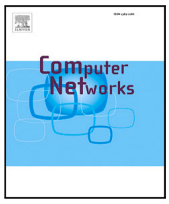
Computer Networks 256 (2025) 110887

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



A²-UAV: Application-Aware resilient edge-assisted UAV networks☆

Andrea Coletta [a], Flavio Giorgi [a], Gaia Maselli [a,*], Matteo Prata [a], Domenicomichele Silvestri [a], Jonathan Ashdown [c], Francesco Restuccia [b]

[a] Department of Computer Science, Sapienza University of Rome, Italy
[b] Institute for the Wireless Internet of Things, Northeastern University, United States
[c] Air Force Research Laboratory, United States

ARTICLE INFO

Keywords:
UAV networks
Edge computing
Network management

ABSTRACT

During advanced surveillance missions, Unmanned Aerial Vehicles (UAVs) usually require the execution of edge-assisted computer vision (CV) tasks. In multi-hop UAV networks, the successful transmission of these tasks to the edge is severely challenged due to severe bandwidth constraints, and the possible node failures. To address these critical challenges, we propose a novel A²-UAV framework that optimizes the number of correctly executed tasks at the edge. In stark contrast with existing art, we take an *application-aware* approach and formulate a novel Application-Aware Task Planning Problem (A²-TPP) to optimize routing, data pre-processing and target assignment for each UAV. Our formulation explicitly takes into account (i) the relationship between CV task accuracy and image compression for the classes of interest based on the available dataset, (ii) the target positions, (iii) the current energy/position of the UAVs, and (iv) the possible node failures. We demonstrate A²-TPP is NP-Hard and propose a polynomial-time algorithm to solve it efficiently. We extensively evaluate A²-UAV through simulation and real-world experiments using a testbed composed by four DJI Mavic Air 2 UAVs. Results on image classification show that A²-UAV attains on average around 38% more accomplished tasks w.r.t. the state of the art, with a 400% improvement in tasks-intensive scenarios. Moreover, we show that our framework is able to reconfigure the network in case of nodes failure.

1. Introduction

Unmanned Aerial Vehicles (UAVs), commonly known as drones, have received significant interest for their potential applications in post-disaster scenarios, where human intervention is challenging or inefficient, due to the vast or harsh area. The key advantage of Unmanned Aerial Vehicles (UAVs) is the combined presence of advanced sensor equipment (*e.g.*, cameras, radars, and GPS), wireless multi-hop networking and mobility in the same device, thus enabling critical applications such as automatic target (*e.g.,* object, person) detection and tracking.

To perform their functions, modern UAVs necessarily depend on the execution of computation-heavy CV tasks to analyze in real time the images of the target area. These tasks usually rely on deep learning (DL) framework using very deep neural networks (DNNs) such as ResNet [1] and DenseNet [2], which are computationally prohibitive for UAVs [3]. To extend UAVs battery lifetime and keep task execution time within acceptable levels, offloading the stream of tasks to neighboring edge servers (*e.g.*, the depot) is a feasible option [4–9]. Fig. 1 shows an example of edge-based task offloading in UAVs, where expensive computational tasks are offloaded to an edge server through multi-hop connection, without relying on any communication infrastructures (e.g., cellular networking) that may often be disrupted in harsh environments [10]. The result of the computation is eventually sent back to the UAVs for control purposes.

Unfortunately, *UAVs networks typically experience limited bandwidth and frequent packet loss* [11]. Prior work on UAV edge task offloading — discussed in details in Section 2 — assumes a single-hop communication between the UAVs and the edge [12,13], or focuses only on networking aspects on a multi-hop communication [14]. All fall short in considering the specific task requirements, which ultimately limits the number of correctly executed tasks. Existing work also rarely uses a testbed to measure performance experimentally.

In stark contrast, we propose A²-UAV, *an **A**pplication-**A**ware (A^2) optimization framework which jointly optimizes **UAV** network deployment and task accuracy*. Our key intuition is that a different image compression will result in a different accuracy for the DNN model. Specifically,

☆ Approved for Public Release; Distribution Unlimited: AFRL-2022-1069.

* Corresponding author.
E-mail addresses: coletta@di.uniroma1.it (A. Coletta), giorgi@di.uniroma1.it (F. Giorgi), maselli@di.uniroma1.it (G. Maselli), prata@di.uniroma1.it (M. Prata), silvestri.d@di.uniroma1.it (D. Silvestri), jonathan.ashdown@us.af.mil (J. Ashdown), frestuc@northeastern.edu (F. Restuccia).

https://doi.org/10.1016/j.comnet.2024.110887
Received 20 February 2024; Received in revised form 26 September 2024; Accepted 29 October 2024
Available online 15 November 2024

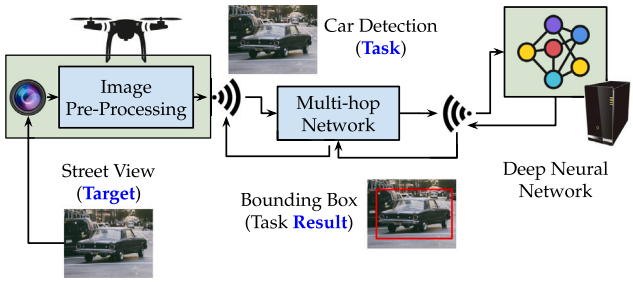


Fig. 1. Edge-based task offloading in UAVs networks.

compressed images have low impact on the network throughput but decrease the DNN accuracy. Conversely, uncompressed images are likely to be classified more correctly but cause higher network overhead. To better highlight this intuition, Fig. 2 shows an example of an original and compressed ($l = 20$) images of *Wildlife* and *Tools* from the ImageNet dataset [15]. Fig. 2.B shows how a hatchet at compression level $l = 5$ is less recognizable with respect to its original version (Fig. 2.A), due to the low contrast with the background and the small object dimension. On the other hand, Fig. 2.D shows how a jaguar at level $l = 20$ is easily distinguishable, thanks to its fur pattern and the higher contrast with the background. We consider lossy compression algorithms for images (e.g., JPEG algorithm [16]), as lossless algorithms do not usually meet the requirements of real-time applications.

Trading-off network load and latency with task accuracy while aiming at maximizing the target coverage is a daunting challenge. A^2-UAV considers all these aspects and balances between images compression and network coverage according to current network conditions and application requirements. In Section 3.3 we show that different applications have starkly different compression-accuracy relationships. Specifically, A^2-UAV maximizes the number of accomplished tasks at the edge, producing a connected coverage formation for the UAVs of the squad and a compression level assignment for the UAVs that satisfies the application requirements. The connected coverage formation is designed such that it jointly maximizes the number of covered targets, minimizes network delay due to inefficient routes or channel contention, and minimizes task miss-classification due to high input compression. We show through simulation and *real-world experiments* with a testbed that Greedy-Application-Aware Task Planning Problem (Greedy-A^2-TPP) attains on the average 38% more accomplished tasks than the state-of-the-art networking-based approach, with a sharp increase (400% more accomplished tasks) when the number of tasks to offload drastically increases.

Handling node failures is a key requirement and major challenge in UAVs networks. As UAVs have usually low computational power and energy UAV networks can suffer from node failures, especially when deployed in vast and harsh area. Node failures not only impact the communication layers, but in the context of a multi-hop network a failure can easily impact also the application layer: task-offloading may be disrupted with severe consequences. Our framework introduces an application-aware re-optimization scheme (see Section 5) aimed at restoring connections and resuming task offloading.

This paper makes the following novel contributions:

- We design A^2-UAV — a novel application-aware framework that optimizes the number of accomplished tasks at the edge by finding the best network deployment and compression level assignment for the considered application. We first design a *Application Aware Task Analyzer* (A^2−TA) to learn the requirements of the tasks, and to map the possible data compression of the UAVs to the expected task accuracy at the edge. Then, we define the A^2-TPP to assign the UAVs to the tasks. We prove the NP-hardness of the problem and formulate a polynomial-time Greedy-A^2-TPP algorithm to solve it efficiently;
- We extend A^2-UAV to handle node failures by applying a cost-efficient re-optimization scheme. Such adaptive A^2-UAV can support network failures as well as a more dynamic nature of targets, i.e., targets may dynamically appear or change position. We evaluate the adaptability of A^2-UAV showing minimal task loss even in the face of environmental changes, such as node failures.
- We study the performance of A^2-UAV with extensive simulations. We analyze six different critical applications for UAVs, including Search-and-Rescue, Maritime and Wildlife monitoring. We show how Greedy-A^2-TPP accomplishes around 38% more tasks with respect to existing network-based approaches, thanks to its application-aware optimization; Whereas, the NP-hard version (Opt-A^2-TPP) attains on average 50% performance increase on restricted problem instances.
- We implement A^2-UAV into a testbed and perform *real-field experiments*. We consider four UAVs and a Jetson Nano board, mounting a Raspberry PI for computation and communication. We execute an image analysis application in which UAVs periodically acquire images from on-board sensors, with different delay requirements. We implement four state-of-the-art image classification models (i.e., DenseNet [2], ResNet152, ResNet50 [1] and MobileNet-V2 [17]), and one object detection model (YoloV4 [18]) which are executed at the edge server on the Jetson Nano board, and we let the UAVs offload tasks through WiFi connection. Experimental results confirm the outstanding performance of A^2-UAV.

This paper extends our previous work [19] introducing an application-aware recovery technique to re-optimize the network in case of UAV failures or dynamic changes in the target points.

2. Related work

A^2-UAV framework jointly optimizes the UAVs deployment and the DL tasks offloading towards the edge. Moreover, A^2-UAV incorporates a novel technique to tolerate UAV failures, and restore the network operations.

Considering the multifaced contribution, we survey prior work according each distinct contribution of our paper.

2.1. Routing in UAV networks

In the last years, networks of UAVs have been increasingly used in several scenarios to offload users' tasks and data to mobile edge servers [20] or enhance ground user connectivity, acting like aerial base stations [21]. Considering the novelty of such networks, several communication issues have been identified and only partially addressed [22]. In particular, in the context of aerial communication and networking several routing protocols have been proposed [23–27] but, to date, a de facto standard does not exist.

In fact, upon on the operative scenarios, different algorithms may have better performance. For example, a geographical scheme [28] may be preferable in case of a single edge and with a good area coverage (i.e., the drones are almost always connected). Instead, in case of a few drones in a vast area a controlled mobility scheme is preferable to deliver packets [29].

2.2. Network restoration in UAVs networks

Several studies have explored the utilization of Unmanned Aerial Vehicles (UAVs) for network restoration, showcasing the potential of drones in addressing node failures [30–32]. Park et al. [31] leveraged drones for network restoration, demonstrating how UAVs can reconnect a stationary ad hoc network severely damaged in a post-disaster scenario. Similarly, Zear et al. [32] propose UAV-NetRest in which they employ UAVs to assist Network Partition Detection and Connectivity Restoration. Such work highlights the versatility of aerial vehicles in

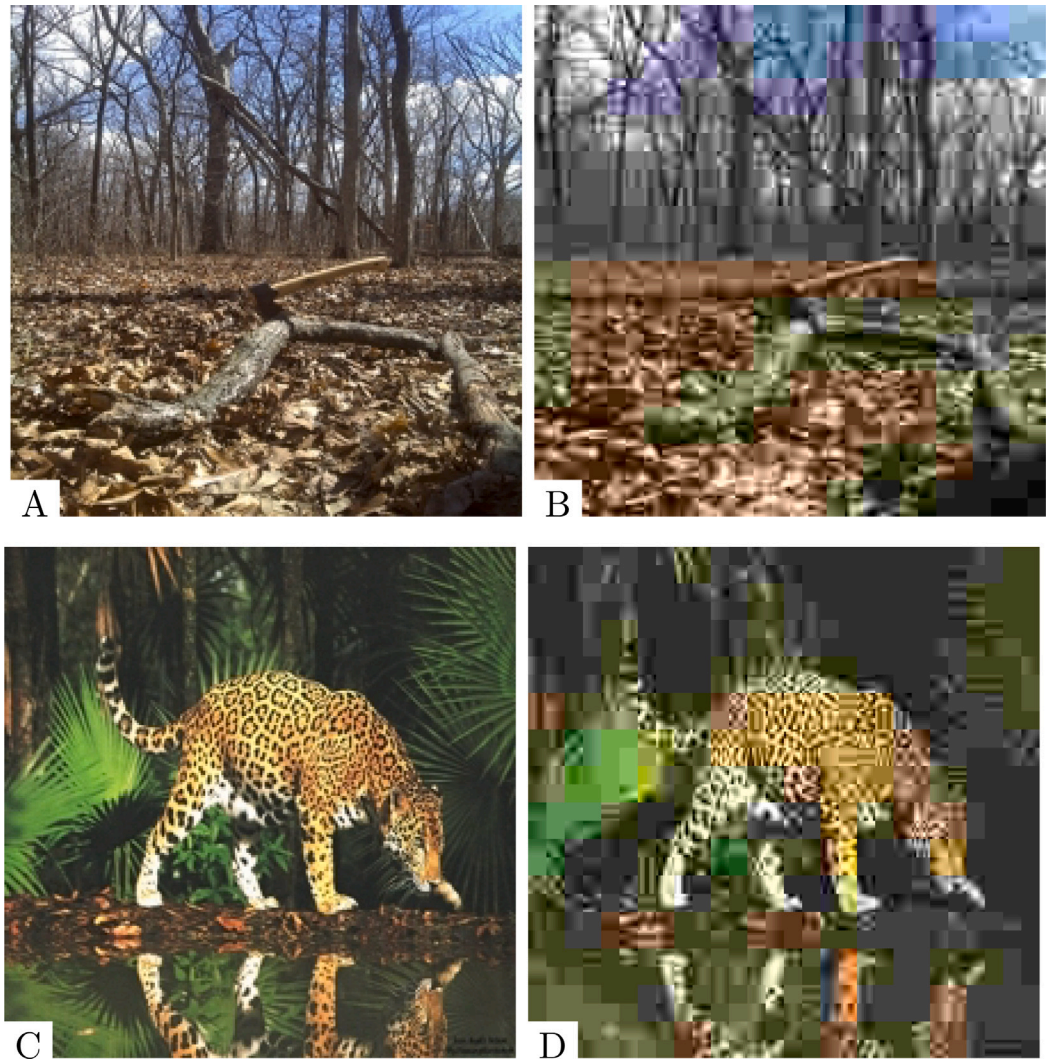


Fig. 2. Examples of application-aware compression. A and C (B and D) represent the images before (after) processing.

mitigating the impact failures in wireless networks, but do not address the possible failures of drones.

Recent effort have been made to optimize network restoration strategies in UAV networks [33,34]. For example, Tian et al. [33] proposed a more optimized approach, emphasizing cooperation among UAVs to detect partitions or separated clusters due to the high node and link dynamic. Similarly, Yin et al. [34] address network restoration keeping into account also the load balance of the new topology and possible delay constraint. However, these approaches do not target our scenario, where a edge-assisted UAV network has the primary goal to offload the collected data to the edge optimizing the application tasks.

2.3. Edge-assisted UAVs networks

Only very recently has the literature considered task offloading in the context of edge-assisted DL-based applications [11,13]. Chuprov et al. [11] show how the performance of the end-line ML systems is affected by the quality of data and network. They consider packet loss and limited bandwidth in a image classification task, and they recommend to stop the system when packet loss reaches 2%–5%. In Section 6.3 we show that our system enables the classification task even with 15% of packet loss. Chen et al. [14] consider a hierarchical offloading of computation tasks. Conversely from us, they focus on the communication and routing of tasks toward a more computational powerful device, without focusing on the specific task requirements. Yang et al. [12] propose a hierarchical DL task execution framework, in which only a few lower layers of a Convolutional Neural Network (CNN) are on the UAVs, while the edge server contains the higher layers of the model, which need more resources. However, a single-hop high-performance 4G network is considered, while we focus on the more challenging scenario of multi-hop connectivity toward the edge. Recently, Callegaro et al. [13] proposed SeReMAS, a framework where the application-, network- and telemetry-based features are used to select and assign UAVs tasks to the most reliable edge servers. However, a single-hop system is considered, and data compression is not explored.

2.4. UAVs deployment algorithms

As one of the output of A^2-UAV is a connected coverage formation, we mention also some prior art on UAVs deployment optimization [14,35–39], which however does not consider task offloading. Natalizio et al. [40] have considered the problem of minimizing networking resources while maximizing the user experience (i.e., perceived quality) when filming sport events. Moreover, [4,5,41] optimize network deployment under continuous or periodic connectivity constraints, but they do not consider critical indicators such as task accuracy with delay constraints, which are critical to the UAVs mission. Recently, Nguyen proposed a Steiner-Tree-Based Algorithm (STBA) for target coverage

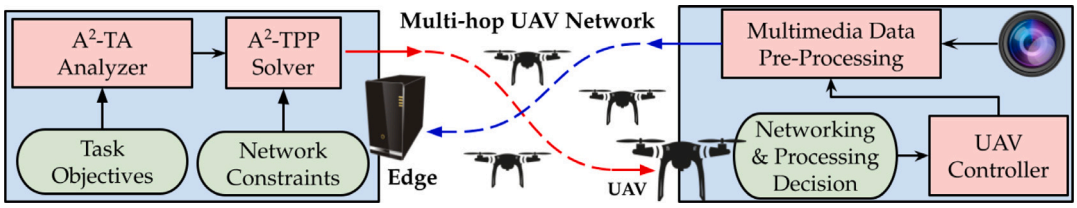


Fig. 3. High-level overview of A²-UAV.

and network connectivity [42], where Fermat points and the node-weighted Steiner tree algorithm are used to find a tree such that most of the targets are covered, and the UAVs are minimized. In Section 6, we consider variants of [42] as performance benchmarks for A²-UAV.

We conclude that our work is the first to address a task-offloading problem keeping as primary goal the application, i.e., we optimize the executed tasks at the edge. Existing work mostly focuses on the network and communication aspects, which not always translate in a good service for the final user.

Our work builds on a previous conference paper [19], and it contributes to a practical problem, proposing a comprehensive framework for edge-assisted UAV networks. With respect to the conference version, we introduce the following contributions: we add a more extensive discussion of related work in Section 2; we introduce a re-optimization scheme to handle node failures and dynamic changes in the targets, in Section 5; we extended the experimental section with a new real-field test to evaluate our new adaptable algorithm in Section 6.3.1; we better discuss model assumptions, including details on collision resolution strategies for UAV networks, on the UAV energy consumption model, and adopted the communication model in Section 3.1.

3. The A²-uav framework

In this section, we give an overview of A²-UAV (Section 3.2) and describe the two key components of A²-UAV: A²−TA (Section 3.3) and A²-TPP (Section 3.4).

Fig. 3 shows a high-level overview of A²-UAV. The Application-Aware Task Analyzer (A²−TA) at the *edge server* learns the relationship between the image compression and the accuracy on a set of classes of interest, and passes its output to the Application-Aware Task Planning Problem (A²-TPP) solver, which jointly optimizes UAVs positions, routing policy, and data compression strategy to maximize the number of correctly executed tasks per unit of time. The optimal network deployment and image compression levels are sent to the *UAVs network*, which moves to the targets, monitors them and streams tasks to the edge through the multi-hop connection. In case of any node failures or changes in the target points we apply the re-optimization technique described in Section 5.

3.1. System model and assumptions

We assume one or more UAVs are deployed over an Area of Interest (AoI), which contains several *targets*, e.g., the location of a vehicle, person, or any entity of interest. Each UAVs is equipped at least with (i) multimedia sensors (e.g., camera and microphone); (ii) a single radio for communication; and (iii) a computational unit. The edge server is equipped with low-latency hardware for DL computation. We do not rely on any communication infrastructures (e.g., 5G) and assume edge offloading is realized through multi-hop communication. A UAV monitors a target by sampling data through its sensors and generates a *task* to be executed at the edge. A task could be "car, bicycle, or bus detection on a video camera frame every 10 frames". The task is then sent to the edge server through a multi-hop connection, where a state-of-the-art DL model is run to perform the task. We assume each task has mission-driven constraints in terms of (i) minimum classification accuracy given a specified DL model; (ii) maximum latency, defined as the time between the task generation and its successful execution. Thus, a task is successfully executed if (i) promptly offloaded to the edge; and (ii) correctly analyzed by the model within a deadline.

Table 1
Table of symbols.

Notation	Description
$\mathcal{U}$	A set of available UAVs of the fleet
$\mathcal{T}$	A set of targets to cover
σ	The edge server
r_{com}	Drone's communication radius
r_{sens}	Drone's sensing radius
l_u	Distance traveled by a drone
δ_u	Drone's overall energy consumption
$\hat{e}^s_{ij}$	Amount of data transmitted through the link between UAV i and UAV j
$\hat{e}^a_{ij}$	Expected task accuracy at the edge, for each task
$\rho_{i,j}$	Estimated channel data rate
$\hat{p}_u$	Position vector assigned by the solver to drone
β_u	Energy spent for each distance unit traveled at constant speed
α_u	Energy spent in a steady position
$\hat{\omega}^u_{ij}$	Drone u monitors the target i with a compression j or not
ϕ_u	UAV initial energy
$Q(s,l)$	A tuple with expected accuracy and data size of the frame of the application scenario s, with compression level l
Ψ	The final connected coverage formation returned by the greedy algorithm
τ_{best}	Best coverage found during an iteration
τ_{par}	Partial coverage to enhance or join with Ψ

Collision avoidance. In this work we do not directly address collision avoidance. When the UAVs transit from the base station to the assigned targets and back, we assume they can adjust their height to avoid obstacles. In fact, UAVs may directly integrate obstacle avoidance systems [43] or they can adopt online collision avoidance mechanisms [44].

Energy model. In our work, we employ a simple energy model to keep the optimization linear and efficient. As shown in the Sections Section 3.4, our model has the advantage to allow energy modes, by simply setting the proper energy expenditure per unit distance traveled by the UAVs. We note that one can easily integrate more complex and non linear models, e.g., those proposed by Goss et al. [45]; or the consumption model reported in existing work on multiple drones [8, 46,47].

Communication model. As mentioned, in the problem formulation we utilize a simplified model considering a free-space line-of-sight communication, adopted for the sake of analytical tractability. In particular we consider that each drone has r_{com} communication radius, using an isotropic antenna with same radiation of signals in all directions. Therefore, the communication range is uniform, and communication is possible in all directions within r_{com}. This model is also used for UAV-to-Edge communication. We envision that this setting is suitable for free-space line-of-sight communication, which are typically of aerial communications, especially in scenarios where operations typically occur in open areas with long distances. While such simplified communication range r_{com} is acceptable for such real-world conditions, more sophisticated models can also be employed if necessary.

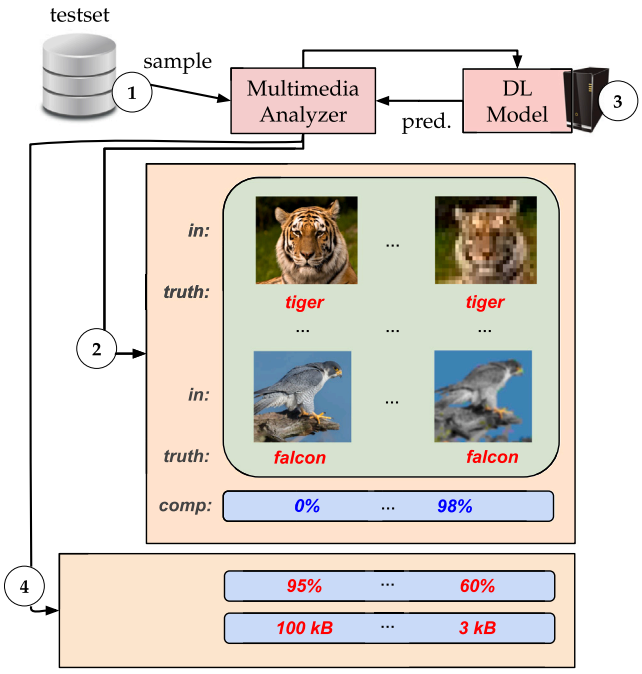


Fig. 4. Main operations of the A²−TA module.

3.2. Overview of A²-UAV

The ultimate goal of A²-UAV is to maximize the number of correctly executed tasks. To approach this challenging issue, and conversely from existing work, *A²-UAV takes into account how the task success is affected by the image compression*. To this end, A²-UAV jointly optimizes the deployment of UAVs and the task offloading to maximize the number of executed tasks. Each UAV is made up of two key modules. First, the *UAV Controller* implements networking and data processing decisions (next position, targets to cover, sampling process, and offloading routes) received from the A²-TPP. Second, the *Multimedia Data Pre-Processing* module samples data and creates tasks by pre-processing collected multimedia data according to the A²-TPP solution.

3.3. Application-Aware Task Analyzer (A²-TA)

The A²−TA module determines the relationship between different image compression levels and task accuracy so that UAVs can reduce the amount of transmitted data and avoid network congestion, while satisfying application requirements.

Fig. 4 shows the workflow of the A²−TA, which is executed before network deployment, by using the datasets of the specific DL application. The A²−TA iterates over the dataset items, creating a new dataset of pairs (*in*, *truth*) for each of the considered compression levels (**step 1** and **step 2**). Each pair is fed to the DL model, which outputs a prediction for *in* (**step 3**). Finally, the predictions are compared with the ground truth, and the model's performances are averaged over all the items according to their compression level (**step 4**). This process allows to map each compression level to the average data size and accuracy obtained with the model.

Formally, let us define the function: $Q : \mathcal{S} \times \mathcal{L} \rightarrow \mathbb{R}^2$ mapping an application scenario in the set $\mathcal{S}$ and a compression level in the discrete set $\mathcal{L} = \{1, .., 100\}$ a tuple in $\mathbb{R}^2$ representing the average *accuracy* and *data size*. This function is learned through the A²−TA module described beforehand. Each sampled image is compressed according the compression level l by the JPEG compression algorithm [16], and it is fed into the DL model to estimate average accuracy and data size.

To give an example, we consider 5 scenarios: **Maritime** (Fireboat, Wreck, Lifeboat, Ocean liner, Speedboat), **Search-and-Rescue (SaR)** (Fire truck, Ambulance, Police van, German shepherd, Pickup truck) **Wildlife** (Kit fox, Polecat, Red wolf, Zebra, Jaguar), **Tools** (Screwdriver, Power drill, Hatchet, Hammer, Chainsaw), **Pets** (Golden retriever, Pomeranian, Guinea pig, Persian cat, Hamster). Fig. 5.a shows the accuracy for the different scenarios as a function of the compression, while Fig. 5.b shows the accuracy for the same scenario when different DL models are used. The figures highlight the need of the A²−TA. For example, images of *Tools* have low accuracy, constraining the compression at level $l = 8$ to achieve at least 40% of accuracy, while *Wildlife* can achieve the same accuracy with higher compression $l = 25$.

3.4. A²-TPP MILP formulation

We formalize A²-TPP as a Mixed Integer Linear Programming (MILP). Hereafter we denote the edge server as σ, the set of targets to monitor as $\mathcal{T}$, the set of available UAVs of the fleet as $\mathcal{U}$. The A²-TPP solver outputs: (i) a connected coverage formation of UAVs, and (ii) the compression level each drone must adopt to capture images when inspecting a target (refer to Table 1 for all symbols).

Definition 3.1. A subset of UAVs $F \subseteq \mathcal{U}$ is deployed according to a connected coverage formation, when some of the UAVs in F are employed to inspect a subset of targets $M \subseteq \mathcal{T}$, while being connected to the base station σ, either directly or through a multi-hop sequence of the other UAVs in F.

Definition 3.2. A task for a drone $u \in F$ covering a target $t \in M$, consists in delivering an image captured from the drone's on-board cameras to the base station σ. It is said to be an accomplished task if two conditions hold: (1) when received at σ, the time since the task was created is not superior to a threshold Δ; (2) when the captured image reaches σ, the DL model outputs a correct prediction for it.

We define the UAV sensing range and communication range as r_{sens} and r_{com}, respectively. We denote with p the position vector of the entities involved, and to (p_u^x, p_u^y) for the x and y coordinates respectively. In particular p_u is the position of UAV u, $\forall u \in \mathcal{U}$ at the beginning of the mission; p_i to the position of the target i, $\forall i \in \mathcal{T}$; p_σ the position of the edge server. We denote with $\hat{p}_u$ the position vector assigned by the solver to drone u, $\forall u \in \mathcal{U}$. We define the distance traveled for each drone as $l_u = |p_u - \hat{p}_u|$, $\forall u \in \mathcal{U}$.

To estimate energy consumption, we define β_u as the energy spent for each distance unit traveled at constant speed, and α_u as the energy spent in a steady position, for a given unit of time. The values of β_u and α_u are estimated through on-field experiments or from technical specifications. The overall energy consumption for UAV u is defined as $\delta_u = l_u \cdot \beta_u + \alpha_u \cdot \Lambda$, $\forall u \in \mathcal{U}$, where Λ is an upper-bound of the time required, once reached the targets, to monitor the targets and complete the mission. We constraint the reachable points according to the UAVs initial energy ϕ_u, as follows:

$$\delta_u + |p_\sigma - \hat{p}_u| \cdot \beta_u \leq \phi_u, \ \forall u \in \mathcal{U} \tag{1}$$

This constraint defines the positions that are reachable as they let the UAVs with enough energy to come back to the edge-server for recharging operations. We use the binary variables $\hat{\omega}_{ij}^u \in \{0, 1\}$, which define if the drone u monitors the target i with compression level j or not. A target i is monitored if an only if the UAV u is close enough to the target position p_i. Formally we want to constraint $\hat{\omega}_{i,j}^u = 1 \iff |\hat{p}_u - p_i| \leq r_{\text{sens}}$ which becomes $\forall u \in \mathcal{U}$, $\forall i \in \mathcal{T}$, $\forall j \in \mathcal{L}$: :

$$\begin{aligned} r_{\text{sens}} &\geq |\hat{p}_u - p_i| - M_{\text{cost}} \cdot (1 - \hat{\omega}_{ij}^u) \\ r_{\text{sens}} &\leq |\hat{p}_u - p_i| + M_{\text{cost}} \cdot \hat{\omega}_{ij}^u \end{aligned} \tag{2}$$

where M_{cost} is a big constant used to model binary variables. In particular, for big enough M_{cost} the constraint is tight only when the

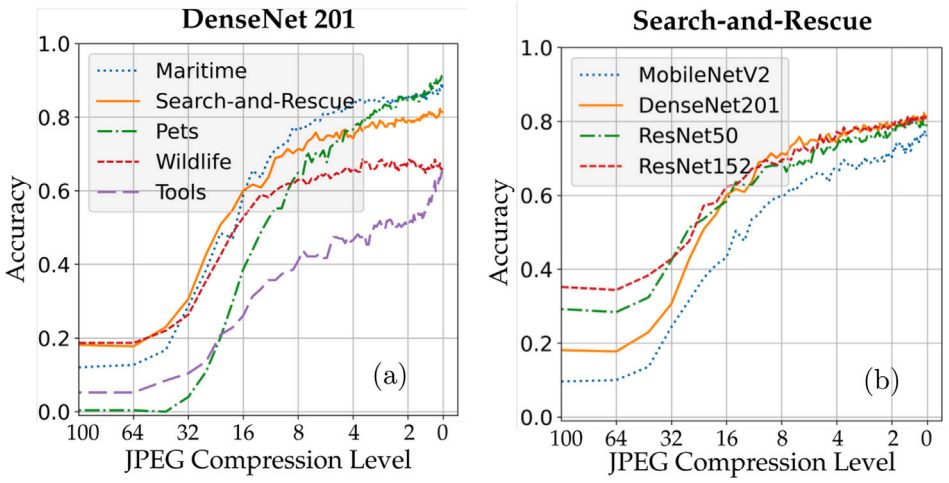


Fig. 5. Accuracy as a function of JPEG compression level.

binary variable takes on a specific value; while it allows the constraint to remain unrestricted when the binary variable takes on the opposite value (i.e., the big constant makes the constraint useless [48]). Next, we enforce that a target is covered by at most one UAV and that each UAV can cover only one target:

$$\sum_{u\in\mathcal{U}}\sum_{j\in\mathcal{L}}\hat{\omega}^u_{ij}\leq 1,\forall i\in\mathcal{T},\quad \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{L}}\hat{\omega}^u_{ij}\leq 1,\forall u\in\mathcal{U} \tag{3}$$

We introduce an extended node set $\mathcal{V}=\mathcal{U}\cup\{\sigma\}$ and we consider all the possible communication paths among nodes, i.e., all the edges $(i,j)\ \forall i,j\in\mathcal{V}$. A binary variable $\hat{\gamma}_{i,j}\in\{0,1\}$, indicates if the nodes i and j are too distant to communicate. The relation $|\hat{p}_i-\hat{p}_j|\geq r_{\text{com}}\Rightarrow\hat{\gamma}_{ij}=0$ is enforced as follows:

$$|\hat{p}_i-\hat{p}_j|\leq r_{\text{com}}+M_{cost}\cdot(1-\hat{\gamma}_{ij}),\ \forall i,j\in\mathcal{V} \tag{4}$$

We define the data frame offloading as a network flow formulation. We introduce a set of variables $\hat{e}^s_{ij}$ defining the amount of data transmitted through the link between UAV i and UAV j. We define $\hat{e}^a_{ij}$ to account for the expected task accuracy at the edge, for each task. We impose that the edge does not generate any outgoing flow, for both data and accuracy flows:

$$\sum_{j\in\mathcal{V}}\hat{e}^s_{\sigma j}\leq 0,\quad \sum_{j\in\mathcal{V}}\hat{e}^a_{\sigma j}\leq 0 \tag{5}$$

Notice that, the *accuracy flow* $\hat{e}^a_{i,j}$ is a variable used solely within the MILP formulation to track the potential accuracy at the edge. In fact, we aim at maximizing the classical accuracy of machine learning tasks performed at the edge with data collected from the targets.

We allow a flow only for between neighboring nodes:

$$\hat{e}^s_{ij}+\hat{e}^a_{ij}\leq\hat{\gamma}_{ij}\cdot M,\ \forall i,j\in\mathcal{U} \tag{6}$$

similarly to Eq. (2), M is a big constant number, used in MILP constraints with binary variables [48]. It makes the inequalities unrestricted when the binary variable takes positive value: i.e., $\hat{e}^s_{ij}+\hat{e}^a_{ij}$ can take any value up to the big enough M. While it restricts the value of the two variables to 0 when $\hat{\gamma}_{ij}$ takes on the opposite value.

The maximum bandwidth allowed between two UAVs is constrained to respect the estimated channel data rate $\rho_{i,j}$:

$$\sum_{j\in\mathcal{V}}\hat{e}^s_{ij}\leq\rho_{i,j},\ \forall i\in\mathcal{U} \tag{7}$$

We specify that a UAV can transmit only towards another UAV, modeling a unicast communication:

$$\sum_{j\in\mathcal{V}}\hat{\gamma}_{ij}\leq 1,\ \forall i\in\mathcal{U} \tag{8}$$

In particular, as the drone communicates towards the edge, this constraint also guarantees that the communication network results into a tree rooted at the edge.

We impose flow conservation as follows:

$$\sum_{k\in\mathcal{V}}\hat{e}^s_{uk}-\sum_{k\in\mathcal{V}}\hat{e}^s_{ku}=\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{L}}b_{i,j}\cdot\hat{\omega}^u_{ij},\ \forall u\in\mathcal{U} \tag{9}$$

which imposes that, for each outgoing edge from u, the flow is increased by expected data size of the target covered by the UAV u. The expected data size is defined by the constant $b_{i,j}$, which is estimated at the beginning of the mission for each possible target i and related compression level j, using A^2-TP algorithm Section 3.3. We also impose that the edge receives all the data produced by the covered targets:

$$\sum_{k\in\mathcal{V}}\hat{e}^s_{k\sigma}=\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{L}}b_{i,j}\cdot\hat{\omega}^k_{ij} \tag{10}$$

To conclude, we constraint the accuracy of the targets at the edge-server, as follows:

$$\sum_{k\in\mathcal{V}}\hat{e}^a_{uk}-\sum_{k\in\mathcal{V}}\hat{e}^a_{ku}=\sum_{t^i_j\in\mathcal{T}'}a_{i,j}\cdot\hat{\omega}^u_{ij},\ \forall u\in\mathcal{U} \tag{11}$$

which models the expected accuracy at the edge for each monitored target. The variable $a_{i,j}$, representing the estimated accuracy, is estimated at the beginning of the mission for each possible target i and related compression level j, using A^2-TP algorithm Section 3.3.

Objective Function: maximize covered targets, DL tasks accuracy, and energy spent by the UAVs:

$$\max\quad\alpha\cdot\sum_{j\in\mathcal{V}}\hat{e}^a_{\sigma j}+\beta\cdot\sum_{i\in\mathcal{T},j\in\mathcal{L},u\in\mathcal{U}}\hat{\omega}^u_{ij}-\eta\cdot\sum_{u\in\mathcal{U}}l_u \tag{12}$$

The term α prioritizes the maximization of the accuracy, while β weights the importance of covering the targets and η minimized the distance traveled by the UAVs.

3.5. Building the UAV network

At the end of the optimization process, the variables of the MILP formulation can be used to construct the network and the flow. For example, $\hat{p}_u$ determine the UAVs position, while $\hat{\omega}^u_{ij}\in\{0,1\}$ determine if the UAVs should cover the target and at which compression level, variables $\hat{e}$ define the communication flow. Notice that, the exact flying paths are managed by the autonomous UAVs which use waypoint flight and collision avoidance protocols, as discussed in Section 3.1.

Theorem 3.1. *The A^2-TPP problem is NP-Hard.*

Proof. We show that A^2-TPP generalizes the *Steiner tree problem with minimum number of Steiner points and bounded edge-length* STPMSPBEL, a known NP-hard problem [49]. Given a set P of n terminal points in a 2-dimensional plane, a positive constant R, and a non-negative integer B, STPMSPBEL asks whether it exists a tree spanning a set of points $P \subseteq Q$ s.t. each edge has a length less than R and the number of Steiner points (i.e., $Q \setminus P$) is less than or equal to B. Any instance of STPMSPBEL can be reduced to an instance of our problem in polynomial time. The set of points P represents our target set $\mathcal{T} \cup \{\sigma\}$, and B defines the number of available UAVs, with communication range equal to R. We consider UAVs with unlimited batteries and one compression level (i.e, $|\mathcal{L}| = 1$). This problem instance finds a solution that maximizes the number of connected targets with the edge server, moving the minimum number of UAVs. If such a solution exists, and covers all the points in P, then it also exists a tree spanning a set of points $P \subseteq Q$, where each edge has length less than R and the number of Steiner points is less then or equal to B. The complexity of the above reduction is polynomial, thus we derive that A^2-TPP problem is at least as hard as the STPMSPBEL problem [49]. □

4. A polynomial time heuristic for A^2-TPP

We propose a greedy heuristic to solve A^2-TPP in polynomial time. We first introduce the algorithm, and then prove its polynomial time complexity.

4.1. Algorithm overview

Greedy-A^2-TPP outputs a *connected coverage formation* — also referred to as *coverage* for brevity — for the UAVs, and a *compression level assignment* for each covered target. Both coverage and compression need to meet the criteria expressed in Eq. (12), that is, optimizing the *number of accomplished tasks*. Our approach is to maximize the number of inspected targets, producing a coverage of minimum congestion, and minimizing task misclassification due to low frame resolution.

Specifically, a coverage $\tau = (V, E, W)$ is a Triangular Steiner Tree [50] in which the set of nodes V represents the positions UAVs must reach to cover the target nodes in M, while staying connected with the base station σ in a multi-hop manner. The set E represents the link between UAVs, thus the routes data streams must follow through the network. The function W maps each $e \in E$ to a weight that represents link's bandwidth. In our implementation we estimate this value empirically. It is assumed that at the base station, communication happens through dedicated transceivers and does not require actual coverage with a drone. Thus, at any time it holds $|V| \leq |\mathcal{U}| + 1$.

4.2. Greedy-A^2-TPP

Algorithm 1 returns a coverage formation Ψ, merging partial coverage formations τ_{par} generated to cover targets using the minimum deployment cost at each iteration. In the initialization phase, we let: $\widehat{\mathcal{T}}$ be the set of covered targets, initially containing only the base station; Ψ be the coverage archived so far; τ_{par} be the partial coverage iteratively grown that is added to Ψ when it cannot be further expanded; c_{par} be the cost of the partial coverage generated so far (**line 1**). The while loop iterates until either all the targets are covered $\mathcal{T} - \widehat{\mathcal{T}} \neq \emptyset$ or the number of UAVs used does not exceed the fleet size (**line 2**). The variables $t_{\text{best}}, \tau_{\text{best}}, c_{\text{best}}$ contain respectively the best target found at each iteration, the coverage including that target and its cost. A for loop over the uncovered targets $\mathcal{T} - \widehat{\mathcal{T}}$ allows to find the best target to add, building new temporary coverage formations τ_{temp} using the targets already covered by τ_{par} (namely the set $C(\tau_{\text{par}})$) and adding to them the candidate target t. The coverage trees are computed solving a Triangular Stainer Tree (TST) problem [50]. Then we evaluate the cost of τ_{temp} (**lines 3–6**). This cost combines the number of drones needed for the coverage, and the loss in accuracy due to the channel contention. We will talk in more detail about how this cost is computed when describing Algorithm 2. Then we check if: (i) τ_{temp} has a lower cost than τ_{best} (ii) and if τ_{temp} can be covered with the remaining UAVs. If both checks go through, then t becomes the best candidate t_{best} and the associated candidate coverage τ_{temp} with its cost c_{temp} are stored into τ_{best} and c_{best} respectively (**lines 7–8**). In case no target was set as a best candidate, (i.e., $t_{\text{best}} = \emptyset$) the while loop breaks. This happens only when the second condition at line 7 is not met for any target, that is no coverage formations can stick to the remaining fleet size constraint. Then, the cost paid to cover only t_{best} that is $c_{\text{best}} - c_{\text{par}}$ is compared to the cost c_{los} of a new line-of-sight (*los*) branch τ_{los} grown using only t_{best} as target. If τ_{los} costs less than the partial grown tree so far τ_{par}, then τ_{par} is merged with the final tree Ψ. Then τ_{los} becomes the new partial connected coverage to grow. Otherwise growing τ_{par} is still convenient, so τ_{best} becomes the new partial deployment including the new target t_{best} and τ_{los} is discarded (**lines 12–19**). When the algorithm terminates (**line 20**) the final coverage Ψ along with all the compression levels assigned to each target are returned.

Algorithm 1: Greedy A^2-TPP

Input: $\mathcal{U}$: set of UAVs, $\mathcal{T}$: set of targets
Output: Ψ a connected coverage formation

1 $\widehat{\mathcal{T}}, \Psi, \tau_{\text{par}}, c_{\text{par}} \leftarrow \{\sigma\}, \{\sigma\}, \{\sigma\}, 0$
2 **while** $\mathcal{T} - \widehat{\mathcal{T}} \neq \emptyset$ **or** $|V_\Psi \cup V_{\text{par}}| < |\mathcal{U}|$ **do**
3 $t_{\text{best}}, \tau_{\text{best}}, c_{\text{best}} \leftarrow \emptyset, \emptyset, \infty$
4 **for** $t \in \mathcal{T} - \widehat{\mathcal{T}}$ **do**
5 $\tau_{\text{temp}} \leftarrow \text{TST}(\{t\} \cup C(\tau_{\text{par}}), r_{\text{com}})$
6 $c_{\text{temp}} \leftarrow \text{Cost}_\alpha(\tau_{\text{temp}}, \tau_{\text{par}}, \Psi, \text{Compression}(\tau_{\text{temp}}))$
7 **if** $c_{\text{temp}} < c_{\text{best}}$ **and** $|V_{\text{temp}}| - 1 \leq |\mathcal{U}| - |V_\Psi \cup V_{\text{par}}|$ **then**
8 $t_{\text{best}}, \tau_{\text{best}}, c_{\text{best}} \leftarrow t, \tau_{\text{temp}}, c_{\text{temp}}$
9 **if** $t_{\text{best}} = \emptyset$ **then**
10 $\Psi \leftarrow \Psi \cup \tau_{\text{par}}$
11 **break**
12 $\tau_{\text{los}} \leftarrow \text{TST}(\{\sigma, t_{\text{best}}\}, r_{\text{com}})$
13 $c_{\text{los}} \leftarrow \text{Cost}_\alpha(\tau_{\text{los}}, \tau_{\text{par}}, \Psi, \text{Compression}(\tau_{\text{los}}))$
14 **if** $c_{\text{los}} < c_{\text{best}} - c_{\text{par}}$ **then**
15 $\Psi \leftarrow \Psi \cup \tau_{\text{par}}$
16 $\tau_{\text{par}}, c_{\text{par}} \leftarrow \tau_{\text{los}}, c_{\text{los}}$
17 **else**
18 $\tau_{\text{par}}, c_{\text{par}} \leftarrow \tau_{\text{best}}, c_{\text{best}}$
19 $\widehat{\mathcal{T}} \leftarrow \widehat{\mathcal{T}} \cup \{t_{\text{best}}\}$
20 $R, \cdot \leftarrow \text{Compression}(\Psi)$
21 **return** Ψ, R

Algorithm 2: Compression

Input: a coverage formation τ_i
Output: R vector with compression levels for all targets in τ_i, L vector with loss in accuracy due to all targets in τ_i

1 sort t by $Q(S(t), *).b \; \forall t \in C(\tau_i)$ in ascending order
2 $\widehat{C}, L, R \leftarrow C(\tau_i), \langle\rangle, \langle\rangle$
3 **for** $t \in \widehat{C}$ **do**
4 $P \leftarrow \text{Shortest-Path}(\tau_i, \sigma, t)$
5 $B \leftarrow \text{Bottleneck}(\tau_i, P)/(|C(\tau_i)| - |\widehat{C}|)$
6 $R(t) \leftarrow \arg\max_{l \in \mathcal{L}} Q(S(t), l).b \; \leq \min\{B;\; Q(S(t), *).b\}$
7 $L(t) \leftarrow Q(S(t), *).a - Q(S(t), R[t]).a$
8 $W_i(e) \leftarrow W_i(e) - Q(S(t), R(t)).b \quad \forall e \in P$
9 $\widehat{C} \leftarrow \widehat{C} - \{t\}$
10 **return** R, L

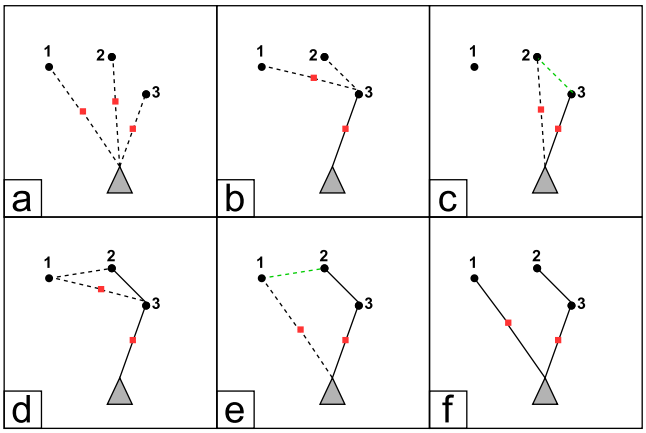


Fig. 6. GREEDY-A^2-TPP algorithm example.

4.3. Assignment of compression levels

Algorithm 2 determines the compression levels for each UAV in F inspecting targets in M. The rationale is to increase the compression of data flowing from a target, based on the bandwidth assigned to it, leaving more bandwidth to targets having more to send. The algorithm iterates over the targets t covered by the candidate input tree, sorted in ascending order based $Q(S(t), *).b$, that is the load produced by the target t according to the task analyzer A^2−TA, belonging to the application scenario $S(t)$ and at the minimum compression level (denoted by $*$) (lines 1–3). At each iteration a bottleneck bandwidth B for the target is computed. This quantity is the bottleneck capacity on the path from the source of flow t, to the destination σ. This value is influenced by the number of targets t shares this path with. The bandwidth allocation function can be thought as slight modification of the Depth First Search (DFS) (**line 5**). To derive the maximum quantity of load that can be transferred from the target t per unit time, we vary the compression level while remaining subject to the flow constraint (**line 6**). We store in the vector R the compression level for each target. We store the loss in accuracy for t subject to compression level $R(t)$, comparing the accuracy due to the best quality $Q(S(t), *).a$ (**line 7**). The weights of the tree are updated considering the used bandwidth (**line 8**). Both the compression levels and the loss for each target are returned.

4.4. Cost of a coverage

The cost of a connected coverage formation is parameterized by α. This exogenous parameter weights the importance given to the accuracy of the tasks. Notice that the importance given to task accuracy opposes to the minimization of the number of UAVs employed. Therefore the cost is a linear combination of the average loss in accuracy, and the percentage of used UAVs to cover the new target in τ_i which was not present in the previous formation τ_{i-1}, the cost is computed as COST_α:

$$\alpha \cdot \frac{\sum_{t \in C(\tau_i)} L(t)}{|C(\tau_i)|} + (1-\alpha) \cdot \frac{|V_i - V_{i-1}| - 1}{|\mathcal{U}| - |V_\Psi \cup V_{i-1}|} \tag{13}$$

4.5. GREEDY-A^2-TPP example execution

Fig. 6 shows an example of execution of GREEDY-A^2-TPP. The gray triangle is the edge server σ. The black dots and red squares represent the target and relay positions, respectively. Fig. 6-a shows three temporary coverage τ_{temp}, each covering a different target. The cost of each of the coverage is compared (algo. 1, **line 7**). Say $\tau_{\text{temp}}\langle\sigma, t_3\rangle$ is the cheapest coverage among them, that is the tree covering σ and t_3. At the subsequent iteration shown in Fig. 6-b, two Triangular Steiner Trees covering σ, t_3 and a new target among the remaining uncovered ones in $\mathcal{T} - \hat{\mathcal{T}}$ (i.e., t_2 and t_1) are proposed. Say the tree $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$ is the cheapest coverage among them. In Fig. 6-c the cost of $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$ is compared with a line of sight coverage $\tau_{\text{los}}\langle\sigma, t_2\rangle$. The cheapest coverage among the two becomes the one to grow from the subsequent iterations (algo. 1, **line 14**). Say the cheapest coverage among them is $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$. In Fig. 6-d we see two grown versions of the tree covering t_1, whereas in Fig. 6-e we see a line of sight coverage of t_1. Say that comparing the cost of $\tau_{\text{temp}}\langle\sigma, t_3, t_2, t_1\rangle$, and $\tau_{\text{los}}\langle\sigma, t_1\rangle$, the cheapest is the line-of-sight version. The tree $\tau_{\text{los}}\langle\sigma, t_1\rangle$ becomes the new tree to grow from the subsequent iterations. $\tau_{\text{temp}}\langle\sigma, t_3, t_2\rangle$ is archived in Ψ. There are no more targets to cover. $\tau_{\text{los}}\langle\sigma, t_1\rangle$ is archived in Ψ the algorithm stops returning Ψ.

4.6. Properties of GREEDY-A^2-TPP

Lemma 4.1. *Computing the compression level assignment has polynomial time complexity of* $O(|\mathcal{U}|^2)$.

Proof (*Proof Sketch*)**.** To measure the cost of a coverage tree τ_i, the set of targets in the tree $C(\tau_i)$ is sorted by their expected transmission load in ascending order. Sorting requires $O(|\mathcal{T}| \log |\mathcal{T}|)$ time complexity. The for loop iterates over the targets in τ_i first computing the bottleneck bandwidth for t, having approximately the cost of a Depth First Search and Shortest Path, that is $O(\log |V_i|)$ for the tree. Iterating over the compression levels to find the highest resolution to fit the bandwidth has constant complexity $|\mathcal{L}|$ i.e., the cardinality of the discrete set of possible compression levels. Iterating over the edges of the path P to update the residual bandwidth has cost $O(\log |V_i|)$. Other assignments have evident constant complexity. By noticing that $|V_i| = O(|\mathcal{U}|)$ the overall time complexity of computing the cost of a coverage tree is $O(|\mathcal{T}| \log |\mathcal{U}|)$. The complexity further simplifies by considering $|\mathcal{T}| = O(|\mathcal{U}|)$, thus resulting in $O(|\mathcal{U}|^2)$. □

Theorem 4.2 (*Time Complexity of* GREEDY-A^2-*TPP*)**.** GREEDY-A^2-*TPP with input* $\mathcal{T}$ *targets sets has polynomial time complexity of* $O(|\mathcal{U}|^6)$.

Proof (*Proof Sketch*)**.** The while loop is executed, in the worst case, until all the targets in $\mathcal{T}$ are included in the final solution Ψ. Within the while loop, a for loop iterates over the set of uncovered targets. For each of them a Triangular Steiner Tree t_{temp} is computed, and the time complexity is bounded by $O(|\mathcal{T}|^4)$ [50]. The cost of each tree is computed with complexity $O(|\mathcal{U}|^2)$ as shown in Lemma 4.1. Once the best candidate target to cover has been chosen, the Triangular Steiner Tree t_{los} of the shortest path towards the target and the relative cost c_{los} are computed. The time complexity to find a stripe can be considered constant in time. The overall time complexity of GREEDY-A^2-TPP is thus given by: $O(|\mathcal{T}|(|\mathcal{T}|^4 + |\mathcal{U}|^2) + |\mathcal{U}|^2)) = O(|\mathcal{U}|^6)$. □

5. Adaptive A^2-TPP

In this section we present *adaptive* A^2-TPP, a seamless integration to A^2-TPP to ensure network's resilience even in the face of unpredictable events. This version of the protocol prioritizes adaptability to accommodate changing mission conditions, including variations in UAVs and targets availability.

Specifically it is designed to handle UAVs failures and the addition of new UAVs to the fleet. Given the inherent susceptibility of UAV networks to node failures, caused by factors such as limited battery or unpredictable adverse environmental conditions, the protocol is designed to swiftly identify affected nodes and restore the network to its original operational integrity.

Moreover, the protocol takes into account the dynamic nature of targets, which may be added, removed, or change positions during a mission. The *adaptive* A^2-TPP ensures that the network remains optimized and ready for evolving missions, guaranteeing resilience in the face of unforeseen circumstances.

Algorithm 3: ADAPTIVE A^2-TPP

Input: $\mathcal{U}_t$: sets with the available UAVs at time t; Ψ_{t-1}, Ψ_t the connected coverage formation at time $t-1$ and t resp.; $\mathcal{T}_t$: set of targets at time t.

Output: Ψ a new connected coverage formation

1 $\Psi_t, R_t \leftarrow$ GREEDY A^2-TPP$(\mathcal{U}_t, \mathcal{T}_t)$

2 $\mathcal{M} \leftarrow$ MINMATCHING(Ψ_t, Ψ_{t-1})

3 **return** $\Psi_t, R_t, \mathcal{M}$

5.1. Algorithm overview

Adaptive A^2-TPP is shown in Algorithm 3 and is triggered at time t when one of the following five events occurs: (i) *UAV failure*, a UAV of the network fails; (ii) *UAV provisioning* a new UAV is added to the fleet (iii) *target removal* a target has no more the need to be monitored; (iv) *target add* a new target was added; (v) *target move*, a target changed its position. We denote by e the reference to the occurred event.

Let $\mathcal{U}_t$ be the sets with the available UAVs at time t. Let Ψ_{t-1}, Ψ_t be the connected coverage formations at time $t-1$ and t respectively, and finally $\mathcal{T}_t$ the sets of targets at time t. Upon trigger event e at time t. The algorithm computes a new connected coverage formation and compression level assignment with the UAVs and targets available at that time (**line 1**). Then the algorithm determines which UAVs, formerly covering Ψ_{t-1}, have to move towards the new formation Ψ_t *minimizing the worst migration time*. Such cost represents the time for the last UAV to connect to the new formation. We implement such migration using a procedure called *MIN-MATCHING*. In particular, we reduce our problem to what in the literature is known as the Bottleneck Matching Problem (BMP) for bipartite graphs [51].

We recall that BMP takes a bipartite graph $G = (V_0, V_1, E)$, and a non-negative edge cost function $c(e) > 0, \ \forall e \in E$, and finds a perfect matching $M \in M_p$ where M_p is the set of subsets of E of cardinality $|V_0| = |V_1|$, of minimal cost, i.e., $\min_{M \in M_p} \max_{e \in M} c(e)$. In our MIN-MATCHING reduction, nodes in V_0 and V_1 are the nodes of the connected formations Ψ_{t-1} and Ψ_t respectively, representing the position of UAVs in the formation. In both the sets, the nodes representing the position of spare UAVs (i.e., UAVs that are not used for granting connectivity) are modeled as duplicated nodes of the base station. The bipartite graph is complete and the cost function of the edges $c((u,w)) > 0, \ \forall e : (u,w) \in E$ represents the time to fly from position u to w.

This reduction allows to minimize the effort needed to migrate from one connected formation to another. The output of the matching algorithm is a map $\mathcal{M}$ from every UAV in the formation Ψ_{t-1} to the new Ψ_t with the least effort in terms of travel time for the fleet (**line 2**). The UAVs on Ψ_{t-1} efficiently migrate towards Ψ_t according to the map $\mathcal{M}$ and compression level R_t.

6. Performance evaluation

We extensively evaluate A^2-UAV through simulation (Section 6.2) as well as real-world experiments (Section 6.3).

6.1. Evaluation setup

Application. We consider a monitoring application where UAVs need to perform image classification or object detection tasks on *target* locations by sampling images at a given frame rate (e.g., 24 frames per second (fps)). We adopt (i) *ResNet-50*, a CNN with 50 layers [1]; (ii) *ResNet-152*, an extended version with 152 layers [1]; (iii) *DenseNet* [2], which consists of a Dense Convolutional Network (i.e., each layer is connected to all the other layers in a feed-forward fashion); (iv) *MobileNet-V2* [17], a new neural architecture for mobile devices; (v) YoloV4, the state-of-the-art model for object detection. All the models were trained on the ImageNet database [15].

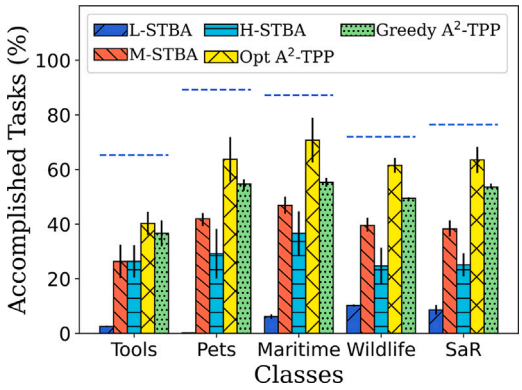


Fig. 7. Accomplished tasks (%), $\Delta = 0.1$sec

Scenarios. To emulate common scenarios for UAVs, we use the five scenarios described in Section 3.3, i.e., *Maritime, Search-and-Rescue, Wildlife, Tools, Pets*. We also design an *Urban* reconnaissance scenario including various objects, such as *wreck, fireboat, ambulance, police van, revolver, crate, packet, backpack, mountain bike, motor scooter*. To ensure repeatability of our experiments, we let the UAVs sample images from a labeled subset of ImageNet. Where not otherwise stated, each target location generates 500 tasks (images) uniformly sampled among these classes.

Metrics. We measure the *Percentage of Accomplished Tasks*, defined as the ratio between the number of successfully completed tasks (according to Definition 3.2) and the number of the generated tasks. The accomplishment of a task is influenced by its deadline Δ. In order to study the performance of A^2-TPP at varying application scenarios, we let Δ vary: low values represent delay critical applications (e.g., intrusion detection), whereas high values, delay tolerant ones (e.g., agriculture). We also measure *Computational Time*, that is the time required by the algorithms to output a connected coverage formation and compression levels for the targets.

Comparison. We evaluate A^2-UAV through real-field experiments and simulation, considering both the optimal solution OPT-A^2-TPP and the greedy algorithm GREEDY-A^2-TPP, against STBA [42]. STBA is a state-of-the-art networking-based approach that is the closest to our work. STBA covers a set of targets while providing network connectivity to the edge server. To find a connected tree, STBA uses a node-weighted Steiner tree algorithm, which computes a set of Fermat points to place relays, and then computes a tree among the targets and the edge-server, minimizing the needed UAVs. To allow for a fair comparison, we enhance STBA with data compression in three variants: 1) H-STBA, which does not compress data, but uses the **H**ighest available quality for collected data ($l = 1$); 2) M-STBA which uses the **M**edium compression ($l = 50$); and 3) L-STBA which uses an extreme compression ($l = 100$) resulting in the **L**owest data quality.

6.2. Simulation results

We used the NS-3 network simulator [52], setting most of parameters in line with the devices used in our real-field experiments (e.g., WiFi interface 802.11n at 2.4 GHz), and testbed measured values (UAVs transmission range is $16m$, sensing radius $1m$, and maximum speed 5 m/s). The simulated area is a square of $500 \times 500m$, with an edge-server positioned in the center of the bottom border. The number of targets varies from 4 to 50, and the number of UAVs from 4 to 50.

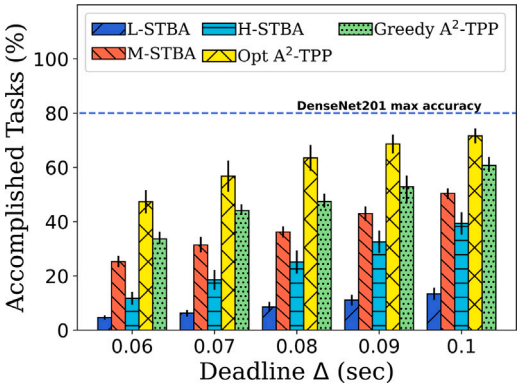


Fig. 8. Accomplished tasks (%) at increasing of Δ.

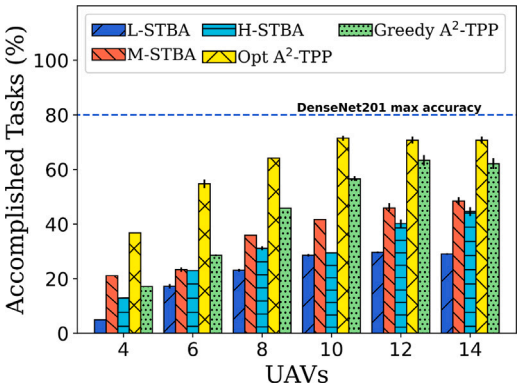


Fig. 9. Accomplished tasks (%) with 6 Targets, $\Delta = 0.1$sec.

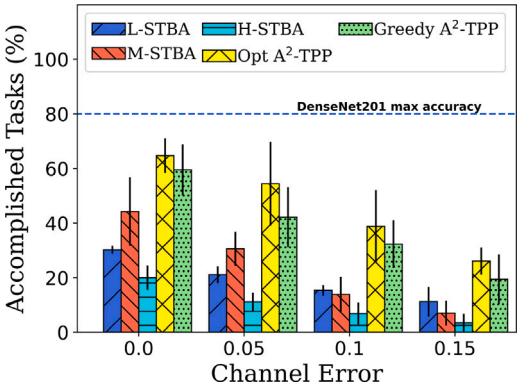


Fig. 10. Accomplished tasks (%), 4 targets, $\Delta = 0.1$sec.

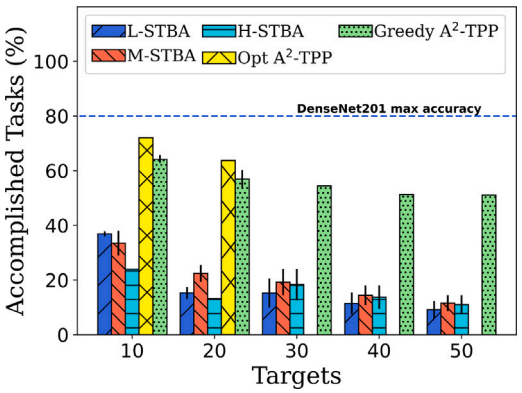


Fig. 11. Accomplished tasks (%), increasing targets.

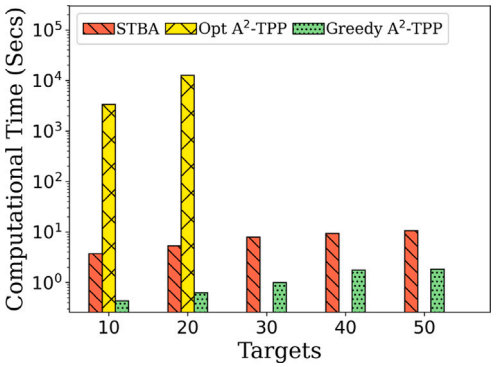


Fig. 12. Computational time (s).

6.2.1. Multiple scenarios

Figs. 7 illustrate the efficacy of our solution for different scenarios, reporting also the theoretical upper bound (blue dotted line). In the most challenging scenario, i.e., *Tools*, with a strict task deadline (Δ = 0.1sec) and DenseNet-201 DL model, Opt-A^2-TPPcompletes 41% of tasks, with an improvement of 52% respect the best STBA variant, i.e., H-STBA, which completes less than 27% of tasks. The theoretical upper-bound for DenseNet-201 in the same scenario is 60%, meaning that under ideal network conditions of zero latency and no compression, the DL model would correctly classify only 60% of the tasks (Fig. 5.a show the complexity of predicting tools images, even for un-compressed images). In the case of *Pets and Maritime*, Opt-A^2-TPP reaches the highest percentage of accomplished tasks — 65% and 70% respectively — where the upper-bounds are 90% and 88%. The improvement with respect to the best STBA variant, i.e., M-STBA, is 55% and 50%. *Pets* require a compression level lower than $l = 50$ (see Fig. 5.a) to achieve satisfactory performance, forcing both Opt-A^2-TPP and Greedy-A^2-TPP to select a medium compression level, more similarly to M-STBA. In *Wildlife* and *Search-and-Rescue* (SaR), the gap between both the A^2-TPP versions and STBA variants increases significantly. Opt-A^2-TPP and Greedy-A^2-TPP complete respectively 60%, 63% and 49% and 54% of tasks, against 39% and 37% of M-STBA. The motivation behind this sharp improvement is the use of the A^2-TA, which understands that even high compressed images can achieve satisfactory performance. Therefore, both our solutions can achieve high accuracy with low network usage, executing the tasks within their deadline $\Delta = 0.1$ seconds. Greedy-A^2-TPP completes 20% and 31% more tasks than M-STBA.

6.2.2. Urban scenario

Fig. 8 shows the performance in the *Urban* scenario as a function of task deadline $\Delta \in \{0.06, 0.07, 0.08, 0.09, 0.1\}$, when DenseNet-201 is employed.

Opt-A^2-TPP accomplishes tasks up to 72% in the case of $\Delta = 0.1$sec, while the best variant M-STBA achieves only 48% of tasks at the same Δ. Opt-A^2-TPPaccomplishes 58% more tasks than M-STBA with the tightest deadline, as it adapts the compression of images to meet the latency constraint. The plot also confirms the performance of Opt-A^2-TPP that outperforms the network-based approaches (i.e., M-STBA) up to 45 – 50%. Greedy-A^2-TPPfollows the Opt-A^2-TPPtrend always remaining widely above the performance of STBA solutions. Fig. 9 shows the percentage of accomplished tasks as function of the number of UAVs, with 6 targets randomly distributed in the area. We employ DenseNet-201, which achieves a maximum accuracy of 80%, and set $\Delta = 0.1$s. Both Opt-A^2-TPP and Greedy-A^2-TPP outperform the STBA variants, as they cover all the targets with only 8 UAVs. Conversely, the STBA variants require at least 10 UAVs to cover all the targets, and achieve lower performance. Opt-A^2-TPP covers 15 – 20% more targets than STBA algorithms, in all the scenarios, completing 69% of tasks

(using 10 UAVs), while the best variant M-STBA accomplishes only 42% of tasks with the same number of UAVs. We can notice how Greedy-A^2-TPP performs better as quickly as the number of UAVs grows, reaching similar performance of Opt-A^2-TPP.

6.2.3. Robustness to channel errors

In Fig. 10 we plot the percentage of accomplished tasks by varying the probability of channel error $\psi \in \{0, 0.05, 0.1, 0.15\}$, in a setting with 20 UAVs and 4 targets. Both Opt-A^2-TPP and Greedy-A^2-TPP are the most robust algorithms, increasing their improvement with respect to STBA variants. Opt-A^2-TPPcompletes up to 170% more tasks than the other approaches. On the other hand, M-STBA and H-STBA experience severe delays and drastic performance reduction due to frequent TCP re-transmissions, which introduce additional data in the network, further overloading communication links.

6.2.4. Scalability

Fig. 11 investigates the percentage of accomplished tasks in a scenario with 50 UAVs, varying the number of targets from 10 to 50. We do not include the Opt-A^2-TPP when the targets are more than 20, due to prohibitive computational time. This result underlies the huge benefit introduced by the polynomial time solution Greedy-A^2-TPP, which scales gracefully when the problem instance grows in complexity. The figure shows that Greedy-A^2-TPP has near optimal performance with 10 targets, accomplishing 63% of the tasks, while L-STBA accomplishes only 38% of them. All the algorithms have a slightly decreasing trend as the number of targets increases, as the UAVs have to offload more tasks with possible network congestion and missed deadlines. The STBA variants quickly drop their performance due to congestion and long delays, while Greedy-A^2-TPP is able to keep satisfactory performance around 50%, trading off compression and accuracy to cover all targets and offload their data. With 50 targets Greedy-A^2-TPP accomplished 5 times the tasks of the best STBA variant. Finally, in Fig. 12 we investigate the computational time. We restrict the time to a maximum of 5 h (18000 s), and we consider no solutions after that time. We consider a general STBA instance without compression levels, as they do not affect the execution time. While Opt-A^2-TPP has very huge computational times even with 10 targets, Greedy-A^2-TPPis 15x faster than the STBA solutions.

6.3. Experimental testbed results

We evaluate the performance through an experimental testbed. The testbed is composed of 4 UAVs and an edge server with dedicated GPU. In the experiments, we consider up to 4 targets placed at a maximum distance of 15 meters from the edge, as shown in Fig. 13. The red triangles represent target locations, while the black and white circle indicates the edge-server location.

We run 10 experiments for each scenario and we average the results. Each UAV includes a DJI Mavic Air 2 drone, mounting a Raspberry PI 4 model B and a powerbank, as shown in Fig. 14. The on-board Raspberry PI, powered by the power bank is used to generate and pre-process tasks, and to offload them to the edge according to the optimization plan. For repeatability and to emulate different scenarios, we sample images from the ImageNet dataset [53].

The edge server is a Jetson Nano board, used to run the DL models and execute tasks. It mounts a Raspberry PI for computation and communication. TCP links are established for reliable connectivity. Considering the limited capabilities of the edge server, we execute only ResNet-50 and MobileNet-V2 on the Jetson Nano, which have approximately 0.03 s of inference time [54]. For DensNet-201, ResNet-152, and YoloV4, we used a laptop with an NVIDIA RTX-2060 Graphics Processing Unit (GPU). Table 2 reports the experimental settings in the *Urban* scenario.

The first set of experiments evaluates the impact of increasing the number of targets (from 1 to 4), with MobileNet-V2. Fig. 15 shows

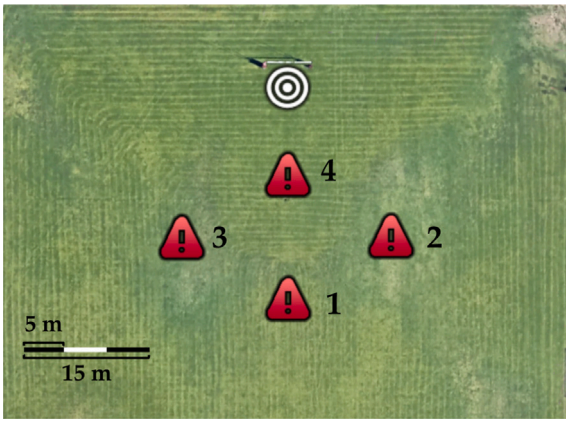


Fig. 13. Scenario.

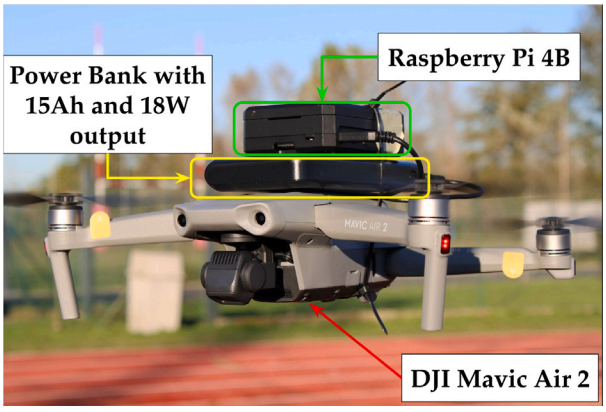


Fig. 14. UAV implementation.

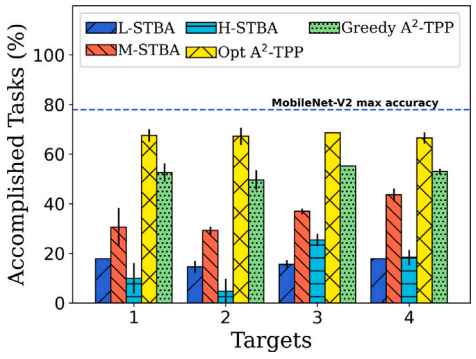


Fig. 15. Accomplished Tasks (%) at increasing targets, MobileNet-V2, $\Delta = 0.4$s.

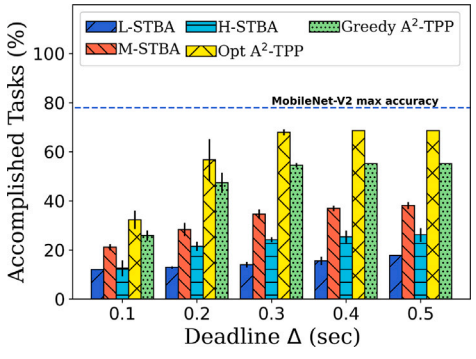


Fig. 16. Accomplished Tasks (%) at increasing of Δ, using 3 targets and MobileNet-V2.

Table 2
Experimental setting.

Field	Value
Time	9:00–18:30 a.m.
Temperature	+4–15 °C
Wind Speed	0.0 to 4.3 m/s
Field Size	65 × 35 (m)
Nr. Of UAVs	4
UAVs Autonomy	18.5 km
Nr. Of Targets	Variable (from 1 to 4)
Humidity	60%–77%
AMSL	2 m

Table 3
Percentage of completed tasks, $\Delta = 0.3$ s.

DL Model	Opt-A^2-TPP	Greedy-A^2-TPP	L-STBA	M-STBA	H-STBA
ResNet50	**66.64**	62.88	25.42	36.14	21.86
ResNet152	**67.89**	65.27	29.93	37.96	24.70
DenseNet201	**70.17**	68.33	32.92	41.87	26.99
MobileNet-v2	**69.29**	57.36	18.37	38.94	21.91
YoloV4	**59.32**	51.3	15.22	31.52	17.18

the percentage of accomplished tasks at the edge-server with a task deadline of $\Delta = 0.4$sec. The plot shows that the Opt-A^2-TPP finds the best trade-off between accuracy and data compression. It completes more than 67% of the tasks, independently of the number of targets. This is close to the maximum performance achievable with the DL model (i.e., 78%), represented by the blue horizontal line. Greedy-A^2-TPPinstead reaches up to 57% accomplished task, with a 20% average improvement over the best STBA version (M-STBA). Conversely, the best STBA variant (i.e., M-STBA) does not complete more than 46% of tasks, independently of the number of targets. In particular, with 2 targets, all STBA variants perform very poorly, completing less than 30% of tasks. The superiority of A^2-UAV in both the approaches (Opt and Greedy) is confirmed by results on the percentage of accomplished tasks by varying the deadline $\Delta \in [0.1, 0.5]$ (see Fig. 16). Opt-A^2-TPP reaches an improvement over the percentage of executed tasks with respect to M-STBA up to 76% when $\Delta = 0.3$sec. The Greedy-A^2-TPP approach instead improves M-STBA results (when Δ=0.3 s) around 50% upholding our intuition. We investigated the performance of Greedy-A^2-TPP and Opt-A^2-TPP also when other DL models are applied. Table 3 summarizes the results in the case of $\Delta = 0.3$sec and 4 targets, for ResNet50, MobileNet-V2 (executed on the Jetson Nano) and ResNet152, DenseNet201 and YoloV4 (executed on a laptop with a dedicated GPU). The results show that both our solutions outperform all STBA variants independently of the applied model. In particular, with DenseNet201 Opt-A^2-TPP has the best performance.

6.3.1. Adaptive A^2-TPP

To test Adaptive A^2-TPP we chose to use the same scenario shown in Fig. 13 described in the previous section. To evaluate the impact of a *UAV failure* event, we measure the impact over the *tasks executed per second*. The failure event affects one of the four UAVs, thus, the entire network must reorganize in order to communicate again with the base station. For a fair comparison, we applied Algorithm 3 also to STBA, in order to compare the results even though it does not encompass a recovery mechanism.

In Fig. 17 shows the variation of executed tasks over time for all the considered algorithms. When a *UAV failure* event happens (vertical red bar in the figure), communication is dropped and so tasks executions. As the figure shows, our algorithm enables faster recovery when compared to the other approaches. In particular adaptive A^2-TPP rebuilds the configuration 5 s earlier with respect to the second best L-STBA. The average time needed to compute the new topology for our solution is 0.00224 ± 0.00043 s and is 0.00653 ± 0.00012 s for all STBA variants. We recall that a reduced computation time enables swift service restoration upon heterogeneous events during mission.

7. Conclusions

Our paper proposes a novel framework called A^2-UAV to optimize the offloading and execution of computational intense CV tasks in multi-hop UAVs networks. A^2-UAV is an *application-aware* optimization framework for reliable and effective offloading. For the first time, we considered the *accuracy* and *delay* requirements of the specific CV task, to jointly optimize task assignment and offloading. Our framework also incorporates a re-optimization scheme to handle any node failures or changes in the target points. Through extensive simulation and real testbed experiments, we demonstrated that A^2-UAV is able to deal with different network conditions and node failures, maximizing the application performance at the edge. A^2-UAV outperforms existing approaches, getting and average improvement w.r.t. the state-of-the-art algorithm of 38%. We share datasets and code with the research community to allow reproducibility.

CRediT authorship contribution statement

Andrea Coletta: Investigation. **Flavio Giorgi:** Investigation. **Gaia Maselli:** Supervision. **Matteo Prata:** Investigation. **Domenicomichele Silvestri:** Investigation. **Jonathan Ashdown:** Supervision. **Francesco Restuccia:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment of support and disclaimer

This work was partially funded by the G5828 "SeaSec: DroNets for Maritime Border and Port Security" project under the NATO's Science for Peace Programme, by the Ateneo 2021 ''DRONES AS A SERVICE for FIRST EMERGENCY RESPONSE'' Sapienza project, and by the National Science Foundation (NSF) grant CNS-2134973 and CNS-2120447, as well as by the National Science Foundation, United States under grants CNS-2134973 and ECCS-2229472, by the Air Force Office of Scientific Research under contract number FA9550-23-1-0261, by the Office of Naval Research under award number N00014-23-1-2221.

Data availability

Data will be made available on request.

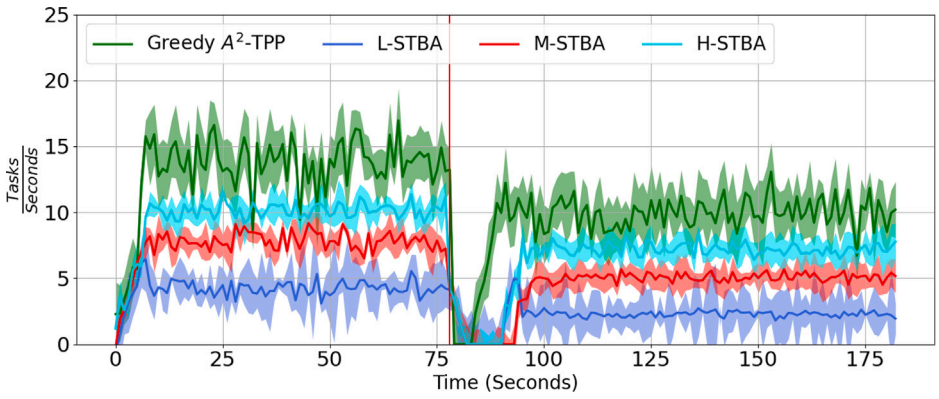


Fig. 17. Task execution ratio during mission. The vertical red line represents the *UAV failure* event.

References

[1] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[2] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.

[3] M.-A. Messous, S.-M. Senouci, H. Sedjelmaci, S. Cherkaoui, A game theory based efficient computation offloading in an UAV network, IEEE Trans. Veh. Technol. 68 (5) (2019) 4964–4974.

[4] J. Scherer, B. Rinner, Multi-robot persistent surveillance with connectivity constraints, IEEE Access 8 (2020) 15093–15109.

[5] N. Bartolini, A. Coletta, M. Prata, C. Serino, On connected deployment of delay-critical FANETs, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2021, pp. 9720–9727.

[6] M. Samir, C. Assi, S. Sharafeddine, D. Ebrahimi, A. Ghrayeb, Age of information aware trajectory planning of UAVs in intelligent transportation systems: A deep learning approach, IEEE Trans. Veh. Technol. 69 (11) (2020) 12382–12395.

[7] Y. Chen, N. Zhao, Z. Ding, M.-S. Alouini, Multiple UAVs as relays: Multi-hop single link versus multiple dual-hop links, IEEE Trans. Wireless Commun. 17 (9) (2018) 6348–6359.

[8] Q. Wu, Y. Zeng, R. Zhang, Joint trajectory and communication design for multi-UAV enabled wireless networks, IEEE Trans. Wireless Commun. 17 (3) (2018) 2109–2121.

[9] S. Hosseinalipour, A. Rahmati, H. Dai, Interference avoidance position planning in dual-hop and multi-hop UAV relay networks, IEEE Trans. Wireless Commun. 19 (11) (2020) 7033–7048.

[10] K. Miranda, A. Molinaro, T. Razafindralambo, A survey on rapidly deployable solutions for post-disaster networks, IEEE Commun. Mag. 54 (4) (2016) 117–123.

[11] S. Chuprov, L. Reznik, A. Obeid, S. Shetty, How degrading network conditions influence machine learning end systems performance? in: IEEE INFOCOM 2022-IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, IEEE, 2022, pp. 1–6.

[12] B. Yang, X. Cao, C. Yuen, L. Qian, Offloading optimization in edge computing for deep learning enabled target tracking by internet-of-UAVs, IEEE Internet Things J. 8 (12) (2020) 9878–9893.

[13] D. Callegaro, M. Levorato, F. Restuccia, SeReMAS: Self-resilient mobile AutonomousSystems through predictive edge computing, 2021, arXiv preprint arXiv:2105.15105.

[14] W. Chen, Z. Su, Q. Xu, T.H. Luan, R. Li, VFC-based cooperative UAV computation task offloading for post-disaster rescue, in: IEEE International Conference on Computer Communications, INFOCOM, IEEE, 2020, pp. 228–236.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.

[16] A.C. Bovik, Handbook of Image and Video Processing, Academic Press, 2010.

[17] M. Sandler, A.G. Howard, M. Zhu, A. Zhmoginov, L. Chen, Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation, 2018, CoRR abs/1801.04381.

[18] A. Bochkovskiy, C. Wang, H.M. Liao, YOLOv4: Optimal speed and accuracy of object detection, 2020, CoRR abs/2004.10934.

[19] A. Coletta, F. Giorgi, G. Maselli, M. Prata, D. Silvestri, J. Ashdown, F. Restuccia, A^2-UAV: Application-aware content and network optimization of edge-assisted UAV systems, in: IEEE INFOCOM 2023-IEEE Conference on Computer Communications, IEEE, 2023.

[20] P.A. Apostolopoulos, G. Fragkos, E.E. Tsiropoulou, S. Papavassiliou, Data offloading in UAV-assisted multi-access edge computing systems under resource uncertainty, IEEE Trans. Mob. Comput. (2021) early access.

[21] E. Arribas, V. Mancuso, V. Cholvi, Coverage optimization with a dynamic network of drone relays, IEEE Trans. Mob. Comput. 19 (10) (2019) 2278–2298.

[22] L. Gupta, R. Jain, G. Vaszkun, Survey of important issues in UAV communication networks, IEEE Commun. Surv. Tutor. 18 (2) (2015) 1123–1152.

[23] A. Baltaci, E. Dinc, M. Ozger, A. Alabbasi, C. Cavdar, D. Schupke, A survey of wireless networks for future aerial communications, IEEE Commun. Surv. Tutor. (2021).

[24] M.Y. Arafat, S. Moh, Routing protocols for unmanned aerial vehicle networks: A survey, IEEE Access 7 (2019) 99694–99720.

[25] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, M. Debbah, A tutorial on UAVs for wireless networks: Applications, challenges, and open problems, IEEE Commun. Surv. Tutor. 21 (3) (2019) 2334–2360.

[26] N. Mansoor, M.I. Hossain, A. Rozario, M. Zareei, A.R. Arreola, A fresh look at routing protocols in unmanned aerial vehicular networks: A survey, IEEE Access (2023).

[27] D.S. Lakew, U. Sa'ad, N.-N. Dao, W. Na, S. Cho, Routing in flying ad hoc networks: A comprehensive survey, IEEE Commun. Surv. Tutor. 22 (2) (2020) 1071–1120.

[28] A. Bujari, C.E. Palazzi, D. Ronzani, A comparison of stateless position-based packet routing algorithms for FANETs, IEEE Trans. Mob. Comput. 17 (11) (2018) 2468–2482.

[29] N. Bartolini, A. Coletta, A. Gennaro, G. Maselli, M. Prata, MAD for FANETs: Movement assisted delivery for flying ad-hoc networks, in: IEEE International Conference on Distributed Computing Systems, ICDCS, 2021.

[30] X. Liu, A. Liu, T. Qiu, B. Dai, T. Wang, L. Yang, Restoring connectivity of damaged sensor networks for long-term survival in hostile environments, IEEE Internet Things J. 7 (2) (2019) 1205–1215.

[31] S.-Y. Park, C.S. Shin, D. Jeong, H. Lee, DroneNetX: Network reconstruction through connectivity probing and relay deployment by multiple UAVs in ad hoc networks, IEEE Trans. Veh. Technol. 67 (11) (2018) 11192–11207.

[32] A. Zear, V. Ranga, Uavs assisted network partition detection and connectivity restoration in wireless sensor and actor networks, Ad Hoc Netw. 130 (2022) 102823.

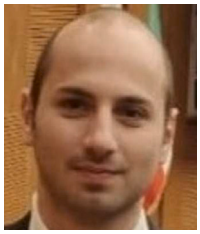
[33] W. Tian, Z. Jiao, M. Liu, M. Zhang, D. Li, Cooperative communication based connectivity recovery for uav networks, in: Proceedings of the ACM Turing Celebration Conference-China, 2019, pp. 1–5.

[34] S. Yin, M.S. Obaidat, X. Liu, H. Zhou, A. Liu, Load-balanced topology rebuilding for disconnected wireless sensor networks with delay constraint, IEEE Trans. Sustain. Comput. 7 (4) (2022) 899–909.

[35] L. Bertizzolo, S. D'oro, L. Ferranti, L. Bonati, E. Demirors, Z. Guan, T. Melodia, S. Pudlewski, SwarmControl: An automated distributed control framework for self-optimizing drone networks, in: IEEE International Conference on Computer Communications, INFOCOM, IEEE, 2020, pp. 1768–1777.

[36] M.T. Rashid, D.Y. Zhang, D. Wang, Socialdrone: An integrated social media and drone sensing system for reliable disaster response, in: IEEE International Conference on Computer Communications, INFOCOM, IEEE, 2020, pp. 218–227.

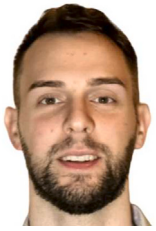
[37] N. Bartolini, A. Coletta, G. Maselli, et al., A multi-trip task assignment for early target inspection in squads of aerial drones, IEEE Trans. Mob. Comput. 20 (11) (2021) 3099–3116.
[38] X. Wang, L. Duan, Dynamic pricing and capacity allocation of UAV-provided mobile services, in: IEEE International Conference on Computer Communications, INFOCOM, IEEE, 2019, pp. 1855–1863.
[39] T. Kimura, M. Ogura, Distributed collaborative 3D-deployment of UAV base stations for on-demand coverage, in: IEEE International Conference on Computer Communications, INFOCOM, IEEE, 2020, pp. 1748–1757.
[40] E. Natalizio, N.R. Zema, E. Yanmaz, L.D. Pugliese, F. Guerriero, Take the field from your smartphone: Leveraging UAVs for event filming, IEEE Trans. Mob. Comput. 19 (8) (2019) 1971–1983.
[41] D. Tateo, J. Banfi, A. Riva, F. Amigoni, A. Bonarini, Multiagent connected path planning: PSPACE-Completeness and how to deal with it, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[42] T.N. Nguyen, B.-H. Liu, S.-Y. Wang, On new approaches of maximum weighted target coverage and sensor connectivity: Hardness and approximation, IEEE Trans. Netw. Sci. Eng. 7 (3) (2019) 1736–1751.
[43] DJI, AGRAS T16, 2019, [Online]. Available: https://www.dji.com/it/t16.
[44] T. Baca, D. Hert, G. Loianno, M. Saska, V. Kumar, Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2018.
[45] K. Goss, R. Musmeci, S. Silvestri, Realistic models for characterizing the performance of unmanned aerial vehicles, in: 26th International Conference on Computer Communication and Networks, ICCCN, Vancouver, BC, Canada, July 31–Aug. 3, 2017, pp. 1–9.
[46] Y. Sun, D. Xu, D.W.K. Ng, L. Dai, R. Schober, Optimal 3D-trajectory design and resource allocation for solar-powered UAV communication systems, IEEE Trans. Commun. 67 (6) (2019) 4281–4298.
[47] F. Shan, J. Huang, R. Xiong, F. Dong, J. Luo, S. Wang, Energy-efficient general poi-visiting by uav with a practical flight energy model, IEEE Trans. Mob. Comput. (2022).
[48] I. Griva, S.G. Nash, A. Sofer, Linear and Nonlinear Optimization 2nd Edition, SIAM, 2008.
[49] G.-H. Lin, G. Xue, Steiner tree problem with minimum number of Steiner points and bounded edge-length, Inform. Process. Lett. 69 (2) (1999) 53–57.
[50] F. Senel, M. Younis, Relay node placement in structurally damaged wireless sensor networks via triangular steiner tree approximation, Comput. Commun. 34 (16) (2011) 1932–1941.
[51] R. Burkard, M. Dell'Amico, S. Martello, Assignment Problems: Revised Reprint, SIAM, 2012.
[52] nsnam, Network Simulator-3 (NS-3), 2021, [Online]. Available: http://www.nsnam.org/.
[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[54] Nvidia, Jetson Nano: Deep learning inference benchmarks, 2021, [Online]. Available: https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks.



Andrea Coletta received his Ph.D. in Computer Science from Sapienza University of Rome in Italy. His research interests include wireless and mobile networks, with particular focus of communication and coordination protocols for aerial drones. His research activities includes the coordination and cooperation algorithms in wireless mobile networks, including Task Assignment, Path-Planning, and Routing Protocols for Aerial Drones. He serves as member of the TPC of several international conferences and workshops, including ACM ICAIF 2022, and the ACM MobiSys student workshop 2021. He has served as reviewer for several journals, such as IEEE IoT Journal, Elsevier Computer Communications, Elsevier Pervasive and Mobile Computing.

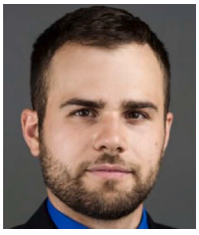
Flavio Giorgi received the M.S. degree in computer science from Sapienza University of Rome in 2021. He is currently pursuing a Ph.D. degree in computer science at Sapienza University of Rome. His research interests include networking, UAV, IoT, wireless sensor networks, optimization.

Gaia Maselli is an associate professor at the Department of Computer Science at Sapienza University of Rome, Italy. She holds a Ph.D. in computer science from the University of Pisa, Italy. Prof. Maselli's current research interests concern design and implementation aspects of mobile networks and wireless communications systems, with particular focus on drones, back-scattering networks and Internet of Things. Other interests include design and performance evaluation of networking protocols for RFID systems. She serves as member of the TPC of several international conferences, is associate editor of Elsevier Computer Communications journal, and serves as reviewer for several journals, such as IEEE TMC, IEEE TON. She participated to many European Community research projects. She is member of IEEE.



Matteo Prata received his Ph.D. in Computer Science from Sapienza University of Rome, Italy. His primary research focuses on computer network performance. Additionally, his interests lie in multi-agent systems, particularly in the context of UAVs networks and financial networks. He actively contributes to the academic community as a reviewer for various flagship conferences and journals in the area of networking and mobile computing.

Domenicomichele Silvestri received the B.S. degree in computer science from the University of Bari "Aldo Moro" in 2016 and the M.S. degree in computer science from Sapienza University of Rome in 2019. He also received the Ph.D. degree in computer science from Sapienza University of Rome in 2023. His research interests include networking, UAV, IoT, wireless sensor networks, optimization. He received the Best Paper Award Runner-Up at IEEE WONS 2023.



Jonathan Ashdown received the B.S., M.S., and Ph.D. degrees from Rensselaer Polytechnic Institute, Troy, NY, USA, in 2006, 2008, and 2012, respectively, all in electrical engineering. He received the Best Unclassified Paper Award at the IEEE Military Communications Conference in 2012. From 2012 to 2015, he worked as an electronics engineer with the Department of Defense (DoD), Naval Information Warfare Center (NIWC) Atlantic, Charleston, SC, USA where he was involved in several basic and applied research projects for the U.S. Navy, mainly in the area of software defined radio. In 2015, he transferred within DoD to the Information Directorate of the Air Force Research Laboratory (AFRL), Rome, NY, USA, where he serves as a senior electronics engineer and is involved in the research and development of advanced emerging communications and networking technologies for the U.S. Air Force.

Francesco Restuccia is an Assistant Professor in the Department of Electrical and Computer Engineering at Northeastern University. He received his Ph.D. in Computer Science from Missouri University of Science and Technology in 2016, and his B.S. and M.S. in Computer Engineering with highest honors from the University of Pisa, Italy in 2009 and 2011, respectively. His research interests lie in the design and experimental evaluation of next-generation edge-assisted data-driven wireless systems. Prof. Restuccia's research is funded by several grants from the US National Science Foundation and the Department of Defense. He received the Office of Naval Research Young Investigator Award, the Air Force Office of Scientific Research Young Investigator Award and the Mario Gerla Award in Computer Science, as well as best paper awards at IEEE INFOCOM and IEEE WOWMOM. Prof. Restuccia has published over 60 papers in top-tier venues in computer networking, as well as co-authoring 16+ U.S. patents and three book chapters. He regularly serves as a TPC member and reviewer for several top-tier ACM and IEEE conferences and journals.