Evidence-QoE-Aware Video Delivery for Real-Time AI Video Chat

Hequn Wang, Wei Bao, Ruoyu Wu, Xinyi Sheng, Bing Bing Zhou, and Albert Y. Zomaya
School of Computer Science, the University of Sydney, Sydney, NSW, Australia
hwan0565@uni.sydney.edu.au, {wei.bao, ruoyu.wu, xinyi.sheng, bing.zhou, albert.zomaya}@sydney.edu.au

***Abstract*—Real-time AI video chat implements vision-language models (VLMs) to answer user queries from live video streams, but it is constrained by response deadlines and wireless uplink bandwidth between the edge device and the VLM server. Existing VLM streaming systems mainly optimize VLM-side visual memory and reasoning, while assuming that video inputs arrive at the VLM server in time. Under bandwidth-limited uplinks, preserving high-quality video transmission increases the queueing delay of the evidence frames and may delay the response. Moreover, frequent changes in high-quality video regions introduce bursts in payload sizes, further degrading response accuracy and timeliness. This paper presents EQAVC, an evidence-QoE-aware video delivery framework for real-time AI video chat. EQAVC models edge AI video chat as a closed-loop evidence delivery system, where the video device uses the server-side semantic feedback to prioritize the quality of important video regions by optimizing the quantization parameter (QP) of corresponding frame blocks. To make effective QP decisions, we design an Evidence-QoE objective that captures the quality of semantic feedback, latency between frame generation and processing, and smoothness of switching high-quality video regions. Based on the Evidence-QoE objective, EQAVC employs a lightweight QP controller to convert delayed semantic feedback into adaptive encoding decisions regarding different video regions. Experiments on real video-chat question-answering (QA) sessions with server-side VLM feedback show that EQAVC outperforms benchmarks with respect to both response accuracy and latency.**

***Index Terms*—Real-time AI Video Chat; Vision-language Models; Semantic-aware Video Encoding**

I. Introduction

Recent advances in wireless networking and mobile video systems have made low-latency visual communication increasingly accessible across mobile sensing [1], video surveillance [2], remote monitoring [3], and interactive video communication [4]. Meanwhile, the rapid development of vision-language models (VLMs) is expanding the audience of real-time video streams beyond human viewers to include AI agents [5], [6]. In a real-time AI video chat system, a camera-equipped edge device continuously captures video inputs and streams them to a server-side VLM, which incrementally updates its visual memory and generates answers in a timely manner, based on the evidence in the video frames available at response time.

Real-time AI video chat and VLM streaming systems have shown the potential for interactive visual reasoning over continuously arriving video frames [7]–[10]. These systems mainly focus on how the server-side model maintains visual memory, samples frames, and generates responses from live video streams. However, in practical edge deployments, the video stream must first pass a constrained and fluctuating wireless uplink before it can be processed by the server-side VLM. When the bitrate of encoded video exceeds the available uplink capacity, congestion and queueing may delay important evidence frames [11]. Therefore, to provide timely responses in the real-time AI video chat, it is a fundamental challenge to deliver high-quality visual evidence under limited and fluctuating uplink bandwidth.

A straightforward approach to adapt timely video delivery to constrained uplink bandwidth is to reduce bandwidth demand by uniformly reducing the encoding quality of the entire video stream. However, reducing the quality of video regions containing critical evidence for visual reasoning leads to degraded response quality. This observation aligns with prior studies on machine-centric video analytics systems, which have shown that full-frame high-quality video is often unnecessary for model consumption; instead, video configurations, regions, patches, or blocks can be adapted according to downstream task accuracy and resource cost [12]–[17]. Motivated by these studies and the challenges of AI video chat system, we implement evidence-aware block-level quantization parameter (QP) control for real-time AI video chat, where the encoder allocates higher quality to regions that are more likely to contribute to the VLM reasoning and fewer bits to less informative regions.

Although semantic feedback from the VLM provides information about the evidence value of different video regions, setting the encoding quality of these regions solely based on such feedback may fail to ensure both accurate and timely VLM responses. First, assigning a lower QP, i.e., higher encoding quality, to evidence-bearing video regions improves their fidelity but also increases the payload size of the encoded video frame [21]. This may delay the frame until a later VLM reasoning round and consequently delay the response. Second, abruptly adjusting the QP for video regions introduces bitrate bursts over payload sizes [22], which may congest the uplink bandwidth and delay the arrival of subsequent video frames at the VLM. Therefore, real-time AI video chat requires evidence-aware QP control that jointly considers evidence fidelity, timely transmission, and smooth QP adjustment.

To address these challenges, we propose EQAVC, an **E**vidence-**Q**oE-Aware video delivery framework for real-time **A**I **V**ideo **C**hat. EQAVC models the AI video chat as an evidence delivery system, where (i) the edge device uploads

979-8-3195-0167-7/26/$31.00 ©2026 IEEE

TABLE I: Positioning of related systems along QoE targets and optimization mechanisms for real-time AI video interaction. ✓: explicitly optimized; △: considered or indirectly involved; –: not addressed.

Related systems	Quality			Timeliness			Stability	
	Task / response accuracy target	Model-side optimization	Edge evidence control	Transport latency	Server inference / token buffer	Evidence deadline / response round	Response-time stability	Evidence-control stability
VideoLLM-online [7], Dispider [9], StreamChat [8], StreamMind [10]	✓	✓	–	–	✓	△	△	–
DDS [14], AccMPEG [15], RegenHance [16], VaVLM [18]	✓	△	✓	✓	△	–	△	–
Chat with AI [19], Artic [20]	✓	△	✓	✓	–	△	△	–
Ours	✓	–	✓	△	–	✓	△	✓

the encoded video frame to the VLM at the edge server, (ii) server-side VLM maintains and updates the visual memory, and returns semantic feedback about video regions that may contain visual evidence, (iii) the edge device incorporates the semantic feedback by adjusting the QP for different blocks of the video frame, and then encodes the video accordingly for transmission. EQAVC is a system with two-layer loops: (i) Each round of the outer loop starts from a question being submitted to the edge server along with the first video frame, and ends with the response being returned; (ii) The inner loop consists of one or more VLM reasoning rounds, in which the VLM first waits for a predefined period or until the response deadline, then sends back the semantic feedback or the final response.

A key principle of EQAVC is that semantic feedback should not be blindly translated into encoding quality of video regions. The returned semantic feedback may be uncertain (i.e., with low confidence) and outdated (i.e., based on a much earlier frame), so following it directly may waste uplink bandwidth for unimportant video blocks. Building on this principle, EQAVC utilizes a lightweight QP controller, called QoE-QP controller, to determine the encoding quality of video frames. When a new video frame is generated, the QoE-QP controller first uses the latest semantic feedback to identify background blocks that likely contain little evidence and protected blocks that may contain more evidence. Then, the QoE-QP controller assigns a conservative QP to background regions to reduce bitrate usage. For protected regions, it evaluates a set of candidate QPs under the Evidence-QoE objective and selects the one that contributes most while still supporting timely transmission. This design ensures high-quality visual evidence is supplied to the VLM in a timely manner and enables accurate and timely VLM responses.

To validate the performance of EQAVC with the QoE-QP controller, we implement EQAVC using a practical video encoder and evaluate it on real video-chat QA sessions with VLM semantic feedback. The results show that EQAVC achieves higher accuracy and lower latency than the rate-adaptive QP and greedy evidence-QP benchmarks. These performance gains are driven by a lower evidence-miss rate and fewer payload bursts, validating the effectiveness of the EQAVC framework and the QoE-QP controller in supporting real-time AI video chat.

II. Background

Real-time AI video interaction has been studied across streaming video understanding, machine-centric video transmission, and AI-oriented real-time communication. These systems share the goal of improving interactive AI services under limited computation and network resources, but they optimize different parts of the edge-cloud pipeline. We organize related work around three dimensions that directly affect edge-cloud AI video chat: *evidence quality*, *evidence timeliness*, and *evidence stability (smoothness)*. Table I positions representative systems, including this study, regarding whether these dimensions are explicitly modeled in the control loop between the video device and server. In the rest of this section, we discuss these lines of work along with the above three dimensions and highlight the existing gap for evidence delivery in real-time AI video chat.

a) Evidence Quality: Traditional video streaming systems define QoE mainly based on perceptual quality and playback impairments, such as startup delay, rebuffering, and quality fluctuation [23]. When the receiver becomes a machine learning model, video delivery shifts from perceptual playback to task-relevant evidence acquisition. Machine-centric systems therefore adapt video configurations, frame filtering, Region of Interest (RoI) retransmission, patch quality, or block-level compression according to downstream analytic accuracy and resource cost [12]–[17]. Recent VLM-based client-server and AI-oriented RTC systems further use model feedback, chat context, uncertainty, or response-important regions to guide visual evidence transmission [18]–[20], [24], [25].

b) Evidence Timeliness: Latency has also been addressed from both the transmission and model sides. Transmission-oriented systems reduce payload or adapt bitrate to alleviate uplink delay, while AI-oriented real-time chat (RTC) systems consider response latency under bandwidth fluctuation [14], [15], [19], [20]. Meanwhile, streaming Video-LLMs optimize model-side delay by changing how visual inputs are selected, buffered, updated, or consumed during online reasoning [7]–[10]. Such optimizations are necessary, but client-side QP control introduces a different timing challenge: a QP configuration changes the size of encoded payload and affects the arrival time of the payload at the server, while the server-side VLM often processes visual content only at pre-scheduled instants.

Therefore, evidence timeliness should be evaluated against the model-reasoning schedule, rather than average transport delay or bitrate reduction alone.

c) Evidence Stability: Most RoI-based delivery systems determine the quality of video regions as a per-frame or per-window decision: once a region is considered important, it is assigned higher fidelity [15]–[17], [19], [20]. This abstraction overlooks the control cost of live AI video chat. Semantic feedback about the RoI needs to be eventually incorporated as QP selection for video regions, so changing high-quality regions also changes the allocation of bitrate across different video frames. Lowering the QP (i.e., increasing the quality) of a region may cause a short-term payload burst. Based on these observations, this study views video delivery from the video device to the edge server as a closed-loop evidence delivery problem, where semantic evidence quality, schedule-aware timeliness, and smoothness of quality adjustment must be jointly considered.

III. System Design

A. System Overview

Fig. 1 gives an overview of the edge-cloud real-time AI video chat system EQAVC, where live visual evidence is continuously delivered from the edge to a server-side VLM and semantic feedback is returned to guide subsequent encoding decisions. EQAVC operates in a closed-loop manner with two-layer loops: (i) During the AI video chat, each round of the outer loop starts from a question being submitted to the edge server along with the first video frame, and ends with the response being returned, marking the beginning and the ending of an *Interaction Round*; (ii) Each interaction round consists of one or many inner loops, each of which corresponds to reasoning Round, where the VLM first waits for a predefined period or until the response deadline, then sends back the semantic feedback or the final response. At the video device, we implement a controller to determine the encoding quality (i.e., QP) for video frames, in order to preserve the evidence in the video under the constrained uplink, and thereby allow the VLM to generate an accurate answer within fewer reasoning rounds (Sec. III-B). To quantify the benefits of different QP decisions toward this purpose, we model it as an Evidence-QoE Objective, which is an optimization problem considering the impact of the encoding quality on both delayed arrival of evidence at the VLM (Sec. III-C) and smoothness of switching video quality (Sec. III-D). In Sec. III-E, we formalize the Evidence-QoE Objective that will guide the decision of our QoE-QP controller.

B. Closed-loop Semantic Evidence Model

We first formalize the semantic-feedback loop that provides the evidence demand for edge-side QP control. Consider one interaction round, where the user issues a query at time $t = 0$ and the edge device continuously captures a live

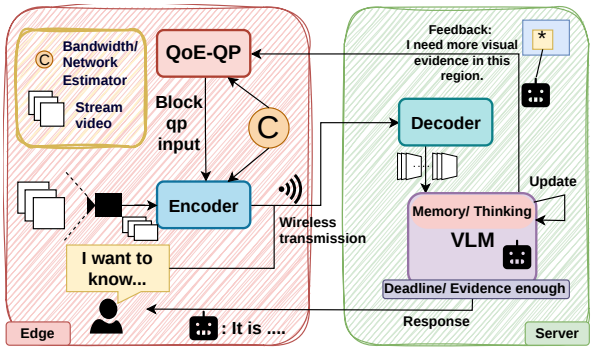


Fig. 1. EQAVC overview

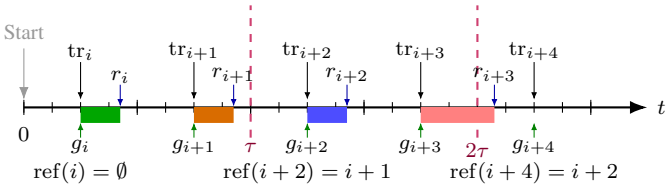


Fig. 2. Example of indexing the reference frame consumed by the reasoning round

video stream over $t \in \mathbb{R}_{\geq 0}$[1] Frames are generated every Δ_f seconds and are indexed by $i \in \{0, \ldots, I-1\}$, where I is the number of frames in the current interaction round, with generation time $g_i = i\Delta_f$. For each generated frame F_i, the edge chooses a block-level QP map, encodes the frame as compressed visual evidence, and uploads it to the edge server through a bandwidth-limited wireless uplink. At the frame-level scheduling step for F_i, the edge selects the QP map before encoding the frame for uplink delivery. We use tr_i as the timing reference for this scheduling step, and r_i to denote the time when F_i becomes available at the edge server. The interval between tr_i and r_i is the transmission delay, which will be formalized in detail in Sec. III-C; here, these symbols are introduced only to specify which feedback and frames are available to the semantic-feedback loop.

The edge server decodes received evidence upon arrival, and the VLM consumes the currently available visual context at scheduled update or reasoning times $t_k = k\tau$, $k \in \mathbb{Z}_{\geq 0}$, where τ is the visual-context consumption period. This scheduled processing model is motivated by practical streaming VLM pipelines, in which the full reasoning model is not continuously invoked upon the arrival of every frame. Instead, incoming visual inputs are sampled and aggregated into visual-context or memory updates. Therefore, the scheduled consumption times represent the reasoning-invocation points at

[1]We assume that the user query and the server-side semantic feedback can be represented by a compact control packet, and their transmission time is therefore ignored. We also assume the propagation delay is negligible in the system. The transmission time of each video frame is non-negligible and depends on both its encoded payload size and the currently available uplink throughput.

which newly arrived evidence can affect the model state. We model these consumption points using the periodic schedule $(0, \tau, 2\tau, \ldots)$; the formulation can also be extended to a non-uniform sequence of consumption times.

We define a hard deadline $D > 0$ for the current interaction round; visual evidence arriving after D is not admitted to the final reasoning process. Based on the current interaction state, the edge server either returns a response or sends semantic feedback to request additional visual evidence. The semantic feedback is generated at the end of a reasoning round, based on the frame processed. Fig. 2 illustrates the reference-frame indexing used by the feedback loop, and we use $\mathrm{ref}(i)$ to denote the index of the reference frame that is used to decide the encoding quality for frame F_i, and define it in two cases. (i) For the frames that are encoded before time τ, when the VLM can first generate and return semantic feedback, no semantic feedback is yet available at the video device. For these frames, we have $\mathrm{tr}_i < \tau$, and we define $\mathrm{ref}(i) = \emptyset$ for these frames. (ii) For frames F_i that are encoded after τ (i.e., $\mathrm{tr}_i \in [\tau, D]$), the video device will use the latest available semantic feedback to encode F_i. Such feedback is the one generated at the end of the last reasoning round ($\lfloor \frac{\mathrm{tr}_i}{\tau} \rfloor \tau$) by VLM analyzing the frame that had arrived most recently. As a result, for all i such that $\mathrm{tr}_i \in [\tau, D]$, the index of the reference frame of frame i is

$$\mathrm{ref}(i) \triangleq \max \left\{ j \in \{0, \ldots, i-1\} \;\middle|\; r_j \le \left\lfloor \frac{\mathrm{tr}_i}{\tau} \right\rfloor \tau \right\}, \quad (1)$$

where the maximum over an empty set is interpreted as $\emptyset$. Hence, $\mathrm{ref}(i) = \emptyset$ covers frames scheduled before the first visual reasoning time and cases where no previous frame has been consumed by the latest eligible reasoning round. This definition makes $\mathrm{ref}(i)$ independent of the candidate QP map currently being evaluated for F_i; it depends only on frames already transmitted before the scheduling step.

We denote the semantic RoI returned by the server-side VLM on the reference frame $F_{\mathrm{ref}(i)}$ as $\hat{\mathcal{R}}_i$, with its associated confidence score denoted by $\rho_i \in [0, 1]$. Since $F_{\mathrm{ref}(i)}$ may differ from the current frame F_i, $\hat{\mathcal{R}}_i$ may be spatially misaligned with F_i, and thereby it needs to be calibrated to the current frame F_i before using it as the reference RoI. Accordingly, we denote the calibrated RoI by $\bar{\mathcal{R}}_i$. If $\mathrm{ref}(i) = \emptyset$, or if the corresponding reasoning round returns no valid semantic RoI, we set $\hat{\mathcal{R}}_i = \bar{\mathcal{R}}_i = \emptyset$ and $\rho_i = 0$. Otherwise,

$$\bar{\mathcal{R}}_i = \mathcal{T}_{\mathrm{TM}}\big(\hat{\mathcal{R}}_i; F_{\mathrm{ref}(i)}, F_i\big), \quad (2)$$

where $\mathcal{T}_{\mathrm{TM}}$ is the template-matching operator based on the normalized cross-correlation coefficient [26].

Given the calibrated feedback $(\bar{\mathcal{R}}_i, \rho_i)$, the video device maps the server-side evidence demand to block-level semantic weights before selecting the QP map. The video device partitions each frame F_i into a fixed grid with N_h blocks along the height dimension and N_w blocks along the width dimension, where $N = N_h N_w$. Let $\mathcal{B}_n$ denote the n-th block, $n \in \{1, \ldots, N\}$, and let $A_n = |\mathcal{B}_n|$ be its area. For each block, the edge selects a QP level $x_{i,n}$ from a feasible set $\mathcal{X} = \{x_{\min}, \ldots, x_{\max}\}$, where a smaller QP means higher visual fidelity (following the video-coding convention). The QP map of frame F_i is denoted by $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,N})$. We measure the semantic importance of each block according to its overlap with the calibrated semantic RoI. Let $A^{\mathrm{o}}_{i,n} = |\mathcal{B}_n \cap \bar{\mathcal{R}}_i|$ be the overlap area between block $\mathcal{B}_n$ and $\bar{\mathcal{R}}_i$. Since the server-side feedback may be uncertain, we combine the RoI-based importance with a spatial prior:

$$\tilde{w}_{i,n} = \rho_i \frac{A^{\mathrm{o}}_{i,n}}{\sum_{j=1}^{N} A^{\mathrm{o}}_{i,j}} + (1 - \rho_i) \frac{A_n}{\sum_{j=1}^{N} A_j}. \quad (3)$$

The normalized semantic evidence weight is $w_{i,n} = \frac{\tilde{w}_{i,n}}{\sum_{j=1}^{N} \tilde{w}_{i,j}}$. If no valid semantic RoI is returned, the edge sets $\rho_i = 0$, and the weight falls back to the spatial prior. For equal-size blocks, the spatial prior becomes $1/N$.

Following [20], we define the normalized evidence fidelity of a block under QP x as $V(x) = \left(\frac{x_{\max} - x}{x_{\max} - x_{\min}} \right)^{\frac{1}{2}}$, $V(x) \in [0, 1]$. The semantic evidence quality acquired from frame F_i is then

$$E_i(\mathbf{x}_i) = \sum_{n=1}^{N} w_{i,n} V(x_{i,n}). \quad (4)$$

This value measures how much useful visual evidence is delivered to the edge server under the current QP map, rather than the perceptual quality of the whole frame.

C. Payload and Schedule-aware Evidence Latency

The QP map affects the closed-loop system by changing the payload (i.e., the encoded frame) size that needs to be transmitted by the uplink, which further determines evidence arrival time. Therefore, we do not treat bitrate as the final QoE objective, but use the encoded payload as the system interface between block-level QP control and schedule-aware evidence latency. For block $\mathcal{B}_n$ in frame F_i, we model the encoded payload contribution under QP x as $u_{i,n}(x) = \alpha_{i,n} \exp\big[-\beta(x - x_{\min}) \big]$, where $\alpha_{i,n}$ is the rate-scale coefficient of block $\mathcal{B}_n$ in the rate–QP payload predictor, and $\beta > 0$ is the rate-QP sensitivity coefficient. A smaller QP produces higher visual fidelity but also occupies more uplink resources. The total visual payload generated by frame F_i is

$$U_i(\mathbf{x}_i) = b_i^{\mathrm{base}} + \sum_{n=1}^{N} u_{i,n}(x_{i,n}), \quad (5)$$

where b_i^{base} denotes the QP-independent payload, including headers and other non-controlled coding overhead. This model is only used to estimate how the selected QP map affects uplink occupancy and evidence arrival.

In streaming AI video chat, evidence latency should be measured against the server-side consumption schedule rather than by transmission time alone. Let tr_i denote its uplink transmission start time and r_i denote its arrival time at the edge server. Since the uplink transmits frames sequentially, frame F_i can start transmission only after it has been generated and after the previous frame has arrived at the edge server, i.e., $\mathrm{tr}_i = \max\{g_i, r_{i-1}\}$. If there is no accumulated

uplink backlog before frame F_i, then $r_{i-1} \le g_i$ and hence $\mathrm{tr}_i = g_i = i\Delta_f$; otherwise, it is transmitted as soon as the previous frame is completed, and $\mathrm{tr}_i = r_{i-1}$. When frame F_i is generated, the edge device knows the predicted usable uplink throughput during the transmission of frame F_i, which is denoted as C_i. The transmission duration of frame F_i under QP map $\mathbf{x}_i$ and the predicted uplink throughput C_i is modeled as

$$\ell_i(\mathbf{x}_i) = \tau_i^0 + \frac{U_i(\mathbf{x}_i)}{C_i}, \tag{6}$$

where τ_i^0 includes decision-independent delay such as packetization. The server arrival time of frame F_i is then $r_i = \mathrm{tr}_i + \ell_i(\mathbf{x}_i)$.

Given the arrival time r_i, the first server-side consumption time that can include frame F_i is $c_i = \left\lceil \frac{r_i}{\tau} \right\rceil \tau$. Let $d_i = \left(\left\lfloor \frac{g_i}{\tau} \right\rfloor + 1\right)\tau$ denote the first possible scheduled visual-context consumption deadline after frame F_i is generated. If $r_i \le d_i$, F_i enters the intended reasoning or memory-update round, whereas $r_i > d_i$ makes it wait until c_i, and arrivals after D cannot affect the final answer. The schedule-aware latency penalty is defined as

$$L_i(\mathbf{x}_i) = [r_i - d_i]^+ + (c_i - r_i)\mathbb{I}(r_i > d_i), \tag{7}$$

where $[z]^+ = \max\{z, 0\}$ and $\mathbb{I}(\cdot)$ is the indicator function. When $r_i \le d_i$, $L_i(\mathbf{x}_i) = 0$. When $r_i > d_i$, the penalty becomes the extra schedule delay caused by missing the intended consumption point.

D. Switching-induced Evidence Instability

Beyond per-frame evidence quality and schedule-aware latency, feedback-driven QP control also introduces temporal instability across consecutive frames. When the semantic RoI changes, the set of blocks protected at high quality may switch accordingly. We model the instability of video-quality adjustment through two asymmetric effects: (1) Bit-burst risk caused by quality increase, and (2) Evidence-continuity loss caused by quality decrease.

For the first effect, the extra payload caused by decreasing the QP of block $\mathcal{B}_n$ relative to the previous QP map is measured as

$$S_i^{\mathrm{b}}(\mathbf{x}_i, \mathbf{x}_{i-1}) = \sum_{n=1}^{N} \gamma_{i-1:i,n}^{\mathrm{b}} \left[\frac{u_{i,n}(x_{i,n}) - u_{i,n}(x_{i-1,n})}{C_i} \right]^+,$$

where $\gamma_{i-1:i,n}^{\mathrm{b}}$ is the bit-burst sensitivity of block $\mathcal{B}_n$ from frame F_{i-1} to frame F_i, computed from texture [27] and inter-frame change features [28]. This term is active only when the current QP is lower than the previous QP, i.e., $x_{i,n} < x_{i-1,n}$, which means that the block is encoded at higher quality than before. The division by C_i converts the extra payload into a transmission-time risk.

For the second effect, we maintain an evidence-continuity state for each block. Let $m_{i,n} \in [0, 1]$ record how much useful evidence has recently been carried by block $\mathcal{B}_n$ after processing frame F_i. It is updated as $m_{i,n} = (1-\mu)m_{i-1,n} + \mu w_{i,n} V(x_{i,n})$, where $\mu \in [0, 1]$ controls how quickly the state follows the current semantic feedback and delivered quality. The evidence-continuity loss caused by quality decrease is then defined as

$$S_i^{\mathrm{c}}(\mathbf{x}_i, \mathbf{x}_{i-1}) = \sum_{n=1}^{N} \gamma_{i-1:i,n}^{\mathrm{c}} m_{i-1,n} \left[V(x_{i-1,n}) - V(x_{i,n}) \right]^+,$$

where $\gamma_{i-1:i,n}^{\mathrm{c}}$ is the evidence-continuity sensitivity of block $\mathcal{B}_n$ from frame F_{i-1} to frame F_i, computed from inter-frame change features [28]. This term is active only when the current QP is higher than the previous QP, namely $x_{i,n} > x_{i-1,n}$, which means that the block is encoded at lower quality than before. It penalizes sudden quality drops on blocks that recently carried useful evidence.

Together, we define the total switching-induced instability of frame F_i as

$$S_i(\mathbf{x}_i, \mathbf{x}_{i-1}) = S_i^{\mathrm{b}}(\mathbf{x}_i, \mathbf{x}_{i-1}) + S_i^{\mathrm{c}}(\mathbf{x}_i, \mathbf{x}_{i-1}). \tag{8}$$

E. Per-frame Evidence-QoE Control Objective

Finally, we formulate the edge-side evidence-QoE objective for block-level QP selection by combining the evidence, timing, and stability models. At each frame scheduling step, the edge selects the block-level QP map by solving

$$\max_{\mathbf{x}_i \in \mathcal{X}^N} \quad E_i(\mathbf{x}_i) - \lambda_L L_i(\mathbf{x}_i) - \lambda_S S_i(\mathbf{x}_i, \mathbf{x}_{i-1}) \tag{9}$$

$$\text{s.t.} \quad r_i(\mathbf{x}_i) \le D, \tag{10}$$

$$x_{\min} \le x_{i,n} \le x_{\max}, \quad n = 1, \ldots, N. \tag{11}$$

Here, $\lambda_L \ge 0$ and $\lambda_S \ge 0$ balance evidence timeliness and switching stability against semantic evidence quality. If no feasible QP map satisfies $r_i(\mathbf{x}_i) \le D$, frame F_i is deadline-infeasible. We then close the current evidence round, and frames $F_i, F_{i+1}, \ldots$ are not optimized or used for the current answer.

IV. Algorithm Design

The Evidence-QoE objective in Eqs. (9)–(11) formulates the block-level QP decision for each frame at the edge device in EQAVC as a per-frame constrained optimization problem. Solving this problem exactly requires a combinatorial search over $\mathcal{X}^N$, which is beyond the online computing budget of a lightweight edge encoder. We therefore design QoE-QP, a deadline-aware controller that makes the block-level QP decision for each frame as shown in Algorithm 1.

For each generated frame, QoE-QP estimates the usable uplink bandwidth, observes the serialized uplink state through r_{i-1}, computes the remaining time before the answer deadline, and reads the latest semantic feedback returned by the server-side VLM when it is available. It then executes three stages. The candidate-guard stage (Lines 3–9) computes the largest encoded payload that can be used for F_i while preserving the current answer deadline and the next-frame service opportunity. The region-identification stage (Lines 10–26) decides which blocks belong to the protected support and which remain in the background. The QP-decision stage (Lines 29–33) filters the candidates by the guard and selects the feasible QP map with the largest Evidence-QoE score.

Algorithm 1 QoE-QP Controller

Input QP set $\mathcal{X}$, support refresh grid $\mathcal{H}$ (defined in Eq. (16)), protection offset grid $\mathcal{D}_q$ (defined in Eq. (18)), deadline D, reservation factor κ_i.

Output Online QP maps $\{\mathbf{x}_i\}$.

Initialize $\mathbf{x}_{-1} = x_{\max}\mathbf{1}$, $P_{-1} = \emptyset$, $\mathcal{R}_{-1} = \emptyset$, $m_{-1,n} = 0$, and $r_{-1} = 0$.
for each generated frame F_i **do**
  # Candidate-guard Stage
  Compute $\mathrm{tr}_i = \max\{g_i, r_{i-1}\}$ and B_i^D by Eq. (12).
  Set $U_i^{\min} = U_i(x_{\max}\mathbf{1})$.
  if $U_i^{\min} > B_i^D$ **then**
    break *(close the current evidence round)*
  end if
  Compute B_i by Eq. (14).
  # Region-identification Stage
  Detect feedback mode η_i by Eq. (15).
  if $\eta_i = 1$ **then**
    Calibrate feedback to $\bar{\mathcal{R}}_i$, compute $\{w_{i,n}\}$, and set $\mathcal{R}_i = \{n \mid A_{i,n}^{\mathrm{o}} > 0\}$.
    Set $\mathcal{H}_i = \{h \in \mathcal{H} \mid h \le \rho_i\}$.
  else
    Reuse the latest valid support and weights, or use the spatial prior, and set $\mathcal{H}_i = \{h_0\}$.
  end if
  Select x_i^{bg} by Eq. (17), initialize $\mathcal{A}_i \leftarrow \{x_i^{\mathrm{bg}}\mathbf{1}\}$.
  Compute $\{\xi_{i,n}\}$ for $P_{i-1} \setminus \mathcal{R}_i$.
  for each $\Delta q \in \mathcal{D}_q \setminus \{\Delta q_0\}$ **do**
    Compute $x_i^{\mathrm{roi}}(\Delta q)$ by Eq. (19).
    Compute $\{\zeta_{i,n}(\Delta q)\}$ for $\mathcal{R}_i \setminus P_{i-1}$.
    for each $h \in \mathcal{H}_i$ **do**
      Construct $P_i^{(h,\Delta q)}$ by Eq. (23).
      Build $\mathbf{x}_i^{(h,\Delta q)}$ by Eq. (24).
      Add $\mathbf{x}_i^{(h,\Delta q)}$ to $\mathcal{A}_i$.
    end for
  end for
  # QP-decision Stage
  Form $\mathcal{A}_i^{\mathrm{feas}}$ by Eq. (26).
  Select $\mathbf{x}_i$ by Eq. (27).
  Encode and upload F_i using $\mathbf{x}_i$.
  Update r_i, $\{m_{i,n}\}$, and P_i.
end for

1) Candidate-guard Stage: When a new frame is generated, QoE-QP first constructs the candidate guard B_i. This guard gives the largest encoded payload of F_i, in bits, that can be admitted to the online search. It combines a proximal guard for the current frame and a distal guard for the next frame.

The proximal guard considers the largest payload that F_i can carry while still reaching the edge server before the hard answer deadline. Let D denote the latest server arrival time at which visual evidence can be admitted to the final reasoning process. Since the uplink serves frames sequentially, the transmission of F_i starts at $\mathrm{tr}_i = \max\{g_i, r_{i-1}\}$. The proximal guard is

$$B_i^D = \left[C_i(D - \mathrm{tr}_i - \tau_i^0)\right]^+. \tag{12}$$

Line 5 checks whether the smallest possible payload is still deadline feasible. Let $U_i^{\min} = b_i^{\mathrm{base}} + \sum_{n=1}^{N} u_{i,n}(x_{\max})$. If $U_i^{\min} > B_i^D$, even the uniform lowest-quality frame cannot arrive before D, and the current evidence round is closed.

The distal guard captures the largest payload that F_i can use while keeping the next frame's upload opportunity feasible under the current throughput estimate. The lookahead term $[C_i(g_{i+1} - \mathrm{tr}_i - \tau_i^0)]^+$ asks the current frame to finish service before F_{i+1} is generated. The fallback term $U_i^{\min}$ ensures that a deadline-feasible frame is not rejected only because this lookahead budget is smaller than the minimum deliverable payload. The distal guard is

$$B_i^{\mathrm{op}} = \max\left\{\left[C_i(g_{i+1} - \mathrm{tr}_i - \tau_i^0)\right]^+, U_i^{\min}\right\}. \tag{13}$$

QoE-QP integrates the two guards as

$$B_i = \min\left\{B_i^D, B_i^{\mathrm{op}}\right\}. \tag{14}$$

Thus, B_i enforces the current answer deadline and avoids starving the next frame unless the uniform fallback already exceeds the distal guard.

2) Region-identification Stage: After the candidate guard is constructed, QoE-QP identifies the protected support and the background support. The rule differs according to whether new semantic feedback has arrived after the previous frame was encoded.

Let $\mathrm{ref}(i)$ be the reference-frame index associated with the latest valid feedback available before encoding F_i, as defined in Sec. III-B. We define

$$\eta_i = \begin{cases} 1, & \mathrm{ref}(i) \neq \emptyset \text{ and } \mathrm{ref}(i) \neq \mathrm{ref}(i-1), \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Here, $\eta_i = 1$ means that QoE-QP uses a newly arrived semantic feedback, either because valid feedback first becomes available or because the reference frame of the latest feedback changes. When $\eta_i = 0$, QoE-QP reuses the latest valid support and weights if they exist. For frames before the first valid feedback, it sets $\mathcal{R}_i = \emptyset$ and uses the spatial prior.

We select h from an evenly discretized grid

$$\mathcal{H} = \{h_0, h_1, \ldots, h_Y\}, \tag{16}$$

where $0 = h_0 < h_1 < \cdots < h_Y = 1$, Y is the number of refresh intervals and $h_y - h_{y-1} = h_{y'} - h_{y'-1}$ for any $y, y' \in \{1, \ldots, Y\}$. Here, $h = 0$ keeps the previous protected support as much as possible, while $h = 1$ fully refreshes the support toward the current semantic demand. QoE-QP calibrates that RoI to F_i, computes $\{w_{i,n}\}$, and sets $\mathcal{R}_i = \{n \mid A_{i,n}^{\mathrm{o}} > 0\}$. It also gates the refresh grid by confidence, $\mathcal{H}_i = \{h \in \mathcal{H} \mid h \le \rho_i\}$. The value $\eta_i = 0$ covers stale feedback and the initial frames before any valid feedback exists. In this case, QoE-QP reuses the latest valid support and weights when available. Otherwise, it sets $\mathcal{R}_i = \emptyset$ and $\mathcal{H}_i = \{h_0\}$, so no support refresh is triggered without a new RoI.

The first quantity computed in this stage is the background QP. For a scalar QP x, $U_i(x\mathbf{1})$ is the predicted payload of the uniform QP map, where $\mathbf{1}$ denotes the N-dimensional all-ones vector. Given B_i, QoE-QP selects

$$x_i^{\mathrm{bg}} = \min\{x \in \mathcal{X} \mid U_i(x\mathbf{1}) \le \kappa_i B_i\}. \tag{17}$$

If this feasible set is empty, $x_i^{\mathrm{bg}} = x_{\max}$. The map $x_i^{\mathrm{bg}}\mathbf{1}$ is always included in the action set as the deliverable background candidate.

The protection offset Δq controls the QP reduction applied to protected blocks relative to background blocks. Its feasible set is

$$\mathcal{D}_q = \{\Delta q_0, \Delta q_1, \ldots, \Delta q_K\}, \tag{18}$$

where $0 = \Delta q_0 < \Delta q_1 < \cdots < \Delta q_K \le x_{\max} - x_{\min}$. A smaller QP gives higher visual fidelity, so a larger Δq allocates higher fidelity to the protected support. The protected QP is clipped to the codec range by Eq. (19). QoE-QP then constructs protection QP candidates. Each candidate uses a protection offset to reduce the QP of blocks that may contain enough response-relevant evidence. For each $\Delta q \in \mathcal{D}_q \setminus \{\Delta q_0\}$,

$$x_i^{\mathrm{roi}}(\Delta q) = \max\{x_{\min}, x_i^{\mathrm{bg}} - \Delta q\}. \tag{19}$$

This equation clips the protected QP to the valid codec range.

The support construction decides which blocks are kept, acquired, and retained. The keep set contains blocks that were protected in the previous frame and are still requested by the current RoI,

$$P_i^{\mathrm{k}} = P_{i-1} \cap \mathcal{R}_i. \tag{20}$$

Newly requested blocks are ranked by evidence gain per burst risk,

$$\zeta_{i,n}(\Delta q) = \frac{w_{i,n}\left[V(x_i^{\mathrm{roi}}(\Delta q)) - V(x_i^{\mathrm{bg}})\right]^+}{\gamma^{\mathrm{b}}_{i-1:i,n}\left[u_{i,n}(x_i^{\mathrm{roi}}(\Delta q)) - u_{i,n}(x_{i-1,n})\right]^+ / C_i}.$$

Previously protected blocks outside the current RoI are ranked by the continuity loss caused by release,

$$\xi_{i,n} = \gamma^{\mathrm{c}}_{i-1:i,n} m_{i-1,n}\left[V(x_{i-1,n}) - V(x_i^{\mathrm{bg}})\right]^+. \tag{21}$$

Let $\mathrm{Top}_k(\mathcal{S}, \phi)$ return the k elements of set $\mathcal{S}$ with the largest score ϕ, with $\mathrm{Top}_0(\mathcal{S}, \phi) = \emptyset$. For each $(h, \Delta q)$,

$$\begin{aligned} P_i^{\mathrm{a}}(h, \Delta q) &= \mathrm{Top}_{\lceil h|\mathcal{R}_i \setminus P_{i-1}|\rceil}\left(\mathcal{R}_i \setminus P_{i-1}, \zeta_{i,n}(\Delta q)\right), \\ P_i^{\mathrm{r}}(h) &= \mathrm{Top}_{\lceil (1-h)|P_{i-1} \setminus \mathcal{R}_i|\rceil}\left(P_{i-1} \setminus \mathcal{R}_i, \xi_{i,n}\right). \end{aligned} \tag{22}$$

The protected support is

$$P_i^{(h,\Delta q)} = P_i^{\mathrm{k}} \cup P_i^{\mathrm{a}}(h, \Delta q) \cup P_i^{\mathrm{r}}(h). \tag{23}$$

Small h values favor continuity by retaining more old blocks. Large h values follow the current semantic demand by acquiring more newly requested blocks.

The candidate QP map is

$$x_{i,n}^{(h,\Delta q)} = \begin{cases} x_i^{\mathrm{roi}}(\Delta q), & n \in P_i^{(h,\Delta q)}, \\ x_i^{\mathrm{bg}}, & \text{otherwise.} \end{cases} \tag{24}$$

The action set constructed in this stage is

$$\mathcal{A}_i = \{x_i^{\mathrm{bg}}\mathbf{1}\} \cup \left\{\mathbf{x}_i^{(h,\Delta q)} \mid h \in \mathcal{H}_i,\ \Delta q \in \mathcal{D}_q \setminus \{\Delta q_0\}\right\}.$$

3) QP-decision Stage: The final stage evaluates only candidates that can be delivered under the current transport state and the current answer deadline (Line 29). For a candidate $\mathbf{x} \in \mathcal{A}_i$, the predicted server arrival time is

$$\hat{r}_i(\mathbf{x}) = \mathrm{tr}_i + \tau_i^0 + \frac{U_i(\mathbf{x})}{C_i}. \tag{25}$$

Line 30 forms the feasible set by applying both the candidate guard and the answer deadline

$$\mathcal{A}_i^{\mathrm{feas}} = \{\mathbf{x} \in \mathcal{A}_i \mid U_i(\mathbf{x}) \le B_i,\ \hat{r}_i(\mathbf{x}) \le D\}. \tag{26}$$

QoE-QP finally chooses

$$\mathbf{x}_i = \arg\max_{\mathbf{x} \in \mathcal{A}_i^{\mathrm{feas}}} \left[E_i(\mathbf{x}) - \lambda_L L_i(\mathbf{x}) - \lambda_S S_i(\mathbf{x}, \mathbf{x}_{i-1})\right]. \tag{27}$$

Here, $L_i(\mathbf{x})$ is evaluated using the predicted arrival time $\hat{r}_i(\mathbf{x})$. Because the maximization is restricted to $\mathcal{A}_i^{\mathrm{feas}}$, a high-evidence candidate cannot be selected if it would exceed the candidate guard or arrive after the answer deadline. After encoding and uploading F_i, Lines 32 to 33 update the server arrival time r_i, the evidence continuity state $\{m_{i,n}\}$, and the protected support P_i for the next frame.

V. Evaluation

A. Experimental Setup

In this section, we evaluate the performance of EQAVC with the QoE-QP controller design against benchmarks for a real-time AI video chat application under constrained uplinks. The code for all experiments in this section is made publicly available at [29].

Video-chat sessions. We design and implement live AI video chat sessions following the real-time video-interaction question-generation protocol of ARTIC [20]. Specifically, we collect 13 publicly available lifestyle-vlog videos, with a total length of over 2.40 hours, and process them as time-stamped video streams. We use Gemini 3.5 Flash [30] to construct localized visual-evidence QA sessions. We manually inspect a random 30% subset of the question sessions and remove or revise ambiguous ones whose answers cannot be determined by the annotated evidence, which leads to 111 final question sessions. Each final question session consists of a user query, query timestamp, evidence interval, and reference answer.

Parameter setting. We implement the VLM at the edge server with Qwen3-VL-Flash [31]. We set $\tau = 0.5$s, so that each reasoning round lasts for 0.5s, or until the response deadline D is reached, whichever occurs first. The edge device captures video at 4 fps. Each frame is partitioned into an 8×8 grid, corresponding to $N = 64$ controllable blocks. QoE-QP controller searches the QP set $\mathcal{X} = \{18, \ldots, 51\}$, the support-refresh ratio set $\mathcal{H} = \{0, 0.25, 0.5, 0.75, 1\}$, and the protection-offset set $\mathcal{D}_q = \{0, 8, 12, 16, 20, 24, 28, 32\}$. We set $\kappa_i = 0.85$, $\mu = 0.2$, $\lambda_L = 1.0$, and $\lambda_S = 0.2$. The rate-scale coefficient $\alpha_{i,n}$ in Sec. III-C is refreshed after each encoded

frame using the observed payload in the online update step (Lines 32–33). The block-level QP map selected by QoE-QP controller is then implemented by FFmpeg/libx264 encoding path [21]. Unless otherwise stated, QoE-QP controller and benchmarks run in the same logical-time streaming pipeline and use the same FFmpeg/libx264 encoding path. For each video v, we set the available uplink capacity to $C_v(\eta) = \eta R_v^{48}$, where R_v^{48} is the measured payload bitrate under uniform QP 48. We evaluate $\eta \in \{1.6, 2.5\}$ and set the hard response deadline D to Δ_D after the end of the annotated evidence interval for each question session s, where $\Delta_D \in \{2\text{s}, 3\text{s}\}$.

Benchmarks. We compare our QoE-QP policy with two QP-control policy benchmarks under the same FFmpeg/libx264 encoding path [21]: (i) **RA-QP**, a semantic-agnostic codec baseline that does not use server-side RoI feedback; (ii) **EA-QP**, a greedy evidence-aware baseline that maps the latest RoI feedback to RoI/qoffset targets using only the evidence-quality term $E_i(\mathbf{x}_i)$ formulated in Sec. III-B, without modeling model-consumption deadlines or QP-map switching instability;

Performance Metrics. We evaluate the performance of QoE-QP controller under EQAVC framework against the benchmarks primarily by response accuracy, response latency, deadline miss, and evidence-frame miss. (i) Response accuracy measures whether the final VLM answer matches the reference answer (Figs. 3(a), 3(b), 4, 5(a), and 5(c)). (ii) Response latency is measured from the later of the query-end and evidence-end timestamps to the start of the final VLM call, excluding VLM inference time (Figs. 3(a) and 4). (iii) Deadline miss and evidence-frame miss are computed over deadline-triggered sessions: the former checks whether the last required evidence frame is absent from the final VLM context, and the latter reports the fraction of annotated evidence frames absent from that context. We also report average evidence-frame bitrate, evidence-frame bit-burst pressure, defined as encoded bits over available frame capacity, and controller decision time excluding VLM inference and video encoding (Figs. 3(b) and 3(c)). (iv) We summarize each method's joint accuracy–latency performance using its mean operating point and a bootstrap covariance ellipse. Let δ_i and a_i denote the latency and accuracy outcome of valid QA session i, respectively. The arithmetic mean of the latency–accuracy vectors $(\delta_i, a_i)^\top$ over all valid sessions is denoted by $\widehat{\boldsymbol{\mu}}$. We then generate M stratified bootstrap replicates [32]. For each $b = 1, \ldots, M$, replicate b is formed by resampling these vectors with replacement within strata while preserving the original stratum sizes. Let $\widehat{\boldsymbol{\mu}}^{*(b)}$ denote the mean latency-accuracy vector of replicate b, and let $\overline{\boldsymbol{\mu}}^*$ denote the mean latency–accuracy vector across the M bootstrap replicates. Then the 2×2 sample covariance matrix is $\widehat{\boldsymbol{\Sigma}}^* = \frac{1}{M-1} \sum_{b=1}^{M} \left(\widehat{\boldsymbol{\mu}}^{*(b)} - \overline{\boldsymbol{\mu}}^*\right) \left(\widehat{\boldsymbol{\mu}}^{*(b)} - \overline{\boldsymbol{\mu}}^*\right)^\top$. Let $\lambda_1 \geq \lambda_2$ be the eigenvalues of $\widehat{\boldsymbol{\Sigma}}^*$, with corresponding unit eigenvectors $(v_{11}, v_{12})^\top$ and $(v_{21}, v_{22})^\top$. Let q_p be the p-quantile of a chi-squared distribution with two degrees of freedom. The full major- and minor-axis lengths and the orientation of the major axis are

$$w = 2\sqrt{q_p \lambda_1},\ h = 2\sqrt{q_p \lambda_2},\ \theta = \arctan(v_{12}/v_{11}). \quad (28)$$

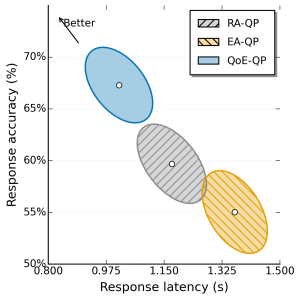
The ellipse is centered at $\widehat{\boldsymbol{\mu}}$, while its orientation θ captures the latency–accuracy covariance across bootstrap resamples: a positive (negative) tilt indicates that higher latency tends to co-occur with higher (lower) accuracy (Figs. 3(a) and 4).

B. End-to-End QoE

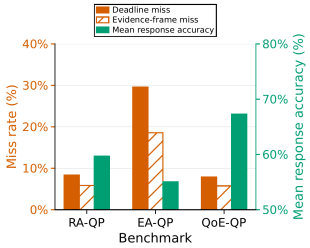
(1) End-to-end performance. We use the method (Eq. (28)) described in Sec. V-A to construct the shaded ellipses around the mean accuracy–latency operating points in Fig. 3(a). It shows that QoE-QP achieves the highest response accuracy and lowest response latency at the same time. It reaches 67.3% response accuracy with 1.01s response latency, compared with 59.7% / 1.17s for RA-QP and 55.0% / 1.36s for EA-QP. The result in Fig. 3(b) separates two failure modes. (i) RA-QP keeps delivery stable, with 8.3% deadline miss and 5.9% evidence-frame miss, but it lacks semantic selectivity and therefore preserves less response-useful evidence. (ii) For EA-QP, it improves the latest RoI greedily, while the resulting payload often cannot be delivered and consumed before the response round, leading to 29.6% deadline miss and 18.6% evidence-frame miss. In contrast, QoE-QP avoids both failure modes, keeping deadline and evidence-frame misses close to RA-QP while achieving an answer accuracy higher than EA-QP.

(2) Burst-induced evidence loss. In Fig. 3(c), the horizontal axis groups evidence frames by bit-burst pressure, defined as the encoded payload divided by the available delivery capacity for the frame, where a larger bit-burst pressure means a heavier scheduling load before the response deadline. Fig. 3(c) illustrates that it is the short-term payload bursts that serve as the primary causes for the two failure modes suffered by benchmarks. As bit-burst pressure increases from 1.00–2.44× to 2.44–5.95×, the evidence-frame miss rate rises from 14.9% to 41.6%, and the deadline miss rate rises from 20.2% to 30.7%. Above 5.95×, the evidence-frame miss rate reaches at least 87.7%. These results show that response-useful evidence must be both preserved and delivered on time. QoE-QP improves QoE by selecting semantic evidence under deadline and payload feasibility constraints, rather than maximizing RoI quality alone.

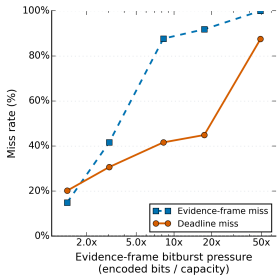
(3) Ablation study. We also use Eq. (28) to construct the ellipses around the mean accuracy–latency operating points in Fig. 4. We ablate the three components summarized in Table II: semantic evidence quality (**w/o E**), schedule-aware latency (**w/o L**), and switching stability (**w/o S**). Removing **E** causes the largest performance drop, reducing accuracy from 73.95% to 68.28% and increasing latency from 0.972 s to 1.151 s. Removing **L** while retaining the candidate guard yields 73.81% accuracy and 1.000 s latency, showing that L_i improves the ranking among feasible QP maps. Removing **S** reduces accuracy to 70.95%, while latency remains close to that of QoE-QP, indicating that switching stability mainly



(a) Accuracy-latency performance.



(b) Miss rate and accuracy.



(c) Bit-burst pressure vs. miss.

Fig. 3. End-to-end QoE of QoE-QP and benchmarks. (a) Accuracy–latency performance ($M = 5000$, $p = 0.80$; stratified by η and Δ_D condition). (b) QoE-QP reduces deadline/evidence misses while improving answer accuracy. (c) Evidence-frame bit-burst pressure and misses across QoE-QP and benchmarks.

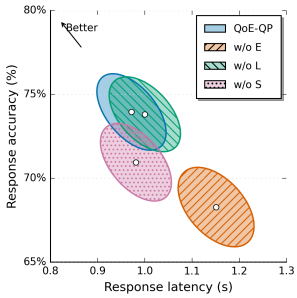
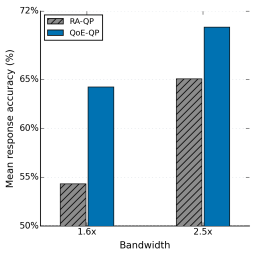
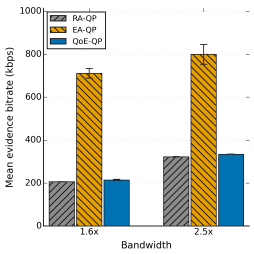


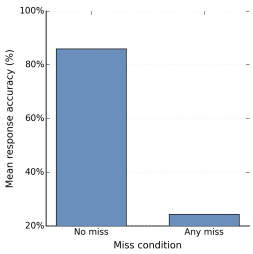
Fig. 4. Accuracy–latency performance of QoE-QP component ablations under the same deadline guard ($M = 5000$, $p = 0.50$, $\Delta_D = 3s$, $\eta = 2.5$).



(a) Accuracy vs. bandwidth.

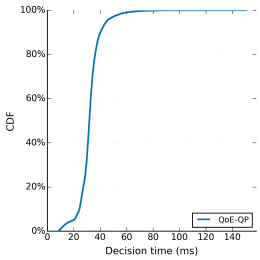


(b) Bitrate vs. bandwidth.

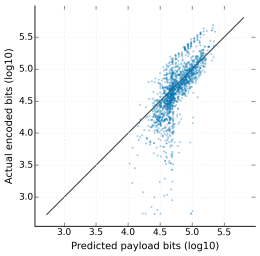


(c) Accuracy vs. miss.

Fig. 5. Evidence-aware QP control improves accuracy with limited bitrate growth. (a) Accuracy under bandwidth stress. (b) Evidence-frame bitrate under bandwidth stress. (c) Accuracy with/without evidence misses.



(a) Decision time.



(b) Predicted vs. actual.

Fig. 6. Online overhead and payload-prediction calibration of QoE-QP. (a) CDF of controller decision time. (b) Predicted vs. actual encoded payload; the diagonal denotes perfect prediction.

TABLE II: QoE-QP ablation variants.

Component	w/o E	w/o L	w/o S
Evidence quality E_i (Sec. III-B)	–	✓	✓
Schedule-aware latency score L_i (Sec. III-C)	✓	–	✓
Switching stability (S_i, h) (Secs. III-D and IV-2)	✓	✓	–
Candidate guard B_i (Sec. IV-1)	✓	✓*	✓

*w/o L partially ablates the latency-aware mechanism: L_i is removed while a minimal bandwidth/deadline feasibility guard is retained; removing both would degenerate the controller toward EA-QP.

suppresses harmful burst-induced transitions. Overall, the results demonstrate the complementary benefits of the three components under the candidate-guarded search.

(4) Source of accuracy gain. Fig. 5 investigates where the accuracy gain of QoE-QP comes from with respect to bandwidth stress, bitrate growth, and evidence availability. We evaluate response accuracy under different bandwidth levels in Fig. 5a, evidence-frame bitrate under the same bandwidth levels in Fig. 5b, and answer accuracy under evidence-miss conditions in Fig. 5c. The gain over RA-QP is larger under tighter bandwidth. QoE-QP improves accuracy from 54.3% to 64.2% at 1.6× bandwidth and from 65.1% to 70.4% at 2.5×. This improvement does not rely on substantially higher bitrate. QoE-QP remains close to RA-QP, using 214.9 kbps versus 206.9 kbps at 1.6× and 334.7 kbps versus 322.7 kbps at 2.5×, while EA-QP consumes 711.2 kbps and 799.6 kbps. Finally, evidence availability directly determines answer quality. Mean accuracy is 85.9% when no required evidence is missed, but drops to 24.4% once any required evidence is missing.

C. Controller Overhead and Payload Prediction

In Fig. 6, we evaluate whether QoE-QP can run in a real-time environment, where each control decision must finish within the frame interval and before the response deadline. The decision time of QoE-QP for a single frame is typically at the tens-of-milliseconds scale, well below the 250 ms frame interval in our 4 fps pipeline and the response deadline D. Thus, the decision time of QoE-QP is not the dominant bottleneck for improving response accuracy and timeliness. The predicted and actual encoded payloads also follow a clear diagonal trend in scale, showing that the rate–QP predictor parameterized by $\alpha_{i,n}$ captures the main payload variation needed by the deadline guard.

VI. Conclusion

This paper studied evidence delivery in real-time AI video chat applications. We proposed EQAVC, an evidence-QoE-aware QP control framework that uses server-side semantic

feedback to guide block-level encoding quality while jointly considering the quality of semantic feedback, the deadline-aware latency of visual evidence, and the smoothness of quality adjustment. Experiments on real video-chat QA sessions show that EQAVC improves response accuracy and latency over greedy evidence preservation while keeping bitrate close to a rate-adaptive baseline. These results demonstrate the effectiveness of EQAVC in supporting accurate and timely AI video chat. Future work could extend the current prototype beyond discrete VLM reasoning rounds to support native continuous streaming inference and further investigate the theoretical properties and performance guarantees of the QoE-QP controller.

ACKNOWLEDGMENT

This research was supported in part by the Australian Research Council Linkage Project (LP230100294).

REFERENCES

[1] H. Chen, B. Guo, Z. Yu, and Q. Han, "Toward real-time and cooperative mobile visual sensing and sharing," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[2] G. Gualdi, A. Prati, and R. Cucchiara, "Video streaming for mobile video surveillance," *IEEE Transactions on Multimedia*, vol. 10, no. 6, pp. 1142–1154, 2008.

[3] A. Casparsen, V.-P. Bui, S. R. Pandey, J. J. Nielsen, and P. Popovski, "Experimental study of low-latency video streaming in an oran setup with generative ai," *IEEE Networking Letters*, 2026.

[4] L. Ge, W. Bao, X. Sheng, D. Yuan, B. B. Zhou, and Z. Wang, "Joint video denoising and super-resolution network for iot cameras," *IEEE Internet of Things Journal*, vol. 11, no. 17, pp. 28 526–28 538, 2024.

[5] Y. Tang, J. Bi, S. Xu, L. Song, S. Liang, T. Wang, D. Zhang, J. An, J. Lin, R. Zhu *et al.*, "Video understanding with large language models: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 36, no. 2, pp. 1355–1376, 2025.

[6] B. Lin, Y. Ye, B. Zhu, J. Cui, M. Ning, P. Jin, and L. Yuan, "Video-llava: Learning united visual representation by alignment before projection," in *Proceedings of the 2024 conference on empirical methods in natural language processing*, 2024, pp. 5971–5984.

[7] J. Chen, Z. Lv, S. Wu, K. Q. Lin, C. Song, D. Gao, J.-W. Liu, Z. Gao, D. Mao, and M. Z. Shou, "Videollm-online: Online video large language model for streaming video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 18 407–18 418.

[8] H. Xiong, Z. Yang, J. Yu, Y. Zhuge, L. Zhang, J. Zhu, and H. Lu, "Streaming video understanding and multi-round interaction with memory-enhanced knowledge," in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 69 332–69 351.

[9] R. Qian, S. Ding, X. Dong, P. Zhang, Y. Zang, Y. Cao, D. Lin, and J. Wang, "Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 24 045–24 055.

[10] X. Ding, H. Wu, Y. Yang, S. Jiang, Q. Zhang, D. Bai, Z. Chen, and T. Cao, "Streammind: Unlocking full frame rate streaming video dialogue through event-gated cognition," in *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2025, pp. 13 448–13 459.

[11] D. Ray, C. Smith, T. Wei, D. Chu, and S. Seshan, "Sqp: Congestion control for low-latency interactive video streaming," *arXiv preprint arXiv:2207.11857*, 2022.

[12] B. Zhang, X. Jin, S. Ratnasamy, J. Wawrzynek, and E. A. Lee, "Awstream: Adaptive wide-area streaming analytics," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, 2018, pp. 236–252.

[13] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. Netravali, "Reducto: On-camera filtering for resource-efficient real-time video analytics," in *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2020, pp. 359–376.

[14] K. Du, A. Pervaiz, X. Yuan, A. Chowdhery, Q. Zhang, H. Hoffmann, and J. Jiang, "Server-driven video streaming for deep learning inference," in *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2020, pp. 557–570.

[15] K. Du, Q. Zhang, A. Arapin, H. Wang, Z. Xia, and J. Jiang, "Accmpeg: Optimizing video encoding for accurate video analytics," in *Proceedings of Machine Learning and Systems*, D. Marculescu, Y. Chi, and C. Wu, Eds., vol. 4, 2022, pp. 450–466.

[16] W. Wang, L. Mi, S. Cen, H. Dai, Y. Li, X. Fu, and Y. Liu, "Region-based content enhancement for {Efficient} video analytics at the edge," in *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*, 2025, pp. 613–633.

[17] Y. Wu, T.-T. Nguyen, L. Liebe, Q. Tau, P. E. Campos, J. Cheng, and D. Lee, "How2compress: Scalable and efficient edge video analytics via adaptive granular video compression," in *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025, pp. 12 006–12 015.

[18] Y. Zhang, H. Wang, Q. Bai, H. Liang, P. Zhu, G.-M. Muntean, and Q. Li, "Vavlm: Toward efficient edge-cloud video analytics with vision-language models," *IEEE Transactions on Broadcasting*, vol. 71, no. 2, pp. 529–541, 2025.

[19] J. Wu, Z. Ren, L. Liu, and X. Zhang, "Chat with ai: The surprising turn of real-time video communication from human to ai," in *Proceedings of the 24th ACM Workshop on Hot Topics in Networks*, 2025, pp. 402–410.

[20] J. Wu, Z. Ren, J. Zhong, L. Liu, and X. Zhang, "Artic: Ai-oriented real-time communication for mllm video assistant," in *Proceedings of the ACM SIGCOMM 2026 Conference*, 2026, pp. 1649–1662.

[21] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 560–576, 2003.

[22] G. Xiang, X. Zhang, X. Huang, F. Yang, C. Zhu, H. Jia, and X. Xie, "Perceptual quality consistency oriented ctu level rate control for hevc intra coding," *IEEE Transactions on Broadcasting*, vol. 68, no. 1, pp. 69–82, 2021.

[23] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proceedings of the conference of the ACM special interest group on data communication*, 2017, pp. 197–210.

[24] C. Qian, X. Yu, Z. Huang, D. Li, Q. Ma, F. Dang, X. Ding, G. Shang, and Z. Yang, "edgevlm: Cloud-edge collaborative real-time vlm based on context transfer," *arXiv preprint arXiv:2508.12638*, 2025.

[25] Y. Li, D. Gumaste, M. K. Turkcan, J. Ghaderi, G. Zussman, and Z. Kostic, "Distributed vlms: Efficient vision-language processing through cloud-edge collaboration," in *2025 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, 2025, pp. 280–286.

[26] J. P. Lewis, "Fast normalized cross-correlation," in *Vision interface*, vol. 10, no. 1, 1995, pp. 120–123.

[27] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

[28] R. Jain and H.-H. Nagel, "On the analysis of accumulative difference pictures from image sequences of real world scenes," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 206–214, 1979.

[29] "Supplementary code," Dropbox shared folder, 2026. [Online]. Available: https://www.dropbox.com/scl/fo/qky9vs3bx6uisknfw4ul6/AKZY3C10X3HsByucafeor7I?rlkey=n8jlrowslmshdbzffu08fmcua&st=ondq8zix&dl=0

[30] "Gemini 3.5 flash," Google Cloud Documentation, 2026. [Online]. Available: https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/gemini/3-5-flash

[31] "Qwen," Qwen.ai, 2026. [Online]. Available: https://qwen.ai/blog?id=99f0335c4ad9ff6153e517418d48535ab6d8afef&from=research.latest-advancements-list

[32] B. Efron and R. Tibshirani, "Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy," *Statistical science*, pp. 54–75, 1986.