Green-EMulTO: a next generation edge-assisted multi-level traffic orchestrator for green computing in consumer autonomous vehicles

Oshin Rawlley
Shashank Gupta
Kashish Mahajan
Shailendra Rathore

This is the accepted version of a paper published in IEEE Transactions on Consumer Electronics:

Rawlley, O., Gupta, S., Mahajan, K. & Rathore, S. (2024) 'Green-EMulTO: a next generation edge-assisted multi-level traffic orchestrator for green computing in consumer autonomous vehicles'. *IEEE Transactions on Consumer Electronics*. DOI: https://doi.org/10.1109/TCE.2024.3442437

Green-EMulTO: A Next Generation Edge-Assisted Multi-Level Traffic Orchestrator for Green Computing in Consumer Autonomous Vehicles

Oshin Rawlley, *Student Member, IEEE,* Shashank Gupta, *Senior Member, IEEE,* Kashish Mahajan, Shailendra Rathore

***Abstract*—Imaging technology in vehicular communications has advanced in the consumer industry. Owing to its ability of sensing ambient environment, decision control, and actuating the vehicle, the technology is able to enhance vision, improve traffic management, and offer convenience to the consumer autonomous vehicles. These vehicles in green computing environments require accurate obstacle detection (OD) and real-time video analytics to enhance on-road perception for forewarned accidents and pollution-free navigation. However, unforeseen obstacles in high vehicle speeds, adverse environments, etc., cause accidents leading to pollution. In this paper, we propose a Green-EMulTO, an Edge-assisted Multilevel Traffic Orchestrator that recognizes obstacles in low latency. It employs a priority queue for the traffic imaging streams, a bandwidth manager for V2X services, and a lightweight DNN model for fast on-device OD. We also introduce a Synergistic service placement and cost minimization algorithm (SSPCM) based on Lyapunov optimization and Markov approximation. It reduces the response latency by addressing the intrusive dynamics of video and unknown network fluctuations in the autonomous driving environment. This orchestrator is designed to provide a pollution-free environment by reducing road accidents thereby satisfying the green vehicular environmental goals. In addition, we develop an autonomous driving platform using NVIDIA JetRacer AI Pro Kit and Jetson Nano for system-level verification. We have also compared it with the benchmark lightweight models (YOLOV5-nano, YOLOv6-nano, YOLOV7-tiny, YOLOv5-small, YOLOv6-small) on such commercial devices. Green-EMulTO witnessed an improvement of up to 20% in accuracy and the training time was reduced to less than 60%. Hence, this orchestrator improved the real-time inference speed over different autonomous driving environments.**

***Index Terms*—Consumer Industry, Green IoV, autonomous driving, traffic orchestrator, imaging technology, video analytics, obstacle detection, low-latency**

I. Introduction

THE potential of Automated Driving Assistant Systems (ADAS) [1] of Autonomous Vehicles (AVs) in consumer industry has played a vital role in enhancing driver's safety and reducing traffic congestion for future Intelligent Transportation Systems (ITS). The essential functionalities of AVs in consumer applications are generally laid down as low-latency sensing, lane merge, real-time video analytics, obstacle detection, etc. Low-latency traffic video analysis aided by vehicle-to-everything (V2X) services is fundamental to smart consumer applications like real-time obstacle detection especially in the stochastic autonomous driving environments [2]. Further, the AVs rely on powerful deep neural networks (DNNs) deployed on edge servers (ES) for obstacle detection and to achieve robust perception on the roads. However, due to heavy network structures and extensive parameters of DNNs, the resource-constrained embedded devices get burdened with enormous computations [3]. Moreover, the deployment of ES for wider spatial coverage also increases the energy consumption [4], [5]. Considering the time-sensitive and spatial variations in AV requests, there is a redundant resource utilization. Due to high-density traffic in urban areas, the larger number of vehicles causes localized congestion where the ES becomes overwhelmed [6]. The varying demands of the vehicles also affect the network resource operations. For e.g., service invocation entails frequent spatial references and contexts, involving not only the dispersed interactions between service provider and client hosts but also the communications with their spatial environments [7]. Additionally, the service status such as workload, running status, etc., and the network environment i.e., traffic stream congestion are subject to temporal variations [8]. Moreover, resources are often over-provisioned to handle peak loads, causing idle times for the resources during off-peak times.

To comply with the green Internet of Vehicles (GIoV) [9] sustainability principles, which couples ITS with a pollution-free integrated traffic management system, it is important to make intelligent decisions on-road using imaging technology [10]. This will avoid traffic accidents and jams thereby, reducing pollution and will establish energy-efficient GIoV. GIoV emphasizes network transmission optimization, efficient traffic management with real-time video analytics, energy consumption minimization, etc., for next-generation IoVs. The main aim is to optimize the configurations for each traffic stream and to maximize the trade-off between accuracy-energy and latency. However, there are certain challenges that are encountered such as time-varying optimal offloading configuration, erratic vehicle flows, unpredictable bandwidths. Therefore, an adaptive framework which is able to adjust the best configuration in dynamic conditions is required.

Manuscript received July 10, 2012. This work is supported by CHANAKYA Fellowships of IITI DRISHTI CPS Foundation under the National Mission on Interdisciplinary Cyber-Physical System (NM-ICPS) of the Department of Science and Technology, Government of India. Grant no: CF-2022-PhD-000002.

O. Rawlley, S. Gupta, K. Mahajan are with Department of Computer Science & Information Systems, BITS Pilani, Rajasthan, India. S. Rathore is with Division of Cyber Security, School of Design and Informatics, Abertay University, UK. (e-mail: p20200063@pilani.bits-pilani.ac.in, shashank.gupta@pilani.bits-pilani.ac.in, f20200995@pilani.bits-pilani.ac.in, s.rathore@abertay.ac.uk)

Moreover, there are three important parameters which motivate us to formulate the optimization function are accuracy, energy, and latency. Our solution proposes the utility function which is the maximum of discrete time slots averaged for all traffic streams. The configurations are adjusted in every slot which are averaged over time.

In the consumer industry, green computing has emerged as a novel paradigm that implements environment-friendly computing, and imaging principles and has a vital role in developing enduring IoV networks [11]. To accomplish GIoV goals, lightweight DNN models should be deployed for detecting obstacles in non-ideal environments on resource-constrained devices. It is also noteworthy that the DNN-equipped AVs and their confluence with V2X services deliver real-time inferences from the traffic videos. For this, the concept of continuous learning has also been incorporated to continuously update the lightweight DNN models [12], [13]. In this regard, the existing studies [14], [15] have also studied energy minimization while applying DNNs, but have overlooked the problem of solving the trade-off between traffic video analytics accuracy under long-term low-latency (LTLL) constraint. In addition, the unruly and diverse scenarios also pose a challenge in detecting the obstacles for traffic video analytics using imaging.

Therefore, to improve the driver's experience and also to contribute to establishing next-generation GIoV, we have proposed green-EMulTO with perception-based safety detection during traffic video stream analytics under the long-term low latency (LTLL) constraint. In order to make the environment pollution-free from a spate of accidents, EMulTO has addressed distinct fundamental smart consumer applications such as traffic video analytics, obstacle detection using imaging for real-time danger detection in low-response times, etc. For that, we employ an edge-assisted lightweight DNN model on the input traffic video streams to enhance the on-road perception for autonomous driving. The manuscript attempts to improve the accuracy variations along with the reduction in the latency of vehicular communication. We have proposed a traffic management framework for differential priority traffic streams having varied latency needs. We employ a bandwidth management system that allocates variable bandwidths according to the application configurations. This framework addresses several challenges such as limited edge capacity, fluctuating network, and indiscreet traffic stream content. We also compare our proposed lightweight model with five nano and tiny versions of the YOLO series to emphasize its suitability and superiority for GIoV in smart consumer imaging applications. The major contributions of our manuscript are listed below:

- We propose an edge-assisted green EMulTO, a traffic orchestrator that analyses real-time traffic generated by onboard edge cameras. It includes a novel lightweight DNN model with improved architecture to detect obstacles in tight stochastic environments.
- To maintain a trade-off between the traffic video analytics accuracy and LTLL, we have developed a synergistic placement and cost minimization (SSPCM) algorithm based on Lyapunov optimization and Markov approximation.
- We have also set up an autonomous driving platform using NVIDIA devices. We validate EMulTO under dynamic environments with different video resolutions. It has achieved an accuracy boost of up to 20% and has reduced training time under 60%, suitable for an energy-efficient GIoV environment.

The proposed framework is anticipated to contribute in enhancing the AV vision with the help of low-latency communication. We have also provided the optimization function for the accuracy energy tradeoff which is a good contributor to the green environment in the consumer industry. The framework intends to save energy while conducting all the operations.

II. System Model

It is vital to analyze road representations and detect obstacles under adverse weather in the input traffic video coming from the end cameras for detecting danger events. For efficient video analysis, we consider a single edge server (ES) shared by a group of S traffic video channels denoted by $V = \{v_1, v_2, ..., n_n\}$. S can share a common narrow uplink channel to offload the video frames onto the server. There are N parallel DNN models $P = \{p_1, p_2, ..., p_n\}$ which take inputs of different sizes are deployed on the ES. res_i refers to the resolution of the input fed into the ith DNN. Let p_0 denote the lightweight DNN model local to each end device. inf_i and $cost_i$ are the input processing time and per-frame cost respectively, for a model p_i. Traffic streams with low input image resolution takes less time for inference (lower inf_i) and are less resource intensive (lower $cost_i$). We divide the time into equally spaced intervals whose duration is adjusted at which updates can be performed to the offloading configuration. *Figure* 1 describes the network architecture of the edge-driven video analytics framework detecting obstacles mainly cars and pedestrians. Tag 1: A *Queue Shuffler* receives the incoming traffic streams from the vehicles and cameras on the road. that. Tag 2: The *Discovery server* provides an active server list that broadcasts all the ES that are ready to receive the requests. Tag 3: The *Queue Shuffler* then, offloads the incoming traffic streams across all the edge nodes to analyze them further for real-time on-road perception. Green-EMulTO primarily employs a multi-queue management mechanism within an edge node that solves the delay problem occurring with a single queue. Tag 4: The *Dynamic Bandwidth Manager* reserves the bandwidth for differential priority streams LC, LT, and UA. Tag 5: The streams are dispatched to the ES. If the estimated delay is more than the pre-defined threshold value, then the *bandwidth manager* initiates the allocation for every queue. It is responsible for altering the bandwidth after re-calculating the adjustments done for differential priority queues to ensure optimal utilization of the local bandwidth. This assists in quick danger detection by informing the driver about adverse conditions on the roads.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3442437

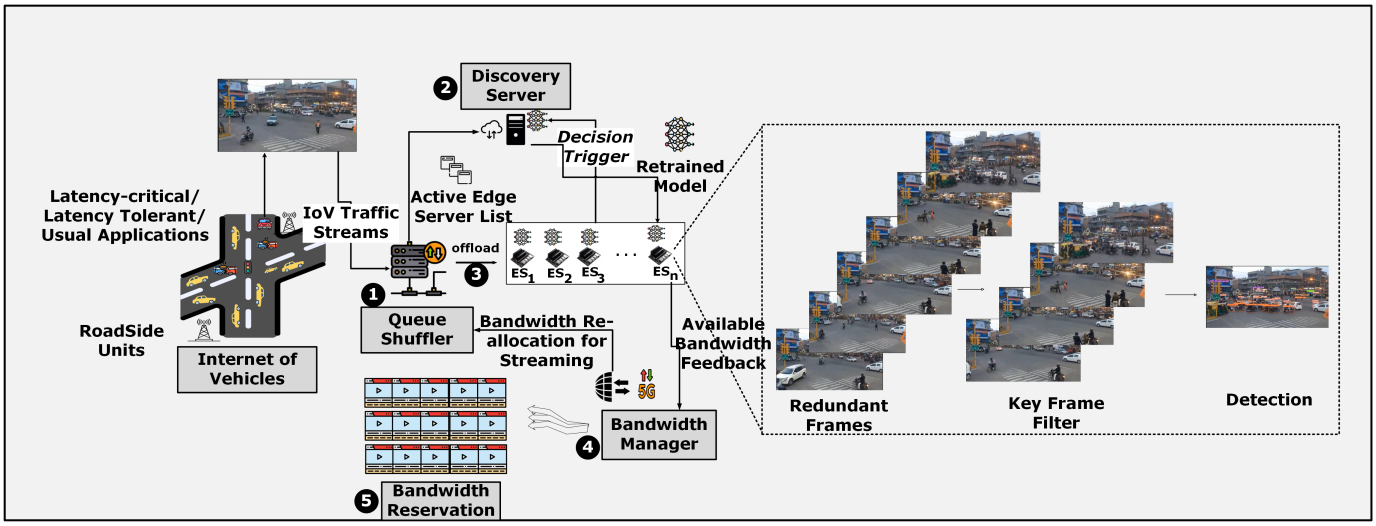

Figure 1: Proposed network architecture of the edge-driven video analytics framework

A. Lightweight DNN Model Architecture

Since the time is divided into equal chunks, we extract the traffic keyframes within each chunk for accurate filtering of the region of interest. Next, we propose to deploy a lightweight DNN model on the edge devices which saves time and accelerates the inference process specifically for LC tasks. With a prior feature extraction process, we can ensure accurate localization for better retraining and classification of vehicular instances. Our object detection model retains the three-fold architecture comprising of the feature extraction network (Shortened backbone of YOLOV4 Backbone), feature enhancement (Modified Feature Fusion Network in Neck), introduction of Leaky ReLU activation, sparse training and model accelerated inference adaptation and the object detection network (Head). Channel pruning has been performed in the backbone to decrease the size of the stack compression model of the residual block for reducing the number of convolutional operations that are involved in the feature extraction process. The PANet network in YOLOv4 was deleted and a combination of SPP and FPN was left intact for performing feature fusion based on our findings. This modification eliminated frequent feature extraction process that caused a lot of redundancy, which would have made it unsuitable for deployment on constrained embedded devices.

The Shortened Backbone of YOLOV4: For connecting two adjacent networks of varying scales, the convolutional layers and up-sampling have been used. The Head part has been left undisturbed, composed of convolutional layers with 3×3 and 1×1 filters followed by a linearized output, which is a regression prediction based on anchor boxes. A feature map is split into two sections, named S1 and S2, respectively. S1 is directly connected to the end of stage portion, whereas, the other part goes through a dense block concatenation. The feature map from the S1 does not need to pass through the transition layer comprising of dense layers. This is how the benefits of DenseNet's feature reuse characteristics are exploited. This method improves the network's learning ability by keeping gradient information intact and solving the vanishing gradient problem. We use CSPBlock in place of ResBlock as the residual network to decrease the repeated gradients and carry out efficient computations. We divide the channels in the characteristic layer by using two CSPBlock modules and simplifying the residual structure to reduce the memory footprint. Gradient's difference can be enhanced by the CSPBlock, improving the learning ability of the convolutional layer. *Modified Feature Fusion Network in Neck*: We have fused the SPP and FPN modules. Our SPP block uses the maximum pooling method for performing concatenation after obtaining the feature map. FPN is basically a feature detector, a method of creating a feature pyramid within the convolution. We mostly use the top-level features for prediction in object detection. However, we introduce FPN to reduce calculations. The low-resolution feature maps having strong semantic information are merged with high resolution maps having weak semantic information to achieve multi-scale prediction. *Decision Trigger*: Unlike earlier works [16] [17] who implemented static model update strategy, we incorporated an adaptive decision trigger that responds to the timely changes in the driving environment and perform fast detections. Moreover, to make the process more adaptive, we ensure to initiate the retraining process only when the detection accuracy of the sampled frames drops. This is done by making the threshold adaptable to both quick response and less burdening on devices [18].

B. Latency Model

The SOTA OD models, such as Faster RCNN and MobileNet, show inference results above 80ms at the edge level. Even the support of *Geforce RTX GPU 2080 Ti* doesn't help in improving the latency [19]. The average inference delay for detecting objects within a single frame must be constrained to within tens of milliseconds in real-time traffic stream analytics. Therefore, we model accuracy, energy, and service latency to optimize on-road video accuracy while ensuring LTLL and optimal energy usage. *Models for Analytics Accuracy*: Our first task is to understand the interdependence between analytics accuracy and image resolution. To do so, we deploy

our proposed DNN model on NVIDIA Jetson Nano, to detect pedestrians in a video clip with various resolutions. F1 score measures the accuracy of compressed frames by comparing the detections to a high-resolution frame. It considers both evaluators i.e., precision and recall. A detection is correct if the label matches and overlaps the ground truth box by at least 70% IOU threshold. *Figure 2* shows the experimental results where the dotted purple line represents the scenario in the time interval *a*, indicating small target objects, whereas the yellow line depicts the scenario in the interval *b*, indicating large targets. The exhibitions reveal that the higher resolution leads to better accuracy, but the gain saturates at higher resolutions. Therefore, the relationship can be modelled as $0.988 - 4.469e^{\frac{(-res)}{200}}$, with an RMSE (Root mean square error) lower than 0.02. Moreover, the impact of resolution on accuracy also changes over time. Though high resolution is essential in small obstacle detection, but less critical for larger ones. Hence, the accuracy model should be adaptively updated conditional on the suffering of varying obstacle size detections. We use $A^r_{(s,t)}(res)$ to depict the accuracy as a function of the resolution denoted by res, for the traffic stream v_s in time interval t. A binary variable $d(s,i,t)$ indicates whether the traffic stream v_s has selected model p_i in time interval t. The frame resolution for a traffic stream v_s in the time interval t is given by $\sum_{i=0}^{N} d(s,i,t)res_i$. *Figure* 2 (b) shows that a higher frame sampling rate leads to better accuracy, with more frames having an F1 score above 0.67. The impact is higher for fast-moving objects in interval x. We approximate the accuracy as a function of the frame rate using a concave function $A^f_{(s,t)}(fps)$, which is modified at the beginning of a time interval according to the target speed in the video. $fps_s(t)$ is the frame sampling rate for traffic stream v_s in time interval t. The effect of the frame sampling rate and the input resolution on accuracy is independent. Therefore, the accuracy corresponding to the configuration for traffic stream v_s in time interval t is given by :

$$A^r_{(s,t)}\left(\sum_{i=0}^{N} d(s,i,t)res_i\right) A^f_{s,t}(fps_s(t)) \tag{1}$$

The formula for calculating average accuracy for the group of S traffic streams in time interval t is as follows:

$$A_{avg}(t) = \frac{1}{S}\sum_{s=1}^{S} A^r_{(s,t)}\left(\sum_{i=0}^{N} d(s,i,t)res_i\right) A^f_{(s,t)}(fps_s(t)) \tag{2}$$

C. Model for Energy Consumption

We are using a Jetson device for computations and inference. Therefore, we calculate (a) energy involved in transmitting data to the ES and (b) energy used in processing the traffic video frames by the proposed model. The amount of energy expended during data transmission is proportional to the amount of data transmitted. The data contained within a single frame is given by $kres^2$ bits where k is a constant of proportionality. If e^s_{trans} is the transmission energy used for one bit for traffic stream v_s, the average energy consumption for transmission in time interval t is given by :

$$E_{trans}(t) = \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N} e^s_{trans}k(d(s,i,t)res_i)^2 fps_s(t) \tag{3}$$

The average energy consumption during the local processing for all traffic streams combined during time interval t can be calculated as :

$$E_{loc}(t) = \frac{1}{S}\sum_{s=1}^{S} d(s,0,t)e^s_{loc}fps_s(t) \tag{4}$$

where e^s_{loc} denotes the energy cost for processing a single frame locally for traffic stream v_s. Combining (2) and (3) gives $E(t)$, i.e., the total average energy consumption for all traffic streams combined in time interval t to be $E_{trans}(t) + E_{loc}(t)$.

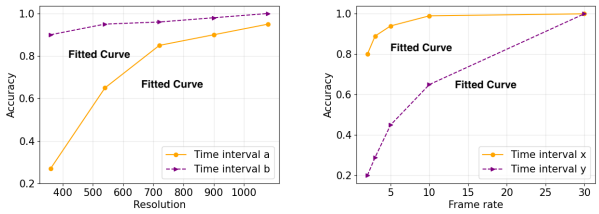

Figure 2: Interplay of Accuracy variations based on configurations

D. Model for Service Latency

Two factors mainly contribute to the global latency calculation for a single frame w.r.t a traffic stream in time interval t i.e., data transmission latency and DNN processing latency. The global latency per frame experienced by the traffic stream v_s in time interval t is given by the formula:

$$Gl_s(t) = \frac{\sum_{i=1}^{N} k(d(s,i,t)res_i)^2}{bw_s(t)} + \sum_{i=0}^{N} d(s,i,t)inf_i \tag{5}$$

, where k ensures that the frame size is in bits. The global average latency (GAL) for the group of S traffic streams in the time interval t is calculated as :

$$Gl_{Avg}(t) = \frac{1}{S}\sum_{s=1}^{S} Gl_s(t) \tag{6}$$

III. Problem Formulation

Video stream analytics in smart consumer applications requires high accuracy. However, it consumes energy and has latency issues. Therefore, we aim to achieve a desirable accuracy for our proposed lightweight DNN model pre-conditioned on LTLL with judicious energy consumption. To realize accurate object detection, it is necessary to have high accuracy in video stream analytics. However, there is a high energy consumption of the device by the deployed model. Hence, we aim to achieve a desirable accuracy for our proposed lightweight DNN model pre-conditioned on LTLL with judicious energy consumption. We formulate the optimization utility function to establish a trade-off between the accuracy and the energy cost and

maximize it for all video streams, as follows:

$$O : max\left\{d, bw, fps\right\} \lim_{T\to+\infty} \frac{1}{T}\sum_{t=0}^{T}\left(A_{avg}\left(t\right) - \Omega E\left(t\right)\right)$$

$$\begin{aligned}
\textit{st:}\quad & C1 : \textstyle\sum_{i=0}^{N} d(s,i,t) = 1, u_s \in V, t \in \tau = \{1,\ldots,T\} \\
& C2 : d(s,i,t) = \{0,1\}, v_s \in V, t \in \tau \\
& C3 : fps_s(t) < \textstyle\sum_{i=0}^{N} d(s,i,t)\frac{1}{infi}v_s \in V, t \in \tau \\
& C4 : \textstyle\sum_{s=1}^{S}\sum_{i=1}^{N} d(s,i,t)cost_i < COSTv_s \in V, t \in \tau \\
& C5 : \textstyle\sum_{s=1}^{S} bw_s(t) = BW(t), v_s \in V, t \in \tau \\
& C6 : \lim_{T\to\infty} \frac{1}{T}\textstyle\sum_{t=0}^{T} Gl_{Avg}(t) < GL_{max}
\end{aligned}$$

Ω decides the relative importance of reducing energy consumption w.r.t to higher accuracy. C_1 and C_2 is that every traffic stream v_s has to select exactly one DNN model in a time interval t. C_3 ensures that the frame sampling rate is less than equal to the frame processing capacity of the DNN. This prevents traffic video frames from accumulating and increasing queueing delay. C_4: memory resources being used collectively by all the traffic stream in t should be within the capacity of the ES denoted by $COST(t)$. C_5: sum of allocated bandwidths to the group of S traffic streams = total uplink bandwidth available in the time interval t, denoted by $BW(t)$. C_6: average LTLL should be < maximum latency threshold. To find an optimal solution to O there are certain challenges: (a) requires network state and traffic video frame characteristics from the near future. (b) its a mixed integer non-linear programming problem, hence an online algorithm to adapt configurations and distribute bandwidth resources independent of future data.

IV. Algorithms

In this section, we present a detailed solution and theoretical analysis to tackle the challenges discussed above. To cater to the precondition of the LTLL, we break down the time-averaged utility maximization problem into a set of on-the-fly minimization problems using the Lyapunov framework. We propose an online lightweight algorithm that forms the adaption strategy based on real-time bandwidth information and characteristics of the video frames without requiring this information beforehand.

A. Using Lyapunov Optimisation (LO) for Problem Transformation

We introduce a virtual queue, whose length is a measure of the exceeded latency till any next point in time. The initial backlog $(VirQ(0))$ is assumed to be zero. The length of the queue as a function of time is described by the following recursive relation:

$$VirQ(t+1) = max(VirQ(t) + \sum_{s=1}^{S} Gl_s(t) - GL_{max}, 0) \quad (7)$$

Always prioritizing high accuracy burdens the virtual queue, causing delays and dissatisfied drivers on the road. We need a system that has queue stability and, in turn, also satisfies C_6. We use a quadratic Lyapunov function: $F(VirQ(t)) = \frac{1}{2}(VirQ(t))^2$, measure of congestion in the virtual queue. The value of the function is directly proportional to the queue backlog. We use single-interval Lyapunov drift to maintain the queue stability by constantly moving towards a less congested queue state.

$$\Delta(VirQ(t)) = E[F(VirQ(t+1)) - F(VirQ(t))|VirQ(t)] \quad (8)$$

$\Delta(VirQ(t))$: expected change in Lyapunov function over one-time interval, given a queue state at the starting of t; also $\Delta(VirQ(t)) \propto \frac{1}{queuestability}\ \Delta(VirQ(t))$ is included for latency queue stability into the trade-off between energy and accuracy, along which we use another term called Lyapunov drift-plus-penalty term:

$$\Delta(VirQ(t)) - \kappa.E[A_{avg}(t) - \Omega E(t)|VirQ(t)] \quad (9)$$

κ decides the trade-off between stabilizing the latency queue and maximizing the utility function. Instead of minimizing the drift-plus-penalty term for every time interval, we use the min-drift-plus-penalty algorithm in LO to minimize its upper bound which is stated by the following lemma. *Lemma 1*: The following statement is valid for all possible values of $VirQ(t)$ over all possible configurations and time intervals.

$$\begin{aligned}
\Delta(VirQ(t)) - \kappa.E[A_{avg}(t) - \Omega E(t)|VirQ(t)] \leq \\
C + VirQ(t)E[(Gl_{Avg}(t) - GL_{max})|VirQ(t)] - \\
\kappa.E[A_{avg}(t) - \Omega E(t)|VirQ(t)]
\end{aligned} \quad (10)$$

where $C = \frac{1}{2}(gl_{max} - GL_{max})^2$ is a constant and $gl_{max} = max_{t\in\tau} Gl_{Avg}(t)$, the maximum average global latency considered over all the time intervals. The new problem O_1 can be formulated as follows:

$$\begin{aligned}
O_1 : min\left\{d, bw, fps\right\} VirQ\left(t\right).Gl_{Avg}\left(t\right) - \\
\kappa.E[A_{avg}\left(t\right) - \Omega E\left(t\right)
\end{aligned} \quad (11)$$

under the constraints C_1,C_2,C_3,C_4,C_5. O_1 is solved using the current information whereas, O requires future information to be solved. $VirQ(t).Gl_{Avg}(t)$ considers average global latency due to data transmission and model performance in t. Accordingly, per frame average global latency minimization has become important because $VirQ(t)$ becomes large. In a nutshell, "During insufficient bandwidth, degraded configuration should be preferred, offsetting the latency increase within the thresholds. The latency queue is independent of future information, thereby satisfying the LTLL constraint and aiding in real-time decisions.

B. Online algorithm based on Lyapunov optimization

Next, the problem O_1 is combinatorial and hence is NP-hard, which makes it unsolvable in polynomial time. Hence, we devise a synergistic service placement and cost minimization (SSPCM) algorithm based on the Markov approximation method to solve this problem. We use: $d(t)$- set $\{d(s,i,t)|\forall p_i \in P, \forall v_s \in V\}$, possessing model selection variables, $fps(t)$- set $fps_s(t)|\forall v_s \in V$ holding the frame rate selection for all traffic streams, and $bw(t)$- set $\{bw_k(t)|\forall v_s \in V\}$ Fixing $d(t)$ results in solving two issues: bandwidth optimization for reduced GAL, and frame rate assignment for configuration maximization. The configuration

Algorithm 1: The SSPCM Algorithm

Data: $VirQ(0) = 0, e^s_{loc}, e^s_{trans}, GL_{max}$,COST

for $t = 0$ *to* T **do**
- Update the accuracy functions : $A^f_{s,t}(fps), A^r_{s,t}(res), \forall s$;
- Obtain $d(t), fps(t), bw(t)$ by solving O_1 using *Algorithm 2*
- Update the virtual queue as:
- $VirQ(t+1) = max(VirQ(t) + \sum_{s=1}^{S} Gl_s(t) - GL_{max}, 0)$

end

Algorithm 2: Single interval optimization for *SSPCM*

Data: $A^r_{s,t}(res), A^f_{s,t}(fps), e^s_{loc}, e^s_{trans}, GL_{max}$,$COST$, initial model selection vector $d(t)$

while T_{max} *iterations have been completed or* $|O'val - Oval| < 0.01$ *for more than 10 iterations* **do**

end

Pick a traffic stream v_s at random and update its model selection vector from $d(s,t)$ to $d'(s,t)$ after selecting a new model p';

if $d^{'}(s,t)$ *is feasible* **then**
- $d^{'}(t) \leftarrow \left\{d(1,t), d(2,t), \ldots, d^{'}(s,t), \ldots d(S,t)\right\}$;
- Get the value of $bw'(t)$ by solving problem O_2;
- Get the value of $fps'(t)$ by solving problem O_3;
- **Update** $\psi \leftarrow \frac{1}{1+e^{\frac{(O'val - Oval)}{h}}}$
- With probability ψ:
- $d(t) \leftarrow d'(t)$,
- $bw(t) \leftarrow bw'(t)$,
- $fps(t) \leftarrow fps^{'}(t)$;
- With probability 1- ψ:
- No updates performed

end

return $\{d(t), fps(t), bw(t)\}$

utility function is unaffected by the dynamic bandwidth distribution. Hence, the bandwidth distribution is only to minimize the GAL denoted as $Gl_Avg(t)$ in the time interval t. If we fix $d(t)$, then we have to focus on solving two problems which are optimizing bandwidth distribution to reduce GAL and assigning frame rates to different traffic streams for maximising the configuration utility. Bandwidth distribution has no role to play in the configuration utility function. Hence the only purpose of bandwidth distribution is to minimize the $Gl_Avg(t)$. With the main aim of minimizing the global latency, we formulate the BW allocation as a new problem: $O_2 : min\{bw(t)\} VirQ(t).Gl_{Avg}(t)$ The optimal bandwidth distribution is given by the following due to the *Karush-Kuhn-Tucker condition (KKT)* condition satisfied by the solution of this problem :

$$Opt - bw_s(t) = \frac{\sqrt{\sum_{i=0}^{N} kd(s,i,t)res_i}}{\sum_{s=1}^{S}\sqrt{\sum_{i=0}^{N} kd(s,i,t)res_i}} \tag{12}$$

We just solved $bw(t)$, and $d(t)$ is pre-fixed. Since the frame sampling rate allocation rate impacts only the configuration accuracy, the second problem can be solved by maximizing the utility function.

$$O_3 : max\{fps(t)\} A_{avg}(t) - \Omega E(t) \tag{13}$$

We take the accuracy function as a function of the frame rate for traffic stream v_s in time interval t as $k_1 - k_2 e^{\frac{-fps_s(t)}{k_3}}$ with k_1, k_2 and k_3 being coefficients of the constants. The dependency between frame rate and accuracy can be modeled as a concave function is well justified. O_3 has a global maxima and the optimal frame sampling rate which achieves it is given as:

$$Opt - fps_s(t) = \left\{ -k_3 log\left(\frac{\Omega e^s_{loc} k_3}{k_2 A^r_{s,t}(\sum_{i=0}^{N} kd(s,i,t)res_i}\right)\right.$$
$$\left.\left((-1 - d(s,0,t))k_3 log\frac{\Omega e^s_{trans} k_3 k \sum_{i=1}^{N}(d(s,i,t)res_i)^2}{k_2.A^r_{(s,t)}\sum_{i=0}^{N} kd(s,i,t)res_i)}\right)\right. \tag{14}$$

We can infer that O_2 and O_3 are solvable once the optimal model selection $d(t)$ is found. But the problem of optimal model selection can not be solved in polynomial time because it is a mix-integer nonlinear problem due to the model selection

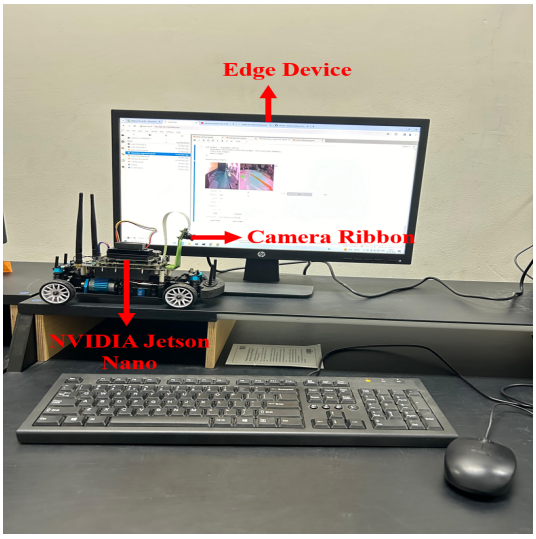

Figure 3: Testbed setup

variables taking binary values. Hence, we will instead solve the problem by finding a good approximation of the optimal solution for model selection using Markov approximation. The online control SSPCM algorithm 1 divides time into T time intervals. The accuracy profiler updates the accuracy models for all traffic streams according to the frame content. The optimal configuration and the bandwidth distribution are found by solving the problem O_1. The latency queue is updated using the average latency for that particular time interval. Problem O_1 is solved using the single-interval optimization for SSPCM as described in *Algorithm* 2. It is an online algorithm that depends upon the dynamically changing bandwidth information

along with the traffic stream video. This helps in deriving a configuration adaptation strategy without relying on global contextual information over a long time.

$d(s,t) = d(s,1,t), d(s,2,t), \ldots, d(s,N,t)$ is the model selection vector for the traffic stream v_s in time interval t. In SSPCM, first, a traffic stream v_s randomly selects a new DNN model p' while, for other traffic streams, the model selection vector remains the same. Similarly, we obtain a new model selection vector for all traffic streams named we obtain a new model selection vector for all traffic streams named $d^{'}(ts)$. Subsequently, now $fps'(t)$ and $bw'(t)$ can be obtained by solving the problems O_2 and O_3. O'_{val} is obtained as new value of the objective function while O_{val} is the value before the update using parameters $\{d(t), fps(t), bw(t)\}$. In the current time interval, the model selection vector for the traffic stream v_s is updated to p' with a probability ψ dependent on $O'_{val} - O_{val}$, conditioned on: ψ is higher when the new configuration $\{d'(t), fps'(t), bw'(t)\}$ results in a lower value of the objective function. This process continues till the maximum number of iterations T_{max} have been reached or $|O'_{val} - O_{val}| < 0.01$ for more than ten iterations. The parameter h mentioned in ψ formula in *Algorithm 2* is the smoothness parameter to establish a trade-off between exploration and exploitation or the amount of randomness in this process. If h is small, the new decision is retained with a larger probability else, if h reduces the objective value, it promotes exploitation. Resultantly, the algorithm takes many iterations to find the global optimum as it may be stuck around a local optimum. If h is large, it promotes exploration, and the new decision is kept with a lower probability if it is better compared to the current decision. This will reduce the number of iterations it takes for the algorithm to converge. A suitable h value ensures that the *Algorithm 2* converges at a super-linear rate.

V. Experimental Settings

A. Testbed Setup

We have used the camera ribbon of Arducam shown in *Figure* 3 having 8MP IMX219 Camera Module for NVIDIA Jetson Nano, Fixed Focus, edge device having 128-core NVIDIA Maxwell GPU Quad-core ARM A57 CPU 4 GB 64-bit LPDDR4 10/100/1000BASE-T Ethernet. We take five resolutions *320p, 400p, 540p, 700p and 1100p* respectively. The network bandwidth has been varied from 25Mbps to 120Mbps for simulations. For simplicity, we take $e^s_{loc} = 5J/frame$ and $e^s_{trans} = 0.510^(- 5)(J)$ for all traffic streams. We compare SSPCM to three benchmark algorithms: *Inflexible scheme* where traffic stream users always select the most expensive configuration, *Delay-centric* which minimizes the average latency in every interval ignoring the energy consumption and accuracy, *Delay-myopic* does not require future information for satisfying the LTLL constraint and imposes it in every time interval. Furthermore, the proposed DNN model is tested on our curated dataset having varied adversarial scenarios. We also present a comparative study with YOLO SOTA models.

B. A snapshot of the Algorithm in execution

Figure 4- 6 depicts how bandwidth share, resolution and frame rate vary with the changing bandwidth and changing characteristics of the traffic video content in smart consumer applications.

C. Analysing the effect of various parameters

1) Convergence: : The convergence of the *Algorithm 2* can be seen in *Figure* 7, which shows the objective value as a function of a number of iterations of *Algorithm 2*. A small value of h leads to an earlier convergence. At $h = 0.05$, *Algorithm 2* converges within 10 iterations. But reducing h too much, leads to faster convergence and might identify a local optimum instead of a global one. If $h \to \infty$, the algorithm keeps on exploring different possible solutions and never converges. A suitable h value ensures that the *Algorithm 2* converges at a super-linear rate.In our experiment, $h = 0.1$ is clearly the best value as it needs the lowest number of iterations.

2) Accuracy and latency trade-off: : *Figure* 8 shows the variation in accuracy of SSPCM between the 800th and 1000th time intervals for different values of GL_{max}. As GL_{max} increases, the fluctuation of the accuracy about the median decreases. This is because for a smaller GL_{max}, delay constraint can be easily violated, leading to huge drops in accuracy to compensate for it. But this is not the case for a large GL_{max} value because the accuracy doesn't have to be compromised a lot. κ also influences the trade-off between accuracy and latency. In *Figure* 9, the value of the queue backlog has been plotted as a function of the time interval for different values of the control parameter κ. All the backlogs converge to a certain value after some time, and LTLL can be satisfied if the number of time intervals increases. A greater value of κ needs more time intervals for convergence, while a smaller κ value makes the service latency smaller in a shorter duration of time.

3) Trade-off between accuracy and energy: : *Figure* 10 depicts the trade-off between time-averaged accuracy and time-averaged energy consumption for different values of the parameter Ω. Value of Ω from 0.001 to 0.002 results in a reduction of 20% in the energy consumption while the loss suffered in the accuracy of vehicle detection is only about 13%. Hence, a good choice of Ω will help save consumption of energy while resulting in only a minor loss in accuracy.

4) Comparison of Algorithm with benchmarks: Figure 11- 12 compare the system latency and accuracy, respectively, as a function of time for four different algorithms. For SSPCM, it can be seen that the algorithm gradually finds an optimal trade-off between latency and accuracy around the 50th time interval.

5) The effect of bandwidth: : In *Figure* 13 , all the algorithms except the Inflexible scheme have a higher accuracy when system bandwidth is increased because it allows for more expensive configurations to be used. The average system latency for both SSPCM and the Delay-myopic scheme is bounded. The impact of bandwidth on time-averaged system latency is shown in *Figure* 14. *Figure* 15 shows that the LTLL constraint is closely satisfied for SSPCM. *Table* I illustrates the performance of the SOTA models with the indicators average precision (AP) and inference time. The AP results are obtained by evaluating the models across all box sizes i.e., *small, medium, and large*. In the table, "Normal" denotes results without the application

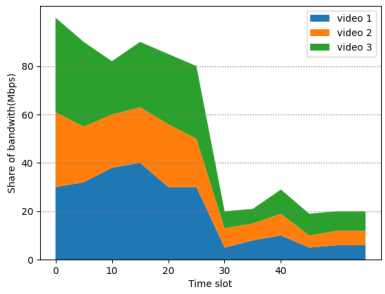

Figure 4: Bandwidth Runtime Inspection

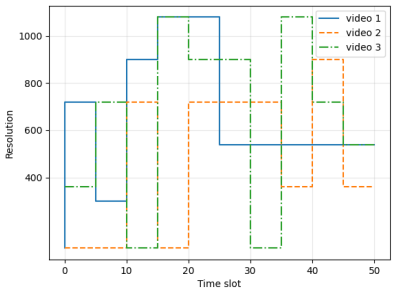

Figure 5: Resolution Runtime Inspection

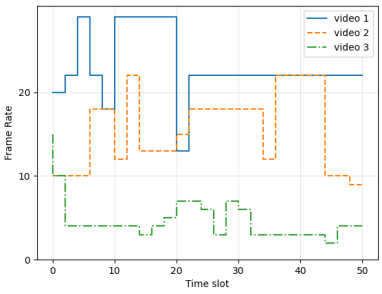

Figure 6: Frame Rate Runtime Inspection

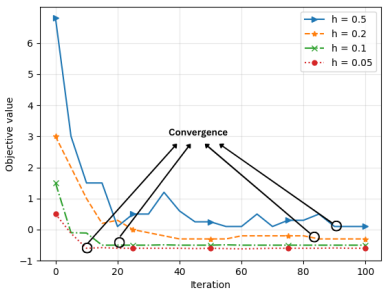

Figure 7: Algorithm 2 Convergence

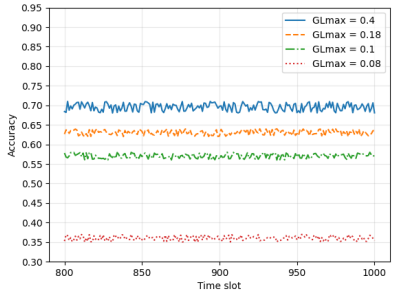

Figure 8: GL_{max} implications on accuracy

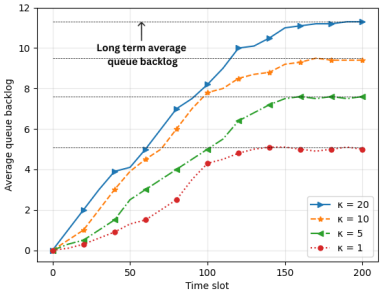

Figure 9: Impact of weight κ

Table I: Inference and precision results of the SOTA models

Model	$Params(M)$	GFLOPS	MAP_{75} (Inference)	MAP_{50}	Normal	AP_S	AP_M	AP_L
YOLOv7-Tiny	6.2	13.8	0.86	0.93	114	0.49	1.70	1.82
YOLOv6-small	18.51	19.09	0.81	0.5	32.37	0.51	0.75	0.88
YOLOv6-Nano	4.3	0.77	0.86	0.93	67	0.50	0.77	0.85
YOLOv5-Nano	1.8	0.76	0.85	0.91	45.1	0.55	0.75	0.75
YOLOv5-small	4.4	0.69	0.74	0.91	32.37	0.4	0.62	0.80
EMulTO	6.1	10.1	0.89	0.93	**30**	5.2	**0.79**	**0.80**

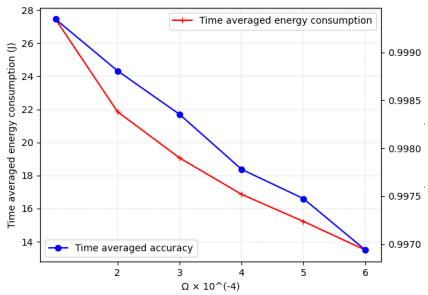

Figure 10: Accuracy and energy tradeoff

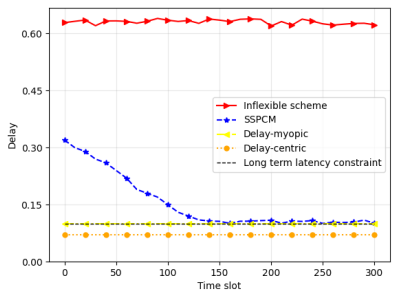

Figure 11: Delay variation

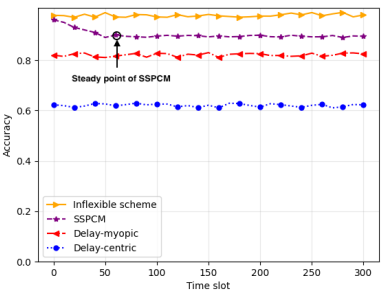

Figure 12: Accuracy variation

of TensorRT optimization. FP16 precision was utilized for the TensorRT iterations. All models were assessed with an input size of 640x640. YOLOv7-Tiny demonstrates superior performance across all threshold values. However, the overall average precision values among all models are closely aligned, excluding the YOLOX variants. Notably, both the tiny and nano versions of YOLOX yield the lowest average precision results, excluding AP_50. While YOLOv7 tiny and nano models boast the highest precision results, their mean confidence scores are the lowest. Conversely, among the tiny models, YOLOv7-Tiny and among the nano models, YOLOv5-Nano and YOLOX-Nano exhibit the highest mean confidence outcomes.

6) Inference Time: - In the context of real-time applications, especially on edge GPU devices, inference time is considered a crucial metric delineating model performance. The inference time experiments encompass the total time taken for model

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3442437

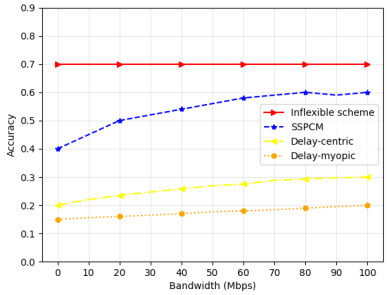

Figure 13: Accuracy vs bandwidth

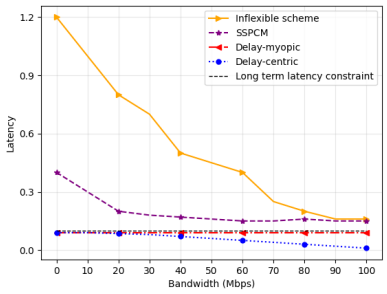

Figure 14: Latency vs bandwidth

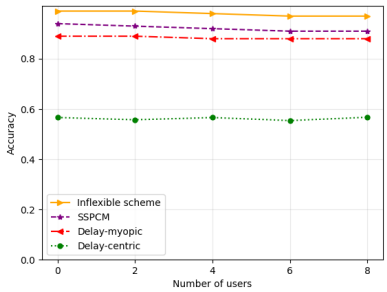

Figure 15: Impact of population

inference and non-max suppression (NMS). *Table* I presents the baseline inference results. Notably, the YOLOv5-Nano model emerges as the fastest, achieving an inference time of 47.6 ms. Among the tiny models, YOLOv6-Tiny demonstrates the shortest inference time at 109ms. Upon conversion to TensorRT, the performance of YOLOv5-Nano and YOLOv6-Nano models becomes closely aligned, with YOLOv6-Nano slightly edging out YOLOv5-Nano by reducing its inference time by approximately 47%. Conversely, the TensorRT-converted YOLOv7-Tiny model exhibits a lower inference time compared to other tiny models. A comparison between the inference times of normal models and their TensorRT-optimized counterparts reveals a significant reduction in inference times with TensorRT optimization. This optimization notably narrows the gap between tiny and nano models, showcasing its efficacy in enhancing real-time inference performance. Table 2 illustrates that YOLOv6-Tiny excelled in achieving the best results for both small and large-sized boxes. On the other hand, YOLOv7-Tiny and YOLOv6-Nano models emerged as the top performers for medium-sized boxes.

D. Energy Consumption

NVIDIA provides the Tegra stats utility which aids in recording the power consumption of the device. The log files are produced which measures the current power incurred by the device. It is found that EMulTO consumes least amount of energy which is comparable with YOLOv5-nano. Except YOLOv7-tiny, the other small and tiny models consume the maximum amount of energy as compared to nano models. By this observation, we can conclude that EMulTO, YOLOv5-nano, YOLOv6-nano, YOLOv7-tiny are fit to be deployed in real-time applications. We also conclude that the memory consumption is not the determining factor as it hardly varies among the models, however, accuracy is deemed as an important parameter for AV applications.

VI. Conclusion

The main objective of this work is to establish an eco-friendly transportation infrastructure for the consumer industry using imaging and V2X technology and to reduce the pollution because of traffic and accidents. In this regard, we have proposed a green-EMulTO that particularly addresses perception-based AV safety aided with seamless V2X service connectivity. We have devised a lightweight DNN model for resource-constrained edge devices on roads, which extracts the most informative frames and a decision initiator to conditionally retrain the model. Keeping in mind the LTLL constraint, we have proposed an online algorithm that selects suitable configurations for traffic video streams according to the bandwidth conditions. It also accounts for analytics accuracy, energy consumption and global latency. We have also deployed this orchestrator on the NVIDIA prototyped testbed. Extensive experimental results have proved the superiority of green-EMulTO in terms of various indexes, as shown in Figures 4-15. However, there are some limitations to the proposed algorithm. The algorithm requires fewer duplicate frames and more filtered data. There should be a keyframe filtering technique to eliminate the redundancy of the traffic streams. Moreover, the algorithm may not generalize well. In our future work, we will consider green-EMulTO to be tested with a wide range of edge devices and scaling it up to multi-vehicle cooperation, intending to achieve interruption-aware perception under LTLL. We also aim to optimize the models during training for achieving better accuracy-latency tradeoff, thereby, envisaging having newly introduced edge devices for vehicular communications and testing the generalizability of our model.

References

[1] Jing Yang, Qinghua Ni, Guiyang Luo, Qi Cheng, Latifa Oukhellou, and Shuangshuang Han. A trustworthy internet of vehicles: The dao to safe, secure and collaborative autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 2023.

[2] Ashu Taneja and Shalli Rani. Quantum-enabled intelligent resource control for reliable communication support in internet-of-vehicles. *IEEE Transactions on Consumer Electronics*, 2024.

[3] Yunlong Li, Xinfeng Zhang, Chen Cui, Shanshe Wang, and Siwei Ma. Fleet: Improving quality of experience for low-latency live video streaming. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(9):5242–5256, 2023.

[4] Junhua Wang, Kun Zhu, and Ekram Hossain. Green internet of vehicles (iov) in the 6g era: Toward sustainable vehicular communications and networking. *IEEE Transactions on Green Communications and Networking*, 6(1):391–423, 2021.

[5] Rahul Yadav, Weizhe Zhang, Omprakash Kaiwartya, Houbing Song, and Shui Yu. Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing. *IEEE Transactions on Vehicular Technology*, 69(12):14198–14211, 2020.

[6] Zhenyu Wen, Renyu Yang, Bin Qian, Yubo Xuan, Lingling Lu, Zheng Wang, Hao Peng, Jie Xu, Albert Y Zomaya, and Rajiv Ranjan. Janus: Latency-aware traffic scheduling for iot data streaming in edge environments. *IEEE Transactions on Services Computing*, 2023.

[7] Yuxin Kong, Peng Yang, and Yan Cheng. Edge-assisted on-device model update for video analytics in adverse environments. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 9051–9060, 2023.

[8] Yan Cheng, Peng Yang, Ning Zhang, and Jiawei Hou. Edge-assisted lightweight region-of-interest extraction and transmission for vehicle perception. In *GLOBECOM 2023-2023 IEEE Global Communications Conference*, pages 1054–1059. IEEE, 2023.

[9] VD Ambeth Kumar, Manish Raghuraman, Abhishek Kumar, Mamoon Rashid, Saqib Hakak, and M Praveen Kumar Reddy. Green-tech cav: Next generation computing for traffic sign and obstacle detection in connected and autonomous vehicles. *IEEE Transactions on Green Communications and Networking*, 6(3):1307–1315, 2022.

[10] Ganesh Gopal Devarajan, U Kumaran, Gopalakrishnan Chandran, Rajendra Prasad Mahapatra, and Ahmed Alkhayyat. Next generation imaging methodology: An intelligent transportation system for consumer industry. *IEEE Transactions on Consumer Electronics*, 2024.

[11] Francesc Guim, Thijs Metsch, Hassnaa Moustafa, Timothy Verrall, David Carrera, Nicola Cadenelli, Jiang Chen, David Doria, Chadie Ghadie, and Raül González Prats. Autonomous lifecycle management for resource-efficient workload orchestration for green edge computing. *IEEE Transactions on Green Communications and Networking*, 6(1):571–582, 2021.

[12] Mehrdad Khani, Ganesh Ananthanarayanan, Kevin Hsieh, Junchen Jiang, Ravi Netravali, Yuanchao Shu, Mohammad Alizadeh, and Victor Bahl. {RECL}: Responsive {Resource-Efficient} continuous learning for video analytics. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 917–932, 2023.

[13] Dawei Li, Serafettin Tasci, Shalini Ghosh, Jingwen Zhu, Junting Zhang, and Larry Heck. Rilod: Near real-time incremental learning for object detection at the edge. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, pages 113–126, 2019.

[14] Bingyi Liu, Yang Sheng, Xun Shao, Yusheng Ji, Weizhen Han, Enshu Wang, and Shengwu Xiong. Collaborative intelligence enabled routing in green iov: A grid and vehicle density prediction based protocol. *IEEE Transactions on Green Communications and Networking*, 2022.

[15] Qian Liu, Rui Luo, Hairong Liang, and Qilie Liu. Energy-efficient joint computation offloading and resource allocation strategy for isac-aided 6g v2x networks. *IEEE Transactions on Green Communications and Networking*, 7(1):413–423, 2023.

[16] Romil Bhardwaj, Zhengxu Xia, Ganesh Ananthanarayanan, Junchen Jiang, Yuanchao Shu, Nikolaos Karianakis, Kevin Hsieh, Paramvir Bahl, and Ion Stoica. Ekya: Continuous learning of video analytics models on edge compute servers. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 119–135, 2022.

[17] Mehrdad Khani, Pouya Hamadanian, Arash Nasr-Esfahany, and Mohammad Alizadeh. Real-time video inference on edge devices via adaptive model streaming. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4572–4582, 2021.

[18] Can Wang, Sheng Zhang, Yu Chen, Zhuzhong Qian, Jie Wu, and Mingjun Xiao. Joint configuration adaptation and bandwidth allocation for edge-based real-time video analytics. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 257–266. IEEE, 2020.

[19] Mengxi Hanyao, Yibo Jin, Zhuzhong Qian, Sheng Zhang, and Sanglu Lu. Edge-assisted online on-device object detection for real-time video analytics. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.