Adaptive Radial Basis Function Neural Network Controller for Autonomous Multirotors

Afshin Banazadeh and Ardalan Samadzadeh

***Abstract*—Limits in the real-time computation of micro-processors on the one hand and high-level controllers that need precise computation in addition to implementation and compiling issues, on the other hand, have caused a great gap between control science and experiments. In this work, an adaptive RBF neural network controller is proposed to control the position and attitude of an autonomous multirotor. The controller is combined with an EKF observer and simulated in real-time flight conditions. In order to check the capabilities of the system, the proposed structure has been successfully applied to a quadrotor and a hexarotor using three data types for getting similar real-time flight results. Based on the results, the proposed structure has accomplished two separate missions with different scenarios, though there exists some error, especially in position estimations which are caused by step-wise characteristics of the desired path.**

***Index Terms*—Radial basis function (RBF), neural network (NN), adaptive control, extended kalman filter (EKF), multirotor.**

I. Introduction

Unmanned Aerial Vehicle (UAV) has provided a useful and economic platform for many applications in a wide range of usage (e.g. military, disaster, rescue operations, civil operations, photography, remote sensing, etc.).

In the class of UAV, multirotor has been concentrated much due to its specific features such as Vertical Take-off Landing (VTOL), and high maneuverability. Although they are applicable in many areas, they have a great potential to extend their abilities and become more autonomous. According to [1] which provides a concept of autonomous control level for UAVs, they have to pass several areas, including Perception, Situational Awareness, Analysis, Coordination, Decision Making, and Capability to become fully autonomous.

By considering these changes in UAVs, their missions and application can be changed completely so that the situation would not be predictable. These changes can be any anomalies in the plan, the structure of the UAV, or the environment. For mission accomplishment, the UAV has to be adapted to such changes. According to [2], using conventional controllers known as linear controllers will not lead to the elimination of the inherent instability of time-varying nonlinear systems such as UAVs with time-varying characteristics or in-flight dynamic changes. To adapt the system to such changes, an intelligent controller is required in the first step. There have been studies in the case of purposing intelligent controllers for multirotor UAVs.

Since the feedback of a multirotor UAV is strict feedback of states it is appropriate for designing backstepping control [3], [4]. In such cases, steady state errors can be eliminated by using the integral of error [5]. Using adaptive methods also, are becoming more popular which let users to implement simple controller which its parameters can be updated through classical methods or the heuristic ones such as combining those techniques with neural networks [6]-[8].

Furthermore, fuzzy logic system (FLS) and neural networks (NN) based controllers have recently caught the attention of researchers [6], [9]-[12].

One step ahead in making a UAV intelligent is to apply vision based controllers using image features to feed high level controllers such as sliding mode and backstepping [13], [14].

As has been mentioned earlier, neural networks have gained attention in the subject of control and this is because of their outstanding ability to approximating nonlinear functions. There are many concepts of neural networks that each have a specific feature that distinguishes them from other types. Among those paradigms exist, Radial Basis Functions Neural Networks (RBFNN) have been demonstrated to be more capable in nonlinear control. They use the Euclidean distance between the input vector and the center of a Gaussian function.

The distinctive feature of the RBFNN is a local approximation to nonlinear input-output mapping. Their main advantages are a short training phase and a reduced sensitivity to the order of presentation of training data. For this, they have been widely used to control a class of nonlinear systems and their stability has been proved [15]. In some cases, they have been combined with other controllers to update their gain or training those [16], [17] and implemented to directly control the system [18].

Most of the methods mentioned earlier have been verified in simulation and no further action including deploying on hardware or real-time flight is done. The level up progress in autonomy demands more actuators and sensors besides of high-level Guidance Navigation Control (GNC) system. In this case, a great amount of data is needed to be processed which is out the capability of a normal, onboard and commercial micro-controller or micro-processor.

One of the approaches toward this situation is to use small, high speed and parallel processors such as Field Programmable Gate Array (FPGA) and Digital Signal Processor (DSP).

Both FPGA and DSP provide parallel processing (all the bit at a time) and having large numbers of gates on a single

Manuscript received March 29, 2021; revised July 20, 2021.

A. Samadzadeh and A. Banazadeh are with the Department of Aerospace Engineering, Sharif University of Technology, Tehran, Iran (e-mail: Banazadeh@sharif.edu).

DOI: 10.7763/IJMO.2021.V11.787

chip (essential for less volume) which are necessary for small UAVs that are constrained to less volume and weight. They also provide cost savings platforms. The software of the systems would also have a huge impact on the overall design, for which code parallelization provides faster speed in referred architectures and other hardware components. Following this path, there are some researches and programs to use these processor architectures as the brain of the autopilot which extends the capabilities of the UAV to carry a camera and using image processing, linking manipulators, or deploying other practical algorithms for the mission along with the control system [19]-[25].

The gap between the results of the simulation and implementation of the algorithm into the processor is the precision of the calculation and some practical limits which exist in the real-time flight such as the time difference between the processor, sensors data acquisition, and digitalized UAV's equations of motion. The software simulations are usually run using double-precision floating-point calculation whereas all the small processors including DSP use single-precision floating-point or fixed-point data with less than 32 bits memory. For an FPGA also, since the designer decides the structure of the processor, there is an option to apply a more accurate data type. On the other hand, by increasing the precision, the more space of the structure is dedicated to data precision, the less space would be available for the algorithm. In this case, fixed-point data are often preferred due to their simple protocol against floating-point data type.

The main question here is, are proposed algorithms generated through single floating-point or fixed-point precision, valid? Besides, some of these methods such as sliding mode need derivation of the data of the states which add undesirable noise and need precise calculations.

Motivated by the above discussion, an RBFNN controller has been adopted for controlling multirotor VTOL UAVs and investigated to be able of running under single floating-point and fixed-point data precision. To reach the results of real-time flight, some conditions and real-time limits have been considered consisting of three-time steps; 1) simulation time step, 2) processor time step, and 3) sensor data acquisition time step. Saturation for each rotor has also been applied in addition to a common Extended Kalman Filter (EKF) which represents the filtering part of the system for accurate estimation of states.

II. Dynamic Model of a Multirotor UAV

A multirotor is a nonlinear dynamic system including high nonlinear terms. The dynamics of the multirotor UAV is the relationships between kinetics and kinematics equations of the system. In the following equations $[x \quad y \quad z]^T$ is the position vector of the UAV in the earth frame illustrated in Fig. 1. Also $[\phi \quad \theta \quad \psi]^T$ represents the roll, pitch and yaw angles. To the model would be derived through the Newton-Euler equation and can be represented as follows [26]:

$$\ddot{x} = \frac{u_1}{m}(\mathrm{C}(\psi)\mathrm{S}(\theta)\mathrm{C}(\phi) + \mathrm{S}(\psi)\mathrm{S}(\phi)) - \frac{k_1\dot{x}}{m} \tag{1}$$

$$\ddot{y} = \frac{u_1}{m}(\mathrm{S}(\psi)\mathrm{S}(\theta)\mathrm{C}(\phi) - \mathrm{C}(\psi)\mathrm{S}(\phi)) - \frac{k_2\dot{y}}{m} \tag{2}$$

$$\ddot{z} = \frac{u_1}{m}(\mathrm{C}(\theta)\mathrm{C}(\phi)) - \mathrm{g} - \frac{k_3\dot{z}}{m} \tag{3}$$

$$\ddot{\phi} = \frac{1}{I_x}\left((I_y - I_z)\dot{\theta}\dot{\psi} + J_r\dot{\theta}\Omega_r + lu_2 - k_4 l\dot{\phi}\right) \tag{4}$$

$$\ddot{\theta} = \frac{1}{I_y}\left((I_z - I_x)\dot{\phi}\dot{\psi} + J_r\dot{\phi}\Omega_r + lu_3 - k_5 l\dot{\theta}\right) \tag{5}$$

$$\ddot{\psi} = \frac{1}{I_y}\left((I_x - I_y)\dot{\phi}\dot{\theta} + Cu_4 - k_6\dot{\psi}\right) \tag{6}$$

where $C(.)$ and $S(.)$ are respectively cos () and sin () and $\mathrm{k}_i, (i \in \{1, \dots, 6\})$ denotes the drag coefficient of each motion. Also I_x, I_y, I_z indicate the diagonal inertial matrix of the UAV and J_r represents inertial moment of the rotors and

$$\Omega_r = \sum_{i=1}^{n}(-1)^i\omega_i \tag{7}$$

which ω_i is the rotational velocity of each rotor and n denotes the number of the rotors. The control inputs of the UAV are related to the ω_i through the following equations:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b & \dots & b \\ c_1^1 lb & c_2^1 lb & c_3^1 lb & c_4^1 lb & \dots & c_n^1 bl \\ c_1^2 lb & c_2^2 lb & c_3^2 lb & c_4^2 lb & \dots & c_n^2 bl \\ -d & d & -d & d & \dots & (-1)^i d \end{bmatrix}_{4\times n} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \\ \dots \\ \omega_n^2 \end{bmatrix}_{n\times 1} \tag{8}$$

where l, b, d stand for the length of each arm of the UAV, lift and drag coefficient of the rotors. The matrix c^1, c^2 are also the geometry coefficient of the related dynamic for the inputs. Equation (8) turns into (9) and (10) for quadrotor and hexarotor as instances.

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ lb & 0 & -lb & 0 \\ 0 & -lb & 0 & lb \\ -k & k & -k & k \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b & b & b \\ \frac{-bl}{2} & -bl & \frac{-bl}{2} & \frac{bl}{2} & bl & \frac{bl}{2} \\ \frac{-bl\sqrt{3}}{2} & 0 & \frac{bl\sqrt{3}}{2} & \frac{bl\sqrt{3}}{2} & 0 & \frac{bl\sqrt{3}}{2} \\ -k & k & -k & k & -k & k \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \\ \omega_5^2 \\ \omega_6^2 \end{bmatrix} \tag{10}$$

Suppose that the studied quadrotor can measure its position, velocity, attitude, and angular velocity through corresponding sensing units. Then the objective is to calculate control inputs $\mathrm{u}_i, (i \in \{1, \dots, 4\})$ using feedback states which the UAV can follow the desired path.

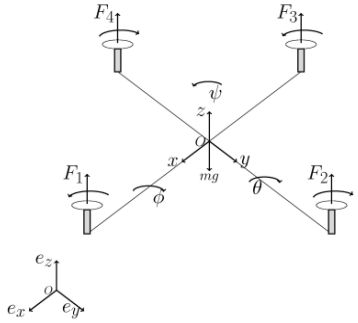

Fig. 1. Quadrotor free body diagram.

III. RBFNN Controller Approach

RBFs has recently drawn much attention due to their good generalization, local decaying feature and a simple network structure that prevent extensive calculation compared to the multilayer feed-forward networks (MFNs).

RFB neural network consists of three layers: 1) input layer, 2) hidden layer, and 3) output layer. Required data enters the network through the first layer and active an activation function in the second layer. The logic of activation is to use Euclidean distance between the input vector and a center matrix. This let valuable data be top rated based on their distance to the desired points. The rated data forms the output by weight matrix based on their rank.

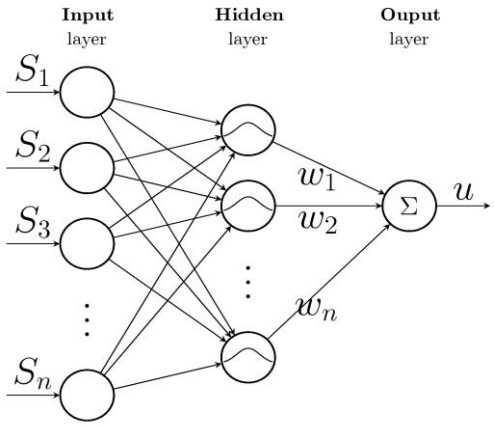

Fig. 2. The scheme of the RBFNN

The typical structure of an RBF is illustrated in Fig. 2. In the mathematical formulation the output which is would be the controller law of the system is calculated as follows:

$$u = \mathbf{w}^T\mathbf{h} \tag{11}$$

where **w** is the weight value of the RBF and **h** is the output of the radial basis activation funcction in the hidden layer and is expressed as $\mathbf{h} = [h_j]^T$, while h_j is Gaussian function value for neuron j in the hidden layer, and

$$h_j = exp(-\frac{||\mathbf{S} - c_j||}{2b_j^2}) \tag{12}$$

where **S** denotes the input vector and $\mathbf{c} = [c_{ij}]$ represents the position of the center of the Gaussian function of neuron j for the ith input. For making neural network flexible there is matrix $\boldsymbol{b} = [b_1, \dots, b_m]^T$ which is the width of Gaussian function for neuron j.

In a simple passive normal Neural Network, all the coefficients are trained before the system execution and inputs are bounded, but when the dynamics of the system change those coefficients need to be updated. One of the approaches to this problem is online learning. In this case, the network tries to change the weight, center, and width matrix in a way to satisfy a performance index function (13).

$$E(t) = \frac{1}{2}(y_d(t) - y(t))^2 \tag{13}$$

where y and y_d stand for the system state and its reference. Using the gradient descent method, the parameters can be updated as follows:

$$\Delta w_j(t) = -\eta_o \frac{\partial E}{\partial w_j} = \eta_o(y_d(t) - y(t))h_j \tag{14}$$

$$w_j(t) = w_j(t-1) + \Delta w_j(t) \tag{15}$$

$$\Delta b_j(t) = -\eta_o \frac{\partial E}{\partial b_j} = \eta_o(y_d(t) - y(t))w_j h_j \frac{||\mathbf{S} - c_j||}{b_j^3} \tag{16}$$

$$b_j(t) = b_j(t-1) + \Delta b_j(t) \tag{17}$$

$$\Delta c_{ji}(t) = -\eta_o \frac{\partial E}{\partial c_{ji}} = \eta_o(y_d(t) - y(t))w_j \frac{S_j - c_{ji}}{b_j^2} \tag{18}$$

$$c_{ij}(t) = c_{ij}(t-1) + \Delta c_{ij}(t) \tag{19}$$

where $\eta_o \in [0,1], o \in [1, \dots, 4]$ is the learning rate of the oth input. In RBFNN, the mentioned coefficient must be chosen based on the scope of the input value. If they were chosen inappropriately, the Gaussian function would not be effectively mapped and the output will be invalid.

After the RBFNN structure is set, the input and performance index function should be expressed in a way to control the system.

A. Altitude and Yaw Dynamics

The dynamic model of multirotor is divided into two sub-systems of fully actuated and underactuated. The fully actuated sub-systems are related to z and ψ states. So through the corresponding inputs, the UAV can track the desired paths of z_d and ψ_d.This means that the objective of the controller is to minimize the error of tracking and the performance index function of referred sub-systems would be

$$E_z(t) = \frac{1}{2}e_z^2 \tag{20}$$

$$E_\psi(t) = \frac{1}{2}e_\psi^2 \tag{21}$$

and the input vector would be chosen subsequently

$$\mathbf{S}_z = [e_z \;\; \dot{e_z}]^T \tag{22}$$

$$\mathbf{S}_\psi = [e_\psi \;\; \dot{e_\psi}]^T \tag{23}$$

By choosing input vector as (22) and (23) the valuable data contains the pattern of satisfying index function is fed to the controller.

B. XY Plane Trajectory

According to the previous subsection, the UAV has two under actuated sub-systems which are in pair of $[x, \theta]$ and $[y, \phi]$ in both sub-systems. The first state is meant to be controlled and the system input directly changes the second state.

The solution to overcome this problem is to make a linear combination of states error of each sub-system. The index function is introduced in (24) and (25) and the input vector would be chosen subsequently in (26) and (27).

$$E_y(t) = \alpha_\phi(e_\phi) + \alpha_y(e_y), \tag{24}$$

$$E_x(t) = \alpha_\theta(e_\theta) - \alpha_x(e_x) \tag{25}$$

$$\mathbf{S}_y(t) = [\alpha_\phi(e_\phi) + \alpha_y(e_y), \quad \alpha_{\dot{\phi}}(\dot{e_\phi}) + \alpha_y(\dot{e_y})]^T \tag{26}$$

$$\mathbf{S}_y(t) = [\alpha_\theta(e_\theta) - \alpha_x(e_x), \quad \alpha_{\dot{\theta}}(\dot{e_\theta}) - \alpha_{\dot{x}}(\dot{e_x})]^T \tag{27}$$

In (24) to (27), the desired angle is zero so that controller is trying to keep the UAV level in most of the time. In these

equations, α gains are selected in a way to make both errors of each sub-system same scale so the importance of both will be in the same level.

The proof of the stability of RBFNN is done in [27].

IV. EKF Observer

So far, the objective was to form a simple controller structure to be able to control the multirotor and could be deployed on proposed processors. Since the simulation condition is supposed to be close to real-time flight, the state feedback to the system contains both transition and measurement noise. This noise should be filtered to achieve pure state data and based on them the controller generates inputs of the system.

One of the optimal state estimators is Kalman Filter(KF) which was published in 1960 [28]. Simple Kalman Filter uses the linearized dynamics equation of the system to predict states in the absence of data and filter both transition and measurement noise around the equilibrium point of the system.

The problem of simple KF is the linearized characteristics of the observer that are efficient only around the equilibrium point of the system. For this situation, some extension of KF has been proposed during history which includes Extended Kalman Filter (EKF). The EKF is the nonlinear version of KF which linearizes equations around the estimated point in each time step.

To express the EKF equation, it is supposed that the form of the nonlinear system and frequent discrete-time measurements are like

$$\dot{\mathbf{x}}(t) = f\big(\mathbf{x}(t),\mathbf{u}(t)\big) + \mathbf{q}(t),\ \mathbf{q}(t) \sim N(0,\mathbf{Q}(t)) \tag{28}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}(t),\ \mathbf{v}(t) \sim N(0,\mathbf{R}_k) \tag{29}$$

where $\mathbf{x}_k = \mathbf{x}(t_k)$ and $h(\mathbf{x}_k)$ is the nonlinear measurement dynamics and $\mathbf{w}(t)$ and $\mathbf{v}(t)$ are process and measurements noise with covariance of Q and R respectively. Since the initial states are not generally measurable, they would be randomly initialized as

$$\mathbf{x}(0) \sim (\overline{\mathbf{x}}_0, \mathbf{P}_0) \tag{30}$$

The structure of EKF consists of two phases; 1) Prediction and 2) Update. In prediction, sensor data are not available and the observer predicts state through the nonlinear model of the system and covariance error is propagated as follows

$$\dot{\hat{\mathbf{x}}}(t) = f\big(\hat{\boldsymbol{x}}(t),\mathbf{u}(t)\big) \tag{31}$$

$$\dot{\mathbf{P}}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T(t) + \mathbf{Q}(t) \tag{32}$$

In (31), $\hat{\boldsymbol{x}}$ denotes the estimation of the states and $\mathbf{F}(t)$ is the

$$\mathbf{F}(t) = \frac{\partial f}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}(t_k),\mathbf{u}_t} \tag{33}$$

In case that data sensors are available observer updates the parameters and estimated states through the following equations:

$$\hat{\boldsymbol{x}}_k|_k = \hat{\boldsymbol{x}}_k|_{k-1} + \mathbf{K}_k(\mathbf{z}_k - h(\hat{\boldsymbol{x}}_k|_{k-1})) \tag{34}$$

$$\mathbf{K}_k = \hat{\boldsymbol{x}}_k|_{k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k|_{k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{35}$$

$$\mathbf{P}_k|_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k|_{k-1} \tag{36}$$

In represented equations, $\mathbf{K}_k$ is the Kalman gain, $\hat{\boldsymbol{x}}_k|_{k-1} \equiv \hat{\boldsymbol{x}}(t_k)$ and $\hat{\boldsymbol{x}}(t_{k-1}) \equiv, \hat{\boldsymbol{x}}_{k-1}|_{k-1}$. Also $\mathbf{H}_k$ is the linearized measurments dynamics. In this paper, since it is supposed that all states are available through the sensors, $h(\mathbf{x}_k) = \mathbf{x}_k$ then $\mathbf{H}_k = \mathbf{I}_{n\times n}$.

V. Simulation Results

In this section, the controller scheme combined with the observer would be evaluated. The simulation test cases consist of one quadrotor and one hexarotor with different desired paths and the way they are generated. In the first scenario, the desired points will be generated as the UAV pass the last desired point and in the other case, the desired path generate as a time function.

In both simulations, three different time steps are considered shown in Table I.

TABLE I: Simulation Time Steps

Time step	Value
Equations of motion	0.001 sec
Processor output frequency	0.005 sec
Sensors frequency	0.05 sec

There is also a saturation function for each rotor to not exceed half of the weight of the UAV considering that the thrust to weight ratio is 2. Besides the equations of motion, other signals in the simulations are pass through a Zero-Order Hold (ZOH) function which represents the discrete characteristics of the processor.

Since both simulations are executed with the same controller, the parameters of the RBFNN are presented in Table II. The number of the hidden layer neurons are chosen based on the related results explained in [27] and consideration of hardware implementation limits.

TABLE II: Parameters of RBFNN

Parameters	Value
Hidden layer neurons (altitude)	6
Hidden layer neurons (attitude)	5
$\eta_i, \quad i \in [1,3,4]$	0.01
η_2	0.0003
$\alpha_{\phi,\theta}$	0.6
$\alpha_{\dot{\phi},\dot{\phi}}$	0.4
$\alpha_{x,y}$	$\frac{0.2}{mg}$
$\alpha_{\dot{x},\dot{y}}$	$\frac{2}{mg}$

The initial value of $\mathbf{w}_0, \mathbf{c}_0$ and $\mathbf{b}_0$ where chosen based on the neural network convergence in a raw simulation.

As well, the parameters of the EKF are set as Table III.

TABLE III: Parameters of EKF

Parameters	Value
P_0	$diag([1\ 1\ 1\ 0.01\ 0.01\ 0.01])^2$
H	$I_{6\times6}$
Q	$0_{6\times6}$
R	$diag([0.32\ 0.32\ 0.32\ 0.045\ 0.045\ 0.045]$
Sensor noise Variance	[0.3 0.3 0.3 0.045 0.045 0.045]

Each simulation is executed three times and data type changes in each. The data types are denoted by *double*, *single* and *fixed*. Here fixed-point data uses 13 bits for the integer part and 19 bits for fractional.

A. Quadrotor

In the first scenario, a quadrotor has been simulated to track some points that are generated after passing the last one. In Table IV the physical parameters of the quadrotors are shown.

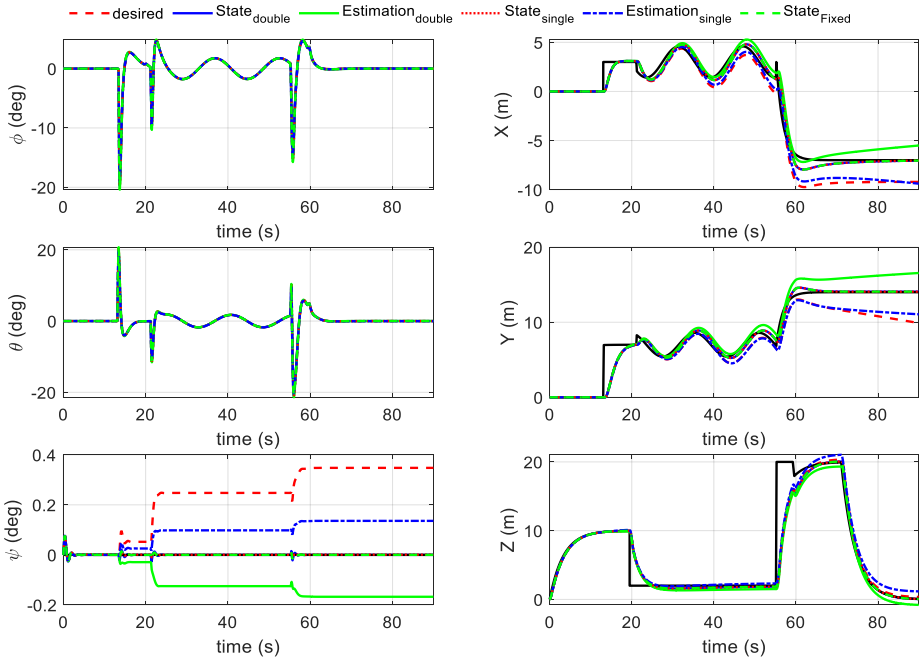

Fig. 3. Position and angle (Estimated and states) of the quadrotor for three data types- first scenario.

As it can be seen in Figs. (3) and (4), the UAV has tracked the desired path and has a considerable offset at the end of the trajectory which is caused by the step-wise desired path and adds error in each part of the trajectory. Nevertheless, the UAV has accomplished its mission in three execution. According to the Fig. (4), although there is a time delay between collecting data sensors and processor calculations, the generated inputs are smooth and there is no chattering like the characteristics of a sliding-mode controller. It should be mentioned that the ψ angle does not follow the desired trajectory and it is for the data noise and error of filtering but the scale of changes are very small consequently so it is acceptable.

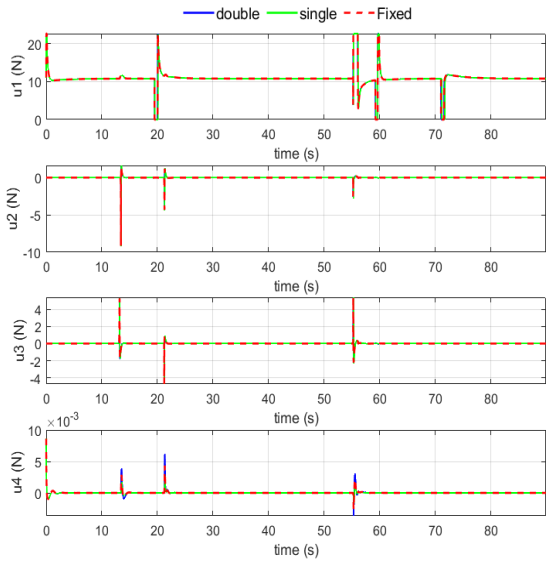

Fig. 4. Applied inputs - first scenario.

B. Hexarotor

In the second scenario, a hexarotor with specified parameters (Table V) and desired trajectory has been simulated. The desired paths after reaching first point (coordinates:[0,3,4]), would be function of time.

TABLE IV: PARAMETERS OF QUADROTOR

Parameters	Value
m	1.1 (kg)
I_x	0.02839 $(kg.m^2)$
I_y	0.03066 $(kg.m^2)$
I_z	0.0439 $(kg.m^2)$
J_r	8.3×10^{-5} $(kg.m^2)$
l	0.32 (m)
b	$5 \times 10^{-5}(kg.m/rad^2)$
d	$2 \times 10^{-5}(kg.m/rad^2)$
$k_i, i = 1,2,3$	0.1
$k_i, i = 4,5,6$	0.12

TABLE V: PARAMETERS OF HEXAROTOR

Parameters	Value
m	0.65 (kg)
I_x	0.03 $(kg.m^2)$
I_y	0.025 $(kg.m^2)$
I_z	0.045 $(kg.m^2)$
J_r	6×10^{-5} $(kg.m^2)$
l	0.165 (m)
b	$3.5 \times 10^{-5}(kg.m/rad^2)$
d	$3 \times 10^{-5}(kg.m/rad^2)$
$k_i, i = 1,2,3$	0.1
$k_i, i = 4,5,6$	0.12

The results of the three execution are presented here. The hexacopter is stable in simulation and has perfectly tracked the trajectory. It can be figured out that at two points hexacopter estimation errors increase unpredictably which coincides with the sudden changes in generated inputs and this means that the observer is sensitive to the sudden changes. Nevertheless, again the inputs and the rotational velocity of each rotor (Fig. (6)) are smooth and all simulations are completed.

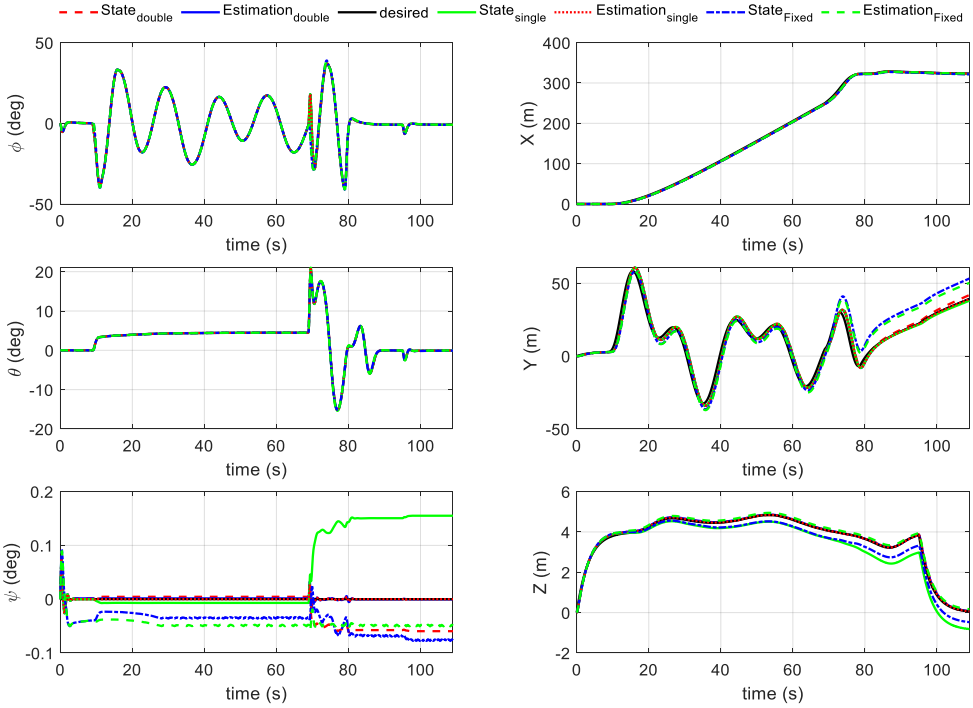

Fig. 5. Position and angle (Estimated and states) of the hexarotor for three data types - second scenario.

VI. Conclusion

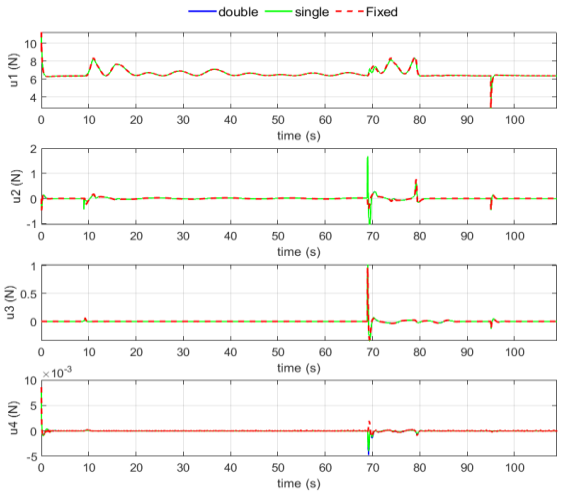

Fig. 6. Applied inputs - second scenario.

In this paper after presenting the equations of motion for multirotor aerial vehicles, an RBFNN controller has been proposed and implemented to control the position and attitude of a quadrotor and a hexarotor in two different scenarios. To achieve real-time flight condition, system states were mixed with noise and an EKF was used to filter the noise. In addition, saturation for each input has been applied. To check the capability of running on the high-level processor such as DSP and FPGA where they use single floating-point and fixed-point data types for calculations, each scenario was executed with corresponding data types. As seen in the results, the proposed structure can control a conventional multirotor. Also, the control-observer system can be implemented on a compatible processor and acceptably accomplish its mission. The results provided herein establish the structure for the adaptive and accurate controller-observer system for a conventional multirotor.

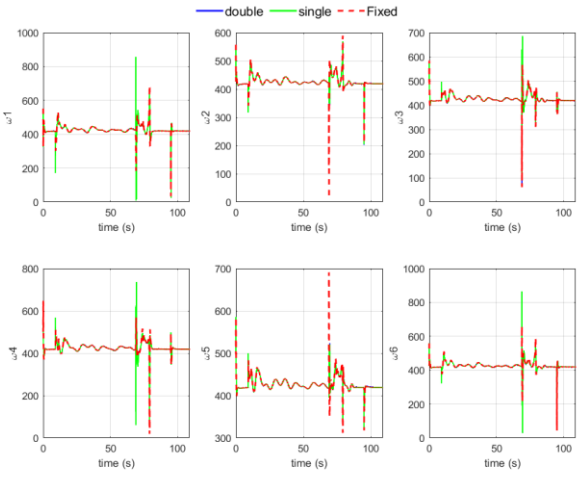

Fig. 7. Rotor rotational velocity- second scenario.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

A. Banazadeh and A. Samadzadeh conceived of the

presented idea. A. Banazadeh designed and directed the research, provided critical feedback, and supervised the findings of this work. A. Samadzadeh designed the model and the computational framework, performed the analytic calculations and numerical simulations. Both A. Banazadeh and A. Samadzadeh authors contributed to the final version of the manuscript.

REFERENCES

[1] T. Shima and S. Rasmussen, "UAV cooperative decision and control: challenges and practical approaches," SIAM, 2009.

[2] S. A. Emami and A. Banazadeh, "Robustness investigation of a ducted-fan aerial vehicle control, using linear, adaptive, and model predictive controllers," *International Journal of Advanced Mechatronic Systems*, vol. 6, pp. 108–117, 2015.

[3] A. A. Saif, M. Dhaifullah, M. Al-Malki, and M. E. Shafie, "Modified backstepping control of quadrotor," *International Multi-conference on Systems, Signals Devices*, pp. 1-6, 2012.

[4] Y. Yali, J. Changhong, and W. Haiwei, "Backstepping control of each channel for a quadrotor aerial robot," in *Proc. 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering*, vol. 3, pp. 403–407, 2010.

[5] M. E. Antonio-Toledo, E. N. Sanchez, A. Y. Alanis, J. A. Flórez, and M. A. Perez-Cisneros, "Real-time integral backstepping with sliding mode control for a quadrotor UAV," *IFAC*, vol. 51, pp. 549–554, 2018.

[6] H. Razmi, "Adaptive neural network based sliding mode altitude control for a quadrotor UAV," *Journal of Central South University*, vol. 25, pp. 2654–2663, 2018.

[7] S. Notter, A. Heckmann, A. Mcfadyen, and F. Gonzalez, "Modelling, simulation and flight test of a model predictive controlled multirotor with heavy slung load," *IFAC*, vol. 49, pp. 182–187, 2016.

[8] B. Wang, Y. Zhang, J-C. Ponsart, and D. Theilliol, "Fault-Tolerant Adaptive Control Allocation for Unmanned Multirotor Helicopter" *IFAC*, vol. 50, pp. 5269-5274, 2017.

[9] H. Housny, E. A. Chater and H. El Fadil, "New deterministic optimization algorithm for fuzzy control tuning design of a quadrotor," in *Proc. 2019 5th International Conference on Optimization and Applications (ICOA)*, pp. 1-6, 2019.

[10] Q. Jing, Z. Chang, H. Chu, Y. Shao, and X. Zhang, "Quadrotor attitude control based on fuzzy sliding mode control theory," in *Proc. 2019 Chinese Control Conference (CCC)*, pp. 8360-8364, 2019.

[11] H. Razmi and S. Afshinfar, "Neural network-based adaptive sliding mode control design for position and attitude control of a quadrotor UAV," *Aerospace Science and Technology*, vol. 91 pp. 12-27, 2019.

[12] Z. Jia, L. Wang, J. Yu, and X. Ai, "Distributed adaptive neural networks leader-following formation control for quadrotors with directed switching topologies," *ISA Transactions*, vol. 3, pp. 93-107, 2019.

[13] M. Shirzadeh, H. Jabbari Asl, A. Amirkhani, and A. A. Jalali, "Vision-based control of a quadrotor utilizing artificial neural networks for tracking of moving targets," *Engineering Applications of Artificial Intelligence*, vol. 58, pp. 34-48, 2017.

[14] Y. Lyu, G. Lai, C. Chen, and Y. Zhang, "Vision-based adaptive neural positioning control of quadrotor aerial robot," *IEEE Access*, vol. 7, pp. 75018- 75031, 2019.

[15] H. Yang, and J. Liu, "An adaptive RBF neural network control method for a class of nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, pp. 457-462, 2018.

[16] M. Zhao, X. Xu, H. Yang, and Z. Pan, "Design of a predictive RBF compensation fuzzy PID controller for 3D laser scanning system," *Applied Sciences*, vol. 10, no. 13, 2020.

[17] M. Shirzadeh, H. Jabbari Asl, A. Amirkhani, and A. A. Jalali, "Vision-based control of a quadrotor utilizing artificial neural networks for tracking of moving targets," *Engineering Applications of Artificial Intelligence*, vol. 58, pp. 34-48, 2017.

[18] Y. Li, N. Sundararajan, and P. Saratchandran, "Neuro-controller design for nonlinear fighter aircraft maneuver using fully tuned RBF networks," *Automatica*, vol. 37, pp. 1293-1301, 2001.

[19] J. Kok, L. F. Gonzalez, and N. Kelson, "FPGA implementation of an evolutionary algorithm for autonomous unmanned aerial vehicle on-board path planning," *IEEE Transactions on Evolutionary Computation*, vol. 17, pp. 272-281, 2013.

[20] B. Eizad, A. Doshi, and A. Postula, "FPGA based stability system for a small-scale quadrotor unmanned aerial vehicle," in *Proc. the 8th FPGA World Conference*, 2011.

[21] K. Boikos and C. Bouganis, "Semi-dense SLAM on an FPGA SoC," in *Proc. 2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 1-4, 2016.

[22] C. Yu and J. Juang, "Application of intelligent systems and DSP to landing controller design," in *Proc. 2013 9th Asian Control Conference (ASCC)*, pp. 1-6, 2013.

[23] P. Gohl, D. Honegger, S. Omari, M. Achtelik, M. Pollefeys, and R. Siegwart, "Omnidirectional visual obstacle detection using embedded FPGA," in *Proc. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3938-3943, 2015.

[24] W. Weihua, W. Peizao, and N. Zhaodong, "A real-time detection algorithm for unmanned aerial vehicle target in infrared search system," in *Proc. 2018 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 1-5, 2018.

[25] Y. Gui, P. Guo, H. Zhang, Z. Lei, X. Zhou, J. Du, and Q. Yu, "Airborne vision-based navigation method for UAV accuracy landing using infrared lamps," *Journal of Intelligent & Robotic Systems*, vol. 72, pp. 197-218, 2013.

[26] S. Bouabdallah, "Design and control of quadrotors with application to autonomous flying," Ph.D. thesis, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland, 2007.

[27] J. Liu, *Radial Basis Function (RBF) Neural Network Control for Mechanical Systems: Design, Analysis and Matlab Simulation*, Springer Berlin Heidelberg, 2013.

[28] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, pp. 35-45, 1960.

A. Banazadeh is an associate professor in the Faculty of Aerospace Engineering at Sharif University of Technology (SUT), where he has taught and researched since 2008. Holding a Bachelor degree in mechanical engineering, a master's degree and a doctorate in aerospace engineering, Dr. Banazadeh has served as an academic advisor to several masters and Doctoral students and is frequently called upon to testify as an expert in his field. He has published over 50 refereed papers in high ranking journals and conferences and has received awards for research excellence from SUT. He is a senior member of AIAA (American Institute of Aeronautics and Astronautics) and a member of ISIRI/TC20 Technical Committee. His current research interests are system identification, uncertainty-based design optimization, nonlinear stability analysis, data-driven control systems, and electric aircraft. He also consults for various companies in the area of modeling, simulation, and design as well as technology demonstration activities.

A. Samadzadeh received his bachelor's degree in aerospace engineering from K.N. Toosi University of Technology in 2018. He serves as a research assistant with the measurement, instrumentation, and system identification laboratory at Sharif University of Technology (SUT). His research interests include nonlinear control, neural networks, and automation.