

# ORBit: An Adaptive Data Shaping Technique for Robust Wireless Video Clip Communication

Clark N. Taylor and Sujit Dey  
Electrical and Computer Engineering  
University of California, San Diego  
{cntaylor, dey}@ece.ucsd.edu

*Abstract*—With the introduction of the low-bitrate video compression standard, MPEG-4, and high bandwidth wireless technologies such as 3G Wireless and WLAN, video communication over wireless channels will soon become a widespread and often used technology. However, video communication generally requires stable and well-known communication conditions, while wireless conditions can vary quickly and dramatically. In this paper, we focus on a specific type of video, video clips, and propose a method, *Out-of-Order Buffering (ORBit)*, which helps to maintain a high quality of received video under widely varying wireless conditions. We introduce an algorithm which judiciously selects information to buffer which will help improve the video quality of the entire video clip. Our results show a significant increase (up to 5dB) in received video quality by using ORBit to achieve robust video clip communication. We also demonstrate that the proposed scheme imposes only a marginal computational overhead on the wireless handheld, thus making the approach feasible for typically CPU-constrained wireless handhelds.

## I. INTRODUCTION

Data communication will soon undergo a fundamental shift in both the method and type of communication required. The method of data communication will change as more data is transmitted over the Internet, with wireless channels being used as a “last hop”. Due to changes in the wireless channel conditions, rapid and severe error, latency, and bandwidth variations can occur over time, something unfamiliar in fixed, wired communication.

The type of data communication that commonly occurs will also change as the amount of multimedia data transmitted over the Internet grows significantly. Multimedia data transmission, in general, requires high bandwidth, low latency, and bounded errors during communication, requirements which are diametrically opposite to the characteristics of wireless channels. To help address this confluence of wireless communication links and multimedia content, this paper focuses on enabling the communication of short digital video clips with a wireless last-hop channel. We focus on video clips because of the large future market for video clips (advertisements, location-aware bulletins, news clips). In addition, video clips provide more opportunities for optimization than the more general streaming video condition.

Current research addressing the confluence of wireless channels and digital video have focused primarily on one of two areas. First, many research works attempt to address the low-bandwidth requirements of wireless channels, resulting in the low-bitrate digital video standard MPEG-4 [1]. Second, much research has been performed on minimizing the effect of wireless channel errors during the transmission of digital video [2], [3], [4], [5].

In this paper, we investigate the effects of buffering video streams in wireless last-hop communication systems. Previous work on the buffering of digital video has primarily focused on ensuring that overflow (too much information for resource-constrained appliances) and underflow (the emptying of the buffer when more information is needed) do not occur during the transmission of variable-bit rate video [6], [7]. However, the above research does not address situations where the data communication channel is itself changing. In this paper, we introduce a new video buffering scheme, *Out-of-Order Buffering (ORBit)*, for video clips that ensures high-quality video transmission even with deteriorating wireless channel conditions. To ensure robust communication, our method buffers key frames from the entire video clip at the beginning of the video transmission, allowing for information sent

at the beginning of a transmission to improve video quality for the entire video clip. We have implemented a video decoder with the ORBit method on an ARM7TDMI processor, demonstrating the feasibility of our method in a CPU-constrained environment. Because our work is primarily concerned with effective buffer usage, and not the actual format of the video clip, this work can be used in conjunction with previous methods used for improving the robustness of compressed video.

The paper is organized as follows. In section II, we present a brief overview of MPEG-4 terminology and techniques. Section III presents the ORBit technique, and an algorithm for deciding what needs to be buffered given current communication constraints. We then present a simulation environment that enables a detailed study of different buffering schemes in section IV. Section V presents the results of using out-of-order buffering on video quality and computation requirements, and compares it with current approaches. Section VI concludes the paper.

## II. BACKGROUND AND TERMINOLOGY

In this section, we present a brief overview of how MPEG-4 video compression works, with particular reference to the terms and methodologies that our work utilizes. Most of the concepts introduced in this section are found in other video compression standards, so this work could also be applied within other video compression formats such as H.263[8].

In general, digital video is composed of a sequence of images, referred to as **frames**. The key observation used by MPEG to compress video is that two frames which are close together temporally (i.e. consecutive frames) will usually have a small difference between them. Therefore, MPEG divides up the frames into two main types: intra-coded and inter-coded frames. **Intra-coded frames (I-frames)**, are essentially images compressed using JPEG-like compression. For the MPEG decoder to decompress these frames, it does not require information from any other frame.

**Inter-coded frames (P-frames)**, on the other hand, depend on other frames to be decoded. The MPEG encoder will take the frame being encoded, compare it to another frame already encoded, and encode the difference between the two frames. To further reduce the differences between the frames, motion estimation/compensation is also used. Therefore, a P-frame will include both motion vectors and difference information. Because the difference between frames is usually quite limited, encoded P-frames contain significantly less information than encoded I-frames, leading to high compression rates for inter-coded frames.

While using inter-coded frames significantly reduces the information required to represent frames, leading to better video compression, the encoding of inter-coded frames also introduces dependencies between different portions of an encoded video stream. Therefore, a transmission error in a specific portion of the encoded video stream can affect not only the frame represented by that portion, but other frames as well. To help minimize this **temporal spreading** of errors, error concealment has been an active topic of research [3], [9]. Error concealment attempts to “cover up” errors introduced in an image so that (1) the error is not so apparent in the frame with an error, and (2) so that the visual effect of the error spreading to other frames is minimized.

This work supported by the SRC, Task # 900.001

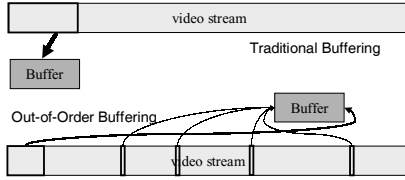


Fig. 1. Traditional and out-of-order buffering (ORBit) techniques

Similarly, due to the use of variable-length coders to help reduce the bit-size of the transmitted video, significant **spatial spreading** of errors can occur. A single bit error can affect not only the erroneous bit, but also cause the decoder to lose “synchronization.” When synchronization is lost, the decoder will not decode the variable-length codes properly until it can be resynchronized. To help minimize the spatial spreading of transmission errors, MPEG-4 divides a video stream into **packets**, and places **resynchronization points** between each packet. Once the decoder finds a resynchronization point, it can decode the bits following the resynchronization point, regardless of any previous errors.

Another key technology used to compress information in the MPEG standard is quantization. For each frame in an MPEG video, a **quantization level** is chosen. The higher the quantization level, the better the compression. However, higher quantization levels also correspond to a lower image quality of the frame when received.

To enable “searching” of MPEG video streams (akin to rewinding and fast-forwarding a VCR), MPEG also introduces the concept of a **Group of Pictures or GOP** header. A GOP header is a location marker in the MPEG video stream which denotes what the time marker of the following frame is (e.g. 3 mins, 5 secs from start of video). To enable a search algorithm to start displaying frames once it finds an acceptable GOP time, the GOP must always be followed by an I-frame.

With an understanding of how MPEG compresses digital video, it is possible to research ways of improving received video quality. In the following section, we present our new method, out-of-order buffering, for improving the video quality of video clips transmitted over wireless channels.

### III. THE ORBIT METHOD

During wireless data communication, the conditions (such as bandwidth allowed, bit error rate, etc) can vary dramatically over time, requiring robust communication of digital video. To allow for more robust communication of video clips, we introduce ORBit, a method that ensures the received video quality of an entire video clip, even with dramatic wireless channel variations during transmission of the video clip. In this section, we explain how ORBit is implemented, followed by a discussion of what types of frames may be useful as ORBit frames how to select what frames to buffer.

As shown in Fig. 1, traditional buffering schemes buffer the next few frames to try and overcome variations in the communication channel. However, by judiciously selecting frames from throughout the video clip, and transmitting them at the beginning of the video clip, we can dramatically increase the received video quality.

To implement the communication of frames from throughout a video clip at the beginning of the video clip (out-of-order buffering), the GOP header from the MPEG-4 standard is used. For each frame selected as an ORBit frame, a GOP header is sent, representing exactly where in the video stream the frame is located, followed by the frame compressed as an I-frame. Therefore, the buffered information at the beginning of a video clip contains all the information about a frame, together with its temporal location in the video stream, allowing the decoder to recreate the OR-

Bit frames independent of any information transmitted later on in the video clip. In addition, because a GOP header is used to denote temporal location, implementation of ORBit requires minimal changes to an MPEG-4 encoder.

To implement the display of ORBit frames during video playback, the ORBit frames are stored in temporal order of display. When the decoder of a video clip decodes a frame at the same temporal location as an ORBit frame, the quality of the decoded frame is compared with the buffered frame, and the higher quality of the two is displayed to the end-user. In addition, the higher quality frame is used as a reference when decompressing further inter-coded frames. To determine which frame is higher quality, the decoder first checks to see if there have been any errors which might affect the currently decoded frame. If errors have occurred then the ORBit frame information is used. If there were no errors affecting the current decoded frame, then the quantization level used to compress the two frames is compared. The frame with the lower quantization level is then used for both display and as the reference frame for further inter-frame decoding.

To ensure the best possible received video quality, special care must be taken in selecting ORBit frames. Two different types of frames that are good candidates for selection as ORBit frames are: (1) *Key Content Frames* and (2) *Periodic Frames*. We discuss the meaning and effects of each of these frames below.

#### A. Possible ORBit Frame Types

**Key Content Frames** are defined as images or frames which represent the content of a portion of the video clip. For example, if as part of a news video clip, a jet is shown streaking across the sky, then a single frame with the jet in the middle of the sky would be sufficient to convey the “content” of that portion of the video clip. Key content frames are especially useful for out-of-order buffering because they provide for a concept of “graceful degradation” of video quality if the wireless channel conditions dramatically deteriorate. In a worst-case situation, a video would degrade into a slide-show of key content frames, one key content frame for each portion of the video clip. The primary disadvantage of key content frames is that they must be identified by the content provider of the video clip.

**Periodic Frames** are frames selected from the video clip on a periodic temporal basis. Because ORBit frames are compressed as I-frames, and I-frames are compressed independent of all other frames, periodic frames can be used to limit the temporal propagation of errors in the received video to the period used to select the periodic frames. For example, in a 30 frames per second video clip, if the encoder selects every 45th frame for out-of-order buffering, the worst-case temporal propagation of errors is 1.5 seconds. Periodic frames are also a good candidate for out-of-order buffering because they can be automatically selected.

While selecting periodic frames as ORBit frames can be very beneficial to the received video quality, buffering consumes storage resources at the video receiver and leads to longer wait-times before the end-user can start viewing video (user-latency). For example, to buffer every 50th frame of a 1 minute long, 30 frames per second video will require an additional 36 frames to be buffered. In the following subsection, we present an algorithm that judiciously selects ORBit frames to maximize the received video quality in a constrained environment.

#### B. Selection of ORBit Frames with Storage and/or User-Latency Constraints

In a storage or user-latency constrained environment, it is important to judiciously allocate the limited resources available for buffering to the buffer information which will maximize the quality of the video viewed by the end user. In this subsection, we present a buffer allocation algorithm that judiciously selects which

frames to buffer to maximize the received video quality given the current communication conditions.

Before execution of our proposed algorithm, it is assumed that the latency conditions of the communication medium (jitter and end-to-end latency), the bandwidth constraints, the user-latency constraint (how long the user will reasonably wait to start viewing the video), and the buffer constraints of the end-user appliance are all known. The first step in executing the algorithm is to determine how many bits can be buffered. The maximum amount that can be buffered is determined by the following formula:

$$\text{MaxBuff} = \min(\text{BuffSize}, (\text{UserLat} - \text{NetLat}) * \text{BW}), \quad (1)$$

where  $\text{MaxBuff}$  is the number of bits allowed for buffering,  $\text{BuffSize}$  is the buffer size constraint of the end-user's appliance,  $\text{UserLat}$  is the desired end-user latency,  $\text{NetLat}$  is the average end-to-end latency of the communication system, and  $\text{BW}$  is the end-to-end bandwidth of the communication system.

Once the maximum number of bits allowed for buffering has been determined, the algorithm must decide which frames should be buffered within that constraint. The frames to be buffered fall into two categories: (1) traditionally buffered frames and (2) ORBit frames. While ORBit frames must be explicitly inserted before the video clip by the algorithm, traditional buffering is achieved by sending video clip information without modification, while delaying the start of the video display for the end-user. Therefore, the buffer allocation algorithm determines how much space is available for ORBit frames, inserts the ORBit frames, and then starts the transmission of the unmodified video clip, thereby controlling both ORBit frame and traditional buffering.

To determine the amount of space available for ORBit frames, the following formula is used:

$$\text{ORBitBuff} = \text{MaxBuff} - \text{JittBuff}, \quad (2)$$

where  $\text{ORBitBuff}$  is the buffer size allowed for ORBit frames,  $\text{MaxBuff}$  is the total buffer allocation found during the first step of the algorithm (see equation 1), and  $\text{JittBuff}$  is the buffer space required to overcome the current latency jitter conditions (see section V for details on determining this number). If  $\text{ORBitBuff}$  is negative, then the algorithm assigns the entire buffer space to traditional buffering. If  $\text{ORBitBuff}$  is positive, then the algorithm will decide which ORBit frames, at what quantization level, should be buffered to maximize the received video quality.

Using experimental results presented in section V, it was determined that the following frame selection criteria best allocated buffer space to maximize the received video quality. First, allocate buffer space for all the key content frames. If the available buffer space will not allow for all key content frames at the lowest quantization level (8 is the lowest quantization level used as it results in a PSNR of around 33dB, a high image quality), then the key content frames can be buffered with a higher quantization level.

Second, as space allows, allocate buffer space for periodic frames. The buffer allocation algorithm must decide on two parameters for the periodic frames: (1) the period used to select the periodic frames, and (2) the quantization level used to compress the frames. The method used for matching the periodic frames selected and their quantization levels with the buffer space available is as follows. An initial period (200 frames) and quantization level (31) is chosen. As more space becomes available, the quantization level is decreased until a quantization level of 9 is reached. If more space is available, then the period used for selecting frames is halved, and the quantization level is increased to 16. The process repeats until the smallest desired selection period, with a low quantization level, is reached. For example, in the video clip we are using, we start with a period of 200 frames, quantization level 31. As space allows, the quantization level is decreased to 9. Then, if more space is available, a periodic spacing of 100 is chosen, with a quantization level of 16, decreasing to 9, after which a period of 50 frames is chosen (the smallest period desired, about 5/3 seconds). With a period of 50 frames, the quantization level starts at

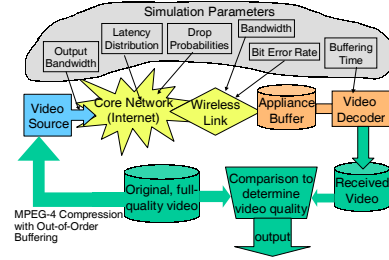


Fig. 2. Functional diagram of exploration environment used to evaluate different buffering schemes

16, and decreases to 8. After a quantization level of 8 is reached, all out-of-order buffered frames desired have been chosen, so the algorithm terminates.

To implement the buffer allocation algorithm, the extra size required for each possible set of ORBit frames, at each quantization level, is computed a-priori and stored in an array. At run-time, the buffer size available for ORBit frames is determined (see equation 2), and the array is searched for the largest value which does not exceed the available size. The frames corresponding to that size are then transmitted after which traditional video clip communication begins.

#### IV. A FRAMEWORK FOR EVALUATING BUFFERING SCHEMES

To evaluate the effects of the ORBit method, and compare it with traditional buffering techniques, we developed a simulation environment capable of representing different core network and wireless link conditions, buffer sizes, ORBit frame selections, and their effect on received video quality [10]. Fig. 2 shows a basic diagram of our simulation environment.

As shown in Fig. 2, the communication system is divided into four parts that can affect the data communication: (1) the video source, (2) the core network (Internet), (3) the wireless link, and (4) the video decoder. What communication conditions are being simulated is controlled by the simulation parameters denoted by the shaded area at the top of the figure. The video source unit transmits a video frame regularly (every 1/30th of a second for a 30 frames per second video). When the video source transmits a packet, the simulator assigns a time that the packet reaches the core network dependent on when the video source generated the packet, the size of the packet, and the "Output Bandwidth" parameter. The core network portion of the simulator adds latency to each packet, and decides whether the packet should be dropped or not. The Wireless Link portion of the simulator corrupts packets according to the "Bit Error Rate" parameter, assuming a binary symmetric channel, together with adding latency to the packets depending on the "Bandwidth" parameter and the order in which the packets arrived. The Video Decoder produces a video to be viewed by the end-user using the packets given to it by the wireless link. However, it does not attempt to display any video until the system time of the simulator is greater than the "Buffering Time" parameter. Selection of the Buffering Time parameter represents the maximum user-latency and appliance buffering constraints.

During the simulation of the core network, it is assumed that the UDP protocol is used, with an RTP (real-time protocol) header added to each packet. Therefore, packet dropping must be considered. To simulate the dropping of packets during the transmission of packets over the Internet, a 2-state Markov Model is used [11]. The model is configured so that in state 0, no packets are dropped, and in state 1, all packets are dropped. Therefore, by modifying the probabilities of the 2-state Markov Model ("Drop Probabilities"), the drop rate and average number of packets dropped at a time may be modified.

To simulate the latency of transmitting a packet across the In-

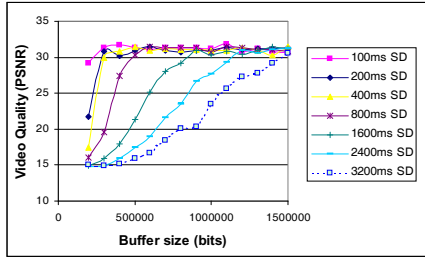


Fig. 3. Effect of traditional buffer size on video quality with differing amounts of jitter

ternet, a left-truncated normal distribution is used. The “Latency Distribution” parameter includes a mean, standard deviation, and left-truncation point, thereby configuring the left-truncated Gaussian distribution to represent the latency of packets across the core network. The left truncation point is used to ensure that unrealistic (negative, for example) latencies are not chosen. By setting the mean of the Gaussian distribution, the user can control the average end-to-end latency for packets transmitted across the core network. By setting the standard deviation of the distribution, the user can control how much latency jitter is present.

To simulate the effect of widely varying wireless channel conditions, the simulator accepts different simulation parameters over time. The final output of the simulator is the video quality seen by the end-user, determined by comparing the received video against the original, full-quality video.

## V. RESULTS

Using the simulation framework explained in section IV, we evaluated the effect of ORBit and traditional buffering on the received video quality. The video clip used to evaluate different buffering schemes is a 29.97 frames per second, 160 by 120 pixel, 68.5 second video stream. The video contains both low-motion and high-motion portions. The video is compressed to obtain a bit rate of 150 kbits/sec, and is divided into packets of approximately 256 bytes each. Unless otherwise specified the parameters used for this simulation were: the output bandwidth was 1E6 bits/s; the latency distribution was left-truncated at 100ms, with a mean of 1000ms, and standard deviation of 300ms; the drop probabilities were set at a 0.125% probability of a drop occurring (transition from state 0 to state 1 in the Markov model), and a 90% probability of multiple drops (staying in state 1); the bandwidth of the wireless link was 160 kbits/s; and the wireless link bit error rate was 1E-6.

### A. Effect of Traditional Buffering on Video Quality

To determine what the effects of traditional buffering are, we simulated different amounts of traditional buffering, with different latency jitter parameters. Fig. 3 represents the results of these simulations, with the x axis representing the available buffer size, the y axis the video quality in terms of PSNR, and each plot representing a different latency standard deviation parameter input to the simulator. Each data point is averaged across 5 different runs of the simulator. It is important to note that for each plot, as the available buffer size increases, the video quality also increases until an “elbow” point is reached. Once this elbow point is reached, an increase in the amount of traditional buffering does not correspond with an increase in video quality. Therefore, in many cases the addition of ORBit frames will not degrade the received video quality due to the decrease in buffer space used for traditional buffering. The elbow point shown in this graph is also used by the buffer allocation algorithm introduced in section III-B to determine how much space should be allocated for traditional buffering.

### B. Effect of ORBit method on Video Quality

To demonstrate the usefulness of ORBit in making video clip communication more robust, we performed simulations using 6 different sets of simulation parameters which represent significant degradation of the wireless channel during video clip transmission. The first three sets of simulation parameters represent when a wireless channel becomes more noisy (due to fades, a physical barrier in the transmission path, etc.), represented by a wireless link bit error rate of 1E-4. Three different noisy channel situations were simulated where a noisy channel was assumed from (1) 30 to 45 seconds, (2) 50 to 65 seconds, and (3) 30 to 65 seconds. Another three sets of simulation parameters were used to simulate a significant decrease in bandwidth availability. When the bandwidth decreases significantly, the base station can drop packets which it knows will not arrive in time to be displayed, thereby causing high packet drop probabilities [12]. Therefore, to simulate a low bandwidth channel condition for part of the simulation, the drop probability was set to 50% for the same three time periods used when simulating a noisy wireless channel.

Fig. V-B demonstrates the effectiveness of using ORBit to increase the received video quality with deteriorating wireless channel conditions. In both graphs, the x axis represents the buffer size consumed, while the different plots represent different strategies used for filling the available buffer size. In Fig. V-B(a), the y axis represents video quality in terms of PSNR, a common video quality metric. In Fig. V-B(b), the y axis represents video quality in terms of *Weighted PSNR* (WPSNR), where the PSNR of the key content frames are weighted more heavily than the PSNR of other frames. WPSNR is used because the key content frames represent the content of the video, and are therefore more important to the user’s perceived video quality than the other frames. Each of the data points in Fig. V-B represents an average across the 6 channel conditions discussed, with 10 data samples for each channel condition.

The results of several different buffering schemes are presented in Fig. V-B. The schemes presented include:

- **Traditional Buffering** only. ORBit is not used.
- **Key Content Frames** only are selected as ORBit frames
- **Periodic Frames** only are selected as ORBit frames, with a period of every 200 or 50 frames, represented by two different plots.
- **Key Content and Periodic Frames** are both buffered, with all frames at the same quantization level, with the period being either 200 or 50 frames.
- **Buffer Allocation Algorithm** uses the algorithm presented in section III-B to decide how to fill up the buffer.

For all the schemes presented, except traditional buffering, the first 400,000 bits of the buffer were consumed using traditional buffering (an approximate “elbow” point from the results shown in Fig. 3.) For all plots except the traditional buffering and buffer allocation algorithm plots, each distinct point represents different quantization levels chosen for the out-of-order buffered frames. For the traditional buffering and buffer allocation algorithm plots, each data point corresponds with an allowed buffer size of 200,000 through 1,500,000 bits, in increments of 100,000.

Several interesting conclusions can be drawn from Fig. V-B. First, even in noisy channel conditions, an increase in traditional buffering size does not correspond with an increase in video quality after the “elbow” point described above. Second, for the simulation bandwidth of 150,000 bits per second, the buffer sizes shown in Fig. V-B correspond to a user-latency time of 1.3 to 10s, well within the acceptable range of current video communications where ORBit is not used. Third, the use of ORBit allows for gains in video quality of approximately 3dB when measuring PSNR and 5dB when measuring WPSNR. Fourth, in both the PSNR and WPSNR graph, the buffer allocation algorithm plot has equal or greater video quality than other, possible buffer allocation

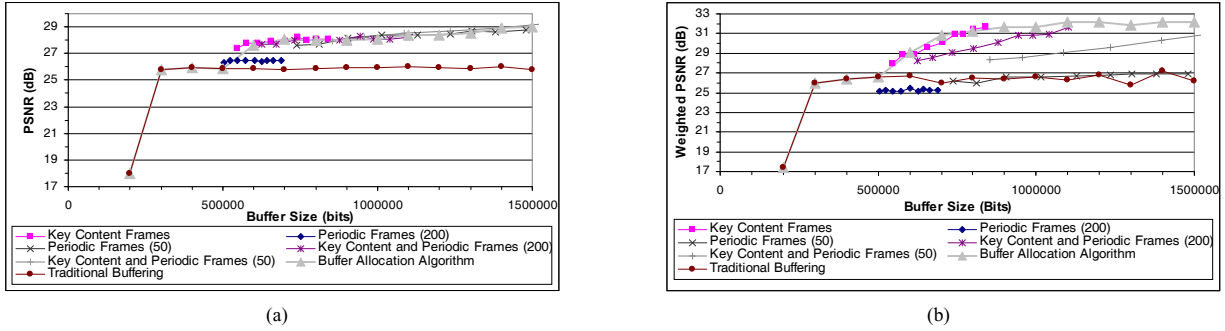


Fig. 4. Effect of buffer size on received video quality using different buffering schemes

Bad channel time	0 sec			10 sec			20 sec		
	w/out	w/	overhead	w/out	w/	overhead	w/out	w/	overhead
600,000 (4s)	25.9%	25.1%	-3.20%	25.1%	25.5%	1.54%	23.9%	24.6%	3.18%
900,000 (6s)	26.3%	27.4%	3.99%	25.7%	26.7%	3.70%	24.9%	26.3%	5.39%
1,200,000 (8s)	26.5%	27.5%	3.48%	26.4%	29.0%	8.97%	24.6%	26.0%	5.40%
1,500,000 (10s)	26.3%	27.7%	5.01%	25.7%	30.7%	16.25%	25.9%	26.7%	2.91%

Fig. 5. Computation load of an ARM7TDMI 200MHz processor with and without ORBit

strategies. Fifth, it is important to note that as the allocated buffer size is increased, the video quality continues to increase when using the buffer allocation algorithm. These conclusions empirically validate the ability of ORBit to enable more robust video communication over noisy wireless channels, at a low overhead cost in terms of extra buffer and latency requirements. The results also demonstrate the effectiveness of the buffer allocation algorithm in selecting ORBit frames, with their corresponding quantization levels.

### C. Computation Costs of ORBit

To evaluate the computational costs of implementing ORBit, we compiled the C code used to implement the evaluation framework described in section IV for an ARM7TDMI processor [13]. The video decoder in the evaluation framework is based off a traditional MPEG-4 video decoder, with slight modifications to implement ORBit. After decoding each frame from the video clip, the decoder checks to see if there is an ORBit frame with the same temporal location as the just-decoded frame. If one exists, the decoder checks to see if the ORBit frame should be inserted or not (as described in section III), and may decode and insert the ORBit frame to improve the display quality of that frame. Using a cycle accurate ARM simulator and the ARM profiler[14], we measured the number of cycles needed to implement both the video decoder and the buffering of packets as they arrive.

Table 5 shows the computational load on the ARM processor for decoding video clips with ORBit versus the case where no ORBit frames are selected. Each row in the table represents how much buffering occurs before playback of the video clip begins in terms of bits (or number of seconds). The columns in the table represent different lengths of time that the channel was in a deteriorated condition (BER of  $1E-4$ ). The sub-columns labeled *w/out* and *w/* show the processor load for decoding a video clip without and with ORBit, respectively. The sub-columns labeled *overhead* show the percentage overhead of using ORBit compared to solely using traditional buffering. The computation loads shown are determined by taking the number of cycles required to buffer and decode the video clip by the total number of cycles available to a 200MHz ARM processor during the decoding of a 63s video clip. Each data point shown is an average across 3 different simulation runs. The results demonstrate that only a marginal computational overhead of about 4.7% is incurred by the handset to decode video clips transmitted using the new ORBit scheme, as opposed to decoding video clips based on traditional buffering.

## VI. CONCLUSION

In this paper, we have introduced a method, ORBit (Out-of-order Buffering), for ensuring high video quality across an entire transmitted video clip, even with dramatically deteriorating wireless channel conditions. We have compared the video quality between using ORBit and traditional buffering schemes. The video quality when using ORBit is shown to be significantly higher than without ORBit, at a marginal computational overhead. We have also presented a methodology for allocating buffer space for ORBit frames, together with selecting which frames, at what quantization level, should be buffered to obtain the highest video quality for the end-user. This methodology enables significant gains in video quality with marginal impact on the user-latency experienced waiting for the video clip to begin playing.

## REFERENCES

- [1] The MPEG Home Page. <http://mpeg.telecomitalia.com/>.
- [2] B. Girod and N. Färber, "Feedback-Based Error Control for Mobile Video Transmission", *Proceedings of the IEEE*, vol. 87, pp. 1707–23, October 1999.
- [3] Y. Wang and Q.-F. Zhu, "Error Control and Concealment for Video Communication: A Review", *Proceedings of the IEEE*, vol. 86, pp. 974–97, May 1998.
- [4] R. Talluri, "Error-Resilient Video Coding in the ISO MPEG-4 Standard", *IEEE Communications Magazine*, vol. 36, pp. 112–19, June 1998.
- [5] J. Brailean, "Wireless Multimedia Utilizing MPEG-4 Error Resilient Tools", in *Proceedings, 1999 Wireless Communications and Networking Conference*, pp. 104–8, September 1999.
- [6] W.-C. Feng, *Buffering Techniques for Delivery of Compressed Video in Video-on-Demand Systems*. Kluwer Academic Publishers, 1997.
- [7] S. Sen, J. L. Rexford, J. K. Dey, J. F. Kurose, and D. F. Towsley, "Online Smoothing of Variable-Bit-Rate Streaming Video", *Transactions on Multimedia*, vol. 2, pp. 37–48, March 2000.
- [8] ITU Telecom, Standardization Sector of ITU, "Video Coding for Low Bitrate Communication, Recommendation H.263 Version2", Feb 1998.
- [9] S. Cen and P. Cosman, "Comparison of Error Concealment Strategies for MPEG Video", in *Proceedings, 1999 Wireless Communications and Networking Conference*, pp. 329–33, September 1999.
- [10] C. N. Taylor and S. Dey, "Report on VideoSim: An Exploration Framework for Video Transmission over the Internet with a Wireless Last-hop", tech. rep., SRC, 2003. Task 900.001, publication # P005943.
- [11] H. Sanneck and G. Carle, "A Framework Model for Packet Loss Metrics Based on Loss Runlengths", in *Proceedings of SPIE - the International Society for Optical Engineering*, vol. 3969, pp. 177–87, 2000.
- [12] J. H. Sarker and S. J. Halme, "Efficiency of the GSM-GPRS Air Interface for Real-Time IP Traffic Flows With and Without Packet Dropping", *Wireless Personal Communications*, vol. 21, pp. 125–40, April 2002.
- [13] The ARM7TDMI Processor. <http://www.arm.com/armtech/ARM7TDMI?OpenDocument>.
- [14] The ARM Development Suite (ADS v1.2). [http://www.arm.com/devtools.nsf/iwpl111/B055BFDC029E3EC880256B9000418908?OpenDocument&style=Dev\\_Tools](http://www.arm.com/devtools.nsf/iwpl111/B055BFDC029E3EC880256B9000418908?OpenDocument&style=Dev_Tools).