Collaborative Smart Environmental Monitoring Using Flying Edge Intelligence

T. Tolga Sarı*, Sabtain Ahmad†, Atakan Aral‡§, Gökhan Seçinti*

*Department of Computer Engineering, Istanbul Technical University, Istanbul, Turkey
† Institute of Information Systems Engineering, Vienna University of Technology, Vienna, Austria
‡ Department of Computing Science, Umeå University, Umeå, Sweden
§ Faculty of Computer Science, University of Vienna, Vienna, Austria
Email: sarita@itu.edu.tr, sabtain.ahmad@tuwien.ac.at, atakan.aral@umu.se, secinti@itu.edu.tr

***Abstract*—Smart environmental monitoring is crucial for public health and ecological balance as it enables us to monitor and react to environmental hazards. However, effective environmental monitoring can be hindered by the lack of infrastructure and high monetary costs. These challenges are even more pronounced in remote areas, where networking and energy sources are often limited or nonexistent. To address these challenges, we utilize UAVs to form a FANET which can provide effective communication infrastructure suitable for environment monitoring. Moreover, we utilize Edge Intelligence at these UAVs to increase the processing speed and reduce the data size that needs to be transmitted. Our results show that, compared to statically placed gateways, our solution is able to attain similar average age of information for monitoring results while also significantly increasing system capacity.**

***Index Terms*—Flying Edge Intelligence, Age of Information, Smart Environmental Monitoring, Flying Ad-Hoc Networks, Value of Information**

I. Introduction

Environmental monitoring is a crucial aspect of sustainable development and preserving our ecosystem and natural resources. With the rapid expansion of human settlements, rural areas are increasingly vulnerable to the effects of climate change, industrial pollution, and other environmental hazards. Smart environmental monitoring (SEM) [1] allows us to collect and analyze data on a range of environmental factors. Moreover, it helps us to identify and address potential health risks in real-time, compared to time-consuming and expensive traditional methods. This is particularly critical in the case of unexpected events such as natural disasters or industrial accidents, where rapid and accurate responses are crucial. However, the vast and diverse nature of rural environments entails several challenges that need to be overcome in order to enable effective SEM.

One of the core challenges of deploying SEM systems in rural areas is their logistic complexity of setting up and maintaining such systems. Rural areas often have limited infrastructure and resources, making it difficult to install and operate the sensors which negatively impacts the ecological balance of the environment. On the other hand, data collection and processing systems, and communication networks that are required for effective monitoring. In addition, the dispersed and often remote nature of rural environments makes it difficult to ensure the reliability and accuracy of data collected by these systems. Furthermore, the size of these environments and their corresponding data size severely limits the bandwidth of wireless transmissions, which limits the scalability of SEM systems.

SEM systems typically consist of multiple IoT devices (sensors) and a remote cloud server. For remote areas monitoring, IoT devices are connected to the gateways in order to allow the data transfer between sensors and the server. In the existing SEM systems, the gateways are deployed in the locations chosen either randomly or through a data-driven approach [2]. The static placement of gateways can have several disadvantages. First, it may result in limited coverage if the gateways are not placed strategically. Second, it can be difficult to scale up or down the monitoring area based on dynamic requirements. Finally, maintenance can be challenging, especially in remote or hard-to-reach areas. We aim to overcome these issues by converting static gateways into mobile ones using unmanned aerial vehicles (UAVs).

Instead, by installing these gateways on UAVs, it is possible to deploy monitoring systems quickly and more cost-effectively over large areas by creating Flying Ad-Hoc Networks (FANETs), without the need for extensive infrastructure or other resources. Furthermore, the customizability of UAVs allows us to configure them with different communication standards, on-board computing units and, multiple sensors to create Flying Edge Intelligence (FEI). Thus, we can adapt existing UAV deployments to different environments easily, which decreases their deployment and maintenance costs. Also, the altitude of the UAVs results in line-of-sight communications that results in more robust communication channel which has less fading effects [3]. As a result, each edge node's (UAV's) coverage improves. This allows for the deployment of fewer edge nodes, which reduces the number of potential points of failure in the system.

In the proposed FEI assisted SEM (FEI-SEM) system in figure 1, edge nodes collect data from sensors in order to extract additional information and shrink the data size with EI processing when their data pool reaches a certain size. This processing also allows them to compensate normally

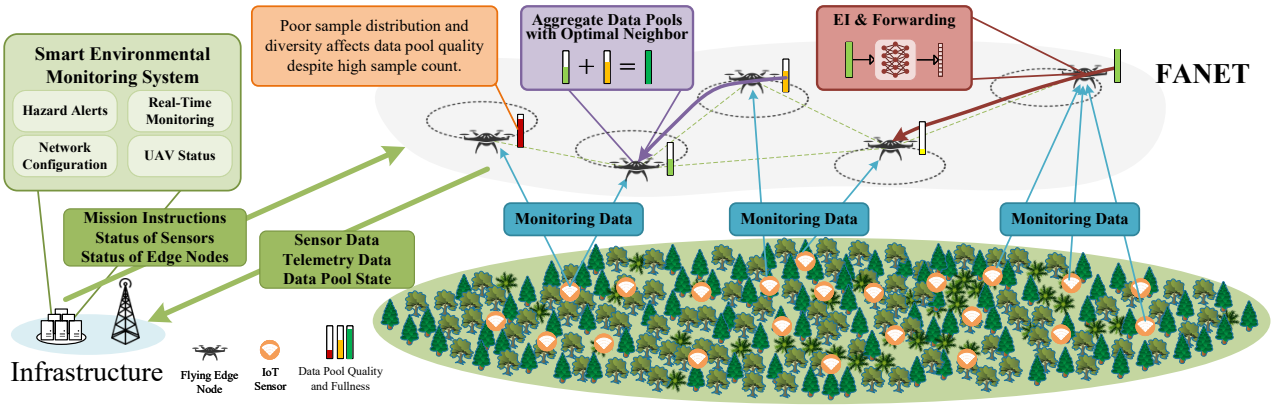


Fig. 1: The proposed FEI-SEM System.

wasted fly-time. Then, the edge nodes forward the results to the SEM server. To ensure optimal performance, it is important to optimize the data collection process at the edge nodes, so that the data pool is diverse and fresh at each iteration. To achieve this, our unique data aggregation method makes each edge node periodically seek to aggregate their data pools with their neighbors, aiming to both increase data quality and decrease Age Of Information (AoI). They select best neighbor by considering, its data pool's geographical dispersion, chronological relevance and distribution fairness. They also consider the AoI impact that a given data pool would make by the time it reaches the SEM server by utilizing well known Value of Information of the Update (VoIU) metric [4], [5].

Our contributions in this paper can be summarized as follows:

- We propose FEI-SEM system that provides increased coverage, capacity and decreased AoI compared to statically placed gateways.
- We propose data pool aggregation scheme, QADA, for FEI-SEM systems, that increases quality of the collected data while reducing its overall AoI.
- We run a detailed analysis through exhaustive OMNET++ simulations, where we compare different topologies, event types, and infrastructures.

The rest of this paper is organized as follows: In section II we introduce our method. In section III we present our evaluation results. Following in the section IV we discuss state of the art for FEI. Lastly in section V we conclude this paper.

II. Quality Aware Data Aggregator (QADA)

In this section, we derive our proposed data pool aggregation scheme and table I shows our notation in our equations. To speed up the data collection process our Quality Aware Data Aggregator (QADA) makes each edge node decide whether to wait for new data or to aggregate its own data points with one of its neighbors every τ seconds. Ultimately, the main objective, defined in (1), is maximizing the quality of the message m_k both in terms of timeliness and intelligence.

TABLE I: Notations

Notation	Meaning
B	Data Pool Size
S_{m_k}	Data Pool of m_k
S	Set of all Sensors
s_i	Sensor i
E	Set of all Edge Nodes
e_k	Edge node k
$U(m_k,t)$	Utility of message m_k at time t
$Q_A(m_k,t)$	VoIU of m_k at time t
$Q_I(m_k)$	Intelligence Quality of m_k
$Q_{loc}(m_k)$	Location Quality of m_k
$Q_{age}(m_k)$	Age Quality of m_k
$\Delta A(m_k)$	Maximum Age Difference in m_k
$Q_{\mathcal{J}}(m_k)$	Fairness in m_k
$\Delta T_k(t)$	Time to Server for e_k's data pool
$\Delta T'_k(t)$	Time to Server after aggregation
$\Delta P_k(t)$	Data pool fill time of e_k
$\Delta P'_k(t)$	Data pool fill time after aggregation
$\overline{R}_k(t)$	Average fill rate of e_k
$\Delta L_k(t)$	Routing time to Server
$N_k(t)$	Neighbors of e_k
$N'_k(t)$	Potential Aggregators of e_k
$D_j(t)$	e_j's Data pool fullness at time t

$$U(m_k,t) = c_1 \cdot Q_A(m_k,t) + c_2 \cdot Q_I(m_k) \tag{1}$$

Where c_1 and c_2 are weighting factors for sub-objectives and, $c_1, c_2 \geq 0$ and $c_1 + c_2 = 1$.

A. Value of information update (VoIU)

Our FEI-SEM system has multiple sensors that it needs to track. Thus, the AoI of each sensor needs to be minimized. $Q_A(m_k,t)$ measures the VoIU metric of each message at the time it arrives at the server. The server contains a vector $A_k(t)$ which stores the current ages of all sensors that belong to S as in (2) where $\Delta u_{k,s_n}$ shows the latest sampling time of the k_{th} sample for sensor s_n.

$$A_k(t) = \begin{pmatrix} p_{s_1,k}(t) \\ . \\ . \\ p_{s_n,k}(t) \end{pmatrix}, p_{s_i,k}(t) = \begin{cases} p_{s_i,k-1}(t) & s_i \notin S_{m_k} \\ t - \Delta u_{k,s_i} & otherwise \end{cases} \tag{2}$$

Then, we can calculate the value of each update as the L^2 norm of the age vectors as in (3). Using L^2 norm causes older

data to contribute more to the norm which also encourages improving the age of the oldest sensors.

$$Q_A(m_k, t) = \frac{\|A_{k-1}(t) - A_k(t)\|_2}{\|A_{k-1}(t)\|_2} \tag{3}$$

B. Quality of Intelligence

In order to create meaningful results, the data batch should include points that are relevant to each other, both geographically and chronologically. To capture the quality of geo-distribution, we utilize intra-cluster dispersion used in Caliński-Harabasz index [6]. This index is basically a measure of how each cluster is packed (intra-cluster dispersion) and how far away the resulting clusters are from each other (inter-cluster dispersion). Since we are concerned with only one cluster, the samples inside the data pool, we omit the inter-cluster parts. Following this, we can calculate intra-cluster deviation (σ_{m_k}) of the data batch simply by taking intra-cluster dispersion square root and dividing it by $B-1$. Moreover, if intra-cluster-deviation is 0, this means that all the points inside the data batch are from a single sensor. Which severely limits the analysis quality of the batch. On the other hand, if intra-cluster dispersion is too high, then this means that the data points are from a large, unrelated area which again limits the analysis quality. Thus, we define the location quality $Q_{loc}(m_k)$ of the data batch for message m_k as shown in (8). Which consists of a Gaussian that measures how close intra-cluster deviation (σ_{m_k}) is to the target deviation. Lastly, μ_0 and, σ_0 are the scaling factor of this Gaussian.

$$Q_{loc}(m_k) = e^{-(\frac{\sigma_{m_k} - \mu_0}{\sigma_0})^2} \tag{4}$$

As a result of (8), if the intra-cluster deviation of the batch coordinates are close to σ_0, then for our use case, its geo-distribution quality is high. Next, we look at the chronological relevance for a data batch using the age difference between the freshest data point and the oldest data point ($\Delta A(m_k)$) in (5).

$$\Delta A(m_k) = max_{1 \le j \le k \le |S_{m_k}|} |p_i(t) - p_j(t)| \tag{5}$$

Then, we derive the age quality of the data batch ($Q_{age}(m_k)$ in (6) by utilizing reverse sigmoid function that has smaller derivative.

$$Q_{age}(m_k) = (1 + e^{\Delta A(m_k)})^{-1} \tag{6}$$

As a result, the analysis quality of the data batch increases as the samples inside are created at relevant times.

Lastly, to also consider data imbalance we also calculate $Q_{\mathcal{J}}(m_k)$, which is the Jain's Fairness Index that utilizes data point count per sensor in m_k as in (7). Where, n_{s_i} shows the number of samples collected from sensor s_i.

$$Q_{\mathcal{J}}(m_k) = \frac{\left(\sum_{s_i \in S_{m_k}} n_{s_i}\right)^2}{B \cdot \sum_{s_i \in S_{m_k}} n_{s_i}^2} \tag{7}$$

It measures evenness of the distribution of the points for a given batch. If a batch has a skewed distribution where it has multiple samples from the same sensor, this decreases the fairness of the data batch. Then, we formulate $Q_I(m_k, t)$ in (8) as the geometric mean of sub-parts of intelligence quality.

$$Q_I(m_k) = GM\big(Q_{age}(m_k), Q_{loc}(m_k), Q_{\mathcal{J}}(m_k)\big) \tag{8}$$

C. Data Aggregation Protocol

To improve the utility of the system, we employ a distributed approach by either waiting for data pool to fill or, forwarding current pool to neighbors to be aggregated and processed. Edge nodes make this decision by estimating time it would take them to send their messages to server which is denoted by $\Delta T_k(t)$ for edge node e_k, as in (9).

$$\Delta T_k(t) = \Delta P_k(t) + \Delta L_k(t) \tag{9}$$

Here, $\Delta P_k(t)$ denotes estimated time that would take to fill the data pool of edge node e_k for current time. Additionally, $\Delta L_k(t)$ is the estimated routing time to server for location of e_k at time t. Our algorithm calculates $\Delta P_k(t)$ as in (10).

$$\Delta P_k(t) = \frac{D_{max} - D_k(t)}{\overline{R}_k(t)} + d_{proc} \tag{10}$$

Where $\Delta P_k(t)$ shows when edge node k will acquire M data points for given time instance t and will process them for μ_{proc} seconds. $D_k(t)$ shows current data point count and $\overline{R}_k(t)$ is the estimated incoming data rate depending on trajectory of the edge node k. Since edge nodes are moving fast and decision time (τ) is smaller than overall data point creation time, using static fill rate for a given time reduces accuracy of the estimation. Instead, we dynamically estimate incoming data rate by having looking ahead $n \cdot \tau$ seconds and taking the average incoming data rate as shown in (11). Where, $\overline{R}_k(t)$ is average data collection rate and $R_k(t)$ current data collection rate of edge node e_k which utilizes data arrival rates (λ) of neighboring sensors.

$$\overline{R}_k(t) = \frac{1}{n+1} \sum_{i=0}^{n} R_k(t + i \cdot \tau) \tag{11}$$

Aside from calculating the expected pool filling time, we also have to consider expected latency to the server. We compute this delay in a simple fashion as in (12) by extracting hop count ($l(e_k, t)$) from the routing tables.

$$\Delta L_k(t) = l(e_k, t) \cdot d_{hop} \tag{12}$$

Next, we develop a greedy algorithm that opportunistically seeks best neighbor to combine data points with. Basically, every τ seconds, an edge node will select the neighbor that reduces time to server sufficiently while providing the best expected utility weighted against its capacity. Edge nodes can only make aggregation decisions if their sample count is higher than δ_D. Then, each edge node selects edge nodes from its neighbors ($N_k(t)$) that provide less than or equal to routing delay to e_k at time t. In other words, potential aggregator set ($N'_k(t)$) does not include edge nodes that increase the routing

time to server as in (13). With this constraint, we enforce the aggregation direction towards the server.

$$N'_k(t) = \{e_j \in N_k(t) : \Delta P_j(t) \leq \Delta P_k(t)\} \tag{13}$$

Following this, QADA finds the best neighbor that provides best maximum weighted aggregated utility as in (14). If the set $N'_k(t)$ is empty, then e_k skips the aggregation for time t.

$$\arg\max_{e_j \in N'_k(t)} \{D_j(t) \cdot U_j(t + \Delta T'_j(t)) + D_k(t) \cdot U_k(t + \Delta T'_j(t))\} \tag{14}$$

Where $\Delta T'_k(t)$ is the estimated time to server *after* aggregation. We simply formulate this as summation of estimated time that edge node e_j needs after combining its pool with e_k, estimated time it would take to route to the server from edge node e_j and time it takes to transmit data points from e_k to e_j in (15).

$$\Delta T'_k(t) = \Delta P'_k(t) + \Delta L_k(t) + d_{hop} \tag{15}$$

Algorithm 1: Highest Quality Aggregator Selection.

input : e_k
output: bestNeighbor

1 **if** $D_k(t) > \delta_D$ **then**
  // Initialize parameters for comparison.
2 bestUtility $\leftarrow 0$
3 bestNeighbor $\leftarrow \emptyset$
  // Loop Potential Aggregators eq. (13)
4 **foreach** j *where* $e_j \in N'_k(t)$ **do**
5 $D'_j(t) \leftarrow D_{max} - D_k(t) - D_j(t)$
6 **if** $D'_j(t) > 0$ **then**
  // Get estimated fill time, eq. (11)
7 $\overline{R}_j(t) \leftarrow$ getAverageRate(e_j)
8 $\Delta P'_j(t) \leftarrow \frac{D'_j(t)}{\overline{R}_j(t)} + d_{proc}$
9 **else**
  // They do not need more data points.
10 $\Delta P'_j(t) \leftarrow d_{proc}$
11 $\Delta L_j(t) \leftarrow$ expectedRoutingTime(e_j)
  // Calculate estimated time to server eq. 15.
12 $\Delta T'_j(t) \leftarrow \Delta P'_j(t) + \Delta L_j(t) + d_{hop}$
  // Calculate aggregated utility.
13 aggrUtility $\leftarrow$ getAggrUtil$(e_j, e_k, \Delta T'_j(t))$
  // If it has sufficient minimum utility.
14 **if** aggrUtility $> \delta_v$ **then**
  // If its quality is better.
15 **if** bestUtility $<$ aggrUtility **then**
16 bestUtility $\leftarrow$ aggrUtility
17 bestNeighbor $\leftarrow e_j$
18 **return** bestNeighbor

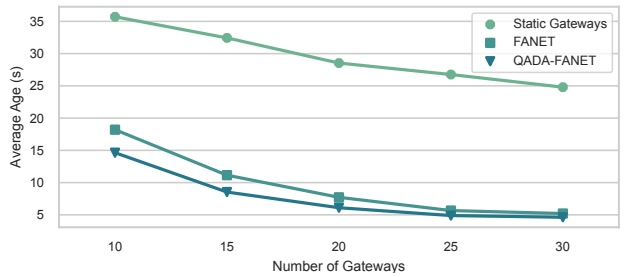
We present our algorithm 1 which describes a process for selecting the highest quality neighbor to aggregate data with, given a current edge node e_k. First, the algorithm checks if the data buffer of the current edge node e_k exceeds a threshold δ_D. If an edge node does not have enough samples, it can not start aggregation process. Following this, our algorithm loops over the neighboring edge nodes of e_k to find best neighbor. If a neighbor requires more hops than e_k, the algorithm excludes this neighbor from the potential aggregators set. Then for each neighbor it calculates the needed data point count after combining data pools in line 8. If this value is negative, it means that combined capacity is higher than batch size and therefore, e_N will not need more data points after aggregation. Thus, $\Delta P'_N$ will be 0 after aggregation (line 13). Next, it calculates estimated time to server after aggregation in line 14. One thing to notice here is, we also need to consider time it takes to transfer data pool to new neighbor thus, we also add d_{hop}. Next, the algorithm naively calculates utility of the potential aggregator using the equation 1 for its own data pool.

Finally, if the potential aggregator results at least δ_v more aggregated utility, algorithm considers it the best neighbor. Following this, e_k sends its data pool to this neighbor to be aggregated. To enable calculations that require more knowledge that an edge node has (lines 5, 7, 8, 11, 13, 14) the ground controller in the system must periodically broadcast relevant mission data from the control link as shown in the figure 1. For starters, it periodically broadcasts age vector off all sensors to all edge nodes in the area. On the other hand, edge nodes also notify the ground controller about their location, current capacity and estimated utility. Only then, the algorithm 1 can function properly since, it requires these global information to be known.

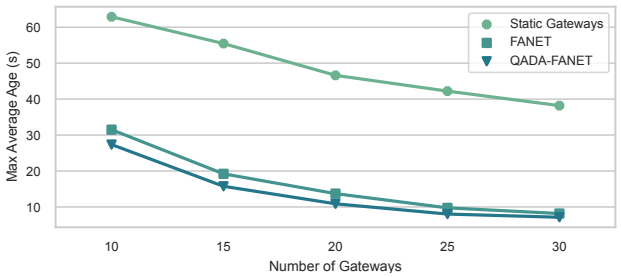
III. Performance Evaluation

Our evaluation methodology consist of creating detailed OMNET++ simulations with INET framework. In this setup, we consider multiple sensors distributed throughout the 25 km^2 area which broadcast their data periodically. These broadcasts then picked up by edge nodes around the environment to be processed and forwarded to cloud in the case of an emergency. In our simulations, we also test static gateways placed throughout the site to create a robust baseline using SWAIN[1] project as inspiration. For this case, the gateways have satellite communication capabilities. Also, since they have much lower altitude compared to UAVs, they experience fading effects higher than the UAVs [3]. Thus, enhanced fading effects reduce their coverage. To model these effects accurately, we employ log-normal model. According to many comprehensive studies [3], [7], [8] log-normal performs well for both satellite, air-to-ground and forest communication links. For our simulations, we take pathloss exponent 2.4 with standard deviation of 1.5 for the cases with UAVs and, 2.7 with standard deviation of 6 for static gateways in the area.

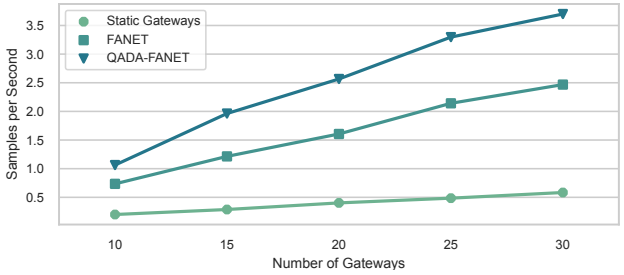
[1]http://swain-project.eu/



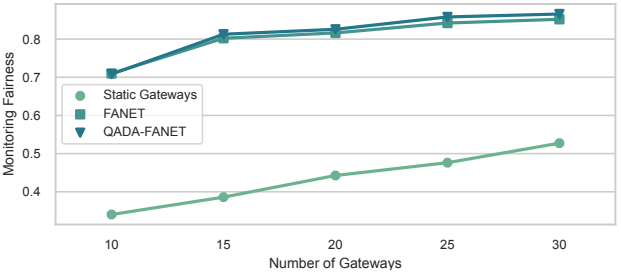
(a) Average Age of information (AoI) of sensors.



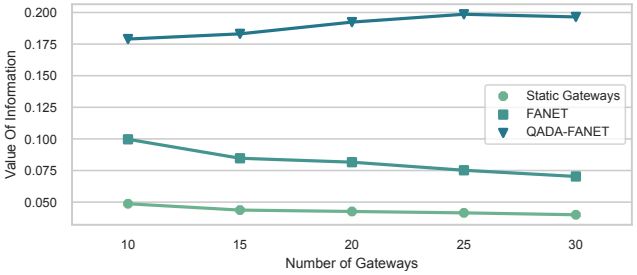
(b) Max Average Age of information (AoI) of sensors.



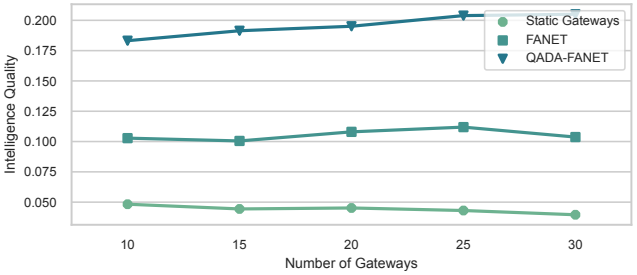
(c) Collected samples per second.



(d) Fairness of the monitoring scheme.



(e) Value of information of updates (VoIU).



(f) Intelligence Quality

Fig. 2: Simulation Results.

We have 3 cases which utilize UAVs: FANET, Random-FANET and QADA-FANET. In FANET, UAVs circle random centers throughout the mission area and listen for sensor broadcasts. When they process their data pools, they forward it to base station via AODV routing protocol. QADA-FANET case uses algorithm 1 to opportunistically select which neighbor to aggregate. For each new simulation seed, gateways and IoT sensors take random coordinates. Mobility for the cases with UAVs is different. For starters, all UAVs have same radius for their circular trajectories but their centers are randomly distributed. Additionally, the rest of the simulation parameters are in table II.

Our initial simulations start with the average age measurements for given simulation cases in figure 2a. The results show that the reduced communication range of the statically placed gateways in the ground severely reduces their ability to track the overall environment. FANET cases, on the other hand, can provide fresh data to server. QADA-FANET, reduces overall AoI by additional 18.1% compared to FANET as it fills its data pool faster by opportunistically aggregating data pools.

Next, we look the average maximum age of the system in figure 2b. Similarly, due to their limited ranges, Static Gateways can not track the environment effectively and as a result, they have the highest data age. Because of VoIU-based aggregation, QADA results in lower AoI compared to FANET case. Following this, we examine sample collection rate of the simulation cases in the figure 2c. The reduced data pool fill time of the QADA substantially increases its sample throughput compared to other cases. This means that QADA collects more fresh data compared to other methods. Next,

TABLE II: Simulation parameters

Parameter	Value
Capacity Threshold (δ_D)	0.1
Cluster Deviation (μ_0)	400 m
Cluster Deviation Mean (σ_0)	400 m
Data Pool Size (B)	24 samples
Decision Period (τ)	2 s
Gateway Count	10-30
Hop Delay (d_{hop})	200 ms
Look-ahead horizon (n)	5
Processing Delay (d_{proc})	250 ms
Satellite Comm. Delay	300 ms
Sensor Data Arrival Rate (λ)	0.2 / s
Sensor Count	100
Trajectory Radius	500m
UAV Altitude	50-250 m
UAV Speed	30-50 m/s
Utility Gain Threshold (δ_v)	0.03
Weight of Q_A (c_1)	0.9
Weight of Q_I (c_2)	0.1

we take a look at the VoIU of the messages that reach to the server for all methods in figure 2e. As expected, updates of QADA has the highest VoIU. Because of this, it can increase the throughput gracefully, without increasing the overall AoI. Static Gateways provide less VoIU because its resulting data pools are less diverse and therefore updates carry information about fewer sensors.

In figure 2d, we measure monitoring fairness of the methods using Jain's Fairness index as in (7). This shows us how proposed methods distribute their resources to track all sensors evenly by using total number of collected points per sensor. The superior coverage of the FANET based result in more fair tracking than the Static Gateways in the ground. Lastly, we look at the Intelligence Quality of the messages in the figure 2f which shows the same trend as the previous measurements. Static Gateways suffer from reduced coverage, FANET case improves this substantially and, by utilizing quality focused aggregating QADA provides even better intelligence Quality. While concluding the evaluation of the methods, we can conclude that it requires a higher number of Static Gateways in the ground to provide relatively similar performance to FANET based cases. Which is not viable due to increased deployment costs coupled with increased ecological disturbance.

IV. Related Work

Flying Edge Intelligence/Computing (FEI/FEC) for IoT systems has been explored in the literature. The main approach in the literature is optimizing/exploiting the locations of the UAVs while considering mission goals. [9] explores energy efficiency and system throughput tradeoff for UAV-IoT systems. After formulating the UAV-IoT data collection problem, they use particle swarm optimization (PSO) method to find optimal UAV altitude, speed and optimal frame length while considering energy efficiency and system throughput. Similarly, the work [10] also formulates UAV assisted MEC system performance while focusing on the completion time and the energy usage of the system. Then, it calculates optimized flight route that maximizes throughput while minimizing the completion time and energy usage by utilizing Sine Cosine Algorithm (SCA). By utilizing MEC, [11] formulates a closed form solution to minimize response delay from Edge nodes. They consider UAV distributions, and computation capabilities such as number of virtual machines in the deployment to reduce service delay of the deployment. On the other hand, [12] proposes RL based dynamic staleness control algorithm that achieves effective information dissemination that minimizes AoI while maximizing value of received data. Lastly, authors of [13] develop UAV assisted Real Time Air-Quality Monitoring System for Landfills. In their deployment, UAVs collect the data and upload it to the cloud via long range Wi-Fi links. Additionally, the authors share their comprehensive air quality measurements.

Different from these works, we utilize FANETs to increase SEM system coverage over rural areas compared to statically placed gateways in the environment. Moreover, we are the first to combine EI processing with collaborative approach between UAVs with by considering intelligence quality of the collected points and their age of information impact to improve AoI, data quality and sample rate of the SEM system.

V. Conclusion

In this work, we introduced an Quality Aware Data Aggregator (QADA) that opportunistically combines data from different edge nodes in Flying Edge Intelligence assisted Smart Environmental Monitoring System (FEI-SEM). With QADA, every edge node of FEI periodically evaluates their data pool quality and informs the ground controller. Then, considering the expected utility by the time their data reaches to the server, Edge Nodes select best peer to aggregate their data pools to speed up the data collection process while also considering the aggregated data pool's quality. They also take the data ages of the sensor's into account to provide data pools with most freshness impact. QADA, achieves lower overall age of information for sensor data while also increasing the overall intelligence quality.

Acknowledgment

This work is supported by Istanbul Technical University, Department of Scientific Research Projects (ITU-BAP, MGA-2022-44272) and by the CHIST-ERA grant CHIST-ERA-19-CES-005 and, by the Austrian Science Fund (FWF): I 5201-N. Additionally, T. Tolga Sarı is funded by BTS Group.

References

[1] S. Alsamhi, F. Afghah, R. Sahal, A. Hawbani, M. A. Al-qaness, B. Lee, and M. Guizani, "Green internet of things using uavs in b5g networks: A review of applications and strategies," *Ad Hoc Networks*, vol. 117, p. 102505, 2021.

[2] S. L. Ullo and G. R. Sinha, "Advances in smart environment monitoring systems using iot and sensors," *Sensors*, vol. 20, no. 11, p. 3113, 2020.

[3] W. Khawaja, I. Guvenc, D. W. Matolak, U.-C. Fiebig, and N. Schneckenburger, "A survey of air-to-ground propagation channel modeling for unmanned aerial vehicles," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2361–2391, 2019.

[4] A. Kosta, N. Pappas, and V. Angelakis, "Age of information: A new concept, metric, and tool," *Foundations and Trends® in Networking*, vol. 12, no. 3, pp. 162–259, 2017.

[5] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "The cost of delay in status updates and their value: Non-linear ageing," *IEEE Transactions on Communications*, vol. 68, no. 8, pp. 4905–4918, 2020.

[6] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, vol. 3, no. 1, pp. 1–27, 1974.

[7] S. Kurt and B. Tavli, "Path-loss modeling for wireless sensor networks: A review of models and comparative evaluations." *IEEE Antennas and Propagation Magazine*, vol. 59, no. 1, pp. 18–37, 2017.

[8] X. Guo and C. Zhao, "Propagation model for 2.4 ghz wireless sensor network in four-year-old young apple orchard," *International Journal of Agricultural and Biological Engineering*, vol. 7, pp. 47–53, 12 2014.

[9] X. Lin, G. Su, B. Chen, H. Wang, and M. Dai, "Striking a balance between system throughput and energy efficiency for uav-iot systems," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 519–10 533, 2019.

[10] C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato, "Completion time and energy optimization in the uav-enabled mobile-edge computing system," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7808–7822, 2020.

[11] Q. Zhang, J. Chen, L. Ji, Z. Feng, Z. Han, and Z. Chen, "Response delay optimization in mobile edge computing enabled uav swarm," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3280–3295, 2020.

[12] A. Aral, M. Erol-Kantarci, and I. Brandić, "Staleness control for edge data analytics," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 4, no. 2, jun 2020. [Online]. Available: https://doi.org/10.1145/3392156

[13] R. Sharma and R. Arya, "Uav based long range environment monitoring system with industry 5.0 perspectives for smart city infrastructure," *Computers & Industrial Engineering*, vol. 168, p. 108066, 2022.