

Deep Continuum Deformation Coordination and Optimization with Safety Guarantees

Harshvardhan Uppaluru and Hossein Rastgoftar

Abstract—In this paper, we develop and present a novel strategy for safe coordination of a large-scale multi-agent team with “*local deformation*” capabilities. Multi-agent coordination is defined by our proposed method as a multi-layer deformation problem specified as a Deep Neural Network (DNN) optimization problem. The proposed DNN consists of p hidden layers, each of which contains artificial neurons representing unique agents. Furthermore, based on the desired positions of the agents of hidden layer k ($k = 1, \dots, p - 1$), the desired deformation of the agents of hidden layer $k + 1$ is planned. In contrast to the available neural network learning problems, our proposed neural network optimization receives time-invariant reference positions of the boundary agents as inputs and trains the weights based on the desired trajectory of the agent team configuration, where the weights are constrained by certain lower and upper bounds to ensure inter-agent collision avoidance. We simulate and provide the results of a large-scale quadcopter team coordination tracking a desired elliptical trajectory to validate the proposed approach.

I. INTRODUCTION

First inspired by natural phenomena, formation flight and cooperative control in Multi-Agent Systems (MAS) have been fascinating and important areas of study for the past 20 years. Research into MAS has led to interesting theoretical problems and potential practical uses in a wide range of situations. MAS formation flight and cooperative control are achieved either in a centralized or decentralized manner. The centralized technique makes use of a central computer that controls every agent in the MAS. However, the decentralized technique, also known as distributed control, allows for computation onboard each agent, and information is shared across neighboring agents [1]. For cooperative multi-agent control, the decentralized method has many benefits, such as low operational costs, fewer system requirements, great robustness, strong adaptability, and flexible scalability.

A. Related Work

With applications ranging from surveillance [2], [3] to formation flying [3], [4], rescue missions [5], wildlife monitoring and exploration [6], precision agriculture [7], cooperative payload delivery [8], [9], and hazardous environment sensing [10], several multi-agent coordination techniques have been researched and presented.

A group of agents, acting as particles of a single virtual rigid body, use the centralized technique of Virtual Structure

*This work has been supported by the National Science Foundation under Award Nos. 2133690 and 1914581.

Authors are with Scalable Move And Resilient Traversability Lab, Aerospace and Mechanical Engineering Department, University of Arizona, Tucson - 85721, USA. Email: {huppaluru, hrastgoftar}@arizona.edu

(VS) [11], [12]. VS is capable of maintaining the rigid geometric relationship between the agents and evolving as a rigid body in a given direction and orientation. Consensus [13], [14] is among the most exhaustively researched cooperative control approaches. In this approach, a team of agents reaches an agreement or consensus regarding some quantities of interest only by communicating with their neighbors. It is a decentralized coordination and control approach and is broadly divided into two categories: leaderless consensus (i.e., consensus without a leader) [15], [16] and leader-follower consensus (i.e., consensus with a leader) [17], [18].

Another decentralized leader-follower method is called Containment Control [19], [20], [21], [22], where the collective motion of all agents is achieved with multiple leaders. The follower agents obtain the desired positions through local communication with in-neighbor agents, and all agents are contained within a particular area defined by geometric constraints. A recent multi-agent coordination approach known as Homogeneous (or Affine) Transformation, is based on the principles of continuum mechanics, where the agents in the system are treated as particles of a deformable body undergoing a homogeneous transformation [23], [24], [25], [26], [27], [28], [29]. This technique ensures that all agents in the system remain inside a bounding envelope and allows for translation, rotation, and shearing of the bounding envelope while ensuring collision avoidance. Homogeneous transformation advances containment control by ensuring inter-agent collision avoidance. To achieve the desired homogeneous transformation in n -D ($n = 1, 2, 3$), $n + 1$ leaders in \mathbb{R}^n communicate with the follower agents via local communication.

B. Contributions

Although homogeneous transformation coordination can allow for aggressive and safe changes to the inter-agent distances, it has “deformation uniformity” problems. This is due to the fact that at any moment t , the deformation of the complete agent team configuration is given by a single Jacobian matrix that can be specified based on unique rotation and shear deformation angles, as well as axial deformations [30]. Therefore, the rotation, axial, and shear deformations must be consistent over the whole MAS arrangement. To overcome this deformation uniformity issue, the existing homogeneous transformation coordination, specified based on a single Jacobian matrix, is advanced in this paper to Deep Continuum Deformation Coordination (DCDC), which allows us to plan safe “*local deformation*” of an agent team without having to change the inter-agent distances between

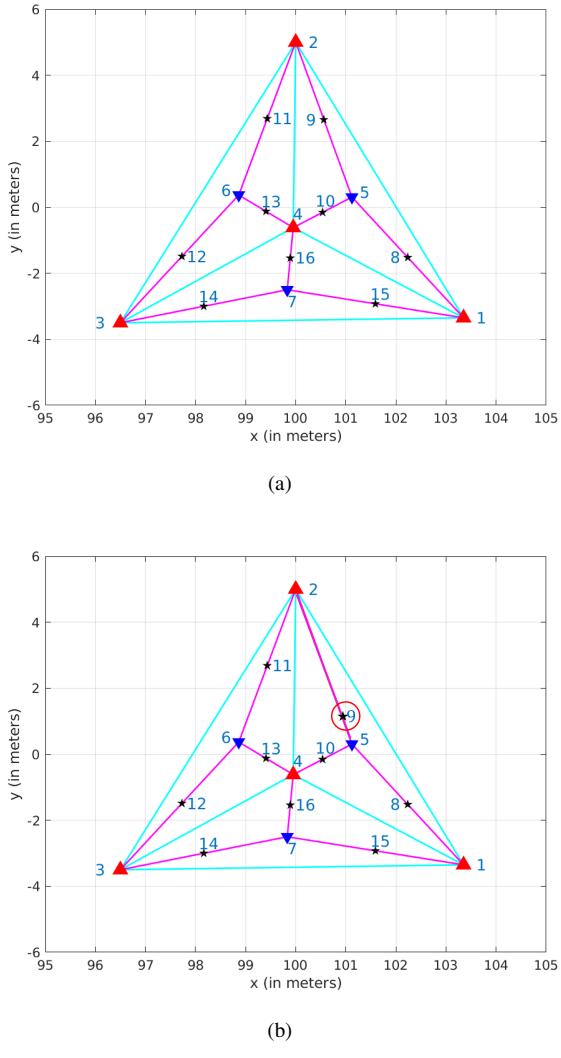


Fig. 1. (a) An example of the reference configuration of a $N = 16$ quadcopter team. As defined in Section II, we have $\mathcal{B} = \{1, 2, 3\}$, $\mathcal{C} = \{4\}$ and $\mathcal{I} = \{5, \dots, 16\}$. The set $\mathcal{L}_1 = \{1, 2, 3, 4\}$ is given by red triangles, $\mathcal{L}_2 = \mathcal{L}_1 \cup \{5, 6, 7\}$ is represented by blue triangles, $\mathcal{L}_3 = \{8, \dots, 16\}$ is denoted by the black stars. Finally, we have $\mathcal{V} = \{1, \dots, 16\}$ as the set that identifies all quadcopters in the team. (b) Local deformation of agent 9 by adjusting weights $\beta_{9,5,3}$ and $\beta_{9,2,3}$.

all agents (See Fig. 1(a),(b)).

DCDC defines multi-agent coordination as a multi-layer continuum deformation constructed using a neural network in which (i) each neuron represents a different agent and (ii) the input layer receives time-invariant reference positions of the boundary agents. In our proposed NN, which has p hidden layers, the desired deformation of the agents of layer $k+1$ is based on the agents' desired positions belonging to hidden layer k ($k = 1, \dots, p-1$). Unlike existing neural network learning methods that obtain the weights and biases based on the training data, our proposed NN optimization assigns weights and biases based on a single input and a time-varying output. More specifically, our goal is to plan the deformation of a multi-agent team by obtaining the weights and biases

of the hidden layers so that the error between the actual position configuration of the agent team and the desired position configuration of the agent team is minimized while guaranteeing inter-agent collision avoidance between every two agents. We provide guarantee conditions for assuring inter-agent collision avoidance by obtaining the inequality and equality constraints on the weights and biases of the hidden layers.

This paper is organized as follows: The preliminaries (Section II) are first introduced, followed by a detailed description of our proposed approach (Section III). We present safety guarantee conditions (Section IV) before presenting simulation results (Section V). Section VI concludes the paper.

II. PRELIMINARIES

Consider a N -agent team (See Fig. 1) moving collectively as particles of a deformable body in 3-D motion space. We use set $\mathcal{V} = \{1, \dots, N\}$ to identify all agents in the team. We express set \mathcal{V} as

$$\mathcal{V} = \mathcal{B} \cup \mathcal{C} \cup \mathcal{I}, \quad (1)$$

where \mathcal{B} defines the primary leader agents that are located at the boundary of the agent team configuration, singleton \mathcal{C} defines a single agent located inside the convex hull defined by \mathcal{B} , and \mathcal{I} identifies the remaining agents, located inside the convex hull defined by \mathcal{B} . Without loss of generality we index agents such that the sets $\mathcal{B} = \{1, \dots, N_L - 1\}$, $\mathcal{C} = \{N_L\}$, and $\mathcal{I} = \{N_L + 1, \dots, N\}$ identify boundary, core, and interior agents, respectively. Set \mathcal{V} is also expressed as

$$\mathcal{V} = \bigcup_{k=1}^p \mathcal{L}_k \quad (2)$$

with subsets \mathcal{L}_1 through \mathcal{L}_p , where $\mathcal{L}_1 = \mathcal{B} \cup \mathcal{C}$. We note that subset \mathcal{L}_k serves as immediate leaders for \mathcal{L}_{k+1} , for $k = 1, \dots, p-1$, which implies that desired positions of the agents belonging to \mathcal{L}_{k+1} are defined based on desired positions of the agents in \mathcal{L}_k . \mathcal{L}_2 through \mathcal{L}_{p-1} are defined as the interior leaders and the set \mathcal{L}_p defines the pure followers that do not serve as immediate leaders for any agents belonging to \mathcal{L}_1 through \mathcal{L}_{p-1} . Furthermore, \mathcal{L}_1 through \mathcal{L}_{p-1} satisfy the following condition:

$$\bigwedge_{k=1}^{p-1} (\mathcal{L}_k \subset \mathcal{L}_{k+1}). \quad (3)$$

More specifically, desired position of agent $i \in \mathcal{L}_{k+1}$ is given by

$$\mathbf{p}_i(t) = \sum_{j \in \mathcal{L}_k} \beta_{i,j,k}(t) \mathbf{p}_j(t), \quad i \in \mathcal{L}_{k+1}, k = 1, \dots, p-1, \quad (4)$$

where $\beta_{i,j,k}(t) \in [0, 1]$, and

$$\sum_{j \in \mathcal{L}_k} \beta_{i,j,k}(t) = 1, \quad i \in \mathcal{L}_{k+1}, k = 1, \dots, p-1. \quad (5)$$

Safety conditions are explained in Section IV. Note that leaders belonging to \mathcal{L}_1 are the *primary leaders* that move independently with the desired trajectories defined by

$$\mathbf{p}_i(t) = \alpha_i(t)\mathbf{a}_i + \mathbf{d}(t), \quad \forall i \in \mathcal{L}_1, \quad (6)$$

where $\mathbf{d}(t)$ specifies the nominal position of the agent team configuration expressed with respect to an inertial coordinate system, \mathbf{a}_i is the constant reference position of primary leader $i \in \mathcal{L}_1$. Note that the reference position of the core leader $i \in \mathcal{C}$ is $\mathbf{0}$, i.e. $\mathbf{a}_{i,0} = \mathbf{0}$.

III. METHODOLOGY

We investigate deep continuum deformation coordination of a N -agent team over a finite time interval $[t_0, t_f]$ and develop a Neural-Network-based (NN-based) optimization method to obtain the desired deformation of the agent team (See Fig. 2), by assigning $\alpha_i(t)$ of the first layer, and $\beta_{i,j,k}(t)$ of every layer $k = 1, \dots, p - 1$. Figure 2 illustrates the schematic of the proposed NN with p hidden layers. Note that the artificial neurons in hidden layer 1 through p represent the agents defined by \mathcal{L}_1 through \mathcal{L}_p , respectively.

Input Layer: As shown in Fig. 2, the input layer generates the reference positions of the boundary agents and core agent, defined by $\mathcal{B} \cup \mathcal{C}$. Note that the reference positions are time-invariant.

Hidden Layers: The first hidden layer, denoted by \mathcal{L}_1 , receives the reference positions of the boundary agents. More specifically, neuron $j \in \mathcal{L}_1$ represents agent $j \in \mathcal{B} \cup \mathcal{C}$, receives reference position \mathbf{a}_j , and returns $\mathbf{p}_j(t)$ by using Eq. (6). Notice that the rigid-body displacement vector $\mathbf{d}(t)$, used in Eq. (6), has the same bias for all neurons in layer \mathcal{L}_1 .

Every hidden layer $k \in \{2, \dots, p\}$ receives desired positions from previous hidden layer $k-1$ and returns the desired positions of the agents defined by \mathcal{L}_k . More specifically, neuron $i \in \mathcal{L}_k$ receives the desired positions of agents of \mathcal{L}_{k-1} and returns $\mathbf{p}_i(t)$ using (4). Note that the bias of a neuron in hidden layers 2 through p is $\mathbf{0}$.

Output Layer: The output layer consists of a single neuron averaging the desired position of the agent team configuration. Therefore, the weights associated with the edges connecting neurons of hidden layer p to the output layer are $\frac{1}{|\mathcal{L}_p|}$ where $|\mathcal{L}_p|$ denotes the cardinality of set \mathcal{L}_p .

Loss function: In this paper, we use the residual sum of squares as the loss function at time t given by the following equation:

$$\text{Loss}(t) = \left\| \frac{1}{|\mathcal{L}_p|} \sum_{i \in \mathcal{L}_p} \mathbf{p}_i(t) - \mathbf{d}(t) \right\|^2, \quad t \in [t_0, t_f]. \quad (7)$$

Training of the Neural Network: To train the network, we determine positive weight parameters $\alpha_i(t)$ and $\beta_{i,j,k}(t)$ at any time $t \in [t_0, t_f]$. To ensure safety of the agent team coordination, $\alpha_i(t)$ and $\beta_{i,j,k}(t)$ are constrained by specific lower and upper bounds that are determined in Section IV.

IV. SAFETY GUARANTEE CONDITIONS

Before proceeding further, we express desired, actual, and reference positions of agent $i \in \mathcal{V}$ as $\mathbf{p}_i(t) = [x_{i,d}(t) \ y_{i,d}(t) \ z_{i,d}(t)]^T$, $\mathbf{r}_i(t) = [x_i(t) \ y_i(t) \ z_i(t)]^T$, and $\mathbf{a}_{i,0} = [x_{i,0} \ y_{i,0} \ z_{i,0}]^T$, respectively. For obtaining the safety guarantee conditions, we make the following assumptions:

Assumption 1. Every agent $i \in \mathcal{V}$ can execute a proper trajectory tracking control such that deviation of the actual trajectory $\mathbf{r}_i(t)$ from the desired trajectory $\mathbf{p}_i(t)$ remains bounded and satisfies the following equation

$$\bigwedge_{i \in \mathcal{V}} \bigwedge_{q \in \{x, y, z\}} |q_{i,d}(t) - q_i(t)| \leq \delta, \quad \forall t, \quad (8)$$

where $\bigwedge_{i \in \mathcal{V}}$ implies “include all” and $\delta > 0$ is constant.

Assumption 2. Every agent $i \in \mathcal{V}$ can be enclosed by a box with side length 2ϵ .

To assure safety of the agent team continuum deformation, the following two safety requirements must be satisfied:

Safety Condition 1. For every two different agents i and j , the inter-agent collision avoidance is guaranteed, if

$$\bigwedge_{i=1}^{N-1} \bigwedge_{j=i+1}^N \bigwedge_{q \in \{x, y, z\}} |q_i(t) - q_j(t)| > 2\epsilon, \quad \forall t, \quad (9)$$

Safety Condition 2. Every follower $i \in \mathcal{L}_{k+1}$ remains inside the convex hull defined by \mathcal{L}_k . We say agents of layer \mathcal{L}_{k+1} are contained by the convex hull defined by \mathcal{L}_k , if

$$\bigwedge_{k=1}^{p-1} \bigwedge_{i \in \mathcal{L}_{k+1}} \bigwedge_{j \in \mathcal{L}_k} (\beta_{i,j,k}(t) \geq 0), \quad \forall t. \quad (10a)$$

$$\bigwedge_{k=1}^{p-1} \bigwedge_{i \in \mathcal{L}_{k+1}} \left(\sum_{j \in \mathcal{L}_k} \beta_{i,j,k}(t) = 1 \right), \quad \forall t. \quad (10b)$$

The Safety Condition 1 is satisfied, if [30]

$$\bigwedge_{i=1}^{N-1} \bigwedge_{j=i+1}^N \bigwedge_{q \in \{x, y, z\}} |q_{i,d}(t) - q_{j,d}(t)| > 2(\delta + \epsilon), \quad \forall t. \quad (11)$$

Note that Eq. (11) provides a sufficient safety condition for inter-agent collision avoidance.

To ensure safety conditions 1 and 2 are satisfied, we constrain $\alpha_{i,k}$ and $\beta_{i,j,k}$ by

$$\bigwedge_{i \in \mathcal{L}_1} (\alpha_{min} \leq \alpha_i(t)), \quad \forall t \quad (12)$$

$$\bigwedge_{k=1}^{p-1} \bigwedge_{j \in \mathcal{L}_k} \bigwedge_{i \in \mathcal{L}_{k+1}} (\beta_{k,min} \leq \beta_{i,j,k} \leq \beta_{k,max}). \quad (13)$$

We first assign $\beta_{k,min}$ and $\beta_{k,max}$ for every layer $k = 1, \dots, p - 1$. Then, for given $\beta_{k,min}$ and $\beta_{k,max}$ at every layer $k = 1, \dots, p - 1$, we obtain α_{min} by solving the following optimization problem:

$$\alpha_{min} = \min_{\alpha \in (0,1)} \alpha \quad (14)$$

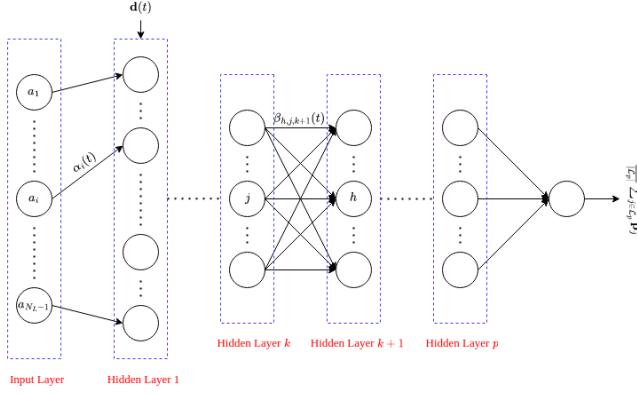


Fig. 2. The structure of the proposed neural network used for MQS continuum deformation optimization.

subject to

$$\bigwedge_{i \in \mathcal{L}_1} \bigwedge_{q \in \{x,y,z\}} (q_{i,d} = \alpha q_{i,0}), \quad (15)$$

$$\bigwedge_{k=1}^{p-1} \bigwedge_{i \in \mathcal{L}_{k+1}} \bigwedge_{q \in \{x,y,z\}} \left(q_{i,d} = \sum_{j \in \mathcal{L}_k} \beta_{i,j,k} q_j \right), \quad (16)$$

$$\begin{aligned} & \bigwedge_{k=1}^{p-1} \bigwedge_{i \in \mathcal{L}_{k+1}} \bigwedge_{\substack{q \in \{x,y,z\} \\ i \neq h}} \left(\sum_{j \in \mathcal{L}_k} |(\beta_{i,j,k} - \beta_{h,j,k}) q_{j,d}| > 2(\delta + \epsilon) \right), \\ & \forall \beta_{i,j,k}, \beta_{h,j,k} \in [\beta_{k,min}, \beta_{k,max}]. \end{aligned} \quad (17)$$

We use Algorithm 1 to solve the above optimization problem.

Algorithm 1 Obtaining safety guarantee condition for Deep Continuum Deformation Coordination and Optimization

- 1: *Get:* $\beta_{k,min}$ and $\beta_{k,max}$ for $k = 1, \dots, p-1$, $\Delta\alpha$, and reference position of every agent $i \in \mathcal{V}$
- 2: *Obtain:* α_{min}
- 3: *Set:* $\alpha = 1$
- 4: **while** Eq. (17) is satisfied **do**
- 5: $\alpha \leftarrow \alpha - \Delta\alpha$
- 6: Update $\mathbf{p}_i(t)$ of agent $i \in \mathcal{L}_1$ using Eq. (15)
- 7: **for** $< k \in \{1, \dots, p-1\}$ **do**
- 8: **for** $< j \in \mathcal{L}_k$ **do**
- 9: **for** $< q \in \{x, y, z\}$ **do**
- 10: **for** $< \beta_{i,j,k} \in [\beta_{k,min}, \beta_{k,max}]$ **do**
- 11: Update $\mathbf{p}_i(t)$ using Eq. (16).
- 12: Check constraint (17).
- 13: **end for**
- 14: **end for**
- 15: **end for**
- 16: **end for**
- 17: **end while**
- 18: $\alpha_{min} \leftarrow \alpha$

Parameter	Value	Units
m	0.5	kg
g	9.81	m/s^2
l	0.25	m
I_r	3.357×10^{-5}	kgm^2
I_x	0.0196	kgm^2
I_y	0.0196	kgm^2
I_z	0.0264	kgm^2
b	3×10^{-5}	Ns^2/rad^2
k	1.1×10^{-6}	Ns^2/rad^2

TABLE I
QUADCOPTER SPECIFICATIONS

V. SIMULATION RESULTS

Here, we present simulation results obtained using PyTorch¹ on a desktop running Ubuntu 20.04 LTS with an Intel i7 11th generation CPU, an NVIDIA GPU, and 16 GB of RAM. We consider the evolution of a multi-quadcopter system (MQS), consisting of $N = 16$ quadcopters with initial formation at time $t = 0$, as shown in Fig. 1a. The quadcopters in the MQS are similar and are modeled using the dynamics established in [31]. The quadcopter parameters originally presented in [32] are listed in Table I.

We consider the desired path of the MQS configuration to be an ellipse as shown in Fig. 3 with major radius $a = 100$ m and minor radius $b = 80$ m. The total travel time to complete the ellipse is denoted by $T = 60$ s. The MQS is distributed over the horizontal plane at $z = 10$ m, whose value is always constant.

The quadcopters' in the MQS are identified by defining the set $\mathcal{V} = \mathcal{B} \cup \mathcal{C} \cup \mathcal{I}$, where $\mathcal{B} = \{1, 2, 3\}$, $\mathcal{C} = \{4\}$, and $\mathcal{I} = \{5, \dots, 16\}$. Alternatively, as stated in Section II, the sets $\mathcal{L}_1 = \{1, \dots, 4\}$, $\mathcal{L}_2 = \{5, \dots, 7\}$ and $\mathcal{L}_3 = \{8, \dots, 16\}$ identify the quadcopters of hidden layers 1, 2, and 3, respectively. We compute the α_i and $\beta_{i,j,k}$ parameters using the approach presented in Section III. The minimum and maximum values assigned for the communication weights, $\alpha_i(t)$ and $\beta_{i,j,k}(t)$ are shown in Table II. Note that the minimum values are assigned such that inter-agent collision avoidance amongst every two quadcopters in the MQS is guaranteed while also satisfying the constraints mentioned in Section IV.

TABLE II
LOWER AND UPPER BOUNDS FOR OBTAINING THE NN WEIGHTS

α_{min}	$\beta_{1,min}$	$\beta_{2,min}$	$\beta_{1,max}$	$\beta_{2,max}$
0.5	0.2	0.35	0.6	0.65

It is desired that the quadcopter configuration follow the desired elliptic trajectories while satisfying the safety conditions presented in Table II. Therefore, we run the neural network described in Section III for 6000 epochs with a learning rate of 0.01. Stochastic Gradient Descent (SGD) has been used along with the loss function mentioned in Section III. The weights and biases of the NN are used to calculate

¹<https://pytorch.org/>

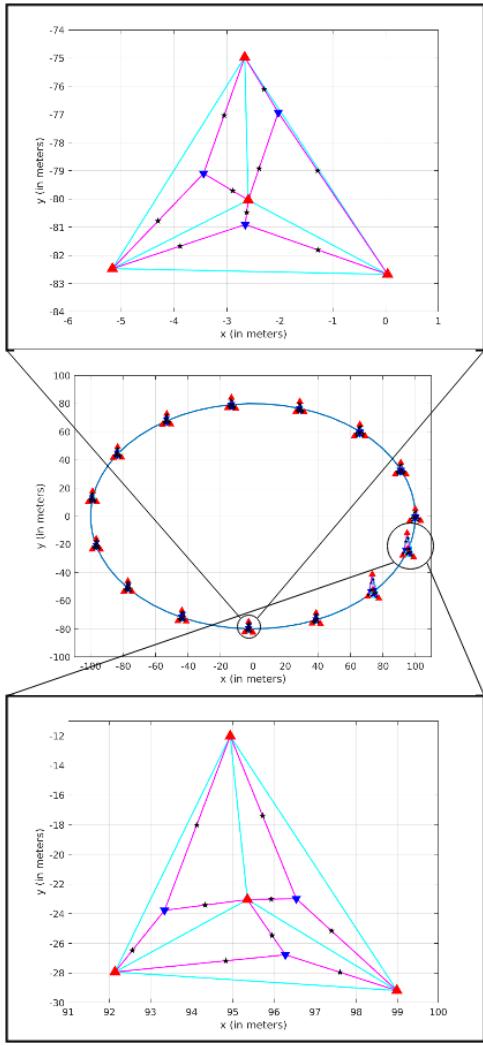


Fig. 3. Desired configuration of the MQS at various times.

the nominal trajectories of each of the quadcopters in the MQS team. Using the input-output feedback linearization control approach [31], Figures 4, 5, 6, and 7 plot the x -position component, the y -position component, the thrust force magnitude, and the rotors' angular speeds of the quadcopter $6 \in \mathcal{L}_2$, respectively.

VI. CONCLUSIONS

This paper has developed and presented a NN-based technique for safe continuum deformation coordination and optimization of a N -quadcopter team tracking a known target trajectory in 3-D motion space. Assuming that there are no obstacles while tracking the desired trajectory and no failures in the MQS, we presented a novel algorithm in which the input to the NN is the constant reference configuration and optimization is done based on the desired trajectory to track so as to obtain the weights that directly correlate to the matrices in homogeneous transformation. Our work also provided safety guarantees ensuring inter-agent collision avoidance. Future work lies in the direction of making the algorithm robust to obstacles in the environment, conducting

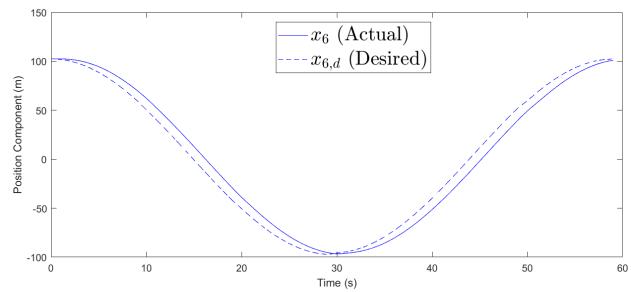


Fig. 4. x component of the actual and desired trajectories of quadcopter 6.

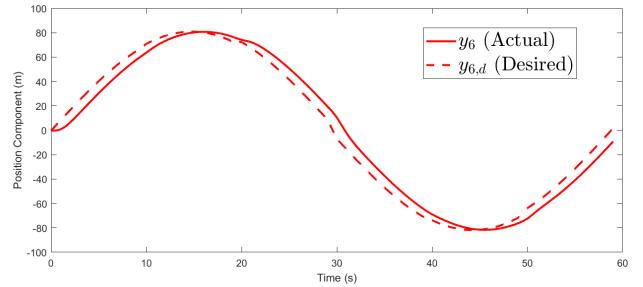


Fig. 5. y component of the actual and desired trajectories of quadcopter 6.

further simulations and flight experiments with obstacles, and incorporating fluid flow navigation [25], [29], [33] as an obstacle avoidance algorithm. An alternate direction would be to obtain the deformation of the MQS formation through a known obstacle, such as windows of different shapes and sizes.

REFERENCES

- [1] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 427–438, 2012.
- [2] S.-L. Du, X.-M. Sun, M. Cao, and W. Wang, "Pursuing an evader through cooperative relaying in multi-agent surveillance networks," *Automatica*, vol. 83, pp. 155–161, 2017.
- [3] L. C. B. Da Silva, R. M. Bernardo, H. A. De Oliveira, and P. F. Rosa, "Multi-uav agent-based coordination for persistent surveillance with dynamic priorities," in *2017 International Conference on Military Technologies (ICMT)*. IEEE, 2017, pp. 765–771.
- [4] S.-h. Ha and S.-d. Chi, "Multi-agent based design of autonomous uavs for both flocking and formation flight," *Journal of Advanced Navigation Technology*, vol. 21, no. 6, pp. 521–528, 2017.
- [5] H. AlTair, T. Taha, J. Dias, and M. Al-Qutayri, "Decision making for multi-objective multi-agent search and rescue missions," in *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*. IEEE, 2017, pp. 1–4.
- [6] J. Witczuk, S. Pagacz, A. Zmarz, and M. Cypel, "Exploring the feasibility of unmanned aerial vehicles and thermal imaging for ungulate surveys in forests-preliminary results," *International Journal of Remote Sensing*, vol. 39, no. 15–16, pp. 5504–5521, 2018.
- [7] D. C. Tsouros, S. Bibi, and P. G. Sarigiannidis, "A review on uav-based applications for precision agriculture," *Information*, vol. 10, no. 11, p. 349, 2019.
- [8] H. Rastgoftar and E. M. Atkins, "Cooperative aerial lift and manipulation (calm)," *Aerospace Science and Technology*, vol. 82, pp. 105–118, 2018.

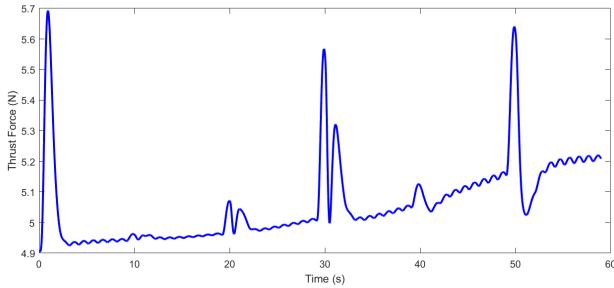


Fig. 6. Thrust of quadcopter 6.

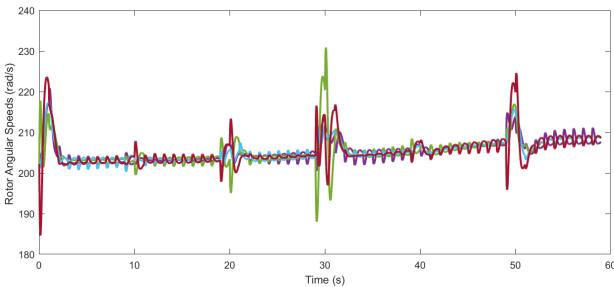


Fig. 7. Rotor angular speeds of quadcopter 6.

- [9] F. Rossomando, C. Rosales, J. Gimenez, L. Salinas, C. Soria, M. Sarcinelli-Filho, and R. Carelli, “Aerial load transportation with multiple quadrotors based on a kinematic controller and a neural smc dynamic compensation,” *Journal of Intelligent & Robotic Systems*, vol. 100, no. 2, pp. 519–530, 2020.
- [10] B. Argrow, D. Lawrence, and E. Rasmussen, “Uav systems for sensor dispersal, telemetry, and visualization in hazardous environments,” in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005, p. 1237.
- [11] M. A. Lewis and K.-H. Tan, “High precision formation control of mobile robots using virtual structures,” *Autonomous robots*, vol. 4, no. 4, pp. 387–403, 1997.
- [12] W. Ren and R. Beard, “Virtual structure based spacecraft formation control with formation feedback,” in *AIAA Guidance, Navigation, and control conference and exhibit*, 2002, p. 4963.
- [13] W. Ren and R. W. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Transactions on automatic control*, vol. 50, no. 5, pp. 655–661, 2005.
- [14] L. Ding, Q.-L. Han, and G. Guo, “Network-based leader-following consensus for distributed multi-agent systems,” *Automatica*, vol. 49, no. 7, pp. 2281–2286, 2013.
- [15] S. Rao and D. Ghose, “Sliding mode control-based autopilots for leaderless consensus of unmanned aerial vehicles,” *IEEE transactions on control systems technology*, vol. 22, no. 5, pp. 1964–1972, 2013.
- [16] W. Ren, “Distributed leaderless consensus algorithms for networked euler-lagrange systems,” *International Journal of Control*, vol. 82, no. 11, pp. 2137–2149, 2009.
- [17] Q. Song, J. Cao, and W. Yu, “Second-order leader-following consensus of nonlinear multi-agent systems via pinning control,” *Systems & Control Letters*, vol. 59, no. 9, pp. 553–562, 2010.
- [18] L. Xu, H. Fan, Z. Dong, and W. Wang, “Robust consensus control for leader-following multi-agent system under switching topologies,” in *2016 International Conference on Cybernetics, Robotics and Control (CRC)*. IEEE, 2016, pp. 27–31.
- [19] G. Ferrari-Trecate, M. Egerstedt, A. Buffa, and M. Ji, “Laplacian sheep: A hybrid, stop-go policy for leader-based containment control,” in *International workshop on hybrid systems: Computation and control*. Springer, 2006, pp. 212–226.
- [20] Z. Li, W. Ren, X. Liu, and M. Fu, “Distributed containment control of multi-agent systems with general linear dynamics in the presence of multiple leaders,” *International Journal of Robust and Nonlinear Control*, vol. 23, no. 5, pp. 534–547, 2013.
- [21] X. Wang, S. Li, and P. Shi, “Distributed finite-time containment

control for double-integrator multiagent systems,” *IEEE Transactions on Cybernetics*, vol. 44, no. 9, pp. 1518–1528, 2013.

- [22] G. Wen, Y. Zhao, Z. Duan, W. Yu, and G. Chen, “Containment of higher-order multi-leader multi-agent systems: A dynamic output approach,” *IEEE Transactions on Automatic Control*, vol. 61, no. 4, pp. 1135–1140, 2015.
- [23] H. Rastgoftar and S. Jayasuriya, “Evolution of multi-agent systems as continua,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 136, no. 4, p. 041014, 2014.
- [24] H. Rastgoftar, *Continuum deformation of multi-agent systems*. Springer, 2016.
- [25] H. Emadi, H. Uppaluru, and H. Rastgoftar, “A physics-based safety recovery approach for fault-resilient multi-quadcopter coordination,” *arXiv preprint arXiv:2110.07777*, 2021.
- [26] H. Emadi, H. Uppaluru, H. Ashrafiou, and H. Rastgoftar, “Collision-free continuum deformation coordination of a multi-quadcopter system using cooperative localization,” in *2022 European Control Conference (ECC)*. IEEE, 2022, pp. 1349–1354.
- [27] H. Emadi, H. Uppaluru, and H. Rastgoftar, “A physics-based safety recovery approach for fault-resilient multi-quadcopter coordination,” in *2022 American Control Conference (ACC)*. IEEE, 2022, pp. 2527–2532.
- [28] M. Romano, P. Kuevor, D. Lukacs, O. Marshall, M. Stevens, H. Rastgoftar, J. Cutler, and E. Atkins, “Experimental evaluation of continuum deformation with a five quadrotor team,” in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 2023–2029.
- [29] H. Uppaluru, H. Emadi, and H. Rastgoftar, “Resilient multi-uas coordination using cooperative localization,” *Aerospace Science and Technology*, vol. 131, p. 107960, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1270963822006344>
- [30] H. Rastgoftar and I. V. Kolmanovsky, “Safe affine transformation-based guidance of a large-scale multiquadcopter system,” *IEEE Transactions on Control of Network Systems*, vol. 8, no. 2, pp. 640–653, 2021.
- [31] H. Rastgoftar, “Real-time deployment of a large-scale multi-quadcopter system (mq5),” *arXiv preprint arXiv:2201.10509*, 2022.
- [32] E. Gopalakrishnan, “Quadcopter flight mechanics model and control algorithms,” *Czech Technical University*, vol. 69, 2017.
- [33] M. Romano, H. Uppaluru, H. Rastgoftar, and E. Atkins, “Quadrotor formation flying resilient to abrupt vehicle failures via a fluid flow navigation function,” *arXiv preprint arXiv:2203.01807*, 2022.