Real-Time HAP-Assisted Vehicular Edge Computing for Rural Areas

Alessandro Traspadini, *Student Member, IEEE*, Marco Giordani, *Member, IEEE*, Giovanni Giambene, *Senior Member, IEEE*, and Michele Zorzi, *Fellow, IEEE*

***Abstract*—Non-Terrestrial Networks (NTNs) are expected to be a key component of 6th generation (6G) networks to support broadband seamless Internet connectivity and expand the coverage even in rural and remote areas. In this context, High Altitude Platforms (HAPs) can act as edge servers to process computational tasks offloaded by energy-constrained terrestrial devices such as Internet of Things (IoT) sensors and ground vehicles (GVs). In this letter, we analyze the opportunity to support Vehicular Edge Computing (VEC) via HAP in a rural scenario where GVs can decide whether to process data onboard or offload them to a HAP. We characterize the system as a set of queues in which computational tasks arrive according to a Poisson arrival process. Then, we assess the optimal VEC offloading factor to maximize the probability of real-time service, given latency and computational capacity constraints.**

***Index Terms*—6G, non-terrestrial networks (NTNs), HAP, vehicular edge computing (VEC), optimization.**

I. Introduction

ONE OF the grand objectives of 6th generation (6G) wireless networks [1] is to enhance broadband Internet coverage in remote areas [2]. The main issues towards this goal are the high costs of terrestrial deployments (especially for fiber-optic backhaul roll-out) as well as the lack of power sources and transport connectivity in rural regions. To overcome these problems, 6G will promote Non-Terrestrial Networks (NTNs) with Unmanned Aerial Vehicles (UAVs), High Altitude Platforms (HAPs), and satellites to provide fast connectivity thanks to their flexibility and inherent support for large coverage [3]. For example, in 2020 a network of HAPs was deployed by Loon to bring Internet connectivity to unserved regions in Kenya. Similarly, in 2021, Airbus has proved the suitability of the solar-powered Zephyr HAP to provide direct-to-device connectivity in the rural areas of Arizona, U.S.

Besides providing connectivity, NTNs can also host edge servers for processing, caching, and/or storing data generated from power-constrained Internet of Things (IoT) devices, thereby supporting the transition of remote areas towards smart cities [4]. Similarly, air/space-borne platforms can support Vehicular Edge Computing (VEC), where ground vehicles (GVs) offload computationally-intensive autonomous driving tasks like object detection and recognition, tracking/trajectory prediction, and/or semantic segmentation [5]. Processing these tasks onboard certain vehicles may be infeasible due to the limited autonomy and to the small capacity of their batteries. While in urban areas GVs can delegate the burden of data processing to roadside units [6], NTNs may represent a valid alternative to serve computational requests in poorly connected rural areas. One of the main requirements of VEC systems is the support for real-time latency to guarantee safe autonomous driving as specified by the standards [7]. While satellites incur large round-trip times, UAVs offer limited computational capacity and short-lived connectivity [8]. Instead, HAPs provide large coverage, and are deployed in the stratosphere at an altitude of 20 km, which helps reduce the propagation delay to less than 1 ms. The effectiveness of HAPs in Mobile Edge Cloud (MEC) networks was studied in [9], where the authors addressed the problems of partial offloading and bandwidth allocation in remote areas without the coverage of ground infrastructure. However the benefit of HAPs for VEC is still an open question, that we will analyze in this letter.

Based on the above introduction, in this letter we present an optimization problem for real-time VEC in a rural scenario, in which multiple GVs can decide whether to process perception data (generated in the form of video frames) onboard or offload them to a HAP. We consider wireless transmissions in the millimeter wave (mmWave) band [10] With respect to our previous work [11], we develop a more accurate model to characterize both GVs' and HAP's queuing systems, and determine a closed-form expression for the average waiting time experienced by computing tasks. We show that it is convenient to offload some VEC processing tasks from GVs to the HAP under common assumptions for the processing capacity and load of the nodes.

II. System Model

A. Problem Formulation

We analyze a vast remote region where n GVs are distributed within an Area of Interest (AoI) of size A. Each GV generates sensory perceptions in the form of video frames of size n_{UL} according to a Poisson process with arrival rate r, as considered in some reference papers to model autonomous driving data [12]. Each frame requires a constant computational load C (e.g., for object detection [13]), which has to be executed fast enough to provide real-time services, i.e., at least at the frame rate of the sensor.

We consider that the GVs can offload a subset η of their computational load to a HAP providing VEC functionalities. According to the split property of the Poisson processes, we can distinguish two independent Poisson processes, namely for frames that are offloaded to the HAP at a rate ηr, and for frames that are processed onboard at a rate $(1-\eta)r$. On one side, local processing may involve long delays, given the limited computational capacity C_{GV} of low-budget car

Manuscript received 14 December 2022; accepted 18 January 2023. Date of publication 23 January 2023; date of current version 11 April 2023. This work was supported in part by the European Union through the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" under Grant PE0000001 (Program "RESTART"). The associate editor coordinating the review of this article and approving it for publication was K. Wang. *(Corresponding author: Alessandro Traspadini.)*

Alessandro Traspadini, Marco Giordani, and Michele Zorzi are with the Department of Information Engineering, University of Padova, 35131 Padua, Italy (e-mail: traspadini@dei.unipd.it; giordani@dei.unipd.it; zorzi@dei.unipd.it).

Giovanni Giambene is with the Department of Information Engineering and Mathematical Sciences, University of Siena, 53100 Siena, Italy (e-mail: giovanni.giambene@unisi.it).

Digital Object Identifier 10.1109/LWC.2023.3238851

models. On the other side, the HAP can count on a higher computing power C_{HAP} than available onboard a vehicle, i.e., $C_{\mathrm{HAP}} \geq C_{\mathrm{GV}}$, thus processing data faster, at the expense of a non-negligible communication delay for offloading data to the HAP server and for the processed data to be returned to the end users. We will investigate the optimized selection of $\eta \in [0, 1]$.

B. Delay Model

This section introduces our delay model for each data frame for both onboard and HAP-assisted processing.

1) Onboard Processing: Each GV is modeled as an M/D/1 queue, and the average delay is equal to

$$\bar{t}_{\mathrm{GV}} = \bar{W}_q(\mathrm{M/D/}\,1) + \left(C/C_{\mathrm{GV}}\right), \tag{1}$$

where $\bar{W}_q(\mathrm{M/D/}\,1)$ is the average waiting time (see Section II-D), while C/C_{GV} is the frame processing time onboard the GV.

2) HAP-Assisted Processing: Since the HAP has less severe space and energy constraints than a GV, it is modeled as an M/D/*c* queue, where *c* servers can process up to *c* frames in parallel. The average delay for processing tasks can be expressed as

$$\bar{t}_{\mathrm{HAP}} = 2\tau_p + t_{\mathrm{UL}} + t_{\mathrm{DL}} + \bar{W}_q(\mathrm{M/D/}\,c) + \left(C/C_{\mathrm{HAP}}\right), \tag{2}$$

where $\bar{W}_q(\mathrm{M/D/}\,c)$ is the average waiting time (see Section II-D), $\tau_p = d/c_l$ is the propagation delay (where d is the distance between the generic GV and the HAP, and c_l is the speed of light), and t_{UL} and t_{DL} are the times to transmit data to and from the HAP, respectively (see Section II-C).

C. HAP-to-Ground Channel Model

According to the 3GPP specifications [14], HAPs may operate through highly directional links at mmWaves, where the large bandwidth available at these frequencies offers the potential for ultra-fast connectivity [10]. With the assumption of an interference-free environment considering transmissions over orthogonal bands, the Signal to Noise Ratio (SNR) between transmitter *i* and receiver *j* can be calculated as:

$$\gamma_{ij} = \frac{\mathrm{EIRP}_i \cdot (G/T)_j}{\mathrm{PL}_{ij} \cdot k \cdot B} \tag{3}$$

where EIRP_i is the effective isotropic radiated power, $(G/T)_j$ is the receiver antenna-gain-to-noise-temperature, PL is the path loss, *k* is the Boltzmann constant, and *B* is the bandwidth. The path loss depends on the frequency (here mmWaves) and on the distance between *i* and *j*, and accounts for additional atmospheric attenuations as described in [15].

Based on the SNR in Eq. (3), we introduce the median ergodic capacity *R* between the generic GV and the HAP as $R = B\log_2(1+\gamma_{ij})$. Then, the transmission times t_{UL} and t_{DL} in Eq. (2), for each data frame, are given by $t_\ell = n_\ell/R_\ell, \quad \ell \in \{\mathrm{UL}, \mathrm{DL}\}$, where n_ℓ is the size of the transmitted data.

D. Queuing Model

Sensory frames are generated at rate *r*, and are queued and eventually processed at the GVs and HAP according to a Poisson process of rate λ, equal to $(1-\eta)r$ and $\eta r n$, respectively; we follow a first-come-first-served (FCFS) discipline. Notably, GVs operate via an M/D/1 queue, while HAPs via an M/D/*c* queue. Following the analysis made in [16], the M/D/*c* state probability p_j for state *j*, i.e., the probability that the queue at the HAP has *j* frames, can be computed by observing that any frame in the system at time $t + 1/\mu$ (where $1/\mu$ is the constant processing time) is either waiting in the queue at time *t* or has arrived in $[t, t + 1/\mu]$. Thus, given that the number of arrivals in an interval of duration $1/\mu$ is Poisson distributed with mean λ/μ, the state probability p_j is:

$$p_j = e^{-G}\frac{G^j}{j!}\sum_{k=0}^{c} p_k + e^{-G}\sum_{k=c+1}^{c+j} p_k \frac{G^{j-k+c}}{(j-k+c)!}, \tag{4}$$

where $G = \lambda/\mu$ is the offered traffic, and μ is the service rate at the HAP and GVs, i.e., C_{HAP}/C and C_{GV}/C, respectively. Eq. (4), combined with the normalization condition $(\sum_{j=0}^{+\infty} p_j = 1)$, generates an infinite system of linear equations that describe the state probabilities of the queue. We adopted the geometric tail approximation proposed in [16] to find a closed-form expression for this system of equations. Therefore, we can write

$$p_j = p_M \tau^{-(j-M)}, \quad j \geq M, \tag{5}$$

where τ is a constant depending on the server utilization, and *M* is a threshold state; we refer the interested reader to [16] to clarify how to compute these values. Based on Eq. (5), the infinite system in Eq. (4) can be reduced to a finite system of $M+1$ linear equations that can be solved easily.

Lemma 1: Based on Eq. (5), an empirical closed-form expression for the average waiting time of an M/D/*c* queue with arrival rate λ and service rate μ is given by

$$\bar{W}_q(\mathrm{M/D/}\,c) = \frac{\sum_{k=c}^{M-1} p_k(k-c) + p_M\left[\frac{M-c+\frac{1}{1-\frac{1}{\tau}}-1}{1-\frac{1}{\tau}}\right]}{\lambda}, \tag{6}$$

where a relatively small value of *M* can already provide an accurate approximation of $\bar{W}_q(\mathrm{M/D/}\,c)$.

Proof: See the Appendix. ∎

III. Optimal Offloading for HAP-Assisted VEC

In our scenario, we require that autonomous driving tasks, especially object detection on the video frames, be executed at least at the same rate at which new frames are generated, i.e., $t_{\max} = 1/r$. We model this aspect through the probability of real-time service P_{RT}, given by

$$P_{\mathrm{RT}}(\eta) = \eta P(t_{\mathrm{HAP}} \leq t_{\max}) + (1-\eta)P(t_{\mathrm{GV}} \leq t_{\max}), \tag{7}$$

where $t_{\max}$ is the maximum tolerable latency, and t_{GV} and t_{HAP} have been defined in Eqs. (1) and (2), respectively. Given the rural scenario, we assume that all GVs experience the same channel condition. Thus, we assume they follow the same policy, and offload data frames with the same probability η (*shared offloading factor*). The value of $P(t_{\mathrm{HAP}} \leq t_{\max})$ in Eq. (7) can be expressed by computing the maximum number of frames in the system that ensure real-time processing, that is

$$f_{\max}^{\mathrm{HAP}} = c\left\lfloor \frac{t_{\max} - t_{\mathrm{UL}} - t_{\mathrm{DL}} - 2\tau_p}{C/C_{\mathrm{HAP}}} \right\rfloor. \tag{8}$$

However, real-time processing is also possible when new offloading requests find the queue at full capacity, but some

processing tasks are already in service and will leave the system soon. We define as Δ_t^{HAP} the percentage of in-service tasks that ensure real-time processing even if more than $f_{\max}^{\mathrm{HAP}}$ requests are in the system. We have

$$\Delta_t^{\mathrm{HAP}} = \Gamma\left(\frac{t_{\max} - t_{\mathrm{UL}} - t_{\mathrm{DL}} - 2\tau_p}{C/C_{\mathrm{HAP}}}\right), \tag{9}$$

where $\Gamma(\mathrm{x}) = \mathrm{x} - \lfloor \mathrm{x} \rfloor$. Similarly, we can solve $P(t_{\mathrm{GV}} \leq t_{\max})$ in Eq. (7) by introducing $f_{\max}^{\mathrm{GV}}$ and Δ_t^{GV} for the GV:

$$f_{\max}^{\mathrm{GV}} = \left\lfloor \frac{t_{\max}}{C/C_{\mathrm{GV}}} \right\rfloor, \quad \Delta_t^{\mathrm{GV}} = \Gamma\left(\frac{t_{\max}}{C/C_{\mathrm{GV}}}\right). \tag{10}$$

Eq. (7) depends on the queuing state probability of both GVs and HAP, based on the model in Section II-D. Using the Poisson Arrivals See Time Averages (PASTA) property, and the probability distribution described in Eqs. (4) and (5), the real-time probability for the HAP can be expressed as

$$P(t_{\mathrm{HAP}} \leq t_{\max}) = \sum_{k=1}^{c} p_{\mathrm{bin}}(k, c, \Delta_t^{\mathrm{HAP}}) \sum_{j=0}^{k-1} p_{f_{\max}^{\mathrm{HAP}}+j} + \sum_{i=0}^{f_{\max}^{\mathrm{HAP}}-1} p_i, \tag{11}$$

where $p_{\mathrm{bin}}(k, c, \Delta_t^{\mathrm{HAP}}) = \binom{c}{k} (\Delta_t^{\mathrm{HAP}})^k (1 - \Delta_t^{\mathrm{HAP}})^{c-k}$ describes the probability that k offloading requests have been processed by at least a fraction Δ_t^{HAP}, and probabilities p_i are as in Eq. (4). Similarly, for the GV we have

$$P(t_{\mathrm{GV}} \leq t_{\max}) = \sum_{i=0}^{f_{\max}^{\mathrm{GV}}-1} p_i + \Delta_t^{\mathrm{GV}} p_{f_{\max}^{\mathrm{GV}}}. \tag{12}$$

The objective of our optimization problem is to choose the optimal offloading factor η, i.e., η^*, that maximizes P_{RT}, i.e.,

$$\arg\max_{\eta} \quad P_{\mathrm{RT}}(\eta), \tag{13a}$$

$$\text{subject to} \quad G_{\mathrm{HAP}} < c, \;\; G_{\mathrm{GV}} < 1 \tag{13b}$$

$$\eta \in [0, 1], \tag{13c}$$

where $G_{\mathrm{HAP}} = \eta r C n / C_{\mathrm{HAP}}$ and $G_{\mathrm{GV}} = (1-\eta) r C / C_{\mathrm{GV}}$ are the offered traffic at the HAP and the GVs, respectively, while Eq. (13b) sets a constraint on η to keep the system stable. Therefore, the optimization problem can be rewritten as

$$\arg\max_{\eta} \quad P_{\mathrm{RT}}(\eta), \tag{14a}$$

$$\text{subject to} \quad \eta \in [\eta_{\min}, \eta_{\max}], \tag{14b}$$

where $\eta_{\min} = \max(0, 1 - C_{\mathrm{GV}}/(rC))$ and $\eta_{\max} = \min(1, cC_{\mathrm{HAP}}/(rnC))$. Notably, C_{GV} (C_{HAP}) set a limit to the lower (upper) bound of η. Given that $P_{\mathrm{RT}}(\eta)$ is a continuous function defined on a closed interval $[\eta_{\min}, \eta_{\max}]$, there exists a solution to the optimization problem in Eq. (14) according to Weierstrass' theorem. Then, the problem is solved numerically using the Brent solver [17].

We can analyze the average time it takes for a GV, which may offload computational tasks to the HAP, to process each video frame. Since, on average, each GV processes a frame in $\bar{t}_{\mathrm{GV}}$, and the latency for an offloaded frame is $\bar{t}_{\mathrm{HAP}}$, then the average latency is evaluated as

$$\bar{t}(\eta) = \eta^* \bar{t}_{\mathrm{HAP}} + (1 - \eta^*) \bar{t}_{\mathrm{GV}}, \tag{15}$$

where η^* is the optimal offloading probability from Eq. (14).

IV. Performance Evaluation

In Section IV-A below, we introduce our simulator and its parameters, while numerical results are provided in Section IV-B.

A. Simulation Setup and Parameters

We analyze a rural/remote scenario with $n \in \{1, \ldots, 200\}$ GVs, uniformly distributed over a large AoI of 1000 km^2. The coverage area of the HAP is large enough to ensure that GVs are always and continuously under coverage in the AoI, even in case of mobility Each GV generates video frames of size $n_{\mathrm{UL}} \in [1, 3]$ Mb from its camera sensor at an average rate $r = 10$ fps. Object detection on these frames requires a constant computational load of $C = 60$ GFLOP per frame, which is computed as the average between the computational performance of two popular object detectors, namely Gaussian YOLO and SqueezeDet+ [18]. If frames are offloaded to the HAP (with probability η^*), eventually the processed output (i.e., the bounding boxes of the detected objects) is returned to the GVs in a packet of a much smaller size than the original frame, i.e., $n_{\mathrm{DL}} = 100$ kb, which implies that $t_{\mathrm{DL}} \ll t_{\mathrm{UL}}$.

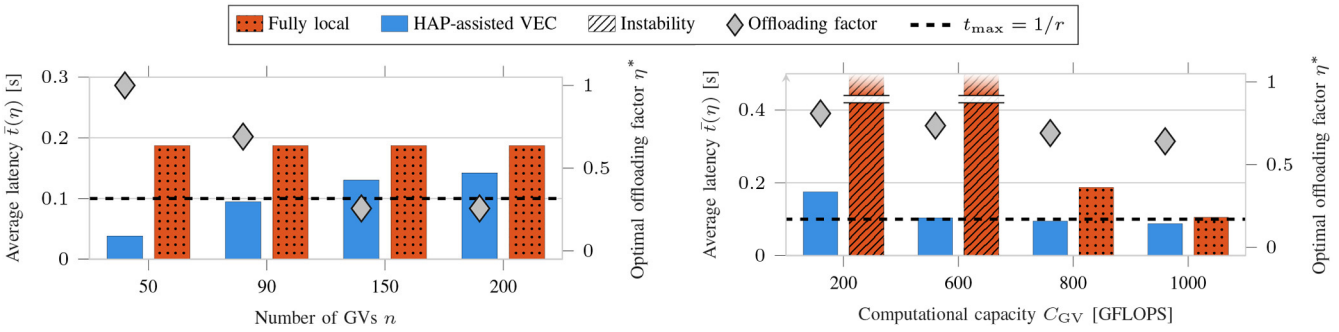
In the simulations, we compare a fully-local scheme in which all data frames are processed onboard (i.e., $\eta = 0$) and the optimal offloading policy (where $\eta = \eta^*$) based on the solution of Eq. (14) with $t_{\max} = 1/r$. GVs have a computational capacity of $C_{\mathrm{GV}} \in \{200, 600, 800, 1000\}$ GFLOPS, whereas the HAP operates via $c = 15$ parallel servers, each of which offers a computational capacity of $C_{\mathrm{HAP}} \in \{3000, 4000, 5000\}$ GFLOPS, so as to simulate different computing conditions. These values are consistent with the capacity of off-the-shelf computing units: for example, the Nvidia GeForce RTX 3080 Mobile CPU (GeForce GTX 1080 GPU), which is compatible with the space/power constraints onboard GVs (HAP), offers a capacity of 300 (9000) GFLOPS.

All devices operate at a carrier frequency of $f_c = 38$ GHz (mmWave) and with a bandwidth $B = 400$ MHz, while for a complete description of the channel parameters in Section II-C we refer the interested reader to [11, Table 1].

B. Numerical Results

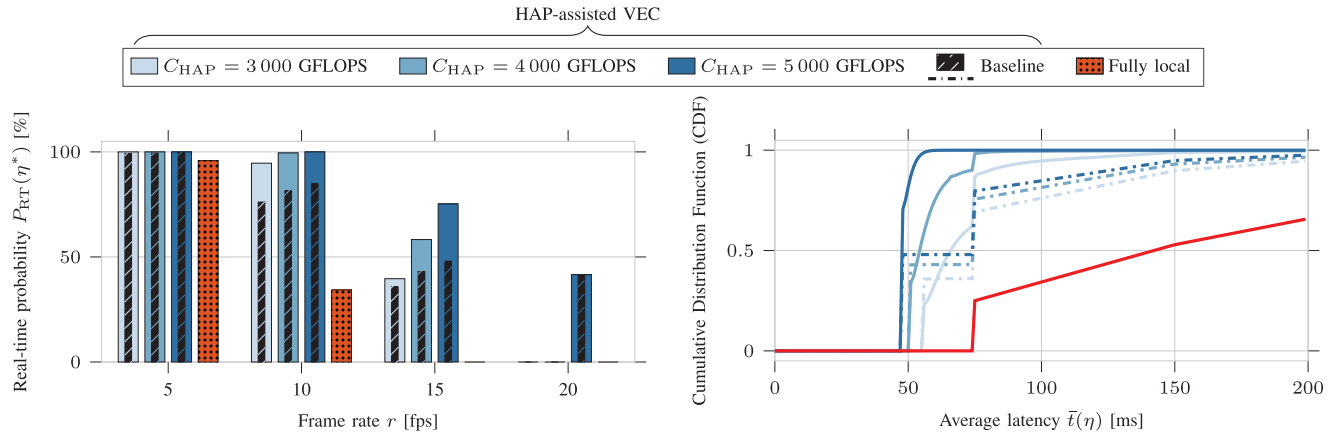
1) Number of Users (GVs): In Fig. 1(a), we evaluate the impact of the number of GVs in terms of VEC performance. We can see that, on average, fully local processing onboard the GVs, with $C_{\mathrm{GV}} = 800$ GFLOPS, requires about 200 ms for each video frame, regardless of the value of n, which is not compatible with real-time services. As expected, the average latency for processing data via the HAP grows with n, due to the more frequent offloading requests and the resulting populated queues as the number of GVs increases. In any case, HAP-assisted VEC with powerful processing can reduce the average latency compared to the fully local scenario (up to almost 5 times when $n = 50$ GVs), despite the communication delay for uploading data frames to the HAP servers and for delivering the processed output to the end nodes.

Interestingly, the optimal offloading probability η^* decreases with the number of GVs. When $n < 150$, the best choice is to offload data with probability $\eta^* > 0.7$. This approach allows to achieve, on average, real-time data processing at the frame rate of the sensors, i.e., $\bar{t}(\eta) < t_{\max}$. On the other hand, when $n \geq 150$, the average latency $\bar{t}(\eta)$ to/from the HAP alone exceeds the latency constraint,



(a) $C_{\mathrm{GV}} = 800$ GFLOPS. (b) $n = 90$ GVs.

Fig. 1. Optimal offloading factor η^* (right axis) and average latency $\bar{t}(\eta)$ (left axis) vs. n and C_{GV}, for $r = 10$ fps and $n_{\mathrm{UL}} = 3$ Mb. Striped bars are interrupted to represent the case of unstable queues where the latency increases indefinitely in the long term.



(a) Real-time probability at the optimal offloading factor η^*. (b) Statistics of the average latency for $r = 10$ fps.

Fig. 2. Real-time probability (left) and average latency (right) vs. C_{HAP} and r, when $C_{\mathrm{GV}} = 800$ GFLOPS, $n = 100$ GVs and $n_{\mathrm{UL}} = 1$ Mb. Dash-dot lines and black striped bars refer to a baseline offloading scheme which balances the load of the HAP and of the GVs equally.

i.e., $P(t_{\mathrm{HAP}} \leq t_{\max}) = 0$. This is due to the fact that the more populated system may overload the available channel bandwidth, thus resulting in longer transmission delays, which makes fully local processing an increasingly desirable option: in these conditions, the optimal offloading factor is as low as $\eta^* = 0.25$.

2) GV's Computational Capacity: We let C_{GV} vary from 200 to 1000 GFLOPS, as considered in [11], in a scenario with $n = 90$ GVs, and plot the optimal offloading factor η^* in Fig. 1(b). We see that η^* decreases as C_{GV} increases, specifically from 0.81 (200 GFLOPS) to 0.64 (1000 GFLOPS). Fully local processing becomes increasingly more attractive as vehicles incorporate more powerful processing hardware, to avoid additional delays for data offloading. Still, for the values of capacity in Fig. 1(b), HAP-assisted VEC remains a more convenient choice to achieve real-time performance, even though the gap in terms of average latency compared to the fully local scenario is only 17% when $C_{\mathrm{GV}} = 1000$ GFLOPS. Notably, Fig. 1(b) shows that fully local processing leads to queue instability if $C_{\mathrm{GV}} \leq 600$ GFLOPS, which further motivates the need for offloading to HAP.

3) Frame Rate: In Fig. 2(a), we study the real-time probability at the optimal offloading factor η^*, as a function of the frame rate r and the computational capacity of the HAP. In general, increasing r allows the sensor to capture data at better resolution, which leads to more accurate autonomous driving perception, at the cost of a higher data rate and complexity. As expected, $P_{\mathrm{RT}}(\eta^*)$ is a decreasing function of r: for $C_{\mathrm{HAP}} = 5000$ GFLOPS, P_{RT} ranges from 1 at $r = 10$ fps to 0.41 at $r = 20$ fps. For $C_{\mathrm{HAP}} < 5000$ GFLOPS, the system is constrained by the capacity of the queues at the HAP servers; in particular, the system becomes unstable as $r \geq 20$ fps. Moreover, for the fully local configurations, $P_{\mathrm{RT}} > 0.95$ only with $r = 5$ fps and $C_{\mathrm{GV}} = 800$ GFLOPS, an indication that the limited computational capacity at the GVs is not enough to support automotive tasks at the frame rate of the sensors.

For comparison, we considered a simple baseline offloading scheme that balances the load of the HAP and the GVs using an offloading factor η_{bl} given by

$$\eta_{\mathrm{bl}} = \left(\frac{n \cdot C_{\mathrm{GV}}}{c \cdot C_{\mathrm{HAP}}} + 1 \right)^{-1}, \tag{16}$$

which guarantees $G_{\mathrm{HAP}}(\eta_{\mathrm{bl}})/c = G_{\mathrm{GV}}(\eta_{\mathrm{bl}})$. Results in Fig. 2(a) show that this baseline can support real-time processing with a probability higher than 0.85 as long as the frame rate is less than 10 fps, otherwise the system becomes unstable. On the other hand, our proposed framework is able to regulate the offloading factor based on the actual capacity of the network, and can offer real-time performance even in more congested scenarios.

In Fig. 2(b), we plot the Cumulative Distribution Function (CDF) of the average latency for $r = 10$ fps. HAP-assisted VEC, even with $C_{\text{HAP}} = 3\,000$ GFLOPs, achieves real-time processing (i.e., $\bar{t}(\eta) \leq 1/r = 100$ ms) with a probability higher than 0.94, vs. 0.34 for fully local processing. The CDF of the baseline shows a trend similar to the fully local scheme, but with a constant gain from 0.44 to 0.54 which depends on C_{HAP}. From Fig. 2(b), it is clear that the CDF shows two steps: the first step (at $\bar{t}(\eta) \simeq 50$ ms) corresponds to the minimum processing time for HAP-assisted VEC, i.e., the time for downlink and uplink transmissions, and the processing time at the HAP; the second step (at $\bar{t}(\eta) \simeq 75$ ms) corresponds to the minimum processing time for fully-local processing, i.e., $C/C_{GV} \simeq 75$ ms.

V. Conclusion and Future Works

In this letter, we analyzed a rural scenario where autonomous GVs capture sensory data to perform real-time object detection, and addressed the following research question: *"Is it feasible to offload and process self-driving-related tasks to HAPs?"* For this purpose, we described both the GVs and the HAP as queues that receive sensory data, and determined a closed-form expression for the average waiting time of data in those queues. We then formalized a VEC optimization problem and compared the case in which data processing is performed onboard the GVs (fully local scenario) vs. the case in which a fraction of the processing load is offloaded to HAP servers. Simulation results showed that the limited computational capacity of GVs is generally not compatible with real-time operations, while an optimal offloading factor exists to minimize the processing time. In particular, real-time performance requires a computational capacity at the HAP higher than 3000 GFLOPS for a frame rate $r \leq 10$ fps.

In future work, we will extend our offloading optimization problem by incorporating the impact of power consumption.

Appendix

Proof of Lemma 1

Let us consider an M/D/c queue with arrival rate λ and service rate μ. The state probability distribution fulfills Eq. (4). The average length of the queue $E[L_q]$ can be written as

$$E[L_q] = \sum_{k=c}^{M-1} p_k(k-c) + \sum_{k=M}^{+\infty} p_k(k-c). \tag{17}$$

Using the approximation in Eq. (5), the second summation in Eq. (17) can be rewritten as

$$\sum_{k=M}^{+\infty} p_k(k-c) = \sum_{k=M}^{+\infty} p_M \tau^{-(k-M)}(k-c) \tag{18}$$

$$= p_M \left[\sum_{j=0}^{+\infty} \tau^{-j} j + \sum_{j=0}^{+\infty} \tau^{-j}(M-c) \right] \tag{19}$$

$$= p_M \left[\sum_{j=0}^{+\infty} \tau^{-j} j + \frac{M-c}{1-\frac{1}{\tau}} \right]. \tag{20}$$

The summation in Eq. (20) is equal to

$$\sum_{j=0}^{+\infty} \tau^{-j} j = \sum_{k=1}^{+\infty} \sum_{j=k}^{+\infty} \tau^{-j} = \sum_{k=1}^{+\infty} \tau^{-k} \sum_{j=0}^{+\infty} \tau^{-j} \tag{21}$$

$$= \sum_{k=1}^{+\infty} \left[\sum_{j=0}^{+\infty} \tau^{-j} - \sum_{j=0}^{k-1} \tau^{-j} \right] \tag{22}$$

$$= \sum_{k=1}^{+\infty} \left[\frac{\left(\frac{1}{\tau}\right)^k}{1-\frac{1}{\tau}} \right] = \frac{\frac{1}{1-\frac{1}{\tau}} - 1}{1-\frac{1}{\tau}}. \tag{23}$$

Finally, $E[L_q]$ can be written from (17), (20), and (23) as

$$E[L_q] = \textstyle\sum_{k=c}^{M-1} p_k(k-c) + p_M \left[\frac{M-c+\frac{1}{1-\frac{1}{\tau}}-1}{1-\frac{1}{\tau}} \right]. \tag{24}$$

where p_0, p_1,..., p_M are computed by a linear system, as explained in Section II-D. The average waiting time in the queue, $E[W_q]$, is derived from Little's Law, i.e., $E[W_q] = E[L_q]/\lambda$, so that we obtain the expression in Eq. (6).

References

[1] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6G networks: Use cases and technologies," *IEEE Commun. Mag.*, vol. 58, no. 3, pp. 55–61, Mar. 2020.

[2] A. Chaoub et al., "6G for bridging the digital divide: Wireless connectivity to remote areas," *IEEE Wireless Commun.*, vol. 29, no. 1, pp. 160–168, Feb. 2022.

[3] M. Giordani and M. Zorzi, "Non-terrestrial networks in the 6G era: Challenges and opportunities," *IEEE Netw.*, vol. 35, no. 2, pp. 244–251, Mar./Apr. 2021.

[4] D. C. Nguyen et al., "6G Internet of Things: A comprehensive survey," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 359–383, Jan. 2022.

[5] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: A survey," *Mobile Netw. Appl.*, vol. 26, no. 3, pp. 1145–1168, Jun. 2021.

[6] A. Belogaev, A. Elokhin, A. Krasilov, E. Khorov, and I. F. Akyildiz, "Cost-effective V2X task offloading in MEC-assisted intelligent transportation systems," *IEEE Access*, vol. 8, pp. 169010–169023, 2020.

[7] G. Velez, Á. Martin, G. Pastor, and E. Mutafungwa, "5G beyond 3GPP release 15 for connected automated mobility in cross-border contexts," *Sensors*, vol. 20, no. 22, p. 6622, Nov. 2020.

[8] M. Ke et al., "An edge computing paradigm for massive IoT connectivity over high-altitude platform networks," *IEEE Wireless Commun.*, vol. 28, no. 5, pp. 102–109, Oct. 2021.

[9] D. S. Lakew, A.-T. Tran, N.-N. Dao, and S. Cho, "Intelligent offloading and resource allocation in HAP-assisted MEC networks," in *Proc. ICTC*, 2021, pp. 1582–1587.

[10] D. Wang, M. Giordani, M.-S. Alouini, and M. Zorzi, "The potential of multilayered hierarchical nonterrestrial networks for 6G: A comparative analysis among networking architectures," *IEEE Veh. Technol. Mag.*, vol. 16, no. 3, pp. 99–107, Sep. 2021.

[11] A. Traspadini, M. Giordani, and M. Zorzi, "UAV/HAP-assisted vehicular edge computing in 6G: Where and what to offload?" in *Proc. EuCNC-6G Summit*, 2022, pp. 178–183.

[12] M. Lundgren, E. Stenborg, L. Svensson, and L. Hammarstrand, "Vehicle self-localization using off-the-shelf sensors and a detailed map," in *Proc. IEEE Intell. Veh. Symp.*, 2014, pp. 522–528.

[13] V. Rossi, P. Testolina, M. Giordani, and M. Zorzi, "On the role of sensor fusion for object detection in future vehicular networks," in *Proc. EuCNC-6G Summit*, 2021, pp. 247–252.

[14] "Solutions for NR to support non-terrestrial networks (NTN)," 3GPP, Sophia Antipolis, France, Rep. TR 38.821 (Release 16), 2020.

[15] M. Giordani and M. Zorzi, "Satellite communication at millimeter waves: A key enabler of the 6G era," in *Proc. IEEE ICNC*, Feb. 2020, pp. 383–388.

[16] H. C. Tijms, *A First Course in Stochastic Models*. New York, NY, USA: Wiley, 2003.

[17] K. R. Gegenfurtner, "PRAXIS: Brent's algorithm for function minimization," *Behav. Res. Methods Instrum. Comput.*, vol. 24, no. 4, pp. 560–564, Dec. 1992.

[18] W. Li and K. Liu, "Confidence-aware object detection based on MobileNetv2 for autonomous driving," *Sensors*, vol. 21, no. 7, p. 2380, Mar. 2021.

Open Access funding provided by 'Università degli Studi di Padova' within the CRUI CARE Agreement