sensors

MDPI

Communication

Prediction of Flight Status of Logistics UAVs Based on an Information Entropy Radial Basis Function Neural Network

Qin Yang *, Zhaofa Ye, Xuzheng Li, Daozhu Wei, Shunhua Chen and Zhirui Li

School of Mechanical Engineering, Hefei University of Technology, Hefei 230009, China;
2019110116@mail.hfut.edu.cn (Z.Y.); 2020110193@mail.hfut.edu.cn (X.L.); weidaozhu@126.com (D.W.);
shchen@hfut.edu.cn (S.C.); lizhirui202105@163.com (Z.L.)
* Correspondence: yangqin@hfut.edu.cn; Tel.: +86-138-5510-0482

Abstract: Aiming at addressing the problems of short battery life, low payload and unmeasured load ratio of logistics Unmanned Aerial Vehicles (UAVs), the Radial Basis Function (RBF) neural network was trained with the flight data of logistics UAV from the Internet of Things to predict the flight status of logistics UAVs. Under the condition that there are few available input samples and the convergence of RBF neural network is not accurate, a dynamic adjustment method of RBF neural network structure based on information entropy is proposed. This method calculates the information entropy of hidden layer neurons and output layer neurons, and quantifies the output information of hidden layer neurons and the interaction information between hidden layer neurons and output layer neurons. The structural design and optimization of RBF neural network were solved by increasing the hidden layer neurons or disconnecting unnecessary connections, according to the connection strength between neurons. The steepest descent learning algorithm was used to correct the parameters of the network structure to ensure the convergence accuracy of the RBF neural network. By predicting the regression values of the flight status of logistics UAVs, it is demonstrated that the information entropy-based RBF neural network proposed in this paper has good approximation ability for the prediction of nonlinear systems.

Keywords: Internet of Things; logistics drones; RBF neural network; information entropy; nonlinear system

check for updates

Citation: Yang, Q.; Ye, Z.; Li, X.; Wei, D.; Chen, S.; Li, Z. Prediction of Flight Status of Logistics UAVs Based on an Information Entropy Radial Basis Function Neural Network. *Sensors* **2021**, *21*, 3651. https://doi.org/10.3390/s21113651

Academic Editor: Enrico Natalizio

Received: 23 April 2021
Accepted: 22 May 2021
Published: 24 May 2021

1. Introduction

The emergence of logistics UAVs solves the problem of goods transportation in remote and underdeveloped areas, which plays an important role in promoting the construction and development of the global economy [1,2]. In addition to this, high level urban traffic conditions are an issue for public management and in order to achieve precise and accurate traffic studies, the real traffic flow conditions in urban areas can be evaluated based on videos acquired by UAVs [3]. However, many issues have not been solved for the use of logistics UAVs, for example, short battery life, limited load, low payload, high operating cost, and the fact they are easily affected by the weather [4,5]. Part of the pulling force generated by an UAV is used to lift and stabilize the weight of fuselage and cargo, to resist wind and to complete forward and backward, left and right roll operations. Under different loads, the measurement of flight stability of logistics UAV needs to be solved. Therefore, the study of UAV flight state prediction is crucial.

Scholarly research on UAV flight state prediction is currently a very hot topic, which can be divided into two categories according to the different principles of UAV prediction: modeled prediction techniques and modeless prediction techniques.

Modeled prediction techniques predict the target trajectory based on the established target equations of motion combined with the predicted current state of the target. Porretta et al. [6] integrated the wind speed and lateral braking force of the aircraft to build a performance model of the aircraft, and used the aircraft navigation and intention information as the input parameters of the model to predict the flight trajectory based on the aircraft flight plan calculation. Prandini et al. [7] used Brownian motion to describe the uncertainty and modeled the aircraft motion with uncertainty, assuming that the aircraft maintains a uniform flight speed, through which the aircraft trajectory is predicted, and then the airspace complexity is evaluated based on the trajectory prediction results. Since the flight plan of the aircraft is not considered in the modeling process, the timeliness of the method is limited. Benavides et al. [8] proposed an UAV trajectory prediction algorithm based on a kinematic model from two perspectives of aircraft horizontal profile and altitude profile, and integrated factors such as fuel consumption and planned flight time, after which, the flight process was simulated by simulation experiments, and it was concluded that the fuel consumption during flight could be reduced and the operation quality of UAV could be improved by trajectory prediction.

It can be seen that the target motion models used by the modal trajectory prediction technique are usually based on certain assumptions, i.e., under certain ideal conditions under which they are built. Since the factors affecting the target trajectory are very complex in the actual operating environment, it is extremely difficult to accurately model this considering all the influencing factors, and the more accurate the modeling is, the less generalizable the model built often is, so multiple motion models may need to be built to match the motion states of the target at each time. Unlike the model trajectory prediction technique, the modeless trajectory prediction technique treats the trajectory data as time series and abstracts the trajectory prediction problem as a time-series prediction problem, thus avoiding the problem of modeling complex motions. Some of the model-free trajectory prediction techniques only use several trajectory points before the current moment to predict future trajectories, such as the asymptotically optimal target trajectory prediction algorithm proposed by Chen et al. [9]. These trajectory prediction methods use less data, less computation, and high generalizability, but they do not make full use of the historical trajectory information of the target, and only predict the trajectory based on the information of several trajectory points before the current moment, and when the predicted trajectory of the target is complex, these methods may not accurately grasp its future motion trend. The historical trajectory of the motion target contains its unique motion law, and some model-free trajectory prediction techniques can improve the accuracy of trajectory prediction by using a large number of historical trajectories of the predicted target, and the more representative trajectory prediction methods include trajectory prediction methods based on Gaussian mixture model, trajectory prediction methods based on neural network and trajectory prediction methods based on Markov model, etc. Wiest et al. [10] used a Gaussian mixture model for vehicle trajectory prediction, converting the trajectory data from a position coordinate representation to a representation by velocity and yaw rate, and bringing the above features into the Gaussian mixture model for training, which allows the prediction model to update more quickly when the direction of travel of the target vehicle is predicted to change. Qiao et al. [11] proposed an adaptive parametric trajectory prediction model based on Hidden Markov Model, which considers the velocity variation of the predicted target and improves the prediction efficiency of the algorithm by density-based processing of the trajectory division. Sanchez-Garcia et al. [12] accelerated the convergence speed of the particle swarm algorithm by improving the specific range of the environmental search to obtain the optimal trajectory faster. Yuan et al. [13] improved the robustness and convergence speed of the algorithm by constructing a geometric space structure to guide the particle swarm motion. Hafez et al. [14] proposed a method that combines control parameterization and time discretization with particle swarm optimization to solve the trajectory optimization problem. Song et al. [15] proposed a new multimodal delayed particle swarm optimization algorithm that reduces the occurrence of local convergence of the trajectory planning and improves the trajectory planning performance.

In general, the advantage of the modeless trajectory prediction technology is that the prediction target itself does not need to be modeled precisely, and it mainly starts from the temporal sequence of trajectory information, and learns the motion law of the prediction target by means of data mining and fitting, so as to achieve the prediction effect.

The development of RBF neural network provides a new method for industrial process modeling and control. Because of its simple structure and strong approximation ability, RBF neural network has been widely used in practical application and theoretical research [16–19]. RBF design is mainly about structure design and learning algorithm design. The design of algorithm has become more and more mature, but the structure design is still an unsolved problem, i.e., the radial basis function. Radial basis function neural network is a kind of efficient feedforward neural network, which has the best approximation performance and global optimal performance superior to other feedforward neural networks, and has simple structure and fast training speed. The RBF neural network is trained by using of experimental input/output data [18]. Due to the small number of input samples, the accuracy of the final convergence decreased. Aiming at the structural change and optimization of RBF neural network in training, Platt [19] proposed a kind of resource allocation network (RAN), which can increase hidden layer neurons, according to the complexity of processing objects. However, the continuous increase will lead to the decrease of calculation efficiency. Lu et al. [20] proposed a kind of minimum resource allocation (MRA network, MRAN), which can actively increase, or reduce the number of neurons in the hidden layer during network training. This has been approved by the structural design method of the neural network [21]. The adjustment of parameters has not been considered after the structural change, therefore, the convergence speed is slow. Lian et al. [22] proposed a self-organizing RBF (SORBF) neural network RBF. RBF is the basis of structural adjustment by continuously approaching the expected error, but it does not take into account the connection information between the hidden layer and the output layer, and the reset of parameters after structural adjustment. The training accuracy is therefore not high, and the convergence time is long. Huang et al. [23] proposed a kind of extended growing and pruning RBF (GGAP-RBF) neural network, by calculating the importance of hidden layer nodes. The nodes are added or deleted, but the initial value setting of network structure parameters needs to refer to the overall sample data, which is not suitable for online learning. Based on the above, it can be seen that structural optimization of RBF neural networks is still an open problem, especially the convergence of the dynamic structural adjustment process of RBF neural networks is still not well solved. Yu et al. [24] proposed an idea based on error correction, if the required performance is not achieved, the input corresponding to the point with the largest error after training is selected as the center of the next newly added neuron. This method can approximate the nonlinear function well using a more streamlined structure, but the method needs further research on the application to the actual process for information processing. Feng et al. used the particle swarm optimization (PSO) algorithm to optimize the center, width, and weights of the RBF network to achieve better modeling accuracy, but due to the global search capability of the algorithm, the training speed is slow and the algorithm is computationally large and complex, which is not conducive to real-time modeling [25,26].

In this paper, an information entropy-based dynamic tuning RBF (D-RBF) method is proposed to address the structural design and optimization of RBF neural networks when the effective input samples are small, and the information entropy-based RBF neural network is demonstrated to have good approximation ability for the prediction of nonlinear systems. The information entropy-based RBF neural network structure proposed in this paper is able to increase the activity and decrease the implicit layer neurons, which makes the D-RBF neural network more compact, has better approximation ability to nonlinear functions, high stability during training, and fast convergence. The theoretical validation of the convergence is given using the steepest descent learning algorithm [27], which proves the effectiveness of the method.

2. Information Entropy

The result of the quantitative measurement of uncertain information is called "information entropy", which is an integration based on probability theory and numerical statistical analysis to express the degree of dispersion of a system, based on the entropy value to determine the evolutionary direction of the system. The multi-dimensional information is quantified and synthesized [28–33], which has broad implications for problems with complexity and uncertainty. The definition formula of information entropy is:

$$H(x) = -\sum_{x \in \chi} p(x) \log p(x) \tag{1}$$

where *p(X)* is the probability of an event, and *H(X)* is a measure of the degree of discrepancy of uncertain information. In a RBF neural network, it can be used to measure the discreteness of a neuron activation data sample in the hidden layer.

The three properties of information entropy are: (1) monotonicity, the higher the probability of occurrence of events is, the lower the information entropy; (2) nonnegativity, information entropy is used to express the quantitative synthesis of information quantity of discrete system, where nonnegativity is a necessity; (3) accumulation, the synthesis of uncertainty quantity of system can be expressed as the sum of uncertainty measurement of discrete events.

(1) Mutual information, which shows the degree of intersection of event *X* and *Y*:

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log_2 \left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{2}$$

where: *p(x)*, *p(y)* is the probability of an event, *p(x, y)* is the joint probability of event *X* and event *Y*, and *I(X, Y)* is the degree of crossover between event *X* and event *Y*.

The sum of the information quantity of event *X* and *Y* alone, where the sum of the information quantity of event *X* and *Y* occurs at the same time. When event *X* and *Y* are completely independent, the interactive information interaction information represents the intersection degree of event *X* and *Y.*

Joint entropy, which measures the uncertainty of the simultaneous occurrence of event *X* and event *Y*.

$$H(X,Y) = -\sum_{y \in Y} \sum_{x \in X} p(x,y) \log\ p(x,y) \tag{3}$$

where *p(x, y)* is the joint probability of occurrence of event *X* and event *Y*, and *H (X, Y)* the measure of uncertainty of simultaneous occurrence of event *X* and event *Y*.

Conditional entropy, which measures the uncertainty of the occurrence of event *Y* at known event *X*.

$$H(Y/X) = \sum_{x \in X} p(x) H(Y/x) \tag{4}$$

where *H(Y/X)* is a measure of uncertainty about the occurrence of event *Y* for a known event *X*, and *p(x)* is the probability of an event occurring.

The relationship between the three is shown in Figure 1:

$$I(X,Y) = H(Y) - H(Y/X) = H(X) - H(X/Y) \tag{5}$$

where *I(X, Y)* is the interaction information of event *X* and event *Y*.

The entropy of neuron *i* reflects the discrete degree of input samples, activated by neuron *i*. the larger the entropy is, the greater the discrete degree of input samples. The strength of the connection between the hidden layer neuron *i* and the output layer neuron is measured by information entropy. The interaction information will be smaller with the smaller impact of the discrete samples activated by neuron *i* on the output neuron *j*, and the disconnection can be considered. The sample of neuron *i* activation become more concentrated and the effect on the output neuron *j* become greater as the interaction information is greater. It is partitioning neuron i, splitting or disconnecting neurons according to the information entropy of the hidden layer neurons and output layer neurons to improve the efficiency of network operation [34].

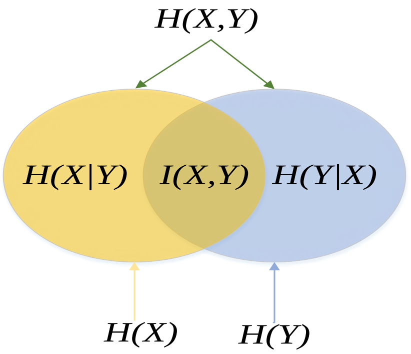


Figure 1. Event X and Y are connected by information entropy.

3. RBF Neural Network

The process of RBF neural network is just like searching a curved surface which can match the sample data in high-dimensional space, and it uses the curved surface to interpolate the test data to find the unknown points [35]. The structure is shown in Figure 2, where n is the number of input data samples, and m is the number of hidden layer nodes. The output data of this experiment is a set of four-dimensional vectors:

$$Y = [y_1, y_2, y_3, y_4] \tag{6}$$

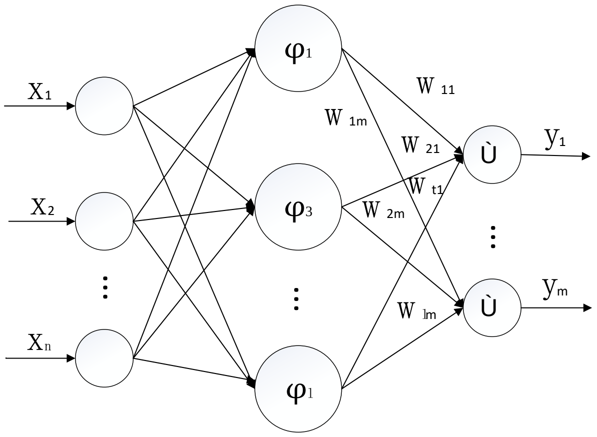


Figure 2. Structure of RBF neural network.

Select a group of test sample input data as:

$$X = [x_1, x_2, \ldots x_n] \tag{7}$$

The center of the i-th hidden layer node is C_i, and the activation function is:

$$exp(-\frac{1}{2\delta_i^2}\|c_i - x_i\|b_i), i = 1, 2, \ldots n \tag{8}$$

where b_i is the connection threshold between the i-th hidden layer neuron and the j-th output neuron, c_i and δ_i are the center and variance of node i of the hidden layer neuron, respectively, x_i is the test sample input data.

The output of RBF neural network is as follows:

$$Y_j = \sum_{i=1}^{M} \omega_{ij} exp(-\frac{1}{2\sigma^2} \|c_i - x_i\|^2 b_i) \tag{9}$$

where ω_{ij} *(i = 1, 2 . . . M, j = 1, 2, 3, 4)* is the connection weight of the *i*-th hidden layer and the *j* output unit, and *M* is the number of hidden layer neurons.

4. D-RBF Neural Network

D-RBF chooses whether to divide neurons or not dynamically according to the connection strength of neuron nodes. Enough hidden layer neurons can ensure that the RBF network approaches any nonlinear function. In order to dynamically adjust the number of neurons in the hidden layer during training, the initial neuron node of this experiment is smaller than the sample dimension. By introducing entropy and information theory, the entropy of neuron node *i* can reflect the discreteness of the sample parameters, activated by the node. As shown in Figure 3, After activating the input sample, we find that the sample becomes more concentrated when the dispersion of node is smaller, and the entropy of node *i* also becomes smaller; on the contrary, the entropy of node *i* is larger. *C* is the central point of the activation function, the yellow point is the input sample before activation, and the blue point is the probability distribution of the input sample after activation.

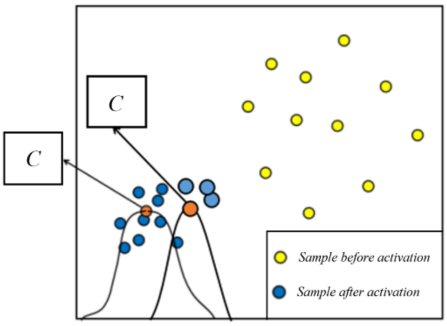


Figure 3. Dispersion of input samples after activation.

As shown in Figure 4, when samples *1, 2... p* are input to hidden layer *i,* the joint probability density function between node *i* and input samples is *f (p1, p2...p6, ai),* and the output of hidden layer neuron *i* is:

$$a_i = \sum_{q=1}^{6} w_{ij} \phi_{qi} \tag{10}$$

where W_{ij} is the connection weight of the *i*-th hidden layer to the *j*-th output unit, φ is the activation function of any hidden node.

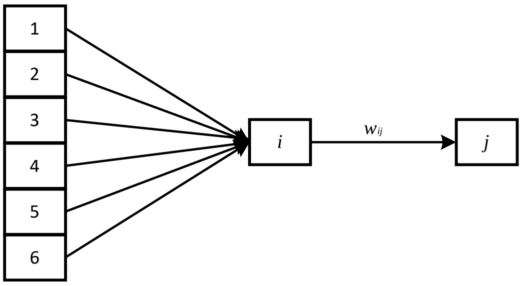


Figure 4. Sample data activation.

As shown in Figure 5, the output a_i of neuron *i* accounts for a small part of the total output y_j, the joint probability density function of M hidden layer neurons and network output y_j, is *f (a1, a2...a*$_M$*, y*$_j$*)*, and the output of output neuron j is:

$$y^j = \sum_{i=1}^{M} w_{ij}\phi\|c_i - p\| \tag{11}$$

where W_{ij} is the connection weight of the i-th hidden layer to the j-th output unit, c_i is the center of neuron node i in the hidden layer, p is the input sample, M is the number of neurons in the hidden layer, φ is the activation function of any hidden node, y^j is the entropy of the network output.

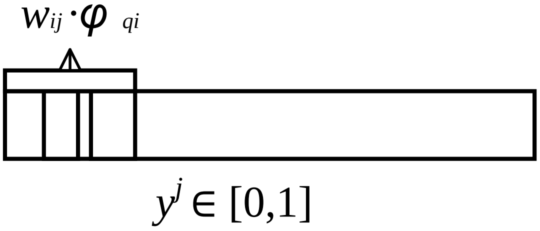


Figure 5. Composition of output neuron *j*.

The entropy of node i represents the degree of information dispersion among the p samples. The degree of comprehensive activation of neuron i to input samples increases when the entropy is smaller, leading to the greater attraction of neuron node i to p samples [36]. The information interaction between node i and node j is in the state of high load calculation, and small data error will lead to incalculable prediction results. In order to ensure the stability of network training, node i, the entropy of node j represents the difference of contribution of M neurons in the hidden layer to the total output. The large entropy indicates that the contribution of M neurons in the hidden layer to the output is large. The hidden layer neuron i with small contribution should be disconnected from the output layer neuron j.

The entropy of neuron node i can be expressed as:

$$H(i) = -\frac{1}{p}\log_2\frac{\phi_{1i}}{\sum_{q=1}^{p}\phi_{qi}} - \frac{1}{p}\log_2\frac{\phi_{2i}}{\sum_{q=1}^{p}\phi_{qi}} - \ldots - \frac{1}{p}\log_2\frac{\phi_{6i}}{\sum_{q=1}^{p}\phi_{qi}} \tag{12}$$

where φ is the activation function of any hidden node and p is the total number of samples.

The entropy of network output y^j is expressed as:

$$H\left(y^j\right) = -\frac{1}{M}\log_2\frac{\sum_{q=1}^{p} w_{1j}\cdot\phi_{q1}}{y^j} - \frac{1}{M}\log_2\frac{\sum_{q=1}^{p} w_{2j}\cdot\phi_{q2}}{y^j} - \ldots - \frac{1}{M}\log_2\frac{\sum_{q=1}^{p} w_{6j}\cdot\phi_{q6}}{y^j} \tag{13}$$

where W_{ij} is the connection weight of the i-th hidden layer to the j-th output unit, M is the number of neurons in the hidden layer, φ is the activation function of any hidden node, and p is the total number of samples, y^j is the entropy of the network output.

The contribution of output of neuron i to total output of network is given as:

$$H\left(i, y^j\right) = -\frac{1}{M}\log_2\frac{\sum_{q=1}^{p} w_{ij}\cdot\phi_{qi}}{y^j} \tag{14}$$

where W_{ij} is the connection weight of the i-th hidden layer to the j-th output unit, M is the number of neurons in the hidden layer, φ is the activation function of any hidden node, and p is the total number of samples, y^j is the entropy of the network output.

The adjustment of the connection relationship between node *i* and node *j* depends on the connection strength of node *i* and node *j I(i, j)*

$$I(i,j) = H(i) - H(i/j) = H(i) - \frac{H(i,y^j)}{H(y^j)} \tag{15}$$

Neuron *i* and *j* are independent of each other. When *I (i, j)* is large, the neuron node *i* and output node *j* have strong information interaction ability. Hidden layer node *i* has great attraction to *p* input samples, and node *i* may be in calculation state of high load [36]. At this time, node *i* is divided into 2 neurons, as shown in Figure 6.

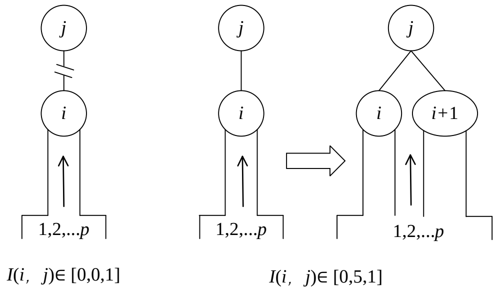


Figure 6. Division or disconnection of neuron I.

5. Proof of Convergence

Dynamic Adjustment of the Network Structure

When the network is training the *M*-group samples, the number of hidden layers is *n*, and the error is *e(k)* = *d(k)* − *y(k)*, *d(k)* is the expected output, *y(k)* is network output. The information association strength *I(i, j)* between the hidden layer node and the output layer node is calculated according to Equation (11). As shown in Figure 7, node *i* is divided into two nodes. The center point of the new node *j* has the same variance and the connection weight becomes half of the original value.

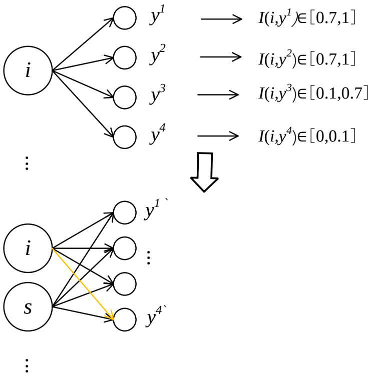


Figure 7. Neuron *i* splits into two, disconnects *i* from y^4.

Before and after the structural adjustment, the original and new output are:

$$y^1 + y^2 + \ldots + y^4 = \sum_{j=1}^{4} w_{ij}\phi\|c_i - p\| \tag{16}$$

$$\begin{aligned} y^{1\backslash} + y^{2\backslash} + \ldots + y^{4\backslash} &= \sum_{j=1}^{4} \tfrac{w_{ij}}{2}\phi\|c_i - p\| - w_{i4}\phi\|c_i - p\| + \sum_{j=1}^{4} \tfrac{w_{sj}}{2}\phi\|c_i - p\| \\ &= \sum_{j=1}^{4} w_{ij}\phi\|c_i - p\| - w_{i4}\phi\|c_i - p\| \end{aligned} \tag{17}$$

where W_{ij} is the connection weight of the i-th hidden layer to the j-th output unit, φ is the activation function of any hidden node before and after structural adjustment, c_i is the center of neuron node i in the hidden layer, p is the total number of samples, and y^j is the entropy of the network output.

Because the connection strength, between neuron i and y^4 output, is less than 0.1, the neuron i has a low activation degree to p samples, a large sample dispersion degree, and very small impact on output, which can be ignored. When network structure is adjusted, the connection between hidden layer neuron and y^4 is ignored. So $y(k) = y'(k)$, the error $e(k) = d(k) - y(k) = d(k) - y'(k)$.

During the training process, the structure of RBF neural network is adjusted, and the output error $e(k)$ of the network is not changed, which increases the convergence speed of the average error E.

$$E = \frac{1}{2p}\sum_{k=1}^{p} e^2(k) \tag{18}$$

where: e is the network output error and p is the number of samples

In the process of training, by splitting the neurons in the hidden layer with high connection strength, the connections with low connection strength are disconnected. Under the premise of constant output value, the stability of the output value of the network is guaranteed and the collection is improved convergence speed [27].

6. Prediction of Regression Value of UAV Flight Status

6.1. Establishment and Analysis of UAV Model

The logistics UAV consists of a power system, main control board, protective devices, mission load and landing gear. The motion state of the drone refers to vertical motion, pitch motion, transverse motion, and roll motion. To put it simply, vertical motion refers to ascending and descending, pitch motion refers to forward and backward motion, transverse motion refers to left and right motion, and roll motion refers to changing direction. The four-rotor motors control the attitude angle in flight, and the change of motor speed controls the change of attitude angle. Four propellers are distributed in a "x" cross pattern. The lift generated by the rotor supports the UAV's own weight and completes various flight maneuvers. The rotor generates lift as well as air resistance, so the choice of wing shape is critical to the amount of lift and drag. The model of the quadrotor transport UAV is shown in Figure 8.

In this experiment, the finite element model of the logistics UAV in ANSYS software is shown in Figure 9. In order to control the number of grids and ensure the quality of grids, the small structures, such as bolts and wires, are cleaned up, and the calculation domain of UAV outflow field is constructed by topology. Because the rotation region of propeller is the key area of calculation, the grid is encrypted to ensure the quality of calculation. Finally, the triangular mesh is 43,752, and the number of volume mesh is 1,156,169. After checking, the mesh quality meets the calculation requirements.

6.2. Flow Field Analysis during Flight

It can be seen from the streamline diagram (Figure 10) that, in the flight process, the downwash flow generated by the rotor is spiraling downward perpendicular to the rotating plane without considering the wind interference, the flow velocity is gradually reduced, and the lift is generated. At the same time, the airflow interference between the rotors is relatively small, which has little impact on the working efficiency of a single rotor. The flow field distribution is reasonable. According to the calculation results, the torque of a single rotor is 0.008N/m. The maximum lifting force is 0.61 kg, which meets the requirements of working conditions (fuselage 0.6 kg, motor 0.1 × 4 kg, load 0.6–1 kg). Within this range, the rotor + motor selection is reasonable. The maximum deformation (0.76 mm) of the UAV rotor is located at the tip of the rotor blade. The maximum stress is 1.4 MPa, which is less than the allowable stress 25 MPa of the plastic used in this paper. From the safety margin formula:

$$MS = \sigma_s/(\sigma_{max} * f) - 1$$

where MS is the safety margin; σ_s is the maximum permissible stress; σ_{max} is the calculated maximum stress; f is the safety factor, which is 1.5 in this paper, and the calculated safety margin at the maximum stress of the rotor blades is 16.85, which is greater than zero. The material safety of rotor is reasonable. In addition, according to cloud Figure 11, the deformation and stress of the structure, at the installation hole in the center, are small, with high safety margin and no damage.

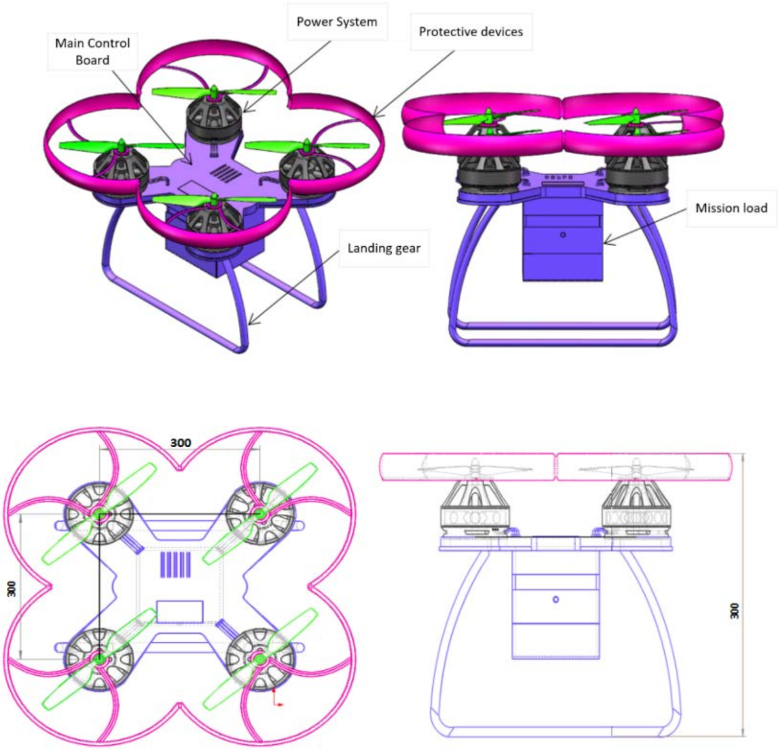


Figure 8. Schematic diagram of rotary wing logistics UAV.

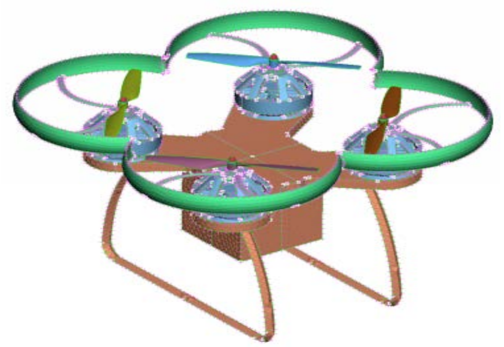


Figure 9. Finite element model of four rotor aircraft.

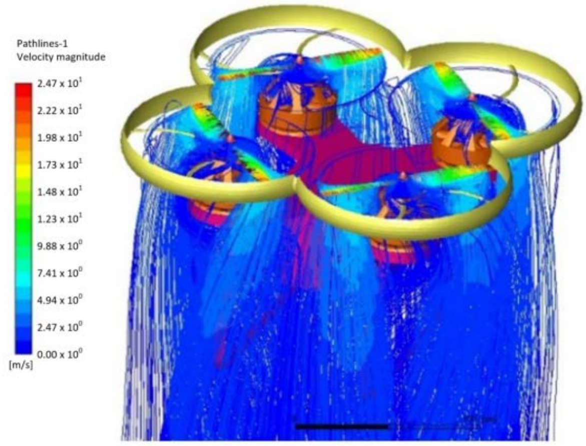
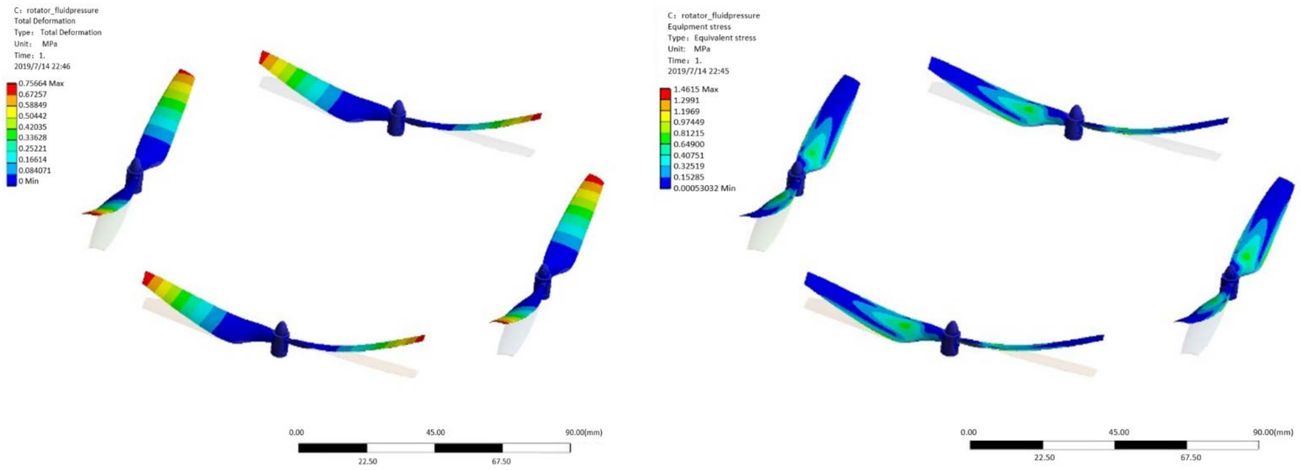


Figure 10. Flight flow field.



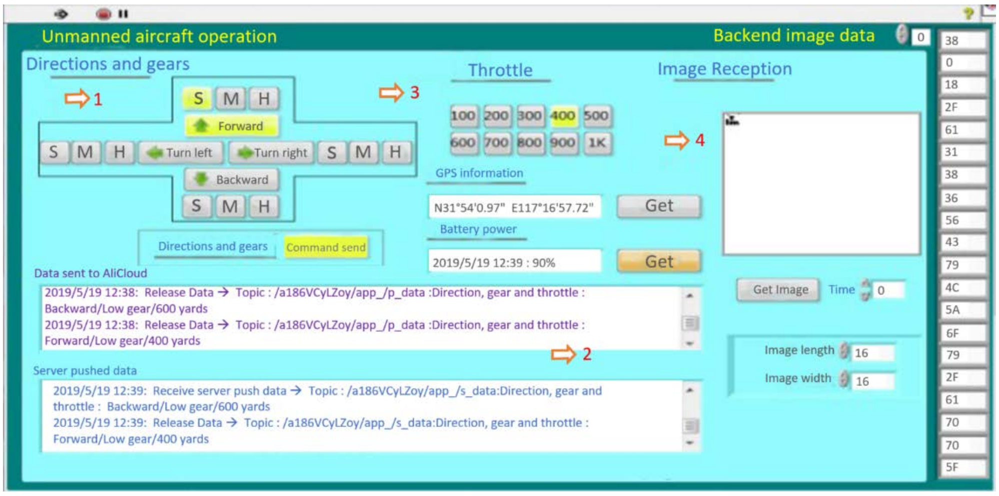
(**a**) (**b**)

Figure 11. Stress cloud map (**a**) and deformation cloud map (**b**) of rotor under lift and self-weight.

6.3. Flight Test

Three types of UAV (the maximum tension produced by battery with motor and propeller is known) are tested. In order to obtain the flight status of the logistics UAV under different weight ratios and different output tensions, in the experiment, the logistics UAV is equipped with different cargo loads to fly at different flight speeds, recording the maximum flight time, and making judgments and records on the flight stability within the maximum flight time. As shown in Figure 12, when the maximum pull force is known, different throttle can obtain different output pull force. The user interface sends commands to control the takeoff, landing and flight of the Internet of Things (IOT) logistics UAV. Part 1 is direction control and flight speed control; Part 2 is user interface sending instructions to cloud platform. After receiving the message, the cloud platform sends the message to the UAV. Part 3 is the quantitative output throttle to control the output pull of logistics UAV.

Table 1 records the flight status of logistics UAVs with different mass ratios and output tensions over the range time. According to the test results of four rotor propeller and the aerodynamic calculation analysis of UAV, the lift drag ratio of logistics UAV was ***L*/*D*** = 8.5. According to the barometric altimeter of logistics UAV, we calculate the time required to rise to 5 m, $t = \sqrt{2s/a}$, and the output tension, ***F* = *ma***. When the logistics UAV flies at a constant speed at a height of 5 m, the flight distance within the flight time is recorded, and the flight speed is obtained ***v* = *s*/*t*.** According to many groups of data, the flight state was recorded. The data was used to train RBF neural network, and the key nodes were selected for flight experiment.



N31°54'0.97" E117°16'57.72"

2019/5/19 12:39 : 90%

Data sent to AliCloud

2019/5/19 12:38: Release Data → Topic : /a186VCyLZoy/app_/p_data :Direction, gear and throttle : Backward/Low gear/600 yards
2019/5/19 12:38: Release Data → Topic : /a186VCyLZoy/app_/p_data :Direction, gear and throttle : Forward/Low gear/400 yards

Server pushed data

2019/5/19 12:39: Receive server push data → Topic : /a186VCyLZoy/app_/s_data:Direction, gear and throttle : Backward/Low gear/600 yards
2019/5/19 12:39: Release Data → Topic : /a186VCyLZoy/app_/s_data:Direction, gear and throttle : Forward/Low gear/400 yards

Figure 12. User interface connects with cloud platform through mqtt protocol to control Internet of things UAV.

Table 1. Experimental data of logistics UAV.

Number	Pulling Force (N)	Output Pull (N)	Flight Resistance (N)	Fuselage Weight (kg)	Flight Speed (m/s)	Flight Time (min)	Flight Condition Weights (Excellent, Moderate, Poor, Noisy)
1	3.2	1.76	0.21	1.1	3.58	9.21	(0.90, 0.05, 0.04, 0.01)
2	3.2	1.92	0.23	1.2	3.9	8.46	(0.87, 0.1, 0.03, 0)
3	3.2	2.08	0.24	1.4	4.14	6.24	(0.68, 0.1, 0.0.08, 0.1)
4	3.2	2.24	0.26	1.0	4.72	7.18	(0.88, 0.1, 0.02, 0)
5	3.2	2.4	0.28	1.7	4.10	3.76	(0.58, 0.08, 0.1, 0.24)
6	3.2	2.88	0.34	1.4	4.65	5.28	(0.74, 0.08, 0.06, 0.12)
7	3.2	3.04	0.36	1.1	6.08	5.46	(0.84, 0.13, 0.03, 0)
8	3.2	3.2	0.38	2.0	5.32	2.19	(0.21, 0.13, 0.17, 0.49)
9	4.0	2.6	0.31	1.0	4.37	7.92	(0.91, 0.06, 0.03, 0)
10	4.0	2.8	0.33	1.2	5.01	6.31	(0.89, 0.07, 0.04, 0)
11	4.0	3.2	0.38	1.6	5.98	5.41	(0.84, 0.09, 0.04, 0.03)
12	4.0	3.4	0.40	1.8	6.08	3.85	(0.71, 0.05, 0.03, 0.21)
13	4.0	3.6	0.42	2.1	6.01	2.77	(0.41, 0.15, 0.06, 0.38)
14	4.0	3.8	0.45	1.6	6.88	4.08	(0.69, 0.21, 0.06, 0.04)
15	4.0	4.0	0.47	1.4	7.04	4.14	(0.76, 0.16, 0.06, 0.02)
16	4.8	3.12	0.37	1.7	5.12	7.36	(0.91, 0.04, 0.05, 0)
17	4.8	3.36	0.40	1.9	6.08	6.01	(0.79, 0.09, 0.04, 0.08)
18	4.8	3.6	0.42	2.8	4.75	3.56	(0.40, 0.01, 0.08, 0.51)
19	4.8	4.08	0.48	2.1	5.63	4.28	(0.51, 0.25, 0.1, 0.14)
20	4.8	4.56	0.54	1.4	7.52	4.83	(0.87, 0.08, 0.05, 0)
21	4.8	4.8	0.56	1.9	7.49	4.27	(0.71, 0.18, 0.07, 0.04)

According to the experimental input/output data, the flight state of logistics UAV is a highly nonlinear change process. RBF neural network is suitable for regression fitting of the flight state of logistics UAV to obtain the best load ratio of UAV.

6.4. Normalization of Experimental Data

Input variables of the experiment are maximum pulling force, output pulling force, flight resistance, load capacity, flight speed and maximum flight time, and the input vector is:

$$x = [F_{\max}, f_{out}, f_{for}, m_{\max}, v_{ave}, t_{\max}] \tag{19}$$

The contents and expressions of the six parameters are inconsistent. The elements in the sample set are scaled to the known maximum and minimum values, and the original data is normalized. The conversion formula is shown in Equation (20). As shown in Table 2, after normalization, the process of finding the optimal output layer weight will become smooth, and the convergence speed will be faster and more stable.

$$x_i = x - x_i^{min} / x_i^{max} - x_i^{min} \tag{20}$$

Table 2. Six nodes and experimental results after the normalization process.

Normalized Processing Nodes	Maximum Pulling Force	Output Pull	Flight Resistance	Weight Capacity	Flight Speed	Endurance Time
1	0.32	0.176	0.021	0.11	0.358	0.921
2	0.32	0.192	0.023	0.12	0.39	0.846
3	0.32	0.208	0.024	0.14	0.414	0.624
4	0.32	0.224	0.026	0.10	0.472	0.718
5	0.32	0.240	0.028	0.17	0.410	0.376

6.5. Simulation Experiment

In this paper, the maximum pulling force, output pulling force, flight resistance, load, flight speed and maximum flight time of logistics UAV are used as input variables of the D-RBF neural network training. The output of network training is a set of four-dimensional vectors to represent the UAV's flight state in the maximum time. 21 sets of data are selected as training samples and 10 sets of data are selected as test samples. In the training process, the convergence time of D-RBF and the number of hidden layer nerves are shown in Figures 13 and 14. In Figure 13 as the number of training per second increases, the error becomes smaller and smaller, and when it exceeds 75 time/s. The change of error is no longer obvious.

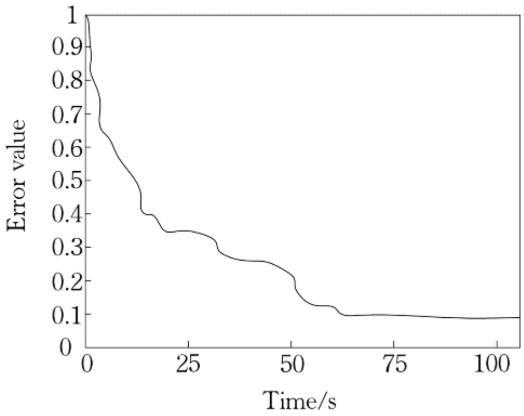


Figure 13. Error variation in convergence process.

In Figure 14, as the number of training per second increases, the left neurons of D-RBF first increase sharply, then tend to grow slowly, and its convergence time is 79.40; the left neurons of SORBF first increase sharply, then fall slightly in the middle, then tend to grow slowly, and its convergence The convergence time of SORBF is 84.21; the left neurons of GGAP-RBF rise sharply first, then fall slightly in the middle, and then tend to grow slowly, and the convergence time is 99.12; the left neurons of RBF do not change, and the convergence time is 236.72.

The evaluation index of network output error is established as:

$$e = \sum_{k=1}^{4} \left| d_p(k) - e_p(k) \right| \tag{21}$$

where $d_p(k)$ is the expected output and $e_p(k)$ is the network output error.

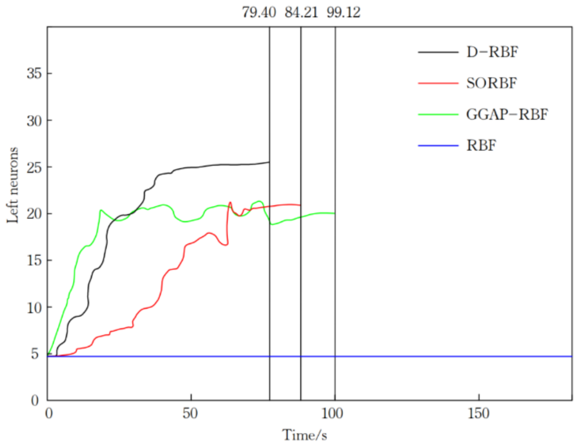


Figure 14. The number of neurons in hidden layer during network training.

The error between the expected value and the output value of ten groups of test samples is shown in Figure 15, In Figure 15 the total error changes significantly as the number of samples increases, but the total error lies in the interval of 0.09 to 0.18. The average performance of D-RBF, SORBF, RBF and GGAP-RBF, trained ten times respectively, is shown in Table 3.

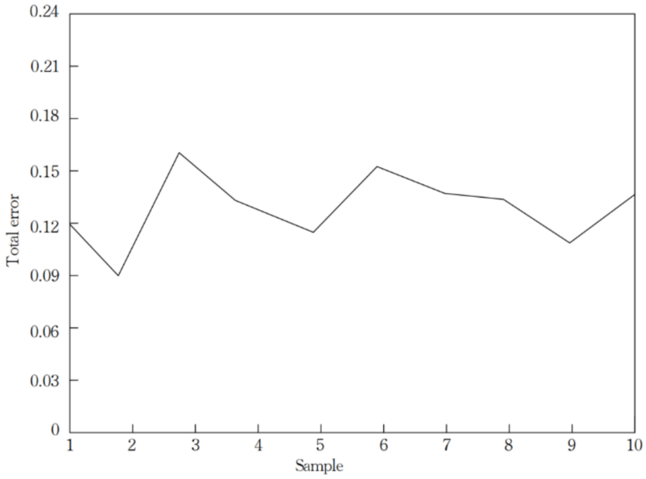


Figure 15. Error values of ten groups of test samples.

Table 3. Average performance comparison.

Function	Algorithm	Expected Error	Detection Error	Final Network	Convergence Time
Regression fitting	D-RBF	0.01	0.0138	24	79.40
	RBF	0.01	0.0326	6	236.72
	SORBF	0.01	0.0163	20	84.21
	GGAP-RBF	0.01	0.0142	19	99.12

The simulation results show that D-RBF can well predict the regression value of the nonlinear system of the logistics UAV. In Table 3, the number of neurons in the final neural network of D-RBF is 24, which is higher than that of RBF, SORBF and GGAP-RBF, the detection error of D-RBF is 0.0138, which is much smaller than that of RBF, SORBF and GGAP-RBF, and the convergence time of D-RBF is 79.40 times, is the smallest. This indicates that the average training time of D-RBF is better than the other three neural network algorithms, and the detection error is the smallest.

In this paper, we take the flight status prediction of UAV as a case study, and propose an information entropy-based RBF neural network structure adjustment and optimization method for the problem of inaccurate convergence that occurs when the effective training samples are small. It solves the problem of dynamic adjustment of RBF neural network structure in training, and D-RBF convergence analysis is provided. Compared with other RBF neural networks, the following conclusions are drawn. The conclusion is as follows:

(1) The D-RBF neural network does not depend on the initial structure of the network. It can dynamically adjust the number of neurons and disconnect the weakest connection, according to the connection strength of hidden layer neurons and output layer neurons. It can respond in real time.

(2) The entropy of hidden layer neuron and output layer neuron is calculated. The output information of hidden layer neuron and the connection strength between hidden layer neuron and output layer neuron are measured, and the mathematical expression is given to realize the dynamic adjustment of network structure.

(3) By experimentally comparing the performance differences among D-RBF, SORBF, RBF, and GGAP-RBF, the convergence speed of the average error is accelerated and D-RBF is proved to have good convergence performance.

(4) The present D-RBF neural network solves the regression prediction of the flight state of the logistics UAV, providing guidance for the research of the flight stability performance of the logistics UAV under different loads and flight speeds.

Author Contributions: Conceptualization, Q.Y.; methodology, Q.Y. and Z.Y.; software, Z.L.; validation, Q.Y. and D.W.; formal analysis, S.C.; resources, Q.Y.; data curation, Z.L.; writing—original draft preparation, Z.L.; writing—review and editing, Z.Y.; visualization, X.L.; supervision, Q.Y.; project administration, Q.Y.; funding acquisition, Q.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by China Scholarship Council, grant number 201706695019 and the National Natural Science Foundation of China, grant number 5177516. The APC was funded by Q.Y.

Institutional Review Board Statement: The study did not involve humans or animals.

Informed Consent Statement: The study did not involve humans.

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

References

1. Sheng, Z.; Yang, S.; Yu, Y.; Vasilakos, A.V.; McCann, J.A.; Leung, K.K. A Survey on the Ietf Protocol Suite for the Internet of Things: Standards, Challenges, and Opportunities. *IEEE Wirel. Commun.* **2013**, *20*, 91–98. [CrossRef]
2. Bodner, D.A.; Wade, J.P.; Watson, W.R.; Kamberov, G.I. Designing an Experiential Learning Environment for Logistics and Systems Engineering. *Procedia Computer Science* **2013**, *16*, 1082–1091. [CrossRef]
3. Salvo, G.; Caruso, L.; Scordo, A. Urban Traffic Analysis through an UAV. *Procedia Soc. Behav. Sci.* **2014**, *111*, 1083–1091. [CrossRef]
4. Perez-Montenegro, C.; Scanavino, M.; Bloise, N.; Capello, E.; Guglieri, G.; Rizzo, A. A Mission Coordinator Approach for a Fleet of UAVs in Urban Scenarios. *Transp. Res. Procedia* **2018**, *35*, 110–119. [CrossRef]
5. Lane, E.L.T.B.L. UAV market takes off for Intelligent Energy in inventory solutions. *Fuel Cells Bull.* **2017**. [CrossRef]
6. Porretta, M.; Dupuy, M.-D.; Schuster, W.; Majumdar, A.; Ochieng, W. Performance Evaluation of a Novel 4D Trajectory Prediction Model for Civil Aircraft. *J. Navig.* **2008**, *61*, 393. [CrossRef]
7. Prandini, M.; Putta, V.; Hu, J. A probabilistic measure of air traffic complexity in 3-D airspace. *Int. J. Adapt. Control Signal Proc.* **2010**, *24*, 813–829. [CrossRef]
8. Benavides, J.V.; Kaneshige, J.; Sharma, S.; Panda, R.; Steglinski, M. Implementation of a Trajectory Prediction Function for Trajectory Based Operations. In Proceedings of the Aiaa Atmospheric Flight Mechanics Conference, Atlanta, GA, USA, 16–20 June 2014.
9. Chen, L.; Shan, Y.; Tian, W.; Li, B.; Cao, D. A Fast and Efficient Double-Tree RRT*-Like Sampling-Based Planner Applying on Mobile Robotic Systems. *IEEE/ASME Trans. Mech.* **2018**, *23*, 2568–2578. [CrossRef]
10. Wiest, J.; Hoffken, M.; Kresel, U.; Dietmayer, K. Probabilistic trajectory prediction with Gaussian mixture models. In Proceedings of the Intelligent Vehicles Symposium (IV), Madrid, Spain, 3–7 June 2012.
11. Qiao, S.; Shen, D.; Wang, X.; Han, N.; Zhu, W. A Self-Adaptive Parameter Selection Trajectory Prediction Approach via Hidden Markov Models. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 284–296. [CrossRef]

12. Sanchez-Garcia, J.; Reina, D.G.; Toral, S.L. A distributed PSO-based exploration algorithm for a UAV network assisting a disaster scenario. *Future Gener. Comput. Syst.* **2018**, *90*, 129–148. [CrossRef]
13. Yuan, W.; Liu, Y.; Wang, H.; Cao, Y. A Geometric Structure-Based Particle Swarm Optimization Algorithm for Multiobjective Problems. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 2516–2537. [CrossRef]
14. Hafez, A.T.; Kamel, M.A. Cooperative Task Assignment and Trajectory Planning of Unmanned Systems Via HFLC and PSO. *Unmanned Syst.* **2019**, *7*, 65–81. [CrossRef]
15. Song, B.; Wang, Z.; Zou, L. On Global Smooth Path Planning for Mobile Robots using a Novel Multimodal Delayed PSO Algorithm. *Cogn. Comput.* **2017**, *9*, 5–17. [CrossRef]
16. Luengo, J.; García, S.; Herrera, F. A study on the use of imputation methods for experimentation with Radial Basis Function Network classifiers handling missing attribute values: The good synergy between RBFNs and EventCovering method. *Neural Netw.* **2009**, *23*, 406–418. [CrossRef] [PubMed]
17. Stefano, F.; Francesco, B.; Vincenzo, P.; Alberto, B.N. A hierarchical RBF online learning algorithm for real-time 3-D scanner. *IEEE Trans. Neural Netw.* **2010**, *21*, 275–285.
18. Bortman, M.; Aladjem, M. A growing and pruning method for radial basis function networks. *IEEE Trans. Neural Networks* **2009**, *20*. [CrossRef]
19. Platt, J. A Resource-Allocating Network for Function Interpolation. *Neural Comput.* **1991**, *3*, 213–225. [CrossRef]
20. Yingwei, L.; Sundararajan, N.; Saratchandran, P. A Sequential Learning Scheme for Function Approximation Using Minimal Radial Basis Function Neural Networks. *Neural Comput.* **1997**, *9*, 461–478. [CrossRef]
21. Panchapakesan, C.; Palaniswami, M.; Ralph, D.; Manzie, C. Effects of moving the center's in an RBF network. *IEEE Trans. Neural Networks* **2002**, *13*, 1299–1307. [CrossRef]
22. Lian, J.; Lee, Y.; Sudhoff, S.D.; Zak, S.H. Self-organizing radial basis function network for real-time approximation of continuous-time dynamical systems. *IEEE Trans. Neural Netw.* **2008**, *19*, 460–474. [CrossRef]
23. Huang, G.-B.; Saratchandran, P.; Sundararajan, N. A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Trans. Neural Netw.* **2005**, *16*, 57–67. [CrossRef]
24. Yu, H.; Reiner, P.D.; Xie, T.; Bartczak, T.; Wilamowski, B.M. An Incremental Design of Radial Basis Function Networks. *IEEE Trans. Neural Networks Learn. Syst.* **2014**, *25*, 1793–1803. [CrossRef]
25. Hao, C.; Yu, G.; Xia, H. Online modeling with tunable RBF network. *IEEE Trans. Cybern.* **2013**, *43*, 935–947.
26. Feng, H.-M. Self-generation RBFNs using evolutional PSO learning. *Neurocomputing* **2006**, *70*, 241–251. [CrossRef]
27. Buzzi, C.; Grippo, L.; Sciandrone, M. Convergent Decomposition Techniques for Training RBF Neural Networks. *Neural Comput.* **2001**, *13*, 1891–1920. [CrossRef]
28. Li, D.; Wang, Z.; Cao, C.; Liu, Y. Information entropy based sample reduction for support vector data description. *Appl. Soft Comput.* **2018**, *71*, 1153–1160. [CrossRef]
29. Harte, J.; Newman, E.A. Maximum information entropy: A foundation for ecological theory. *Trends Ecol. Evol.* **2014**, *29*, 384–389. [CrossRef]
30. Zheng, K.; Wang, X. Feature selection method with joint maximal information entropy between features and class. *Pattern Recognit.* **2018**, *77*, 20–29. [CrossRef]
31. He, Q.P. *Information Entropy Based Indices for Variable Selection Performance Assessment*; Computer Aided Chemical Engineering: San Diego, CA, USA, 2018; pp. 2281–2286.
32. Majumdar, S.; Mukherjee, N.; Roy, A.K. Information entropy and complexity measure in generalized Kratzer potential. *Chem. Phys. Lett.* **2018**, *716*, 257–264. [CrossRef]
33. Santos, W.D.D. The entropic and symbolic components of information. *BioSystems* **2019**, *182*, 17–20. [CrossRef]
34. Neves, G.; Cooke, S.F.; Bliss, T.V.P. Synaptic plasticity, memory and the hippocampus: A neural network approach to causality. *Nat. Rev. Neurosci.* **2008**, *9*, 65–75. [CrossRef] [PubMed]
35. Broomhead, D.S.; Lowe, D. Multivariable Functional. Interpolation and Adaptative Networks. *Complex Syst.* **1988**, *2*, 321–355.
36. Erdelj, M.; Król, M.; Natalizio, E. Wireless Sensor Networks and Multi-UAV systems for natural disaster management. *Comput. Netw.* **2017**, *124*, 72–86. [CrossRef]