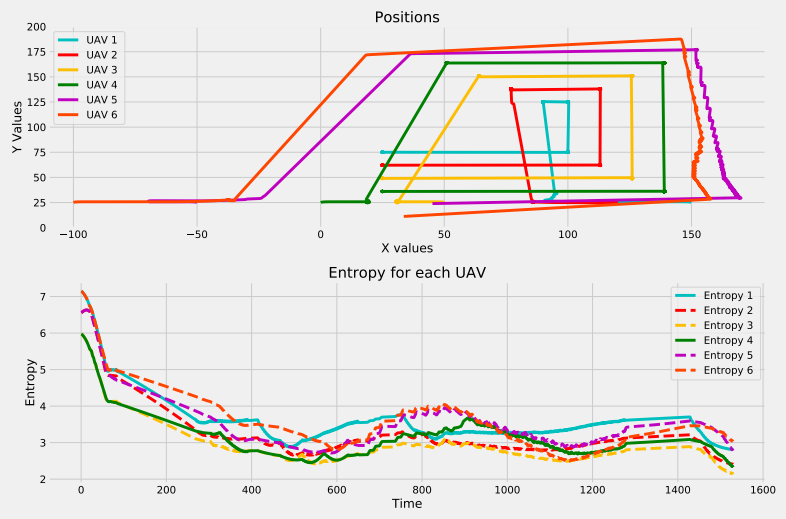


Figure 11. Multiple waypoint navigation with six UAVs.

5.3. Scalability and Stability Discussions

As new UAV agents are added to the system, they populate the environment in a linear pattern. As the multi-agent UAV team grows, so must the entropy threshold value. Close inspection of the entropy values for each UAV shows that as they move towards the first objective, they also reduce their entropy by decreasing the distance between each other. However, the system's overall entropy ends up relatively higher than in the case with fewer agents. This makes sense due to the fact that system entropy takes agent distance from each other into consideration. If they are apart from each other, the entropy will be higher; if they are close to each other, the entropy will be lower. Further, the entropy value limits in a group with more UAVs will be higher than a group with fewer UAVs. Thus, entropy threshold values definitely need to be adjusted when changing the number of UAVs in the group.

Moreover, the entropy threshold needs to be adjusted considering the stability of the system. The entropy threshold forms a "bubble" that the UAV group must stay inside.

A "bubble" that works for a group of three UAVs is physically too small, for example, for a group of 20 UAVs to occupy. Therefore, the entropy threshold must be increased based on how many UAVs are in the group in order to satisfy stability.

Analysis for stability was performed by running simulations with different numbers of UAV agents in the group. Entropy threshold values for groups of 3, 6, 9, 12, 15, 18, and 20 were calculated (Table 3), and the correlation between the number of UAVs and the entropy threshold is formulated with a linear equation as:

$$y = 1.11x - 2.57 \tag{17}$$

Using Equation (17), entropy threshold values for 5, 8, 11, 14, 17, and 19 UAVs were calculated and trials were run. In all cases, the mission was completed successfully using the calculated entropy threshold, and stability of the system was confirmed.

Table 3. Entropy threshold values by number of UAVs.

Number of UAVs	Entropy Threshold Value
3	1
6	3.9
9	7.3
12	10.6
15	14.3
18	18
20	19.7

6. Conclusions

In this paper, details of a novel, entropy-based distributed behavior model and parameter analysis of that behavior model for three simulated UAVs were presented. The UAV team was able to use the developed model to take off from different locations and then group together using Tsallis entropy. When the entropy of the system is high, the UAVs try to come closer to each other; that is called the grouping phase. Once entropy is less than a predefined threshold, the UAVs move to the global goal; this is called the mission phase. In this paper, the global goal was defined as navigating to a waypoint location. Results of three UAVs and six UAVs following multiple waypoints show robustness of the distributed behavior model to control multi-agent UAVs. One immediate future work for our study will be to compare the developed algorithm with other well-known methods or approaches in the literature. Future work will also focus on implementation of obstacle avoidance to determine robustness in complex environments, as well as implementation on real hardware.

Author Contributions: Conceptualization, software, validation, methodology, investigation, and writing—original draft preparation, L.F.; supervision and project administration, D.S.S.J.; software and validation, J.C.; supervision, project administration, and writing—review and editing, H.E.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the AFWERX, U.S. Air Force SBIR/STTR grant (Award Number: FA864921P1400).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mukhamediev, R.I.; Symagulov, A.; Kuchin, Y.; Zaitseva, E.; Bekbotayeva, A.; Yakunin, K.; Assanov, I.; Levashenko, V.; Popova, Y.; Akzhalova, A.; et al. Review of Some Applications of Unmanned Aerial Vehicles Technology in the Resource-Rich Country. *Appl. Sci.* **2021**, *11*, 10171.
2. Noor, N.M.; Abdullah, A.; Hashim, M. Remote sensing UAV/drones and its applications for urban areas: A review. In *IOP Conference Series: Earth and Environmental Science*; IOP Publishing: Bristol, UK, 2018; Volume 169, p. 012003.
3. Shakhatreh, H.; Sawalmeh, A.H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N.S.; Khreishah, A.; Guizani, M. Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges. *IEEE Access* **2019**, *7*, 48572–48634.
4. Barnes, L.; Fields, M.; Valavanis, K. Unmanned ground vehicle swarm formation control using potential fields. In Proceedings of the 2007 Mediterranean Conference on Control & Automation, Athens, Greece, 27–29 June 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 1–8.
5. Deng, Q.; Yu, J.; Wang, N. Cooperative task assignment of multiple heterogeneous unmanned aerial vehicles using a modified genetic algorithm with multi-type genes. *Chin. J. Aeronaut.* **2013**, *26*, 1238–1250.
6. Das, A.; Kol, P.; Lundberg, C.; Doelling, K.; Sevil, H.E.; Lewis, F. A Rapid Situational Awareness Development Framework for Heterogeneous Manned-Unmanned Teams. In Proceedings of the NAECON 2018-IEEE National Aerospace and Electronics Conference, Dayton, OH, USA, 23–26 July 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 417–424.
7. Das, A.N.; Doelling, K.; Lundberg, C.; Sevil, H.E.; Lewis, F. A Mixed Reality Based Hybrid Swarm Control Architecture for Manned-Unmanned Teaming (MUM-T). In Proceedings of the ASME 2017 International Mechanical Engineering Congress and Exposition (IMECE2017), Tampa, FL, USA, 3–9 November 2017; IMECE2017-72076.
8. Madey, G.R.; Blake, M.B.; Poellabauer, C.; Lu, H.; McCune, R.R.; Wei, Y. Applying DDDAS principles to command, control and mission planning for UAV swarms. *Procedia Comput. Sci.* **2012**, *9*, 1177–1186.
9. MacKenzie, D.C. Collaborative tasking of tightly constrained multi-robot missions. In Proceedings of the Multi-Robot Systems: From Swarms to Intelligent Automata: 2003 International Workshop on Multi-Robot Systems, 17–19 March 2003, Naval Research Laboratory, Washington, DC, USA; Volume 2, pp. 39–50.
10. Lundberg, C.L.; Sevil, H.E.; Das, A. A VisualSfM based Rapid 3-D Modeling Framework using Swarm of UAVs. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12–15 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 22–29.
11. Sauter, J.; Matthews, R.; Robinson, J.; Moody, J.; Riddle, S. Swarming unmanned air and ground systems for surveillance and base protection. In Proceedings of the AIAA Infotech@ Aerospace Conference and AIAA Unmanned... Unlimited Conference, Seattle, WA, USA, 6–9 April 2009, 2009; p. 1850.
12. Dasgupta, P. A multiagent swarming system for distributed automatic target recognition using unmanned aerial vehicles. *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* **2008**, *38*, 549–563.
13. Frew, E.; Xiao, X.; Spry, S.; McGee, T.; Kim, Z.; Tisdale, J.; Sengupta, R.; Hedrick, J.K. Flight demonstrations of self-directed collaborative navigation of small unmanned aircraft. In Proceedings of the AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit, Chicago, IL, USA, 20–23 September 2004; p. 6608.
14. Hinchey, M.G.; Sterritt, R.; Rouff, C. Swarms and swarm intelligence. *Computer* **2007**, *40*, 111–113.
15. Kolling, A.; Walker, P.; Chakraborty, N.; Sycara, K.; Lewis, M. Human interaction with robot swarms: A survey. *IEEE Trans. Hum. Mach. Syst.* **2015**, *46*, 9–26.
16. Rizk, Y.; Awad, M.; Tunstel, E.W. Cooperative heterogeneous multi-robot systems: A survey. *ACM Comput. Surv. (CSUR)* **2019**, *52*, 1–31.
17. Balch, T.; Arkin, R.C. Behavior-based formation control for multirobot teams. *IEEE Trans. Robot. Autom.* **1998**, *14*, 926–939.
18. Monteiro, S.; Bicho, E. A dynamical systems approach to behavior-based formation control. In Proceedings of the 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Washington, DC, USA , 11–15 May 2002; IEEE: Piscataway, NJ, USA, 2002; Volume 3, pp. 2606–2611.
19. Lawton, J.R.; Beard, R.W.; Young, B.J. A decentralized approach to formation maneuvers. *IEEE Trans. Robot. Autom.* **2003**, *19*, 933–941.
20. Fredslund, J.; Mataric, M.J. A general algorithm for robot formations using local sensing and minimal communication. *IEEE Trans. Robot. Autom.* **2002**, *18*, 837–846.
21. Caglioti, V.; Citterio, A.; Fossati, A. Cooperative, distributed localization in multi-robot systems: A minimum-entropy approach. In Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06), Prague, Czech Republic, 15–16 June 2006; IEEE: Piscataway, NJ, USA, 2006; pp. 25–30.
22. Monteiro, S.; Bicho, E. Attractor dynamics approach to formation control: Theory and application. *Auton. Robot.* **2010**, *29*, 331–355.
23. Xu, D.; Zhang, X.; Zhu, Z.; Chen, C.; Yang, P. Behavior-based formation control of swarm robots. *Math. Probl. Eng.* **2014**, *2014*, 205759.
24. Olfati-Saber, R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. Autom. Control* **2006**, *51*, 401–420.
25. Vásárhelyi, G.; Virágh, C.; Somorjai, G.; Nepusz, T.; Eiben, A.E.; Vicsek, T. Optimized flocking of autonomous drones in confined environments. *Sci. Robot.* **2018**, *3*.
26. Mirjalili, S. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Comput. Appl.* **2016**, *27*, 1053–1073.

27. Arafat, M.Y.; Moh, S. Localization and clustering based on swarm intelligence in UAV networks for emergency communications. *IEEE Internet Things J.* **2019**, *6*, 8958–8976.
28. Kamel, M.A.; Yu, X.; Zhang, Y. Real-Time Fault-Tolerant Formation Control of Multiple WMRs Based on Hybrid GA-PSO Algorithm. In *IEEE Transactions on Automation Science and Engineering*; IEEE: Piscataway, NJ, USA, 2020.
29. Zhang, J.; Yan, J.; Zhang, P. Multi-UAV Formation Control Based on a Novel Back-Stepping Approach. *IEEE Trans. Veh. Technol.* **2020**, *69*, 2437–2448.
30. Liu, W.; Gao, Z. A distributed flocking control strategy for UAV groups. *Comput. Commun.* **2020**, *153*, 95–101.
31. Neto, V.E.; Sarcinelli-Filho, M.; Brandão, A.S. Trajectory-tracking of a Heterogeneous Formation Using Null Space-Based Control. In Proceedings of the 2019 International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 11–14 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 187–195.
32. Lee, G.; Chwa, D. Decentralized behavior-based formation control of multiple robots considering obstacle avoidance. *Intell. Serv. Robot.* **2018**, *11*, 127–138.
33. Qu, X.; Wan, Y.; Zhou, P.; Li, L. Consensus-Based Formation of Second-Order Multi-Agent Systems via Linear-Transformation-Based Partial Stability Approach. *IEEE Access* **2019**, *7*, 165420–165427.
34. Fu, X.; Pan, J.; Wang, H.; Gao, X. A Formation Maintenance and Reconstruction Method of UAV Swarm based on Distributed Control. *Aerospace Sci. Technol.* **2020**, *104*, 105981.
35. Alonso-Mora, J.; Montijano, E.; Nägeli, T.; Hilliges, O.; Schwager, M.; Rus, D. Distributed multi-robot formation control in dynamic environments. *Auton. Robot.* **2019**, *43*, 1079–1100.
36. Fathian, K.; Safaoui, S.; Summers, T.H.; Gans, N.R. Robust 3D distributed formation control with collision avoidance and application to multirotor aerial vehicles. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 9209–9215.
37. Speck, C.; Bucci, D.J. Distributed UAV swarm formation control via object-focused, multi-objective SARSA. In Proceedings of the 2018 Annual American Control Conference (ACC), Milwaukee, WI, USA, 27–29 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 6596–6601.
38. Du, H.; Zhu, W.; Wen, G.; Duan, Z.; Lü, J. Distributed formation control of multiple quadrotor aircraft based on nonsmooth consensus algorithms. *IEEE Trans. Cybern.* **2017**, *49*, 342–353.
39. Yang, X.; Fan, X. A distributed formation control scheme with obstacle avoidance for multiagent systems. *Math. Probl. Eng.* **2019**, *2019*, 3252303.
40. Pickem, D.; Glotfelter, P.; Wang, L.; Mote, M.; Ames, A.; Feron, E.; Egerstedt, M. The robotarium: A remotely accessible swarm robotics research testbed. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1699–1706.
41. Cofta, P.; Ledziński, D.; Śmigiel, S.; Gackowska, M. Cross-Entropy as a Metric for the Robustness of Drone Swarms. *Entropy* **2020**, *22*, 597.
42. Albani, D.; Manoni, T.; Arik, A.; Nardi, D.; Trianni, V. Field coverage for weed mapping: Toward experiments with a UAV swarm. In Proceedings of the International Conference on Bio-inspired Information and Communication, Pittsburgh, PA, USA, 13–14 March 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 132–146.
43. Schranz, M.; Umlauft, M.; Sende, M.; Elmenreich, W. Swarm Robotic Behaviors and Current Applications. *Front. Robot. AI* **2020**, *7*, 36.
44. Arnold, R.D.; Yamaguchi, H.; Tanaka, T. Search and rescue with autonomous flying robots through behavior-based cooperative intelligence. *J. Int. Humanit. Action* **2018**, *3*, 18.
45. Kamel, M.A.; Yu, X.; Zhang, Y. Formation control and coordination of multiple unmanned ground vehicles in normal and faulty situations: A review. *Annu. Rev. Control.* **2020**, *49*, 128–144.
46. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 621–635.
47. Furrer, F.; Burri, M.; Achtelik, M.; Siegwart, R. Rotors—A modular gazebo mav simulator framework. In *Robot Operating System (ROS)*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 595–625.
48. Microsoft. Airsim/Simpleflight. Available online: https://microsoft.github.io/AirSim/simple_flight/ (accessed on 29 June 2022).
49. Silano, G.; Oppido, P.; Iannelli, L. Software-in-the-loop simulation for improving flight control system design: A quadrotor case study. In Proceedings of the 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), Bari, Italy, 6–9 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 466–471.
50. Tsallis, C. Possible generalization of Boltzmann-Gibbs statistics. *J. Stat. Phys.* **1988**, *52*, 479–487.
51. Can, F.C.; Bayram, Ç.; Toksoy, A.K.; Avşar, H.; Özdemir, S. Characterization of swarm behavior through pair-wise interactions by Tsallis entropy. In Proceedings of the 2nd Indian International Conference on Artificial Intelligence, Pune, India, 20–22 December 2005.