An Adaptive and Reliable Forward Error Correction Mechanism for Real-time Video Delivery from UAVs

Batoul Sarvi
School of Computer Science,
University College Cork,
Cork, Ireland
b.sarvi@cs.ucc.ie

Cormac J. Sreenan
School of Computer Science,
University College Cork,
Cork, Ireland
cjs@cs.ucc.ie

Ahmed Zahran
School of Computer Science,
University College Cork,
Cork, Ireland
a.zahran@cs.ucc.ie

Abstract— **This paper introduces Adaptive and Reliable Forward Error Correction (AR_FEC), a novel mechanism that is designed to ensure reliable and efficient real-time video delivery for Unmanned Aerial Vehicles (UAVs). UAVs communicate using wireless channels that are inherently unreliable. AR_FEC is an application-layer solution that is inspired by edge computing. It incorporates several elements, specifically, adaptive Forward Error Correction (FEC), dynamic video quality, and Unequal Error Protection (UEP). AR_FEC dynamically adjusts the number of redundancy frames per video Group of Picture (GoP) based on packet loss rate, recent average round-trip times, and cross-layer information such as bandwidth usage and network traffic load. Moreover, AR_FEC reacts to congestion via a dynamic video quality algorithm. By adaptively determining suitable redundancy for the video I-frames and adjusting video quality, significant improvements are achieved. Comprehensive simulations demonstrate that the proposed AR_FEC protocol offers excellent performance for the number of delivered frames, latency, throughput, and delivery of frames, within the delay constraints for real-time multimedia streaming from UAVs.**

Keywords*— *UAV, Error Correction, Multimedia communication, FEC, video streaming

I. INTRODUCTION

The use of Unmanned Aerial Vehicles (UAVs) has experienced unprecedented growth over the past decade [1] due to their flexibility in accessing remote areas and relatively low operating costs [2]. Integrating cameras with UAVs enables them to record high-quality videos and/or seamlessly share them with control stations or operators for inspection and further analysis. More recently, interest has grown in using consumer-grade UAVs for automated flight and remote inspection, examples include environmental monitoring, precision agriculture, search and rescue operation, target tracking, and firefighting [3]. In such scenarios, the availability of a consistently high-quality video for analysis is crucial to ensure safe and correct operation. The nature of these applications is that video will be analyzed and actions taken, often within strict time bounds, to ensure appropriate control of the UAV flight and sensing activities. Consequently, minimal latency is more important than absolute reliability, unlike the case of HTTP Adaptive Streaming over TCP/QUIC, which, gives 100% reliability but with high latency (even with lower-latency variants such as HLS, the latency is at least 5 seconds [4]). The use of network-based edge computing to support low-latency video analysis is likely to be widely used with this class of UAVs, together with some on-device edge computing for control and other purposes. UAVs depend on wireless communications, which due to their inherent lack of reliability, presents a research challenge for the scenarios outlined. In particular, impairments or delays in the delivery of the video stream can result in a failure in the analysis of the content, and thus in the correct operation of the UAV. This is especially challenging due to the fact that UAVs are usually mobile, and the distance between the UAV and its control station changes as a consequence, also impacting on the quality of wireless communication.

The implementation of error control mechanisms is crucial to ensuring efficient video transmission over wireless. These mechanisms, which include Automatic Repeat Request (ARQ), Forward Error Correction (FEC), and Network Coding, can be applied at different points in the protocol stack [5], and are often optimized using cross-layer information [6]. In a real-time context, ARQ is not suitable due to retransmission delays; instead, FEC is better suited, despite the additional overhead for transmitting redundant data [6], [7]. To address concerns related to adding redundancy to limited and vulnerable UAV networks, this paper proposes using adaptive redundancy in FECs. Our approach seeks to maximize the delivered video quality while avoiding excessive redundancy. It is designed to operate at the application layer, where it can exploit information related to the structure of the video, specifically the type of video frames. It comprises an adaptation algorithm running at the edge-device, i.e., the UAV, and a streaming protocol with explicit feedback from the receiver, i.e., the control station. The proposed method also includes an adaptive video quality module that attempts to deliver video frames even in low quality and protect other types of frames without additive (redundancy) packets.

We present a novel approach for the reliable delivery of real-time video from UAVs, called Adaptive and Reliable Forward Error Correction (AR_FEC). This seeks to predict the most suitable amount of FEC redundancy, taking into account information about the video and the wireless link conditions. Through simulation, this is demonstrated to yield improved video quality at the control station. The rest of this paper is organized as follows: In Section 2, an overview of the existing error control schemes for multimedia communication related to UAVs is provided. In Section 3, we describe our solution, AR_FEC, including the system model for UAV networks, and the adaptive and reliable FEC modules. Section 4 describes our simulation methodology and presents the

---

This publication has emanated from research supported with the financial support of Science Foundation Ireland under Grant No. 18/CRT/6222: the SFI Centre for Research Training in Advanced Networks for Sustainable Societies (ADVANCE); and in part by SFI Grant Number 13/RC/2077 P2.

results of our comprehensive performance evaluation. Section 5 presents the conclusions and future work.

II. RELATED WORK

In this section, we review related published work on reliable and multimedia communications, including the use of edge computing, for UAVs.

A. Multimedia communication with UAVs

In terms of video communication on UAV networks, there are some papers that are focused on distinct aspects of UAV networking. In [8], researchers proposed a parameter optimization mechanism for UAV intelligent video analysis. An "optimal strategy library" is designed so that the parameters for video encoding and FEC can be optimized for real-time UAV videos. These values are prepared based on selected video streams and do not take account of variable channel conditions.

In the context of energy consumption in UAV networks, [9] studied comparing the performance of FEC and ARQ algorithms at the application layer. It found that FEC with adaptive redundancy demonstrates superior energy efficiency compared to ARQ in UAV networks with unstable channel quality. These results justify our research focus on implementing adaptive FEC at the application layer, in respect to relative energy consumption.

Other authors focus on the use of network-edge and device-edge computing to support video on UAVs. In [10], four strategies are introduced based on a combination of UAV-based processing and network-edge processing using a set of machine learning algorithms. Another use case appears in [11], where the performance boost from using a network-edge for video analysis is especially significant for longer videos. Additionally, in [12], the use of a network-edge server to offload UAV computation is analyzed. The common aspect between all of them is on balancing UAV-based and network-edge processing to conserve bandwidth, enhance scalability, and maintain result accuracy and latency.

B. Error Control Mechanisms for UAVs

The subject of error control for network communication has been widely studied, but less so in the incredibly challenging context of UAVs. We focus on the use of FEC, due to the low latency delivery requirements of real-time video.

In [5], the authors describe a cross-layer view for improving the transmission reliability of streaming media transmission on a cellular-connected UAV. They propose an integrated client-server-ground&user framework within which they implement a series of algorithms, including one for FEC. This FEC algorithm defines four streaming modes, each corresponding to defined network conditions, based on end-to-end delay and packet loss rate parameters. Unlike our work, the use of a predefined set of operational states and FEC, limits the solution's ability to respond to highly variable link conditions. Furthermore, their approach is limited to a maximum correction ability of 2% of PLR.

In [13], the authors investigate the reliability and efficiency of employing broadcast transmissions from a UAV to multiple network access points, with packet-level FEC. Their experimental results have shown that broadcast transmissions with packet-level FEC achieve a higher packet delivery ratio than unicast transmissions. The proposed framework in [14], also uses broadcast from a UAV to multiple access points for increasing reliability, with a FEC scheme in the application layer to remove the necessity of retransmissions. While attractive in principle, the use of broadcast does not scale for larger numbers of transmitters, and the assumption that a UAV will always be within range of multiple network access points is not reasonable, unless perhaps for indoor scenarios.

The paper [15] presents an adaptive error control framework for a cognitive radio-based UAV network, seeking to minimize the energy consumption of UAVs to increase their operating life span. Selecting the type of error control mechanism is one of the approaches in this framework. To ensure the integrity of the data, they can opt between ARQ and FEC, which depends on the selected radio channel. The network topology is hop-to-hop delivery via UAVs, each of which can choose the error coding scheme that is either ARQ or an FEC, out of a set of 70 different BCH codes. This work does not address the real-time, high-bandwidth, and high-quality characteristics of video delivery.

The authors of [16] proposed a QoE-aware adaptive mechanism for improving video quality over UAVs by utilizing motion vectors details, FEC, and Fuzzy logic through a cross-layer adaptive video-aware mechanism. However, this approach requires pre-calculation of the required redundancy, limiting its use in more dynamic network scenarios.

Table I lists publications that discuss FEC on UAVs. We were unable to locate papers that share the goals of our work and that provided sufficient algorithmic details and/or code to allow us to reproduce their solution for comparison.

Table I: Summary of related published papers

Ref.	**Year**	**Focus**	**Error control mechanism**	**Evaluation**	**Code**
[9]	2023	Unicast	Comparing AL-FEC and AL-ARQ	Analysis	No
[5]	2021	Framework	Adaptive AL-FEC	Testbed	No
[13]	2020	Broadcast	AL-FEC	Testbed	No
[8]	2020	Tracking	Adaptive Video encoding and AL-FEC	Simulation	No
[15]	2019	Multiple UAVs as sender and relay	Selecting between ARQ and FEC	Analysis	No
[14]	2017	Broadcast	AL-FEC	Analysis	No
[16]	2014	Unicast	Adaptive AL-FEC	Simulation	No

III. RELIABLE FORWARD ERROR CORRECTION METHOD ON UAV VIDEO TRANSMISSION

A. AR_FEC Overview

AR_FEC is designed to reliably stream high-quality real-time video from a mobile UAV to a designated station. AR_FEC achieves this goal by adaptively changing the video quality and FEC overhead as the UAV navigates its assigned path. The adaptation decisions are based on monitored packet losses and delays for previously transmitted GoPs to estimate the network capacity and losses. Fig. 1 shows a system block diagram of our proposed method. The estimated parameters are used to decide the FEC level and the streamed quality of the next GoP. The key features are summarized as follows:

- Employing device-edge computing on the UAV for the FEC mechanism, thus ensuring timely adaptation to network conditions;
- Using unequal error protection for video I-frames, thus reducing the amount of FEC overhead required while achieving sufficient video quality;
- Employing adaptive error protection ratio to enable high correction ability with less bandwidth usage based on the network condition;
- Adjusting the video quality to ensure real-time communication by avoiding link congestion;
- Providing a real-time error correction module at the receiver (control station).

B. AR_FEC Design

1) AR_FEC Network Monitoring

AR_FEC monitors the network performance by measuring packet loss ratio (PLR) and network delay (D) for every GoP. To simplify our representation, we will use " t " as an index for the next GoP to be sent. Hence, PLR_t and PLR_{t-1}represent the PLR of next and current GoP, respectively.

The sending node, i.e., UAV, tracks PLR_i through feedback mechanism from the receiver. When the edge station receives the GoP$_{t-1}$ I-frame and its FEC packets, it calculates the percentage of lost packets and sends it back to the UAV using real-time control protocol (RTCP) feedback. The sender stores the last received PLR and its corresponding index and uses them to adaptively calculate PLR_t. It is important to note that packets carrying PLR information are also prone to channel degradation, including loss and delay.

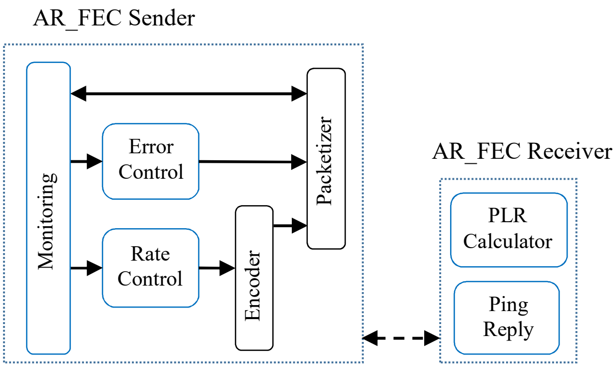


Fig. 1: System blocks of the AR_FEC algorithm

To track the network delay, the UAV sends a PING request message to the station, which subsequently sends the PING reply back. The PING request is sent at an arbitrary time within the GoP. However, sending it closer to the end of the current GoP increases the probability of missing the PING reply when deciding for the next GoP. Note that the request packet will be queued at the MAC layer after all previously transmitted video packets. Hence, it may experience a long delay due to deteriorated network conditions. In case of excessive delay, AR_FEC would be deprived of having an accurate estimate of network rate, which is required for the rate adaptation component.

2) AR_FEC Error Protection

We consider the video compression technique to exclusively utilize I-frames and P-frames, where the latter are dependent solely on the former. This reliance on predictive frames allows for real-time encoding and decoding of video frames during capture by the UAV or reception at the station. However, if an I-frame is lost, the entire GoP becomes uninterpretable. To ensure reliable video delivery, AR_FEC protects the packets containing I-frames.

AR_FEC uses Read-Solomon (RS) code [17] to determine the FEC overhead of I-frames based on an estimate for PLR_t, denoted as PLR_t^e. The RS-FEC algorithm transmits N packets comprising K source I-frame packets and R repair (FEC) packets. The RS code can correct up to N-K errors when the error locations are known. By adding N-K redundant packets to K source packets in the application layer, the original K source packets can be recovered if the receiver receives at least K packets. Note that adding more than K redundant packets is unreasonable, as it aims to protect redundant packets. Therefore, the receiver would not be able to decode an I-frame if the PLR exceeded 50% of N corresponding packets.

When the receiver sends PLR_{t-1} as part of its monitoring component, the sender would have to estimate PLR_t^e, if the RTCP feedback packet is not lost or delayed. Additionally, the sender should adjust the amount of overhead in response to changes in network conditions to avoid sending excessive FEC overhead if network condition is improving or vice versa. In the presence of PLR_{t-1} at the sender, PLR_t^e is calculated as:

$$PLR_t^e = \begin{cases} \frac{PLR_{t-1}^e}{2^\omega} , & PLR_{t-1} = 0\% \\ PLR_{t-1} , & PLR_{t-1} < 50\% \\ Congestion , & PLR_{t-1} \geq 50\% \end{cases} \quad (1)$$

This design implies using the latest captured network loss when RS-FEC can be used to recover the data. In the case of excessive packet loss, AR_FEC would consider the system under congestion, which is handled as described below. Otherwise, it reduces FEC overhead in an exponential fashion as the loss drops to zero. The exponential value represents ω and is an AR_FEC parameter. If PLR_{t-1} is missed or delayed, AR_FEC estimates PLR_t^e using the most recently received PLR and the previously predicted PLR based on follow:

$$PLR_t^e = \begin{cases} PLR_{t-\Delta} + \omega , & PLR_{t-1}^e \leq PLR_{t-\Delta} \\ PLR_{t-1}^e + \omega , & PLR_{t-1}^e > PLR_{t-\Delta} \end{cases} , \quad (2)$$

where Δ is the number of consecutive delayed ping reply packets and $PLR_{t-\Delta}$ shows the latest recieved PLR. This is a usage of the Additive Increase and Multiple Decrease (AIMD) approach on the AR_FEC algorithm to swiftly adjust and adapt to deteriorating network conditions or temporary network degradation.

PLR_t^e is further updated based on the measured D_t and its availability. Note that PLR_t^e is identified at the beginning of the GoP while D_t represents a later probe for network conditions. If D_t is not available, an Adaptive Application Threshold, denoted as λ^a, is calculated by the sender as follows:

$$\lambda^a = \lambda\,(1 - \alpha \times \Delta)\ , \tag{3}$$

where λ is the max acceptable application delay threshold, α is a tunable AR_FEC parameter, and Δ represents the number of consecutive delayed ping reply packets. For example, if the last received ping reply is D_{t-2}, then $\Delta = 2$. Consequently, λ^a will adaptively decrease as network conditions deteriorate. AR_FEC compares the λ^a with the last received $D_{t-\Delta}$. If $D_{t-\Delta} > \lambda^a$, network channel has been congested or is in the early stages of congestion. Hence, AR_FEC would enter congestion mode. Otherwise, the sender monitors the trend of last received 5 delay samples, i.e., $D_{t-\Delta-5}$ to $D_{t-\Delta}$ and calculates their sample mean and standard deviation, denoted as μ and σ, respectively, as follow:

$$\sigma = \sqrt{\frac{\sum_{i=t-\Delta-5}^{t-\Delta} (D_i - \mu)^2}{5}} \tag{4}$$

If $D_{t-\Delta}$ exceeds $\mu + 2\sigma$, PLR_t^e is increased by ω%, where ω represents an AR_FEC parameter. Once the PLR_t^e is estimated, the number of FEC packets for GoP$_t$'s I-frame is calculated as:

$$p_t^{FEC} = \frac{p_t^{I-frame}}{1 - PLR_t^e} - p_t^{I-frame}\ , \tag{5}$$

where $p_t^{I-frame}$ represents the number of packets required to send the I-frame.

3) AR_FEC Quality adaptation

AR_FEC quality adaptation enables the algorithm to deliver high quality video in good network conditions and switch to low quality as network conditions deteriorate. Initially, the UAV sends the GoP$_1$ at a low-quality and $PLR_1^e = 5\%$. AR_FEC estimates the network rate during transmitting GoP$_t$, denoted as R_t^e, and is calculated as:

$$R_t^e = \left({}^{B_t}\!/_{T_t} \right)\ , \tag{6}$$

where B_t represents the number of transmitted bytes for the I-frame and FEC included, and T_t represents the duration starting when the first byte of the I-frame is transmitted until the RTCP message carrying the PLR feedback is received at the sender. Noting that I-frame tends to be large in size, their transmission is considered a good probe for the network throughput. AR_FEC will switch up the quality of the GoP$_t$ if R_t^e is less than the estimated network rate for GoP$_{t-1}$ including the additional FEC based on PLR_t^e; i.e.,

$$R_t^e \le \beta * R_{t-1}^e \quad , \quad \beta = \frac{D_{t-2}}{D_{t-1}}\ , \tag{7}$$

where β is a safety factor that is an AR_FEC parameter and its default value is 1. AR_FEC will downgrade the video quality from high to low if:

- The system is identified as in congestion when PLR_t^e or D_t are deemed high as explained above. In this mode, AR_FEC will also set PLR_t^e to 50%. This means that we send low-quality video with high correction ability.

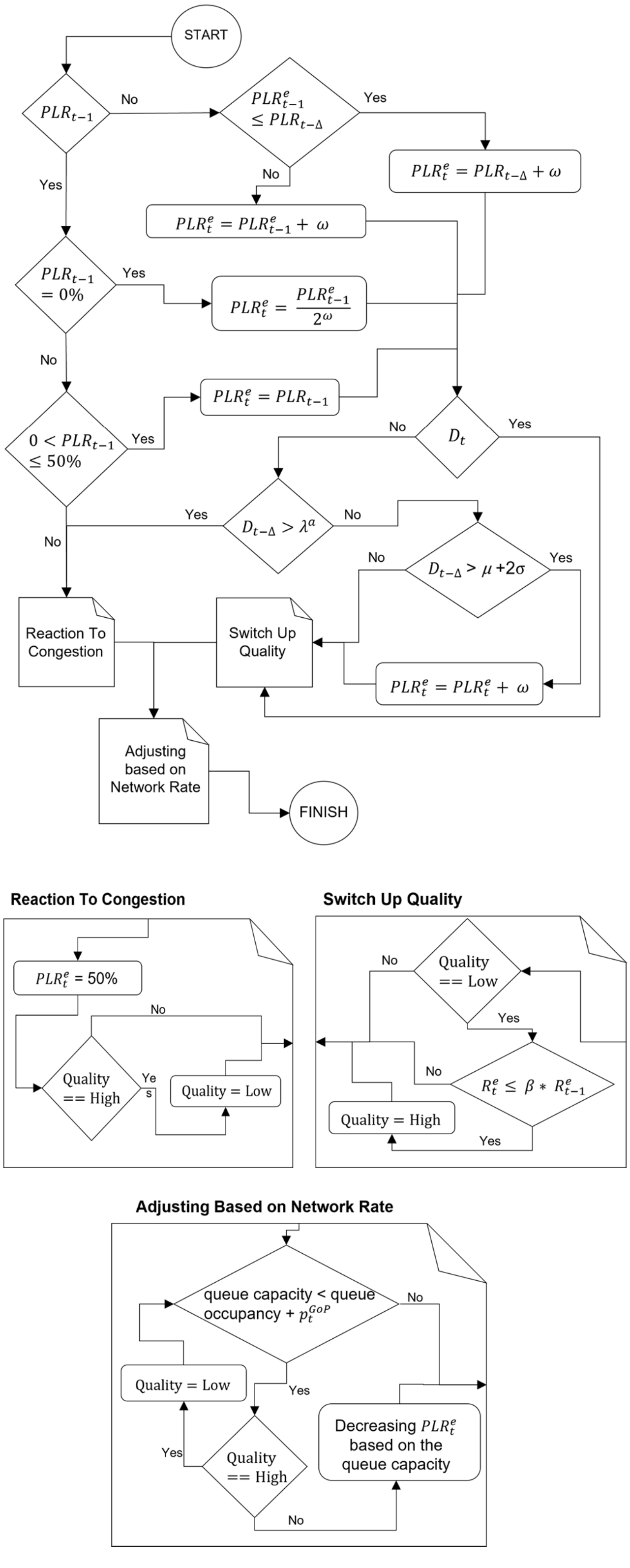


Fig. 2: AR_FEC Algorithm

- AR_FEC expects a link layer buffer overflow. The overflow condition is defined as (queue capacity < queue occupancy + p_t^{GoP}). Note that AR_FEC would also adjust the PLR_t^e if the packets of GoP$_t$ at low-quality will cause buffer overflow. The update value is determined by the max value for PLR_t^e that avoids the overflow.

Fig. 2 illustrates the whole processing of AR_FEC. The AR_FEC mechanism aims to enhance video quality while minimizing network overhead and conserving resources. The proposed algorithm could respond quickly to the fluctuating network conditions on a UAV with high mobility through the optimized redundancy on FEC.

IV. EVALUATION

This section begins by providing an overview of the evaluation setup[1]. Subsequently, we present the performance results obtained from testing the various scenarios.

A. Evaluation Setup and Performance Metrics

Our evaluation is based on NS3 [18] simulations using the topology shown in Fig. 3. The receiving station is located in the center, and the UAV follows a square path using the popular WaypointMobilityModel mobility model. The nodes communicate over the 802.11n [19] wireless standard operating at 2.4 GHz. The traffic is based on real traces from a real Drone Footage video [20]. The no audio version of this video is encoded using FFmpeg [21] to two qualities, including 4K and 720p, at 30 fps using H.264 video compression. The core framework for sending video frames in our application is an extension of Evalvid-ns3 [22]. Besides, we extended it to enable RTCP feedback and RTP headers. We also implemented FEC and quality adaptation modules on it. The application traffic is served over UDP to ensure real-time operation. Table II shows the simulation parameters.

We compare the performance of AR_FEC to a number of variants, including (1) *No_FEC-quality* schemes that send the video at a fixed quality without FEC. These schemes are denoted No_FEC-720p and No_FEC-4K. (2) *AR_FEC-quality* schemes that send the video at a fixed quality with FEC. These schemes are denoted AR_FEC-720p and AR_FEC-4K.

Table II: Simulation Parameters

Parameters	**Value**
Distance	400m-600m-800m
Mobility Model	WaypointMobilityModel
Error Rate Model	TableBasedErrorRateModel
Video Quality	720p and 4K
Frame Rate	30 fps
Codec	H.264
Propagation Loss Model	FriisPropagationLossModel
Propagation Delay Model	ConstantSpeedPropagationDelayModel
Wireless Standard	802.11n – 2.4 GHz
Transport Layer	UDP
Rate Control Algorithms	MinstrelHtWifiManager

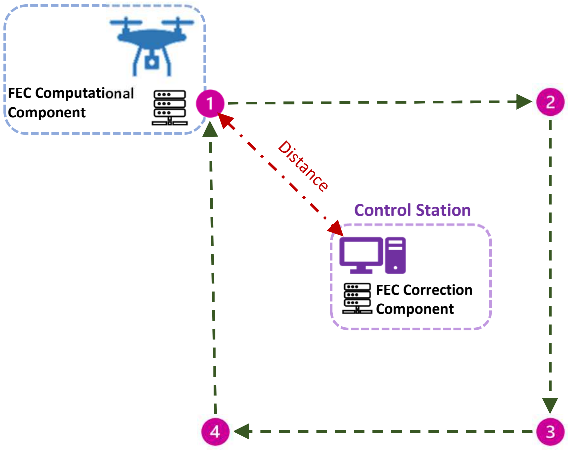


Fig. 3: Evaluation Topology

Our performance evaluation considers different application and network metrics, including:

1. **Number of decodable frames for every quality at the receiver:** This refers to the number of frames that can be successfully decoded at the receiver for each quality level. A frame is considered undecodable if any of its packets are lost and cannot be recovered, or if it is received after its designated playout time.
2. **Frame latency:** This measures the time taken to transmit and receive an entire frame. It represents the delay experienced from the moment a frame is sent until it is fully received.
3. **Throughput (Goodput):** This is the rate at which useful packets are received during the entire simulation duration. It excludes lost and delayed packets from the calculation, providing a measure of the actual data transfer rate.

In our performance evaluation, we consider different scenarios, including testing the performance for different maximum distances between the UAV and the receiving station, including 400 m, 600 m, and 800 m. Furthermore, we look at how the application threshold (λ) affects streaming performance over a distance of 800 meters between the transmitter and receiver. We have tuned AR_FEC parameters, and the shown results are based on α=0.8 and a uniform increment (ω) of 2 for PLR_t^e. Because of limited space availability, the outcomes of tuning the α and ω variables are not included in this context. The simulated time is set to 72 seconds, and the presented outcomes are derived from conducting 10 iterations for each configuration. To optimize the performance, the timing of sending Ping packets is adjusted, and the evaluation results are obtained by sending Ping packet in the middle of each GoP.

B. Impact of Operating Distance

Number of decodable frames: We evaluate the performance of considered schemes for different maximum operating distances. In this section, we set the application threshold (λ) to 1s. Fig. 4 (a-c) demonstrates the number and quality of transmitted and received frames for the compared

---

[1] Our NS3 code is available from https://github.com/BatoulSarvi/AR-FEC

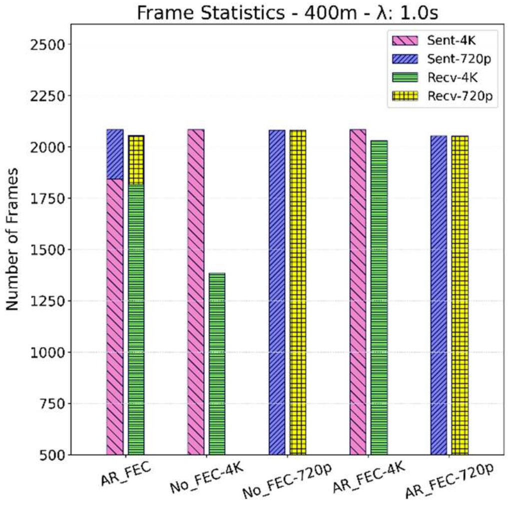
schemes when the UAV's max distance is set to 400m, 600m, and 800m, respectively. The figures illustrate that AR_FEC strikes a good balance between streaming reliability and quality. Streaming a low-quality video exhibits no loss for 400m and 600m but a minimal loss of 9 frames is encountered at the 800m case, which is rectified when using FEC. In contrast, when streaming a high-quality video, frame losses occur at all distances. These losses reached 72% at 800m without FEC, while using FEC reduced this loss to 56%. Notably, when AR_FEC is employed, there is a significant decrease in frame loss. At 800m, the frame loss is reduced to 1% - 7% for low-quality and high-quality videos, respectively. The important point is that the adaptation component of AR_FEC significantly enhances the number of high-quality frames compared to AR_FEC-4K. Notably, at distances of 600m and 800m, it results in an 8% and 5% increase in high-quality frames, respectively.

Throughput: Fig. 5 depicts the throughput of the compared schemes at different distances. In the 400m scenario, AR_FEC-4K achieves the highest throughput by delivering the most high-quality video frames (Fig. 4a). However, in the other two scenarios, AR_FEC exhibits the highest throughput by delivering a large number of frames, particularly high-quality frames. Remarkably, AR_FEC accomplishes this performance with minimal overhead. Table III presents the FEC packet ratios for different approaches utilizing FEC mechanisms at varying distances. Notably, AR_FEC consistently demonstrates the lowest FEC packet ratio across all distances. Additionally, it achieves the highest ratio of high-quality frames compared to AR_FEC-4K (as shown in Fig. 4a, 4b). Importantly, AR_FEC maintains less than 5% frame loss while delivering over 56% high-quality frames at 800m. Furthermore, the FEC packet ratio for this mechanism accounts for only 1.26% of the total packets.

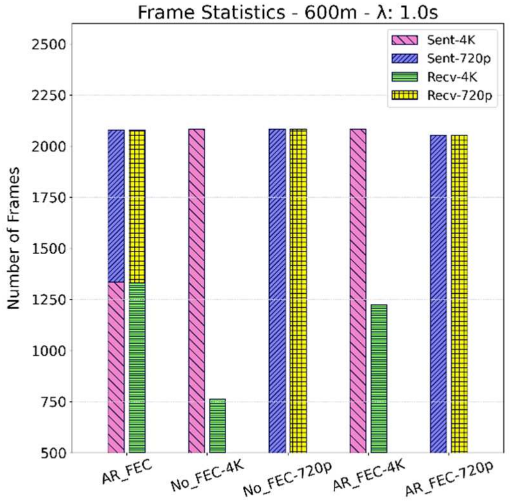
Frame latency: Fig. 6 illustrates the average frame latency of the compared schemes at various distances. It is observed that schemes delivering high-quality video experience higher frame latency. However, AR_FEC demonstrates lower latency across different distances compared to No_FEC-4K and AR_FEC-4K. Notably, as the distance increases to 800m, the average frame latency for AR_FEC is 0.074 s, while it is 0.154 s for AR_FEC-4K and 0.217 s for No_FEC-4K. This reduction in latency is attributed to the adaptive nature of AR_FEC, which reduces quality as network conditions deteriorate. Consequently, it prevents the accumulation of wireless interface queues. This adaptive behaviour also contributes to a reduction in the variance of frame latency, as depicted in Fig. 6.

Table III: FEC packet ratio (%) – λ =1.0s

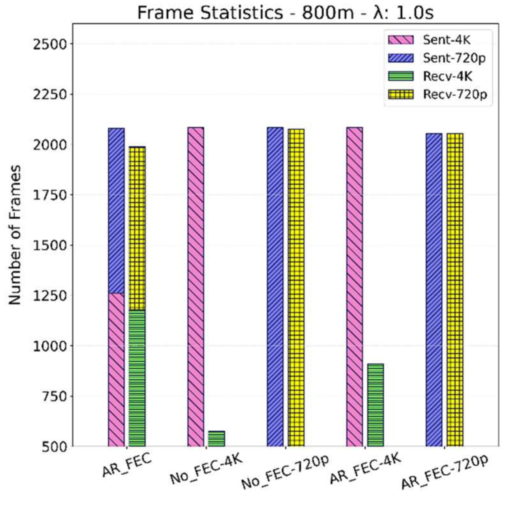
Distance **Algorithm**	**400 m**	**600 m**	**800m**
AR_FEC	0.54	0.68	1.26
AR_FEC–720p	1.45	1.45	1.45
AR_FEC–4K	1.02	15.68	18.56



(a) Distance = 400m



(b) Distance = 600m



(c) Distance = 800m

Fig. 4: Received frame quality for different distances

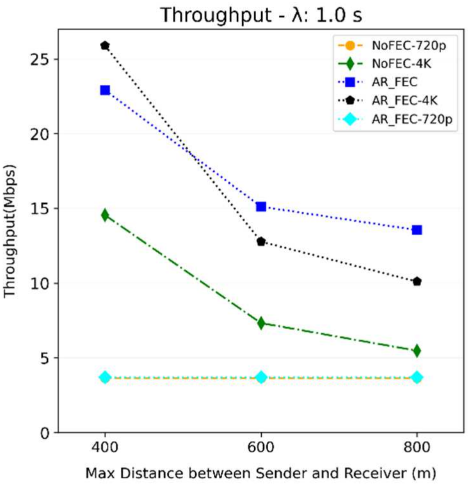


Fig. 5: Throughput on different distances at λ=1.0s

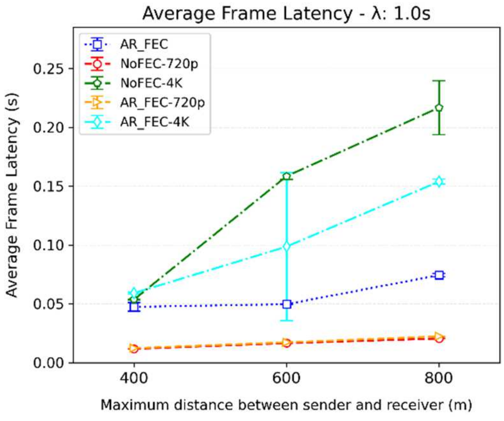


Fig. 6: Average Frame Latency on different distances at λ=1.0s

C. *Impact of Application Threshold*

This section explores the influence of the application threshold (λ) on streaming performance on the 800m as distance between sender and receiver. It is important to note that λ affects both the decisions made by the sender and the perception of lost frames by the receiver. If an error control mechanism does not consider the delay constraint parameter, decreasing the maximum application threshold can have a detrimental impact on the quality of the received video. When λ is reduced during video transmission, a higher PLR is expected due to delayed packets. Therefore, if an error control algorithm fails to deliver the same frames consistently across different application thresholds, it can be deemed unsuitable for time-sensitive video streaming.

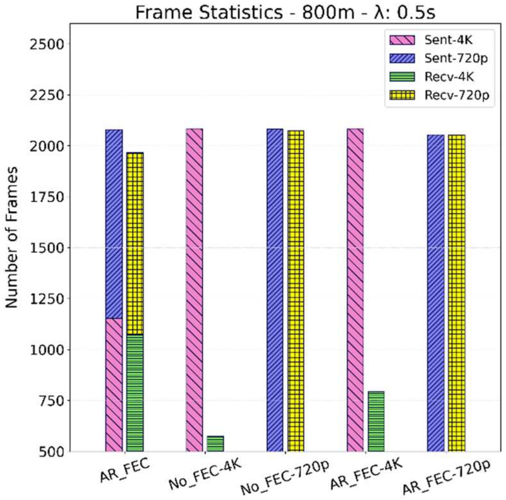
Number of decodable frames: Fig. 7 (a-b) demonstrates how the application threshold affects the number of received frames and their quality for different values of λ (0.5s and 1.5s). In AR_FEC-4K, reducing λ from 1.5s to 0.5s yields a 9% decrease in received frames. The reason for this reduction is to deliver all frames in high-quality with a lower delay, and it is impossible without quality adaptation at a high bit error rate. That is why this reduction is less than 1% for AR_FEC when λ decreases from 1.5s to 0.5s. Moreover, Fig. 7 shows that although the AR_FEC provides a slight decrease in the number of delivered frames at the lowest application threshold, it delivered more than 94% of the frames at λ=0.5s which more than 52% of them have high quality. These findings emphasize the crucial role of video adaptation in maintaining reliable video transmission.

Throughput: Fig. 8 presents the throughput of the compared schemes at different application thresholds over the 800m distance. The figure provides evidence that AR_FEC maintains a relatively stable number of delivered frames across varying application thresholds. In contrast, AR_FEC-4K demonstrates a moderate increase in throughput, reaching 10.8 Mbps from 8 Mbps, when transitioning the application threshold from a 0.5s to a 1.5s. This increase can be attributed to a 9% rise in the number of high-quality frames, as depicted in Fig. 7. It is noteworthy that AR_FEC achieves the highest throughput among the compared schemes while maintaining minimal overhead. Table IV displays the FEC packet ratios for different approaches employing FEC mechanisms at various application thresholds. Generally, there should be a decreasing overhead when we have an increase in the maximum application threshold. This stems from a decrease in PLR because of delayed packets. In contrast, the table reveals a consistent overhead in AR_FEC-720p as all frames have low-quality and there are no losses/delayed I-frame packets that can be recovered by the 1% FEC packets (as a minimum value) for each GoP. Additionally, AR_FEC-720p has a lower number of total packets compared to AR_FEC, resulting in a higher FEC packet ratio for AR_FEC-720p. AR_FEC achieves the highest throughput with the lowest redundancies. On the other hand, AR_FEC-4K exhibits the highest FEC ratio, resulting in approximately 19% overhead while achieving around 8 Mbps throughput at a 0.5s application threshold. Moreover, the FEC packet ratio shows a mere 2% difference between λ=0.5s and λ=1.5s in the case of AR_FEC-4K. However, Fig. 8 illustrates a significant difference of around 3 Mbps in throughput for this scheme. This further emphasizes the importance of the adaptive quality module in AR_FEC, as it plays a crucial role in achieving optimal performance.

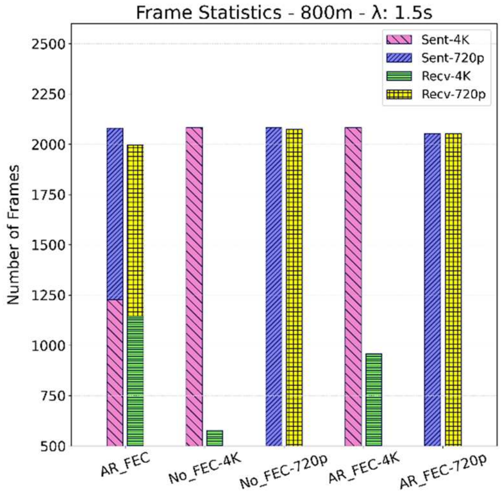
Frame latency: Fig. 9 presents a comparison of the average frame latency among different schemes at various application thresholds within an 800m distance. Despite a significant disparity in the number of delivered 4K frames at λ=0.5s between the AR_FEC method and No_FEC-4K as well as AR_FEC-4K (as depicted in Fig. 7a), AR_FEC demonstrates the lowest latency among them. Additionally, it is worth noting that as video streaming becomes more sensitive to delay, AR_FEC experiences a minor decrease in average latency from 0.074s to 0.054s. In contrast, AR_FEC-4K encounters a substantial decline from 0.217s to 0.063s when transitioning from λ=1.5s to λ=0.5s, respectively. These findings indicate that AR_FEC exhibits enhanced tolerance towards strict delay constraints in the presence of high error rates. Consequently, it can provide reliable video streaming with the highest quality in erroneous real-time environments.

Table IV: FEC packet ratio (%) – Distance=800m

λ **Algorithm**	**0.5s**	**1.0s**	**1.5s**
AR_FEC	1.33	1.26	1.22
AR_FEC–720p	1.45	1.45	1.45
AR_FEC–4K	19.54	18.56	17.94



(a) $\lambda = 0.5s$



(b) $\lambda = 1.5s$

Fig. 7: Received frame quality on different λ at 800m as distance

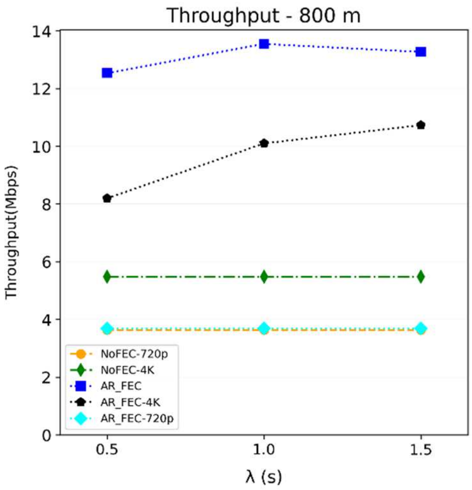


Fig. 8: Throughput on different λ at 800m as distance

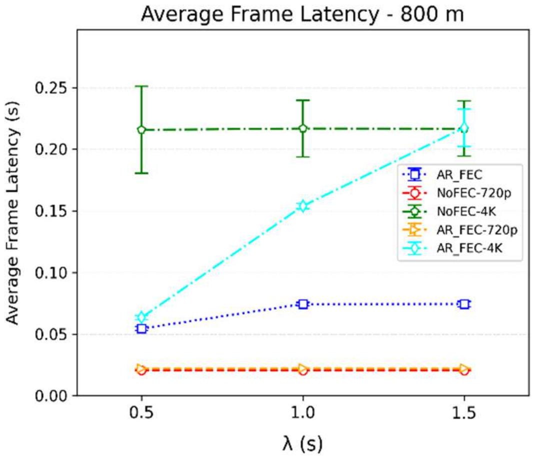


Fig. 9: Average Frame Latency on different λ at 800m as distance

V. CONCLUSION

In this paper, we presented a new adaptive FEC mechanism that is designed for reliable and high-quality multimedia communication over UAVs. We first proposed an adaptive cross-layer FEC scheme in the application layer which adaptively tunes the redundancy level of FEC based on the current packet loss rate on network and estimated traffic load. Then, we introduced a module which dynamically determines the quality of video frames based on the wireless link channel and the predicted data of the queue on the lower layer. Finally, an unequal error protection mechanism was applied to the video packets to increase the video transmission efficiency and the perceived quality of the video at the control station. This is a real-time correction method based on edge computing principles, with video analysis on the network edge, and adaptation on the device-edge. The simulation results validated the efficacy of the proposed AR_FEC method compared to environments without FEC and environment with FEC but without quality adaptation in terms of frame delivery, average latency, and throughput. The results revealed that the AR_FEC protocol was resilient to the strict delay constraints. Moreover, it can provide reliable video communication at the maximum capability of high-quality for error-prone real-time multimedia communications over UAVs while being latency-sensitive.

There are several avenues for future work in the context of cross-layer design for video communication over wireless networks. One is to use multiple queues with different priorities for I-frame and P/B-frame packets. This could be useful in mitigating network congestion by prioritizing I-frame packets during periods of high traffic, or to select I-frames that are more important than the others. Machine learning techniques could also be employed to predict the appropriate level of FEC redundancy more precisely for a given network condition. Evaluation of the energy consumption of the proposed adaptive system would also enhance sustainability and enable the development of energy-conscious strategies for improved efficiency in UAV streaming applications.

REFERENCES

[1] A. Sharma et al., “Communication and networking technologies for UAVs: A survey,” J. Netw. Comput. Appl., vol. 168, p. 102739, 2020, doi: https://doi.org/10.1016/j.jnca.2020.102739.

[2] A. I. Hentati and L. C. Fourati, “Comprehensive survey of UAVs communication networks,” Comput. Stand. Interfaces, vol. 72, p. 103451, 2020, doi: https://doi.org/10.1016/j.csi.2020.103451.

[3] A. Vashisth and R. S. Batth, “An Overview, Survey, and Challenges in UAVs Communication Network,” in 2020 International Conference on Intelligent Engineering and Management (ICIEM), Jun. 2020, pp. 342–347, doi: 10.1109/ICIEM48762.2020.9160197.

[4] K. Durak, M. N. Akcay, Y. K. Erinc, B. Pekel, and A. C. Begen, “Evaluating the Performance of Apple’s Low-Latency HLS,” in 2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP), 2020, pp. 1–6, doi: 10.1109/MMSP48831.2020.9287117.

[5] Z. Liu and Y. Jiang, “Cross-Layer Design for UAV-Based Streaming Media Transmission,” IEEE Trans. Circuits Syst. Video Technol., p. 14, 2021, doi: 10.1109/TCSVT.2021.3125864.

[6] B. Sarvi, H. R. Rabiee, and K. Mizanian, “An adaptive cross-layer error control protocol for wireless multimedia sensor networks,” Ad Hoc Networks, vol. 56, 2017, doi: 10.1016/j.adhoc.2016.12.008.

[7] M. Shabaniyan and E. Akhtarkavan, “A way to Reduce Effects of Packet Loss in Video Streaming Using Multiple Description Coding,” J. Comput. Robot., vol. 13, no. 1, pp. 55–62, 2020.

[8] Y. XIE, Z. LI, J. QI, K. ZHANG, B. ZHANG, and F. QI, “Optimal video communication strategy for intelligent video analysis in unmanned aerial vehicle applications,” Chinese J. Aeronaut., vol. 33, no. 11, pp. 2921–2929, 2020, doi: 10.1016/j.cja.2020.06.009.

[9] X. Chen, Q. Zhou, Z. Wang, Q. Sun, and M. Xu, “Reliable and energy-efficient UAV-assisted air-to-ground transmission: Design, modeling and analysis,” Comput. Commun., vol. 204, pp. 66–77, 2023, doi: https://doi.org/10.1016/j.comcom.2023.03.019.

[10] J. Wang et al., “Bandwidth-efficient live video analytics for drones via edge computing,” Proc. - 2018 3rd ACM/IEEE Symp. Edge Comput. SEC 2018, pp. 159–173, 2018, doi: 10.1109/SEC.2018.00019.

[11] N. H. Motlagh, M. Bagaa, and T. Taleb, “UAV-Based IoT Platform: A Crowd Surveillance Use Case,” IEEE Commun. Mag., vol. 55, no. 2, pp. 128–134, 2017, doi: 10.1109/MCOM.2017.1600587CM.

[12] G. Wu, Y. Miao, Y. Zhang, and A. Barnawi, “Energy efficient for UAV-enabled mobile edge computing networks: Intelligent task prediction and offloading,” Comput. Commun., vol. 150, pp. 556–562, 2020, doi: https://doi.org/10.1016/j.comcom.2019.11.037.

[13] K. Ikeda, Y. Imai, H. Yomo, Y. Kondo, and H. Yokoyama, “Data Transmissions from a Drone Using Multi-AP Wireless LAN: A Field Trial,” Proc. - Int. Conf. Comput. Commun. Networks, ICCCN, p. 2, 2020, doi: 10.1109/ICCCN49398.2020.9209589.

[14] B. Van Den Bergh, A. Chiumento, and S. Pollin, “Ultra-reliable IEEE 802.11 for UAV video streaming: From network to application,” Lect. Notes Electr. Eng., vol. 397, pp. 637–647, 2017, doi: 10.1007/978-981-10-1627-1_50.

[15] N. U. Hasan, W. Ejaz, U. Farooq, I. Baig, and M. Zghaibeh, “Adaptive Error Control Framework for a Multihop Cognitive Radio based UAVs for Disaster Management,” 2019 IEEE 5th Int. Conf. Mechatronics Syst. Robot. ICMSR 2019, pp. 87–91, 2019, doi: 10.1109/ICMSR.2019.8835473.

[16] R. Immich, E. Cerqueira, and M. Curado, “Improving Video QoE in unmanned aerial vehicles using an adaptive FEC mechanism,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8611, pp. 198–216, 2014, doi: 10.1007/978-3-319-10834-6_11.

[17] V. Roca, J. Peltotalo, J. Lacan, and S. Peltotalo, “Reed-Solomon Forward Error Correction (FEC) Schemes,” no. 5510. RFC Editor, Apr. 2009, doi: 10.17487/RFC5510.

[18] “The Network Simulator - NS3,” 2023. https://www.nsnam.org/.

[19] M. A. Khan, I. M. Qureshi, and F. Khanzada, “A hybrid communication scheme for efficient and low-cost deployment of future flying AD-HOC network (Fanet),” Drones, vol. 3, no. 1, pp. 1–20, 2019, doi: 10.3390/drones3010016.

[20] LoungeV Films, “Epic Drone Footage Video,” 2023. https://www.youtube.com/watch?v=R3AKlscrjmQ&t=320s.

[21] “FFmpeg,” 2023. http://ffmpeg.org/.

[22] B. Pinheiro, “Evalvid-NS3,” 2017. https://gitlab.com/gercom-ufpa/evalvid-ns3.