Abstract - In live production, humans are required to make real-time editorial decisions based on multiple, simultaneous streams of audio and video. If latencies between these streams become noticeable, the production can be affected. As such live production systems must be designed so as to keep latencies within tolerable levels. As broadcasters move to IP-based systems for live production, there are new potential sources of latency which must be understood and designed for. In this paper we identify specific latency requirements for live production, model the sources of latency in an IP-based system, and outline a systems approach for achieving latencies that are as good, or better, than the typical latencies found in SDI-based systems.

Requirements are Driven by Human Factors

In many broadcast applications (delivery of program streams, for example) delays on the order of multiple seconds are routinely experienced and go completely unnoticed by the party receiving those streams. However, in live production, the latency requirements are much more stringent. This is because humans are observing and interacting with multiple video and audio signals from the same event in real time. Delays between signals, if large enough, can be noticeable and distract production crews. This in turn can lead to errors during a live broadcast. In order to reduce the potential for on-air errors, live production systems must be designed such that latencies are below human perceptual tolerances.

As human perceptual thresholds vary depending on the type of latency, it is important to consider each type of latency separately. For example, we illustrate here three types of latency which need to be managed in a live production system.

Tactile to Visual Latency. This type of latency is defined by delay between when a human pushes a button (or twists a knob) and when they see a corresponding change in video. In most SDI routers the latency between the push of a “take” button and the video switch is two frames of 1080i59.94 or 67 msec. Therefore, as a matter of practice we know that 67 msec is beneath the human perceptual threshold for tactile to visual latency.

Audio to Visual Latency. Research by Kanabus et al. has shown that humans are can perceive delays between auditory and visual signals when the delay between the signals is greater than ±5 msec. [1] Practice within broadcast facilities correlates well with this research. As an example the CBS recommended practice for “end-to-end audio to video offset (lip sync)” is ±4 msec. [2]

Visual to Visual Latency. This describes the latency between two visual signals. For example if two cameras, A and B, are focused on the same live event, but A has a longer transmission path, B may arrive at the input to a production switcher faster than A. The difference in arrival times at the production switcher defines the relative latency between the two signals. Kanabus et al. also measured that delay between two visual stimuli that humans could perceive is on the order of 80 msec. This also correlates well with live production system designs for SDI based systems where a typical rule of thumb is to keep delay under 100 msec, or three frames of 1080i59.94.

As a goal then, in order that a production crew working a live event perceives no delay and an “instant response” to button pushes, latencies need to be kept within the above mentioned values. These maximum latency targets are independent of video format and of the transport technology used in the system. Rather, they are based entirely on the ability of the human brain to detect differences in the arrival times of various stimuli.

Modeling the Overall System Latency

In order to design IP live production systems whose latencies are below the thresholds outlined above, it is important to model the factors contributing to latency. Here we model system latency with a particular focus on the latency between two visual signals. As such our goal is to keep the differential latency between any video signal (including live event itself which is the original visual signal) under 100 msec. We model the following components of latency in an IP-based system:

1. Jitter Buffering

IP based systems inherently have some level of jitter. In order to account for packet jitter, buffering must be employed. This buffering introduces latency. Fundamentally there are two components of jitter which we will model separately. The first component is jitter due to the transmitter which we model as $J_T$. The second
component, also known as “packet variable delay”, occurs at the ingress port of an IP switch where packets are buffered before being switched. We model this second component of jitter as \( J_{PVD} \). Using these two variables and modeling the number of stages of IP switching in the system as \( n_r \), we can model the latency due to jitter in an IP system with the equation: \( J_T + (J_{PVD} \times n_r) \).

II. Redundancy Buffering (SMPTE ST 2022-7)

In many IP applications, dual network paths are employed with RTP packet level redundancy. One well known method of implementing packet level redundancy in transport applications within broadcast is based on SMPTE ST 2022-7:2013. In this scheme, one must buffer packets to account for differential skew between the dual network paths. Because this standard was designed with video transport (long-haul) in mind, at the time of this writing, the minimum acceptable buffer in a SMPTE ST 2022-7:2013 compliant implementation must be at least 10 msec.\(^1\)

Within a facility studio or mobile truck in live production, a smaller skew between network paths can be designed for effective packet level redundancy. However, strictly speaking, such a design would violate the ST 2022-7:2013 buffer specification. Whatever the buffer size, in a latency budget, one must account for the maximum buffer delay which we will denote by the variable, \( R \). If \( n_r \) represents the number of devices in the signal path that employ packet level redundancy, then the total amount of latency due to the deployment of redundancy in the live production system can be represented as \( R \times n_r \).

III. Codec Delay

While not necessary in all cases, there are use cases where compression can be introduced into the production signal chain. For example when one wishes to transmit a 12 Gbps UHD signal over a 10Gbps Ethernet pipe, codec technology must be employed. The encode/decode process introduces latency every time an encode or decode occurs in the signal path. We model the delay incurred by one encode/decode cycle as \( C \). We further model the number of encode/decode cycles in the signal path as \( n_c \). Thus the total signal path delay due to use of codecs in the signal path is \( C \times n_c \).

IV. Other Processing Delay

In addition to the above two delay components, processing such as up/down cross conversion and frame synchronization introduce additional delay. In practice the amount of delay introduced via processing varies with the type of processing. However, here, to simplify the model, we model processing delay as \( P \), and the number of processing elements in the signal chain we model with \( n_p \). Therefore the total latency due to processing delay can be expressed by \( P \times n_p \).

V. Multiviewer Delay

Though multiviewer tile creation is just a specific case of “processing delay”, such tile creation and display is a fundamental part of live production. Therefore we call it out as a separate delay. Here we model multiviewer latency by the variable \( M \). This is a delay component that does not occur at every stage of the network, but at the endpoint.

VI. LCD Display Delay

Finally, before an operator can see and react, we must consider the latency of the LCD monitor itself which is used in the control room. Whereas CRT based displays had very low latency, LCDs can have significant latency, sometimes on the order of multiple frames due to LCD response time and the processing requirements of modern monitors including scaling, temporal ghost cancellation and other image enhancement techniques. We model the display delay of LCD monitors with the variable \( D \).

Of course this model is simplified, but the above variables can be used to state a model for the overall system latency of a live production system as follows:

\[
\text{Total system latency} = J_T + (J_{PVD} \times n_j) + R \times n_r + C \times n_c + P \times n_p + M + D
\]

Using the above model and given the above maximum tolerable “visual to visual latency” for professional live production, system design must be such that:

\[
J_T + (J_{PVD} \times n_j) + R \times n_r + C \times n_c + P \times n_p + M + D \leq 100 \text{ msec}
\]

USING THE MODEL: CHARACTERIZING EACH VARIABLE

The above model has eleven variables, all of which can be designed so as to contribute to low system latency. Each of these variables can be characterized based on available products and technologies. Once this is done, particular variables can be optimized in order to achieve a desired system-level result. For illustrative purposes in this paper, we will use (2) to solve for \( C \), codec delay. However in general, (2) could be used to solve for any of the eleven variables.

In our illustration we must first outline our assumptions about the ten variables other than \( C \). These assumptions, outlined below, are based on our own experience and input from multiple AIMS members:\(^2\):

- \( M \) (multiviewer delay). While multiviewer delays can vary by manufacturer, they are well characterized and typically specified. Depending on the multiviewer,

---

\(^1\) Class A – Low Skew as specified in SMPTE ST 2022-7:2013

\(^2\) The concepts behind this paper were shared and refined within the AIMS (Alliance for IP Media Solutions, aimsalliance.org) technical working group, and specific contributors are acknowledged at the end of this paper.
Delays can be as short as one field (16 ms for 1080i59.94) and as long as three to four frames (over 100 ms). Therefore choice of multiviewer for low latency system design is critical. Most multiviewers used in live production applications typically have a latency of one frame. Here, we will assume that a multiviewer tuned for live production is used and therefore the multi-viewer has one frame of latency. Therefore, for the purposes of the analysis below, M will equal the frame rate of the video format. Note that with some formats such as 1080i59.94, M equals 33 msec, which uses up one-third of the 100 msec latency budget by itself. However, since most multiviewers in live production have a delay of one frame, the impact of multiviewer delay diminishes as the frame rate increases.

- **D (LCD display delay).** Because so many LCD displays have built in scaling and other image processing, latencies can be on the order of three to four frames of 1080i59.94. The very lowest latency monitors have latencies on the order of 8-10 msec [3], but typically, professional broadcast LCD monitors have latencies on the order of one frame. For this analysis we will use a display latency D of one frame. For 1080i59.94 this would be 33 msec. For 1080p60 it would be 17 msec.

- **J_F (jitter based on transmitter profile).** The amount of jitter buffering required can be greatly affected by the profile of the transmitter. For example, in the case of a CPU based IP transmitter with no standard or traffic profile to dictate otherwise, it is entirely conceivable that data is transmitted in bursts and that time from first pixel of video to when that pixel is transmitted could be close to the order of a frame time, or over 10 msec. In this illustration, we will assume that transmitter has been chosen that is better suited for a live production application where the overall system latency budget is limited to 100 msec. Specifically, we will assume a transmitter which is capable of creating IP packets with the absolute minimal buffer. In this case, J_F is defined by the time to accumulate one packet of data at the video sample rate minus the time to transmit one MTU and the Ethernet line rate. In the case of 1.5 Gbps HD, 10 bit video being mapped into a 1540 octet MTU transmitted over a 10 Gbps Ethernet link, J_F works out to be 5.5 µsec.\(^3\)

- **J_{PVD} (jitter due to packet variable delay).** The second component of jitter to be characterized is jitter introduced as packets traverse a network, sometimes referred to as PVD (packet variable delay). We model this component of latency due to jitter with the variable, J_{PVD}. This is typically a factor when IP switches are managing multiple flows per port. Essentially, as packets are buffered on arrival at the ingress port of a switch, there is additional jitter based on the order and timing of the packets entering and exiting the buffer. This introduces greater variability in the timing between packets of an individual stream. In a well-designed network, jitter due to PVD can be small, but a highly conservative number is to assume 10 packets per each flow at a port’s ingress point. [4] Since the delay per 1540 octet packet on the wire of a system with 10G links is 1.2 µsec, using our 10 packets of delay assumption, the J_{PVD} latency per network segment would be 1.2 µsec/packet times 10 packets, or 12 µsec.

- **R (redundancy).** As stated above, to support RTP packet level redundancy using SMPTE ST 2022-7:2013, the minimum buffer size required is 10 msec. However this specification was designed and intended for WAN applications where the differential skew between flows could be greater than in a LAN environment. For this analysis we will assume the case of an OB (outside broadcast) truck parked outside a stadium, in which case all camera signals are local. In such an environment a much smaller buffer could be used, minimizing delay. Our real world experiments and consultation with leading IP switch vendors shows that a buffer depth of 100 µsec is sufficient for such LAN environments. Therefore, in this illustration, we will assume 100 µsec as the value for R.

- **P (processing).** Because processing delay varies by device, it is difficult to model with a single variable. That said one can characterize overall processing delay of a particular signal path by first characterizing the delay of each device and adding those delays. For the purposes of this illustration, we will assume that one processing function requiring a full frame of delay (such as would be typical with a DVE or digital video effects engine). Furthermore, we will assume that other stages of the system have processing elements with a total delay per stage equal to four lines of video. As an example, in the case of a three stage system with 1080i59.94 video, this equates to a processing delay, P, of 33.06 msec.\(^4\)

---

\(^3\) J_F = 6.7 µsec + 1.2 µsec

\(^4\) 33 msec of frame delay + (2 x 29.7 µsec of delay for two stages with 4 lines of delay each)

\(^5\) One should note that graphics processing delays are one class of processing that can greatly exceed even one frame, ranging between three and nine frames. Technical directors (TDs) in live production must be careful to avoid switching between a camera source and the output of the graphics device which is processing that same source. While this can be managed as a special case, operation would be impaired if such management were required on many signals.
USING THE MODEL TO DETERMINE THE REQUIRED LATENCY OF A SPECIFIC VARIABLE

Once we have characterized the latency of all but one variable, one can then use (2) to solve for the maximum allowable latency of the remaining variable. In our illustration, we are seeking to solve for the maximum allowable value of \( C \), codec delay. There are two primary reasons why, historically, compressed video has not been used in live production as a primary routing format other than for remote production: image quality concerns, and increased latency. While the model proposed in this paper does not answer questions regarding image quality, as we will illustrate, it can be used to determine the maximum tolerable latency, \( C \), of a codec.

To apply the model, we first take (2) and solve for \( C \) which yields:

\[
C \leq \frac{100\text{msec} - D - J_r - (J_{PV} \times n_c)}{n_c} \tag{3}
\]

Then we must pick our video format as many of our assumptions above were expressed as “frames” and “lines of video”, and convert “frames” and “lines” to units of time. For the purposes of this illustration we will use a UHD format at a 50 Hz frame rate, 2160p50. This is an especially pertinent frame rate as real world live production systems using both light compression technology and this frame rate exist [5]. However the same methodology can be applied to other frame rates as well. We also assume a 10 Gbps Ethernet speed which affects the calculation of \( J_r \) and \( J_{PV} \) per our previous analysis, though relatively speaking; the impact between rates such as 10 Gbps and 25 Gbps is small as compared to other factors.

After determining our frame rate and Ethernet rate, we then apply that rate to our system assumptions from the previous section:

- \( M \) (multiviewer latency) = one frame = 20 msec
- \( D \) (display latency) = one frame = 20 msec
- \( J_r \) (latency due to jitter at transmitter) = 2 \( \mu \)sec\(^6\)
- \( J_{PV} \) (latency due to jitter from packet variable delay) = 12 \( \mu \)sec
- \( n_r = 5 \) stages\(^7\)
- \( R \) (latency due to 2022-7 redundancy with buffering appropriate for LAN applications) = 100 \( \mu \)sec/stage
- \( n_R = 5 \) stages
- \( P \times n_p \). Here we take the assumption made in the previous section where we have one stage with a delay of one frame delay, and we have \( (n_p - 1) \) stages each with a delay of four lines. This equates to 20.47 msec\(^8\) which we would substitute into the equation for \( P \times n_p \).

- \( n_c = 5 \) stages (of encode-decode).

Plugging the above values into (3) yields:

\[ C \leq 7.8 \text{ msec} \]

In other words, according to our model, a five stage IP based live production system using 10 Gbps Ethernet technology and 2160p50 video could, under the assumptions stated above, deliver an overall visual-to-visual system latency that would be acceptable to a typical production crew, so long as the latency of each codec stage were under 7.8 msec. This is but one example of an application of the mathematical model described in (2), but it is illustrative of how the model can be used. As further validation of the model, the author and his colleagues have used this model to successfully design IP-based production systems whose visual-to-visual latency performance has been perceptually identical to SDI based systems.

CONCLUSION

The fundamental latency requirements for live production are invariant with changing network transmission speeds and video frame rates. They are based on human factors. The mathematical model described in (2) can be used to characterize total system latency of IP based live production systems. By carefully designing for each variable in the model, practical live production systems can be built using IP technology.

ACKNOWLEDGEMENTS

The author would like to thank his many colleagues at Grass Valley, a Belden Brand for their input and dialogue which helped shape the development of this model for total system latency. Of particular note is Chuck Meyer. Chuck’s thoughts and guidance were both essential and instrumental. The author would also like to thank the members of the AIMS technical working group who provided great insight and input, particularly Greg Coppa and Bob Seidel of CBS, Thomas Edwards of Fox Networks, Mike Overton of Tektronix, Andreas Hildebrand of Lawo, Gerhard Lang of VizRT, and Markus Weber of Avid.

REFERENCES


\(^6\) the time to accumulate one packet of data at the video sample rate (3.4 \( \mu \)sec at 2160p50 line sample rate) minus the time to transmit one MTU at the Ethernet line rate (1.2 \( \mu \)sec with 10 Gbps Ethernet), which is 2 \( \mu \)sec rounded to one significant digit

\(^7\) The system referenced in [5] has fewer stages than five, but five is the number that some US networks have informally estimated as the expected maximum number of stages for IP based live production systems where latency within human factors tolerances is required.

\(^8\) 20 msec + (4 x 117 \( \mu \)sec) = 20.47 msec