Detailed Technical Instructions for CADTC

1: *Basic concept explanation.*

Response: Firstly, the concept of conditional independence. In general multi-agent systems, the reward function depends on all agents' actions. Therefore, the optimal policy for the multi-agent system can be expressed as $\pi(a_1, a_2, ..., a_N|o_1, o_2, ..., o_N)$, where o_i refers to the observation of the ith agent and a_i refers to the action of the ith agent ($1 \leq i \leq N$). Since the dimensions of $\pi(a_1, a_2, ..., a_N|o_1, o_2, ..., o_N)$ can be huge, to simplify the policy optimization, one common approach is to decouple the joint policy by assuming "conditional independence" of actions from different agents, i.e., $\pi(a_i, a_{-i}|o_i, o_{-i}) = \pi(a_i|o_i)\pi(a_{-i}|o_{-i})$, $\forall i$, where a_{-i} represents the actions of all the agent except agent i. Conditional independence means that each agent can make decision independently and solely relies on local observation instead of taking the impact from other agents into consideration, which dramatically reduces the policy space. The study regarding the topic of centralized training with decentralized execution in the deep RL domain, such as MADDPG, are based on the "conditional independence" assumption.

Secondly, we provide detailed explanations of our proposed CADTC algorithm. In reality, the behavior of an agent has non-negligible impacts on others. Consequently, simply assuming "conditional independence" cannot achieve the optimal performance. Therefore, in this letter, we proposed a communication-assisted framework, which allows each agent to share certain messages among its neighbors to enhance the cooperation among agents. Moreover, our proposed method can be trained and executed without a centralized controller, i.e., in a distributed manner. More implementation details regarding our proposed CADTC algorithm are as follows:

(1) Network structure: The critic network is implemented by the fully-connected neural network (FNN) with three hidden layers. The actor network is a combination of three modules, where each module is implemented as an neural network. The encode module utilizes FNN with two layers, and the policy module utilizes FNN with three hidden layers. The information extraction module uses bi-directional Gated Recurrent Unit (GRU) network with one hidden layer. GRU is designed to process sequential data, such as time series or natural language, which has feedback connections that allow them to retain information from previous time steps, enabling them to capture temporal dependencies. We use GRU to process the messages from different agents.

(2) Learning phases: It consists of the sample collection, actor update and critic update. During the sample collection phase, all agents interact with the environment to generate samples. Each UAV agent firstly acquires the locations of its nearest M_{d}^{SN} SNs, and then share its own location to its nearest M_{d}^{UAV} UAVs. According to the observation, each UAV agent encodes message locally, and then transmit the

message to its nearest M_{d}^{UAV} UAVs. After receiving messages from the nearest M_{d}^{UAV} UAVs, each agent makes decision based on received message with the information extraction module and the policy module. After executing the action, the reward is computed and transmitted to each UAV agent and the five-tuple ($\boldsymbol{o}_{n,t}$, $\mathcal{M}_{n,t}$, $\boldsymbol{a}_{n,t}$, $r_{n,t}$, $\boldsymbol{o}_{n,t+1}$) are stored in an experience replay buffer. During actor update and critic update phases, the gradient are computed and the network parameters are updated.

(3) Execution phases: It is identical to the sample collection duration as in learning phases, except storing the five-tuple data into the replay buffer.

2: *Complexity analysis.*

Response: The complexity of CADTC can be reflected by the number of parameters of the agent network and that of the critic network. Therefore, we analyze the number of parameters to discuss the complexity of the algorithm. Firstly, we analyze the number of parameters of the agent network. Let U_l^{enc} denote the number of neurons in the lth layer of the message encoder module with L^{enc} layers, where $1 \leq l \leq L^{enc}$. Let U_l^{ext} denote the number of neurons in the lth layer of the information extraction module with L^{ext} layers, where $1 \leq l \leq L^{ext}$. Let U_l^{pol} denote the number of neurons in the lth layer of the policy module with L^{pol} layers, where $1 \leq l \leq L^{pol}$. Then the total number of parameters of the agent network of our CAVDN is given by $\sum_{l=2}^{L^{enc}-1}(U_{l-1}^{enc}U_l^{enc} + U_l^{enc}U_{l+1}^{enc}) + \sum_{l=2}^{L^{ext}-1}(U_{l-1}^{ext}U_l^{ext} + U_l^{ext}U_{l+1}^{ext}) + \sum_{l=2}^{L^{pol}-1}(U_{l-1}^{pol}U_l^{pol} + U_l^{pol}U_{l+1}^{pol})$. Secondly, we analyze the number of parameters of the critic network. Let U_l^{cri} denote the number of neurons in the lth layer of the message encoder module with L^{cri} layers, where $1 \leq l \leq L^{cri}$. Then, the number of parameters of the critic network of our CADTC is given by $\sum_{l=2}^{L^{cri}-1}(U_{l-1}^{cri}U_l^{cric} + U_l^{cri}U_{l+1}^{cri})$.

During training, the actor and critic networks of all agents share the same network structure and parameters. With the growth of the number of UAVs N, the times of network update (i.e., Lines 12-13, in our Algorithm 1) will increases linearly with respect to N. Thus, the complexity of the algorithm during training can be expressed as $\mathcal{O}(N(\sum_{l=2}^{L^{enc}-1}(U_{l-1}^{enc}U_l^{enc} + U_l^{enc}U_{l+1}^{enc}) + \sum_{l=2}^{L^{ext}-1}(U_{l-1}^{ext}U_l^{ext} + U_l^{ext}U_{l+1}^{ext}) + \sum_{l=2}^{L^{pol}-1}(U_{l-1}^{pol}U_l^{pol} + U_l^{pol}U_{l+1}^{pol})) + \sum_{l=2}^{L^{cri}-1}(U_{l-1}^{cri}U_l^{cric} + U_l^{cri}U_{l+1}^{cri}))$.

During execution, CADTC operates on each agent in a decentralized manner. Therefore, the complexity does not grows when N increases. On the other hand, the critic network is removed during training. Thus, the complexity can be expressed as $\mathcal{O}(\sum_{l=2}^{L^{enc}-1}(U_{l-1}^{enc}U_l^{enc} + U_l^{enc}U_{l+1}^{enc}) + \sum_{l=2}^{L^{ext}-1}(U_{l-1}^{ext}U_l^{ext} + U_l^{ext}U_{l+1}^{ext}) + \sum_{l=2}^{L^{pol}-1}(U_{l-1}^{pol}U_l^{pol} + U_l^{pol}U_{l+1}^{pol}))$.

Besides, the parameters of the actor network and that of the critic network are shared among all agents, thus the complexity of the algorithm does not grow as the number of agents increases.

3: *Synchronization issue.*

Response: We indeed have the implicit assumption that all UAVs should synchronize when they obtain the observation of the environment, so that the AoI and the position obtained by each agent are correct. This means that all UAVs should wait each other to finish each execution, and the synchronization of all

UAVs can be supported by the BSs.

In our paper, the execution time including five parts: observation obtaining time, encoding time, message sharing time, policy decision time, and UAV flying time. Since the architecture and parameters of the agent network for each agent are the same, thus the encoding time and policy decision time are the same. In our system model, we have set the same flying time slot for all UAVs. Thus, the observation obtaining time and message sharing time are the source resulting in non-synchronization due to the wireless communication channel.

Nonetheless, the observation and the shared message are with small data size. For the observation data size of SNs, $S_{obs}^{SN} = 2M_{\mathrm{d}}^{SN}S_{FLOAT} = 128$ bits in our settings, where $M_{\mathrm{d}}^{SN} = 2$ refers to the number of the nearest observable SNs, and $S_{FLOAT} = 32$ representing the size of float data. For the observation data size of UAVs, $S_{obs}^{UAV} = 2M_{\mathrm{d}}^{UAV}S_{FLOAT} = 128$ bits, where $M_{\mathrm{d}}^{UAV} = 2$ refers to the number of the nearest observable UAVs. For the data size of each shared message, $S_{mes} = S_{FLOAT}U_o^{enc}M_{\mathrm{d}}^{UAV} = 8192$ bits, where $U_o^{enc} = 128$ represents the number of neurons in the output layer of the message encoder module. Thus, considering time slot $\delta = 1$ s, the data rate of signaling overhead for each UAV is only $S_{mes} + S_{obs}^{SN} + S_{obs}^{UAV} \approx 8$ Kbps, which can be safely neglected.

Besides, UAVs do not need frequent synchronization, with duration larger than the time slot $\delta = 1$ s in our paper. It is also notable that the synchronization among UAVs can be realized by using primary and secondary synchronization signals (PSS/SSS) via BS as in [A].

[A] Fotouhi, Azade, et al. "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges." *IEEE Communications surveys & tutorials* 21.4 (2019): 3417-3442.

4: *Metric selection.*

Response: According to the definition of AoI, it includes two parts: the transmission delay and the waiting time to schedule. The transmission delay is the duration for the forwarding sampled data from the SNs to the BS, while the waiting time refers to the duration after the sampled data is saved into the local buffer of SNs and before it is started to be forwarded to the BS.

In our letter, the UAV has limited communication range, and thus it has to fly around and schedules to serve SNs within its communication range. We optimize the trajectories of UAVs, and thus we optimize the scheduling of SNs implicitly. Therefore, the AoI in this letter considers not only transmission delay, but also waiting time, which is not the same as the delay.

If AoI only considers the transmission delay, then the algorithm will results in a scenario that all UAVs hover over the top of some SNs. This is because the transmission delay is minimized when the distance between each pair of UAV and SN is minimized.

5: *Convergence.*

Response: Firstly, we provide the convergence information of the proposed solution and all baselines. As shown in Fig.1, we compare the average reward achieved by different methods versus the number of

episodes during training. We can see that the reward curves of all methods firstly increase rapidly and finally converges after about 20000 epochs of training.

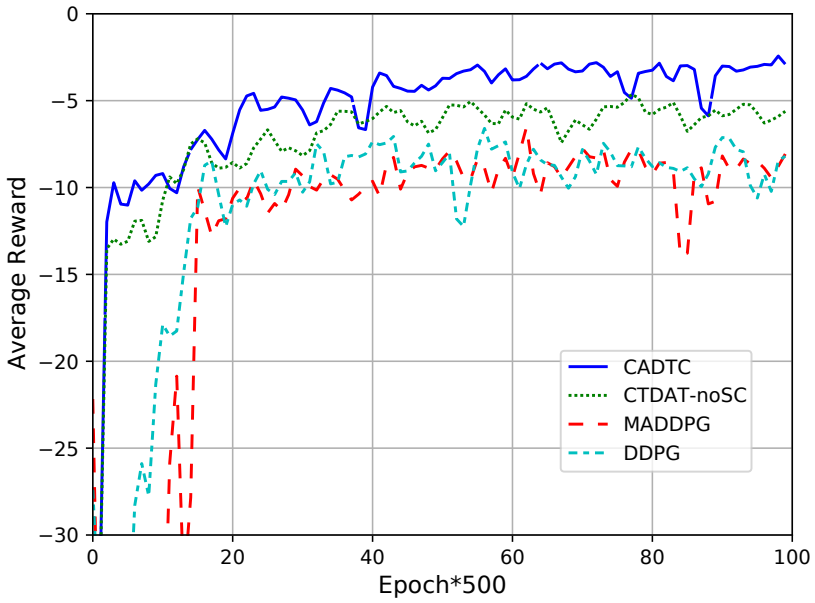

Fig. 1. The convergence information of different methods. The reward is averaged over all UAVs and timesteps in each episode. λ=0.5.

Secondly, we discuss the complexity of our proposed CADTC solution. As we discussed in our response to Comment 3 of Reviewer 1, we have analyzed the complexity of CADTC. During training, the actor and critic networks of all agents share the same network structure and parameters. With the growth of the number of UAVs N, the times of network update (i.e., Lines 12-13, in our Algorithm 1) will increases linearly with respect to N. Therefore, when the number of UAVs are large, the complexity is large during training. Nonetheless, the training can be conducted in simulated environment with enough computing capability. It is worth noting that CADTC operates on each agent in a decentralized manner during execution. Therefore, during execution, the complexity does not grows when N increases.