Trading Off Distortion for Delay for Video Transmissions in Wireless Networks

Zi Feng*, George Papageorgiou*, Srikanth V. Krishnamurthy*, Ramesh Govindan†, Tom La Porta‡
*UC Riverside {*zfeng, gpapag, krish*}*@cs.ucr.edu*, †USC *ramesh@usc.edu*, ‡Penn State University *tlp@cse.psu.edu*

***Abstract*—The end-user experience in viewing a video depends on the distortion; however, also of importance is the delay experienced by the packets of the video flow since it impacts the timeliness of the information contained and the playback rate at the receiver. Unfortunately, these performance metrics are in conflict with each other in a wireless network. Packet losses can be minimized by perfectly avoiding interference by separating transmissions in time or frequency; however, this decreases the rate at which transmissions occur, and this increases delay. Relaxing the requirement for interference avoidance can lead to packet losses and thus increase distortion, but can decrease the delay for those packets that are delivered. In this paper, we investigate this trade-off between distortion and delay for video. To understand the trade-off between video quality and packet delay, we develop an analytical framework that accounts for characteristics of the network (e.g. interference, channel variations) and the video content (motion level), assuming as a basis, a simple channel access policy that provides flexibility in managing the interference in the network. We validate our model via extensive simulations. Surprisingly, we find that the trade-off depends on the specific features of the video flow: it is better to trade-off high delay for low distortion with fast motion video, but not with slow motion video. Specifically, for an increase in PSNR (a metric that quantifies distortion) from 20 to 25 dB, the penalty in terms of the increase in mean delay with fast motion video is 91 times that with slow motion video. Our simulation results further quantify the trade-offs in various scenarios.**

I. Introduction

Video communications have become much more popular and prevalent today due to two factors. First, wireless devices have become much more sophisticated (e.g., tablets); second, there are many video applications such as YouTube and streaming services (e.g., Amazon Instant Video) that have become immensely popular. Not only is video of interest in the commercial world, it is also of importance in other contexts such as disaster recovery, tactical networks, and surveillance.

The user experience with respect to a video flow depends on the experienced distortion. The end-to-end video distortion is affected by both the encoding process at the source and the wireless channel induced errors and interference. Also of importance is the packet delay experienced in transferring a video clip; for example, a rescue operation in disaster recovery may depend on the timely delivery of the information. The play back at a receiver could get affected due to large delays experienced by packets (and thus, video frames) [1].

Unfortunately, the two performance metrics viz., distortion and packet delay are in conflict with each other. The channel access mechanism at the link layer affects both of these performance metrics. Among the plethora of such mechanisms, there are those that minimize interference by dispersing transmissions in the frequency or time domain at the cost of higher packet delay. On the other hand, there are access mechanisms that allow the concurrent usage of the channel by multiple transmitters, to decrease packet delay at the expense of potential higher interference. Our goal is to capture the impact of interference management by an access mechanism on the trade-off between packet delay and distortion in wireless multi-hop networks.

Towards this end, we consider a simple channel access scheme (described later) that allows us to do so. Specifically, it allows us to develop a tractable analytical framework that computes the expected values of the video distortion and packet delay experienced in a video flow, taking into account the characteristics of the channel (e.g. interference) and the video content (motion level). In brief, the scheme is characterized by an "access probability" that represents the likelihood with which a node transmits packets on the shared medium. In order to be able to represent the whole gamut of channel access mechanisms, we introduce a scalar parameter which we call *aggressiveness* (α); this is used to tune the channel access probability to the medium. A low value of α (≤ 1) corresponds to a case with low interference wherein the nodes access the channel in such a way as to avoid collisions; the distortion in this case is low. However, the delay is high. If α is high, the delay is lowered. At the same time however, the probability of a collision and therefore of a packet loss increases, thereby resulting in higher video distortion. We provide more details on the channel access model later in the paper.

Our key finding in this work is that the characteristics of the video traffic and in particular, the motion level (described below) affects the distortion versus delay trade-off. The characteristics of the video traffic depend on the video content and the encoder that is used. The motion level of a video clip can be computed through appropriate detection algorithms; typically these algorithms classify a video clip as a fast motion or a slow motion video. In order to capture the effect of the motion level on the distortion vs delay trade-off, we introduce a second parameter in our model, which we call *sensitivity* (s). The sensitivity represents the robustness of the decoder to packet losses. When a video clip is characterized by high motion levels, the output at the encoder exhibits high variability. Therefore, transmission induced errors have a significant impact on the quality of the decoded signal, since it is more difficult to compensate for the lost information. In such cases, the robustness of the decoder to packet losses is small and consequently the sensitivity to such losses is high. The contrary holds for slow motion level video clips.

In summary, our contributions and key findings are:

- We develop an analytical framework to capture the trade-off between distortion and timeliness. The framework computes the expected values of the video distortion and the transfer delay of a video clip while accounting for sys-

978-1-4673-5946-7/13/$31.00 ©2013 IEEE

tem parameters both at the lower link level (interference, channel induced errors) and the application semantics (motion levels and structure of video content).
- We demonstrate, through extensive simulations, the validity of our analytical model. We then quantify via both analysis and simulations the distortion versus delay trade-off for different types of video flows (fast versus slow motion) in a variety of scenarios.
- Our key observation is that trading off high delay for low distortion is important for fast motion video, but not for slow motion video. We find that if the PSNR (Peak Signal to Noise ratio) requirement for a video clip is increased from 20dB to 25dB, the fast motion video clip suffers from a delay increase penalty that is 91 times higher than the penalty incurred with slow motion video. This shows that slow motion video is able to better tolerate packet losses than fast motion video and thus, should be handled differently in a wireless network.

Our work in perspective: Our work demonstrates that application semantics determine the appropriateness of specific protocols in a wireless network. Specifically, we show that interference has a much more significant effect on fast motion video as compared to slow motion video. Channel access schemes that account for this can drastically improve the performance of video flows. One could also envision transmission of fast motion video on congestion free paths to ensure reliability.

Our analytical framework not only provides an understanding of the distortion versus delay trade-off for video flows, but also provides a quick way of obtaining performance results. Simulations on the other hand, take much longer time to provide the same results. To the best of our knowledge, this is the first analysis that characterizes the distortion versus delay trade-offs for video flows in wireless networks.

Organization: In Section II we present related work. The channel access scheme considered for our analysis is described in Section III. Our analysis is detailed in Section IV. In Section V we validate our analysis via simulations and discuss the main implications of our results. We conclude in Section VI.

II. Related Work

Standards like the MPEG-4 [2] and the H.264/AVC [3] provide guidelines on how a video clip should be encoded for transmission over a communication system. Predictive source encoding where motion estimation and motion compensation play important roles in the process, is very popular for video compression. Typically, each video clip is separated into a repetitive structure called a Group of Pictures (GOP). Each GOP consists of *I*, *P* and *B* frames (*B* frames are optional). An *I*-frame can be decoded independently of any other information within the GOP and each of the *P*-frame or *B*-frame use the *I*-frame as a reference to encode information [1]. In the following, we assume an *IPP...P* encoding structure for each GOP.

Handling missing frames is critical for the decoder since frame losses affect the video quality perceived by the end user. Typically, the last decoded frame is substituted for a frame that is lost at the receiver [1], [4]. With this method, the difference between the substitute frame and the original (lost) frame determines the error and hence, the video quality perceived by the user. We discuss this further when we present our analysis and articulate the relationship between the Peak-Signal-to-Noise-Ratio or PSNR (a common metric to assess video quality) and the distance between the missing and substitute frames.

While protocols have been designed towards trying to achieve an optimal trade-off between video quality and delay in wireless networks, there has not been a formal analytical assessment of this trade-off; these protocols do not account for video semantics. In [5], a priority-based video stream scheduling algorithm which considers channel conditions, frame types and traffic burstiness is proposed. In [6] the authors propose a cross layer error control scheme for Fine Granularity Scalable (FGS) video transmission in an IEEE 802.11a network; the scheme computes the optimal combination of modulation and the FEC rate for a video flow. In [7], a cross layer architecture is designed for multicast streaming in wireless LANs. These efforts have only considered single hop wireless networks and to reiterate, have not assessed the impact of video semantics (fast versus slow motion video) on this trade-off.

The impact of packet losses and delay on the video quality depends on the motion level in the video clip. Video motion detection algorithms can be used during the video encoding process (e.g., [8]).Tools such as PhysMo [9] and AForge [10] can also be used to determine the motion level in a video clip.

III. Channel Access System Model

There is a trade-off between the video quality and the timely delivery of a video clip from a source to a destination node. Spreading parallel transmissions in the frequency or time domain eliminates collisions and can provide the best video quality (since the packet losses are then minimized), but may increase the delay in transferring the video content. Many approaches have been designed towards avoiding collisions; these include scheduled access schemes, random access schemes with carrier sensing and exponential backoffs etc.

On the other hand, allowing concurrent transmissions, reduces the delay but increases packet losses due to collisions and interference, and therefore introduce higher video distortion. Under high load, some random access schemes exhibit this behavior. In order to explore the space between mechanisms that manage interference in different ways, we consider a simple channel access scheme that is described below. With this scheme, by tuning a parameter (*aggressiveness*), we are able to roughly characterize the whole gamut of channel access mechanisms. Importantly, the scheme allows us to characterize the trade-off between distortion and packet delay analytically.

We consider a slotted time system, where all the nodes are assumed to be synchronized. We set the slot duration to be equal to the transmission time of the largest allowable packet in the system. At the beginning of each time-slot, each node decides on whether or not it will access the channel in that slot, based on an access probability p_a, given by:

$$p_\alpha = \alpha \cdot p_r, \tag{1}$$

where, p_r is a reference access probability and α is the *aggressiveness*. The reference probability p_r is equal to the frequency with which a node accesses the channel when a perfect schedule is used. In such a case, based on the topology of the network, each node accesses the channel periodically. If this period is n (typically equal to the maximum of the sizes of

the cliques [11] which the node belongs to, as discussed later), then the reference probability is:

$$p_r = \frac{1}{n}. \tag{2}$$

In the simple case where the network topology is a single clique (all nodes can directly communicate with each other), n is equal to the clique size. In general however, a node may belong to a plurality of cliques of different sizes. Computing a transmission schedule for nodes that belong to cliques of different sizes is NP-hard. A perfect schedule guarantees collision free transmissions for each node at the expense of long packet delays (e.g., see [12]).

When $\alpha = 1$, $p_a = p_r$ and access is as per the reference scheme. However, note that since the access is probabilistic, collisions could still occur. When $\alpha > 1$, a node accesses the channel more aggressively compared to the reference scheme. This results in smaller packet delays but also in increased interference and a higher probability of packet collisions. For $0 < \alpha < 1$, the node accesses the channel less frequently; here, the packet delay increases compared to the reference case but packet collisions are avoided with high probability.

Consider a simple case where two nodes that are one hop away belong to different cliques. Let the size of the clique that the first node belongs to, be n_1 and the size of the clique that the second node belongs to, be $n_2 < n_1$. Here, if a perfect schedule is implemented the first node will access channel much less frequently as compared to the second node. Similarly with the reference approach, the exchange of packets between these two nodes is slowed down by the first clique, since

$$p_{r,1} = \frac{1}{n_1} < \frac{1}{n_2} = p_{r,2}. \tag{3}$$

Increasing the aggressiveness of the first node by setting $\alpha > 1$ causes the first node to access the channel with probability

$$p_{\alpha,1} = \alpha \cdot p_{r,1} > p_{r,1} \tag{4}$$

This reduces the packet delay between the nodes, but possibly at the expense of increased collisions in the first clique (depending on the access probabilities of nodes in that clique).

Note here that p_r is particular to each node i ($p_{r,i}$); it is essentially the reciprocal of the maximum of the sizes of the cliques to which the node belongs. To compute the reference access probabilities $p_{r,i}$, maximal clique enumeration is essential. A maximal clique is a clique in which all the composers are connected to each other and there is no other clique that contains this clique. There are various algorithms that compute the maximal cliques in an undirected graph. Any approach could be used here. In our performance evaluations in Section V we use the well-known Bron-Kerbosch algorithm [13]. Computationally less complex algorithms can be used in practice at the cost of optimality (e.g., [14]).

Other assumptions: We do not consider retransmissions or higher layer protocol effects to keep the analysis and the study simple. We will consider these issues in the future work. Our work however, does not require a node to have something to send all the time. Nodes could also be simultaneously forwarding multiple video streams.

IV. Our Analytical Framework

We develop an analytical framework to compute the expected distortion of a video flow over a multi-hop static wireless network in the presence of interference from other video connections. We first compute the expected value of the Signal-to-Interference-and-Noise-Ratio (SINR) which we use to find the packet success rate. We then translate packet losses to video frame losses and hence to video transmission distortion.

A. SINR Computation

Consider the communication between a pair of nodes i and j that are one hop away. The SINR for this communication is:

$$\text{SINR} = \frac{P_{ij}}{N + \sum_{k \in \mathcal{I}} P_{kj}}, \tag{5}$$

where, N is the noise power, $\mathcal{I}$ is the set of interfering nodes and P_{sr} is the received power at node r of a signal transmitted by node s, for any pair of nodes s, r. Using the Friis propagation loss model [15], the received power P_{sr} can be written as:

$$P_{sr} = \frac{P_s G_s G_r}{L} \cdot \left(\frac{\lambda}{4\pi}\right)^2 \cdot \frac{1}{d_{sr}^2}, \tag{6}$$

where P_s is the transmission power, G_s and G_r are the transmission and reception gains respectively, λ is the wavelength of the signal, d_{sr} is the distance between the sender s and the receiver r and L is the system loss. We assume that all nodes use the same transmission power P, i.e. $P_s = P$ for all s.

To compute the expected value of the SINR we assume that R is the maximum communication range. If a uniform node distribution is assumed, the probability density function of the distance is given by:

$$f_d(t) = \frac{2t}{R^2}, \quad 0 < t < R. \tag{7}$$

The expected value of the SINR can then be computed as:

$$\text{E}[\text{SINR}] = \sum_{l=0}^{m} \text{E}[\text{SINR} \mid l \text{ interferers}] \cdot P\{l \text{ interferers}\}, \tag{8}$$

where we assume that the number of neighbors is at most m.

The number of interferers follows a binomial distribution with parameters m and p_α:

$$P\{l \text{ interferers}\} = \binom{m}{l} p_\alpha^l \ (1-p_\alpha)^{m-l}, \ l = 0, 1, \ldots m, \tag{9}$$

where p_α is given by (1). In general, the network topology may be such that the neighbors of the receiving node belong to cliques of different sizes, and therefore access the channel with different probabilities. In that case, we can compute the probability that l interferers exist, in (9), considering $p_\alpha^{\min}$ and $p_\alpha^{\max}$ as the minimum and maximum values, respectively, of the access probabilities from among the neighbors of the receiver. Using these values for the access probability, we can determine lower and upper bounds for the E[SINR].

Since SINR takes positive values, its conditional expected value given the number of interferers can be computed as:

$$\text{E}[\text{SINR} \mid l \text{ interferers}] = \int_0^\infty P\{\text{SINR} > x \mid l \text{ interferers}\} \ dx \tag{10}$$

Using (5) and (6), the conditional tail distribution of SINR can

be written as:

$$P\{\ \text{SINR} > x \mid l \text{ interferers}\}$$
$$= P\left\{ \frac{\left(\frac{1}{d_{ij}}\right)^2}{\frac{N\cdot L\cdot(4\pi)^2}{P\cdot G_t\cdot G_r\cdot\lambda^2} + \sum_{k=1}^{l}\left(\frac{1}{d_{kj}}\right)^2} > x \mid l \text{ interferers} \right\}$$
$$= \underbrace{\int_0^R \cdots \int_0^R}_{l\ \ times} P\left\{ d_{ij} < \left(\frac{xNL(4\pi)^2}{PG_tG_r\cdot\lambda^2} + \sum_{k=1}^{l} t_k^{-2} \right)^{-\frac{1}{2}} \right.$$
$$\left. \mid l \text{ interferers} \right\} \cdot f(t_1,\ldots,t_l)\ dt_1\ldots dt_l, \qquad (11)$$

where $f(\cdot,\cdot,\ldots,\cdot)$ is the joint probability density function of the distances from the l interferers to the destination node j. Using (7) and assuming statistical independence between these distances we have that:

$$f(t_1,\ldots,t_l) = \prod_{k=1}^{l}\frac{2t_k}{R^2} = t_1\cdots t_l\cdot\left(\frac{2}{R^2}\right)^l. \qquad (12)$$

Using (7) and (12), (11) becomes

$$P\{\ \text{SINR} > x \mid l \text{ interferers}\}$$
$$= \left(\frac{2}{R^2}\right)^{l+1}\underbrace{\int_0^R\cdots\int_0^R}_{l\ \ times}\int_0^{U(x)} t\cdot t_1\cdots t_l\ dt\ dt_1\ldots dt_l, \qquad (13)$$

where $U(x) = \left(\frac{xNL(4\pi)^2}{PG_tG_r\cdot\lambda^2} + x\sum_{k=1}^{l} t_k^{-2}\right)^{-\frac{1}{2}}$. From (10) and (13) we have:

$$\text{E}\left[\text{SINR} \mid l \text{ interferers}\right]$$
$$= \left(\frac{2}{R^2}\right)^{l+1}\int_0^{\infty}\underbrace{\int_0^R\cdots\int_0^R}_{l\ \ times}\int_0^{U(x)} t\cdot t_1\cdots t_l\ dt\ dt_1\ldots\ dt_l\ dx. \qquad (14)$$

From (8), (9), and (14), we get

$$\text{E}\left[\text{SINR}\right] = \sum_{l=0}^{m}\left\{\binom{m}{l}\ p_\alpha^l\ (1-p_\alpha)^{m-l}\right.$$
$$\left.\times\left(\frac{2}{R^2}\right)^{l+1}\int_0^{\infty}\underbrace{\int_0^R\cdots\int_0^R}_{l\ \ times}\int_0^{U(x)} t\cdot t_1\cdots t_l\ dt\ dt_1\ldots\ dt_l\ dx\right\}, \qquad (15)$$

which we solve numerically. We point out here that we have the tail distribution for SINR, the outermost integral from 0 to ∞ converges fairly quickly i.e., as x increases $P\{SINR > x\} \to 0$.

B. Packet Success Rate

To compute the packet success rate we consider the IEEE 802.11b physical layer for high-rate (11Mbps), where complementary code keying (CCK) is adopted. We consider this version of the protocol since the analysis in [16], [17], directly provides the packet success rate; note that it is easy to incorporate other versions of 802.11 (a, g), by considering the appropriate modulation/encoding schemes. The packet success rate, p_s, as a function of the expected value S of the SINR is:

$$p_s(S) = [1 - P_{e,1}(S)]^B \qquad (16)$$

where

$$P_{e,1}(S) = 1 - \left(\int_{-\sqrt{2S}}^{\infty}\left[2\Phi\left(x+\sqrt{2S}\right)-1\right]^{(N-1)} \times \frac{\exp\left(-x^2/2\right)}{\sqrt{2\pi}}\ dx\right)^2. \qquad (17)$$

In the above, B is the length of the packet in bits, $N = K/2$, K is the number of biorthogonal signals used and $S = \text{E[SINR]}$. Equations (16) and (17) provide a mapping from the expected value of the SINR to the packet success rate p_s.

C. Video Frame Success Rate

We map the packet success rate p_s to the video frame (referred to as simply 'frame') success rate P_f, which denotes the probability a frame is successfully transmitted over one hop. As was mentioned in Section II, we assume that each GOP has an *IPP...P*-structure.

If n is the number of packets in each frame, then to successfully decode a frame, (a) the first packet of that frame needs to be successfully received, and (b) $0 \le s \le n-1$ of the remaining $n-1$ packets need also be received successfully. The success probability of a frame is given by:

$$P_f = p_s\ \sum_{i=s}^{n-1}\binom{n-1}{i} p_s^i\,(1-p_s)^{n-1-i}. \qquad (18)$$

We call the parameter s the *sensitivity* of the decoder to packet losses. It is the minimum number of packets that the decoder needs to receive without errors in order to decode the corresponding frame correctly. As discussed in Section V, the sensitivity is associated with the video content itself and specifically with the motion level. When a video clip is characterized by high (or fast) motion, the sensitivity s has a higher value compared to a low (or slow) motion video. This is because in a high motion video clip, the difference between successive frames in the GOP structure is large and the loss of a frame has a higher impact on the overall video quality.

In general, the *I*-frame is much larger than a *P*-frame. Thus, the number of packets in the *I* and *P* frames differ. As a result, the frame success probabilities for an *I* and a *P*-frame also differ. We denote by P_I the success probability of an *I*-frame and by P_P the corresponding success probability of a *P*-frame.

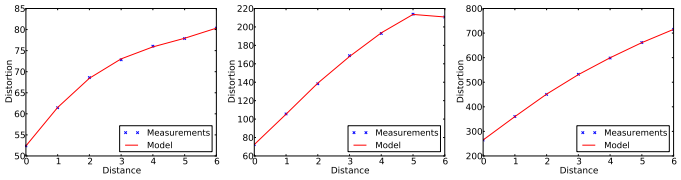
We have validated this model via extensive experiments using the EvalVid tool.

D. Distortion

Let the GOP structure contain $G-1$ *P*-frames that follow the *I*-frame. We consider predictive source coding where, if the i^{th} frame is the first lost frame in a GOP, then the i^{th} frame and all its successors in the GOP are replaced by the $(i-1)^{st}$ frame at the decoder. If the *I*-frame of the GOP cannot be decoded correctly, then the whole GOP is considered unrecoverable and is ignored. In this case, these lost video frames are replaced by the most recent frame from a previous GOP that is correctly received. In all cases, the similarity between the missing frames and the reference frame (substitute frame) affects the distortion [18].

We compute the video distortion as the *mean square error* of the difference between the missing frame and the substitute frame. Initially, we focus on a one-hop transmission and then extend our analysis for multi-hop connections. Specifically, we have the following cases:

Case 1 – Intra-GOP distortion: The *I*-frame of the current GOP is successfully received. The distortion for the current GOP depends on which, if any, of the *P*-frames of the GOP



(a) Low motion (b) Medium motion (c) High motion

Fig. 1: Average distortion with distance.

cannot be decoded without errors. If the first unrecoverable *P*-frame is the i^{th} frame in the GOP, the corresponding distortion is given by [4]:

$$d_i = (G-i)\frac{i \cdot G \cdot d_{min} + (G-i-1)\cdot d_{max}}{(G-1)\cdot G}, \tag{19}$$

for $i = 1, 2, \ldots, (G-1)$, where $d_{\max}$ is the maximum distortion when the first frame is lost and d_{min} is the minimum distortion when the last frame is lost. The values of $d_{\max}$ and $d_{\min}$ depend on the actual video content and have to be evaluated experimentally. The probability P_i that the i^{th} frame is lost is

$$P_i = P_I P_P^{i-1}(1-P_P), \quad i = 1, 2, \ldots, (G-1). \tag{20}$$

Using (19) and (20), the expected value of the distortion can be computed to be:

$$D^{(1)} = \sum_{i=1}^{G} d_i \cdot P_i \tag{21}$$

Case 2 – Inter-GOP distortion: The *I*-frame of the current GOP is lost and a frame from a previous GOP is used as the reference frame. In this case, the difference between the reference frame and the missing frames determine the distortion.

Similar to the work in [18], we expect to see the motion characteristics of the video affecting the distortion. To capture the dependence of the inter-GOP distortion on the motion level of the video we perform a set of experiments and we use the collected results to statistically describe this association.

Specifically, we select a set of video clips from [19] and categorize them into three groups according to their motion level: low, medium and high. All video clips have 300 frames each, with a frame rate of 30 frames per second. We use FFmpeg [20] to convert the video clips from the initial, uncompressed YUV format to the MP4 format. Then, we artificially create video frame losses in order to achieve reference frame substitutions from various distances. Finally, we use the Evalvid toolset [21] to measure the corresponding video distortion.

In Fig. 1 the dependence of the average distortion on the distance between the missing frame and the substitute is shown for the three categories. In order to use these empirical results in other experiments, we approximate the observed curves with polynomials of degree 5 using a multinomial regression. In particular, we define the approximate distortion $D^{(2)}$ as a function of the distance d:

$$D^{(2)}(d) = a_5 d^5 + a_4 d^4 + a_3 d^3 + a_2 d^2 + a_1 d^1 + a_0 d^0 \tag{22}$$

and compute the coefficients $a_0, \ldots, a_5$, through the regression.

Case 3 – Initial GOP: The *I*-frame of the current and all previous GOPs (including the first GOP) are lost. In this case the distortion D is maximized. If $\{D_{\max}^{(1)}, D_{\max}^{(2)}, \ldots, D_{\max}^{(\|\mathcal{G}\|)}\}$, where $\mathcal{G}$ is the set of all GOPs in the video clip, is the set of the maximum distortion values in all GOPs, then

$$D^{(3)} = \max_{k \in \mathcal{G}} D_{\max}^{(k)}. \tag{23}$$

E. Single-hop Transmission

Suppose the video clip has N GOPs and each GOP consists of an *I*-frame followed by $G-1$ *P*-frames. For each GOP of the video clip define the state $S_i, i = 1, 2, \ldots, N$ such that

$$S_i \in \mathcal{S} = \{0, 1, 2, \ldots, (G-1), G\}. \tag{24}$$

The state S_i for the i^{th} GOP indicates which is the first unrecoverable frame in that GOP. Specifically,

$$S_i = \begin{cases} 0, & I\text{-frame is lost,} \\ k, & k^{th}\ P\text{-frame is lost, } 1 \le k \le (G-1), \\ G, & \text{none of the frames is lost,} \end{cases} \tag{25}$$

for $i = 1, 2, \ldots, N$. The initial state for each GOP is G. The transition probability $p_i(G, q)$ of the state S_i from G to $q \in \mathcal{S}$ is

$$p_i(G,q) = \begin{cases} 1-P_I, & q = 0, \\ P_I P_P^{k-1}(1-P_P), & q = k, 1 \le k \le (G-1), \\ P_I P_P^{G-1}, & q = G, \end{cases} \tag{26}$$

for $i = 1, 2, \ldots, N$.

To compute the expected value of the distortion for the transmission of the video clip over the wireless channel we need to consider the states of all the GOPs. We define the vector $\boldsymbol{S}$ as

$$\boldsymbol{S} = (S_1, S_2, \ldots, S_N) \in \mathcal{S} \times \mathcal{S} \times \cdots \times \mathcal{S}. \tag{27}$$

The initial state of $\boldsymbol{S}$ is $\boldsymbol{G} = (G, G, \ldots, G)$ and its transition probability $p(\boldsymbol{G}, \boldsymbol{q})$ to a new state $\boldsymbol{q} = (q_1, q_2, \ldots, q_N)$ is

$$p(\boldsymbol{G}, \boldsymbol{q}) = \prod_{i=1}^{N} p_i(G, q_i). \tag{28}$$

The overall distortion for the video clip transmission depends on the final state $\boldsymbol{q}$. As was discussed earlier in Case 2, the distortion of a GOP may depend not only on the frame losses in that GOP but on losses in previous GOPs as well. Therefore, if D_i is the distortion of the i^{th} GOP, it is a function of the vector $\boldsymbol{q}$ and not only of the i^{th} component of $\boldsymbol{q}$. We define the random variable $\boldsymbol{D}(\boldsymbol{q})$ as:

$$\boldsymbol{D}(\boldsymbol{q}) = (D_1(\boldsymbol{q}), D_2(\boldsymbol{q}), \ldots, D_N(\boldsymbol{q})) \tag{29}$$

consisting of the distortions of each of the GOPs of the video clip. Using (28) we have:

$$\mathrm{E}[\boldsymbol{D}] = (\mathrm{E}[D_1], \mathrm{E}[D_2], \ldots, \mathrm{E}[D_N]) = \sum_{\boldsymbol{q}} p(\boldsymbol{G}, \boldsymbol{q}) \boldsymbol{D}(\boldsymbol{q}) \tag{30}$$

The average distortion that corresponds to the video file is

$$\bar{D} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{E}[D_i]. \tag{31}$$

F. Multi-hop Transmission

For the case of a multi-hop transmission we need to compute the transition probability from the initial state $\boldsymbol{G} = (G, G, \ldots, G)$ at the source node to a final state $\boldsymbol{q} = (q_1, q_2, \ldots, q_N)$ at the destination node. To do this we first compute the transition probability $p_i^{(j)}(n, m)$ of the i^{th} GOP (the likelihood that the first unrecoverable frame is m after this hop, given that it was n upon reaching this hop) at the j^{th} hop along the path from the source to the destination.

$$p_i^{(j)}(n,m) = \begin{cases} 0, & \text{if } m > n, \\ 1, & \text{if } m = n = 0, \\ P_I P_P^m, & \text{if } m = n \text{ and } n > 0, \\ 1-P_I, & \text{if } m = 0 \text{ and } n > 0, \\ P_I P_P^{m-1}(1-P_P), & \text{if } 0 < m < n, \end{cases} \tag{32}$$

for $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, k$, where N is the total number of GOPs and k is the number of hops along the path.

Then, the transition probability for the j^{th} hop is :

$$p^{(j)}(\boldsymbol{n}, \boldsymbol{m}) = \prod_{i=1}^{N} p_i^{(j)}(n_i, m_i). \quad (33)$$

for $\boldsymbol{n} = (n_1, n_2, \ldots, n_N)$ and $\boldsymbol{m} = (m_1, m_2, \ldots, m_N)$, where n_i is the current state of the i^{th} GOP and m_i is the next state of the i^{th} GOP.

For the multi-hop transmission, the transition probability $p_{\text{mu}}(\boldsymbol{G}, \boldsymbol{q})$ from the initial state $\boldsymbol{G} = (G, G, \ldots, G)$ at the source node to a final state $\boldsymbol{q} = (q_1, q_2, \ldots, q_N)$ at the destination node, is given by

$$p_{\text{mu}}(\boldsymbol{G}, \boldsymbol{q}) = \sum_{\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots \boldsymbol{q}_{k-1}} p^{(1)}(\boldsymbol{G}, \boldsymbol{q}_1) p^{(2)}(\boldsymbol{q}_1, \boldsymbol{q}_2) \cdots p^{(k)}(\boldsymbol{q}_{k-1}, \boldsymbol{q}) \quad (34)$$

As is the case for the single hop transmission, the overall distortion depends on the final state $\boldsymbol{q}$. Therefore, the average distortion $\bar{D}$ can be computed using (29), (30) and (31), where instead of using the one-hop transition probability $p(\boldsymbol{G}, \boldsymbol{q})$ we use the multi-hop transition probability $p_{\text{mu}}(\boldsymbol{G}, \boldsymbol{q})$.

G. Mapping Distortion to PSNR

In all the results we present in the sequel, we use the Peak Signal-to-Noise Ratio (PSNR) which is an objective video quality measure [1]. The relationship between distortion and PSNR (in dB) is given by [1]:

$$PSNR = 10 \log_{10} \frac{255}{\sqrt{\text{Distortion}}} \quad (35)$$

H. Delay

For the single hop communication, where a node accesses the channel with probability p_α give by (1), the average delay of a packet is given as the expected value of a geometric distribution with parameter p_α:

$$\text{E[Delay]} = \text{time\_slot\_duration} \cdot \frac{1}{p_\alpha}. \quad (36)$$

In the multi-hop case, a packet successfully received by the destination traverses each intermediate hop. At each hop, the delay is given by (36). Hence, the overall delay is:

$$\text{E[Delay]} = \text{time\_slot\_duration} \cdot \sum_i \frac{1}{p_\alpha^{(i)}}, \quad (37)$$

where $p_\alpha^{(i)}$ is the access probability at the i^{th} hop. This can be directly used to compute the delay incurred in transferring the entire video clip.

Key Observations: It is evident from (37) that as p_α increases, the delay decreases. As a consequence however, the likelihood of an increased interference (see (9)) and thus, frame loss increases. However, for a given frame loss rate, the distortion depends on the sensitivity (see Section IV-C). For slow motion video, a given value of the frame loss rate results in lower distortion than for fast motion video. Thus, for a given distortion requirement, it is possible to achieve a lower delay through more aggressive scheduling for slow motion video as compared to fast motion video.

V. Evaluations

In this section, we seek to quantify the trade-off between distortion and delay. We also validate our analytical model via extensive simulations. The analytical results take an order of magnitude time (minutes) less than the simulation counterparts (hours); the simulation is complicated by the presence of both the network functions (channel access, PHY), and the application semantics (video).

Simulation set up and metrics: We use the network simulator ns-3 [22]. We modify the IEEE 802.11 module therein, to implement our scheme. We implement a slotted-time system to control the shared medium where each node is granted access to the channel with probability $p_\alpha = \alpha \cdot p_r$. As was described in Section III, the reference probability $p_r = \frac{1}{n}$ where n is the maximum clique this node belongs to. In our implementation, the network topology information is collected when the network is set up and the Bron-Kerbosch clique enumeration algorithm [13] is used to compute the maximal cliques across the network.

We also use EvalVid [21], which is a popular tool-set for evaluating the quality of video transmitted over a real or simulated network. The tool allows us to gather performance statistics with metrics such as the Peak-Signal-to-Noise-Ratio (PSNR) and the Mean Opinion Score (MOS) [23]. The PSNR metric compares the maximum possible signal energy to the error energy. The PSNR of each frame can be mapped to the MOS, which is quantified on a scale of five grades (from "bad" to "excellent"). Note that the lower the PSNR and MOS, the higher the distortion; we use these metrics since the tool we use provides these measures directly.

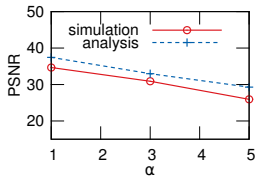
For each video flow in a simulation, we produce a sequence of files. We start with an initial video file (taken from [19]), composed of YUV frames. Using EvalVid we transform the YUV video to the MP4 format and then to the MPEG4 format. To simulate how the video would be transmitted over the real network, EvalVid provides a tool called `mp4trace` which creates a log from an attempted MPEG4 video transmission over a real network. We use this log file as an input to our ns-3 implementation. In the end, based on the log file, the EvalVid tool-set and the packet losses produced by the simulation, we reconstruct the video file as it is 'supposed to be' received at the destination node in a real network. Comparing the reconstructed video with the reference video file, we can measure the video quality degradation caused by the transmission over the network. Finally, note that we use the AForge tool to classify a video flow as fast or slow motion video [10].

The video encoding parameters we use are in Table I. We focus on three metrics: (i) the PSNR, which is an objective quality measure, (ii) the MOS, which is a subjective quality metric, and (iii) the delay distribution of each video flow.

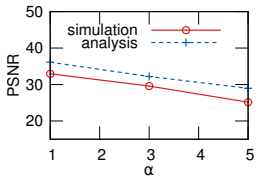
An Example: First, we illustrate our approach by considering the simple network topology in Fig. 2(d). Since the topology is known, we refine our analysis to this specific topology (use topology specific parameters rather than estimated averages) to compute the average values of PSNR and delay. We use a video clip with slow motion for this experiment. There are two maximum cliques in this network: "clique-1" of size 8 and "clique-2" of size 5. In this scenario, there are two video traffic

TABLE I: Video Encoding Parameters

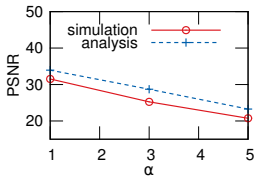
GOP Size	15
Frame Size	CIF (352×288)
Frames per second	30
MTU	1000



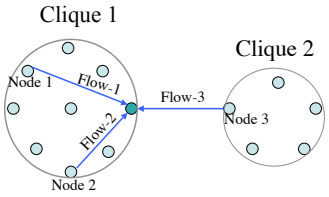
(a) Average PSNR for flow-1



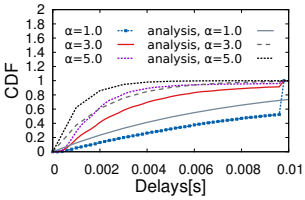
(b) Average PSNR for flow-2



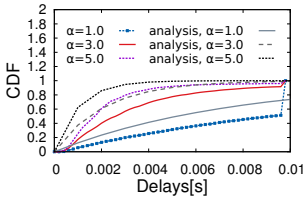
(c) Average PSNR for flow-3



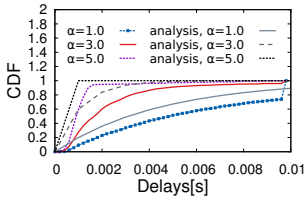
(d) Example network topology.



(e) Delay distribution for flow-1



(f) Delay distribution for flow-2



(g) Delay distribution for flow-3

Fig. 2: Average PSNR and distribution of the end-to-end delay for a simple network topology.

flows originating in "clique-1" and one video traffic flow from "clique-2" both destined to the same receiver. The reference access rates of these flows are $p_{r,1} = p_{r,2} = \frac{1}{8}$, and $p_{r,3} = \frac{1}{5}$.

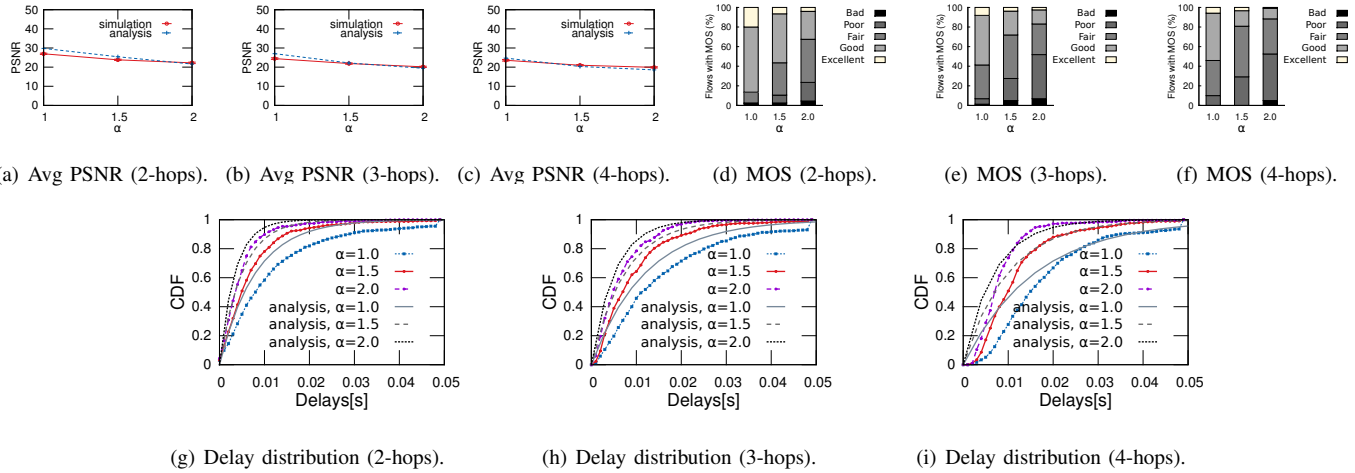
In Fig. 2 the average PSNR and the cumulative distribution of the end-to-end packet delay is shown for each of the three video flows for different values of the aggressiveness α. The effect of the aggressiveness α on PSNR is shown in Figs. 2(a), 2(b) and 2(c). As α increases, each transmitting node accesses the channel more frequently thus increasing the number of packet collisions and lowering the video quality. On the other hand, this increase in α has the opposite effect on the packet delay. As shown in Figs. 2(e), 2(f) and 2(g) the larger the value of α, the less the delay that each packet experiences in the network. For example, if we focus on flow-1, we notice from Fig. 2(a) that the average PSNR is about 35dB when $\alpha = 1.0$. The average PSNR drops to 26dB when $\alpha = 5.0$, i.e. the PSNR is decreased by 26%. On the other hand, Fig. 2(e) shows that when $\alpha = 1.0$, 30% of the video packets are delayed by 0.004 seconds, while 90% of the packets are delayed by 0.004 seconds when α increases to 5.0. If a PSNR value of 26dB is acceptable for a specific application, the aggressiveness α can be tuned to be equal to 5.0 to minimize the delay. However, if the video application has strict constraints regarding the video quality, a lower value of α can be used at the expense of larger delays. Finally, note that the analytical results match very well with simulation results, both for the PSNR and the delay distribution.

More general cases: Next, we consider general, randomly generated network topologies. In all our experiments, we focus on wireless multi-hop networks that cover a geographical area of 800×400 m^2. This area is separated into eight sub-areas, each of which is of size 200×200 m^2. In each sub-area, the nodes are distributed according to a Poisson random field. On the average there are 40 nodes in the network. Each node uses our MAC protocol stack with the ns-3 implementation of the IEEE 802.11b physical layer and the Friss propagation model [15]. The maximum transmission range is about 150 m. We run the experiments for 40 different random topologies and for each video flow, we compute the average PSNR, the MOS of each flow, and the end-to-end delay. The PSNR that is computed is the average PSNR from among all the transmissions. The MOS results show the percentage of flows that fall in the five scales of MOS ratings ("Bad", "Poor", "Fair", "Good", "Excellent"). For the delay we find the cumulative distribution of the end-to-end delay for each flow.

Slow Motion Videos: The first set of experiments are for transmissions of video files with slow motion levels. We consider three different cases, where the flows consist of (i) 2 hops, (ii) 3 hops and (iii) 4 hops. We omit the results from the 1 hop case due to space constraints; the results are similar to that with the example described earlier. In each case, we create flows by choosing the source-destination pairs at random from among the nodes that satisfy in each case, our constraint regarding the hop-count. To determine saturation conditions, we increase the number of flows gradually considering an one-hop scenario, with $\alpha = 1.0$. We find here that 10-12 concurrent transmissions are where the network achieves its capacity with acceptable quality for each video flow. If we consider each hop in a multi-path connection as leading to a single transmission, the total number of concurrent transmissions in the multi-hop case can be roughly estimated to be `hop_count`$\times$`number_of_flows` (is likely to be true for the bottleneck links). Thus, to approximately reach the capacity, we use 5 flows for the 2-hop experiments, 4 flows for the 3-hop experiments and 3 flows for the 4-hop experiments.

First, we notice from Fig. 3 that the analytical model provides a good estimation of the video quality and the delay in all three cases and for the entire range of the considered values of the aggressiveness α. Another observation is that for multi-hop video transmissions, increasing α carefully can yield significant delay gains if one could bear a slight decrease in PSNR. For example, in Fig. 3(c) if we allow the average PSNR value to drop from 24dB to 20dB (by increasing α), a large improvement in delay is possible as shown in Fig. 3(i). When $\alpha = 1.0$, only 27.7% packets are delayed by 0.01 seconds; when $\alpha = 2.0$, 73.6% packets are delayed by 0.01 seconds.

In Fig. 3(d)-3(f) the percentage of flows within the five MOS ratings for the three cases are shown. An increase in α decreases the percentage of flows with 'excellent' quality. Simultaneously,



(a) Avg PSNR (2-hops). (b) Avg PSNR (3-hops). (c) Avg PSNR (4-hops). (d) MOS (2-hops). (e) MOS (3-hops). (f) MOS (4-hops).

(g) Delay distribution (2-hops). (h) Delay distribution (3-hops). (i) Delay distribution (4-hops).

Fig. 3: Average PSNR, delay distribution and MOS results for slow motion videos.

the percentage of flows with 'poor' and 'bad' quality increases. Furthermore, as the path length increases, the percentage of flows with higher quality decreases as one might expect.

As discussed in Section IV, the parameter s in (18) indicates the *sensitivity* of the decoder to packet losses. When computing the analytical results of the first set of experiments for slow motion video transmissions, we set s to 0. Since the motion level of the video content is low, the difference from frame to frame inside a GOP and across the GOPs is expected to be small. In this case, the substitution of a missing video frame by a previously, correctly received frame incurs minimum distortion. Therefore, the decoder is less sensitive to packet losses as compared to fast motion video as we see next.

Fast motion videos: We repeat the same set of experiments with fast motion video clips. The PSNR, delay and MOS results are shown in Fig. 4. For this set of experiments we keep the same number of concurrent flows as we had for slow motion video transmissions, i.e. 5 flows for 2-hop, 4 flows for 3-hop, and 3 flows for 4-hop connections.

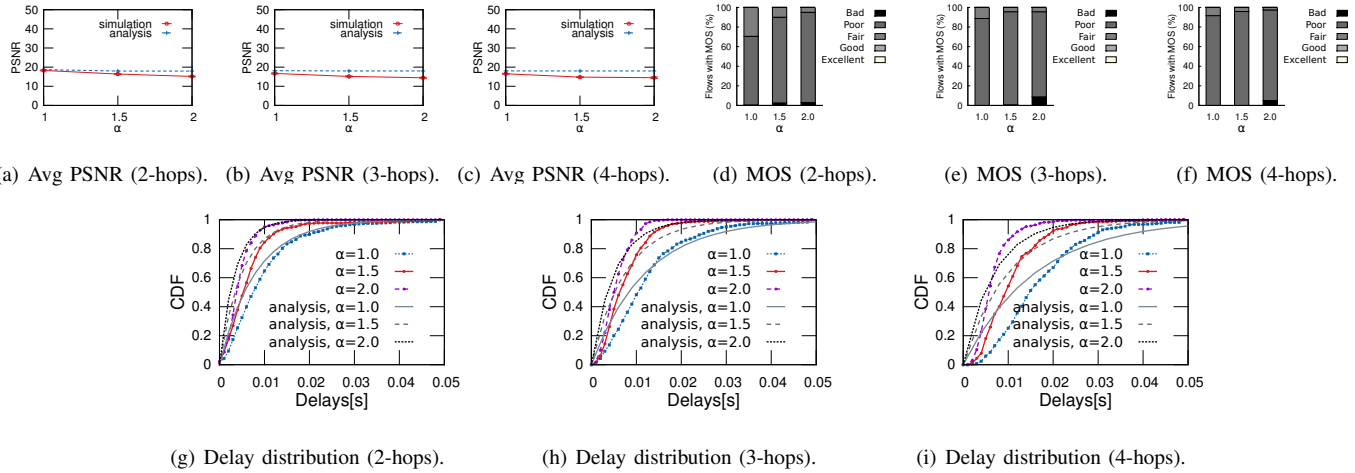
The results in Fig. 4 again show that our analytical model provides a good estimation of the video quality and delay for fast motion transmissions as well. In Figs. 4(a)-4(c) we observe that even when α is 1.0, the average PSNR value is much lower than the average PSNR value in Figs. 3(a)-3(c). This is because the fast motion video reconstruction is much more sensitive to packet losses as compared to slow motion video. Also as shown in Fig. 4(a), when $\alpha = 2.0$, the average PSNR value is as low as 15dB. If we want to keep the average PSNR value above 20dB, we need to tune α towards smaller values for fast motion video transmissions. With regards to the delay, the general effect of α is similar to that with slow motion video. Comparing the results in Figs. 4(a)-4(c), the decrease in video quality is not as prominent with an increase of path length as with slow motion. This is because the video distortion is already high (approaching the maximum) and thus, the averaged PSNR value does drop much further when the path length grows.

Figs. 4(d)-4(f) show the percentage of flows within each category of MOS for fast motion video transmissions. We observe that very few flows are of quality higher than "Good". This is again due to the sensitivity of fast motion video decoding to packet losses. As shown in Fig. 4(e) even with $\alpha = 1.0$, half of the flows are of "fair" quality. This means that we should tune α more conservatively.

We wish to point out here that in the case of fast motion video content, the sensitivity of the decoder to packet losses is higher. A substitution of a missing frame by another frame, even from the same GOP, typically results in a significant increase in distortion. Therefore, we set s to 8 for fast motion video.

Delay for a target PSNR : Next, we seek to estimate the delay incurred for a target PSNR value with both slow and fast motion video. Specifically, we seek to determine the delay increase penalties with fast and slow motion video, for increasing the PSNR value (decreasing distortion). We keep 10 flows in the network and tune the α to different levels, *targeting* average PSNR values of 20dB and 25dB (to reflect a desired video quality). Fig. 5(a) shows the values of α needed to reach the targeted PSNR values for both slow and fast motion video clips. To reach a PSNR of 20dB, we need to tune α to 1.0 for fast motion video but can be more aggressive and set α to 3.0 for slow motion video. To achieve a PSNR value of 25dB, for fast motion video, α should be 0.2, which is a value smaller than 1.0. (In general, α should be tuned towards smaller values if the video is of fast motion, since it is more sensitive to packet losses.) Fig. 5(b) shows the average packet delays associated with the targeted PSNR values of 20dB and 25dB for both slow and fast motion video clips. We see that the packet delays for slow motion video is much lower compared to that for fast motion video. To achieve a target PSNR of 20dB, average delays of 0.015 seconds and 0.003 seconds are incurred for fast and slow motion video, respectively. If the target PSNR is now increased to 25dB, the average delay increases to 0.005 seconds for slow motion video, and a staggering 3.2 seconds for fast motion video (increases by 99.8 %). *The delay increase penalty is 91 times higher with fast motion video as compared to slow motion video, for achieving this PSNR increase.* Thus, if one were to tolerate a slightly lower PSNR requirement with fast motion video, the delay could be drastically reduced. However, this reduction is not that prominent for slow motion video.

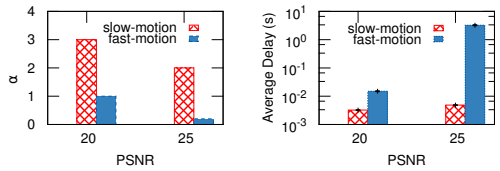
Visual quality: Finally, from the reconstructed videos at the receiver for both slow and a fast motion we have visually verified that the video quality of the slow motion video is clearly better than the fast motion video, for each considered



(a) Avg PSNR (2-hops). (b) Avg PSNR (3-hops). (c) Avg PSNR (4-hops). (d) MOS (2-hops). (e) MOS (3-hops). (f) MOS (4-hops).

(g) Delay distribution (2-hops). (h) Delay distribution (3-hops). (i) Delay distribution (4-hops).

Fig. 4: Average PSNR, delay distribution and MOS results for fast motion videos.



(a) α vs PSNR. (b) Delay vs PSNR.

Fig. 5: Value of α and mean delays for target PSNR.

value of α. This is consistent with the previously presented PSNR and MOS measurements results. However, due to space constraints we do not present snapshots of these videos.

VI. Conclusions

Given the increasing popularity of video communications over wireless networks, we examine two key performance metrics associated with video flows, viz., distortion and delay. Unfortunately, these conflict with each other; one can increase distortion at the expense of delay and vice versa. Towards understanding the trade-off between distortion and delay we develop an analytical framework. We validate our framework with extensive simulation experiments. Since the sensitivities of the decoder to packet losses with respect to fast and slow motion video differ, we find that the motion level of a video clip affects this trade-off. Specifically, we find that for a target video quality, much lower delays are viable for slow-motion video clips. Our findings can be used as a design guideline for new protocols for networks that carry a considerable amount of video traffic.

Acknowledgment

This work was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] A. C. Bovik, *The Essential Guide to Video Processing*. Academic Press, 2009.
[2] ISO/IEC JTC1/SC29/WG11, "ISO/IEC 14496 – Coding of audio-visual objects," http://mpeg.chiariglione.org/standards/mpeg-4/mpeg-4.htm.
[3] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
[4] M. T. Ivrlac, R. L.-U. Choi, E. G. Steinbach, and J. A. Nossek, "Models and analysis of streaming video transmission over wireless fading channels," *Signal Proessing: Image Communication*, vol. 24, no. 8, pp. 651–665, Sep. 2009.
[5] R. S. Tupelly and J. Zhang, "Opportunistic scheduling for streaming video in wireless networks," in *Hopkins University*, 2003.
[6] X. Xu, "Fine-granular-scalability video streaming over wireless LANs using cross layer error control," in *ICASSP*, Montreal, Canada, May 2004.
[7] J. Villalon, P. Cuenca, L. Orozco-Barbosa, Y. S. Y. Seok, and T. Turletti, "Cross-layer architecture for adaptive video multicast streaming over multirate wireless LANs," *IEEE JSAC*, vol. 25, no. 4, May 2007.
[8] J. R. Renno, N. Lazarevic-McManus, D. Makris, and G. A. Jones, "Evaluating motion detection algorithms: Issues and results," in *Proceedings of the 6^{th} IEEE International Workshop on Visual Surveillance*, 2006.
[9] "PhysMo: Video motion analysis," http://physmo.sourceforge.net.
[10] "AForge.NET," http://www.aforgenet.com/framework/features/motion_detection/_2.0.html.
[11] E. A. Akkoyunlu, "The enumeration of maximal cliques of large graphs," *SIAM Journal on Computing*, vol. 2, no. 1, pp. 1–6, 1973.
[12] H. Luo, S. Lu, and V. Bharghavan, "A new model for packet scheduling in multihop wireless networks," in *ACM MobiCom*, 2000.
[13] C. Bron and J. Kerbosch, "Finding all cliques of an undirected graph," *Communications of the ACM*, vol. 16, no. 9, pp. 575–577, Sep. 1973.
[14] R. Gupta, J. Walrand, and O. Goldschmidt, "Maximal cliques in unit disk graphs: Polynomial approximation," in *Proceedings of the 2^{nd} International Network Optimization Conference (INOC)*, 2005.
[15] H. Friis, "A note on a simple transmission formula," *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, May 1946.
[16] M. Pursley and T. Royster Iv, "Properties and performance of the IEEE 802.11b complementary-code-key signal sets," *IEEE Trans. Commun.*, vol. 57, no. 2, pp. 440–449, Feb. 2009.
[17] G. Pei and T. Henderson, "Validation of ns-3 802.11b PHY model," May 2009. [Online]. Available: http://www.nsnam.org/~pei/80211b.pdf
[18] Y. Wang, M. Claypool, and R. Kinicki, "Impact of reference distance for motion compensation prediction on video quality," in *Proceedings of ACM/SPIE Multimedia Computing and Networking (MMCN)*, 2007.
[19] "YUV CIF reference videos (lossless H.264 encoded)," http://www2.tkn.tu-berlin.de/research/evalvid/cif.html.
[20] "FFmpeg," http://ffmpeg.org/.
[21] "EvalVid with GPAC," http://www2.tkn.tu-berlin.de/research/evalvid/EvalVid/docevalvid.html.
[22] "The network simulator, ns-3," http://www.nsnam.org/.
[23] L. Hanzo, P. J. Cherriman, and J. Streit, *Wireless Video Communications - Second to Third Generation Systems and Beyond*. Wiley-IEEE, 2001.