Challenges in Relaying Video Back to Mission Control

CONTENTS

White Paper
Revision 3
July 2020
By Christopher Fadeley

Using a customizable hardware-accelerated encoder is essential to delivering the high compression ratio and guaranteed speed needed for critical manned and unmanned video streaming applications.

1. Introduction

The change in emphasis to a more avionics approach to the military theatre has introduced a new set of challenges in terms of technological feasibility. It is often requested to have an active video feed sent large distances back to ground, with low latency and in high fidelity. These criteria must always be met. The video feed is typically used for monitoring and/or recording of the mission, so if the video feed arrives too late or in poor quality, the resulting images are of no use to control staff and the mission can be compromised.

The real challenge is dealing with the low bandwidth available to stream video. All transmissions must be sent wirelessly, and ground control may be a long distance away, especially when it involves remote controlled Unmanned Aerial Vehicle (UAV) or intelligence gathering missions (Figure 1). Wireless transmission is usually performed in an atypical method with limited bandwidth, like a cellular relay or satellite transmission. This often puts a strict limitation on the data transfer rate of video.

Figure 1: UAVs need the feed streamed remotely in as timely a fashion as possible, otherwise the mission may be compromised.

LRaw video by definition is lossless but is also highly wasteful in the amount of data it takes to display. A 1080p 30fps raw video has a data rate upwards of 200MB/s (megabytes per second). Raw video has its place in the military field in the form of live local viewing/recording and/or GPGPU processing on the fly. But in applications where the video needs to be

on the fly. But in applications where the video needs to be streamed remotely, raw video is simply infeasible. The solution is to compress the video, but this comes with its own set of challenges. Compressing video can cause a drop in quality. The bigger challenge is that video must be sent in a timely fashion and compression introduces latency. Latency is the time between the camera capturing the video and the time that data is actually displayed to the end user. Compression takes time, especially if high quality and/or high resolution is required. The best way to ensure low latency with reasonable compression is to use dedicated compression hardware. And given the military field, the hardware must be ruggedized to survive in harsh environments and still consume low power. Keeping that low power while compressing with expected results can be difficult. Hence the hardware must also be computationally efficient.

2. Encoding

The current most widely adopted standard for compressing video is H.264/MPEG-4 Part 10 AVC. The process of compression is also called "encoding". H.264 is the most popular codec because it offers a high compression ratio with highly configurable options and is offered in many existing products (dedicated hardware, in GPUs, CPUs, software, etc.). The strict bandwidth limitations make the configuration of H.264 a necessity. Encoders typically encode to a constant or variable bitrate. Constant bitrates are easy to understand. The video being encoded is always encoded at the user defined bitrate with no regard for the video being captured.

This allows an end user to strictly define their video stream to their known bandwidth ensuring no transmission overflow since the video is guaranteed to be a certain size. Variable bitrate is much more complex since it encodes based on what the actual video data is. A video stream with high motion (every frame different from the last) is difficult to encode. Compression at a basic level relies heavily on the concept of redundant data through time.

For example, a zip file looks at the binary data of a file and sees where data is being repeated. If the file being compressed is a document containing the letter 'A' repeated 1000times, the encoder can analyze this data and store the file as 'A:1000' instead of the letter being repeated so many times. This saves valuable space. In the case of video, a repeated black screen is quite easy to encode since it is the same pixel data over and over. Variable bitrate encoding can be set to a target bitrate which it tried to meet; however, it is used only as a median. This means when a video is easy to encode, a variable bitrate will reduce the bitrate and in turn optimize the bandwidth available. The same goes for sudden difficult to encode video (fast moving motion video) where the bitrate may increase to ensure a decent quality.

If a video stream is difficult to encode, there is a chance the variable bitrate encoder will surpass the available bandwidth. This could result in potential loss of video (dropped frames or buffered delayed frames). Therefore, constant bitrate settings are most often used for the military space Variable bitrate encoders do have the option to enact a strict maximum size in order to limit this potential for overflow, but this can cause more strain on the encoder which is unnecessary for most military applications and ultimately the variability of the bitrate is an unnecessary addition to an already complex system. H.264 also has various profiles or ways in which the encoding is handled – the 3 primary profiles are Baseline, Main and High. Baseline is computationally simple and fast to encode. Main and High and more features (like B-frames) making the resulting compression ratio better, but at the cost of computation time and hence latency.

3. Time is of the Essence

Compression ratios keep getting better and better, but this always comes at the cost of simplicity and processing time. Anything that becomes more computationally difficult typically takes longer to process. The process to receive video from a sensor and send it back to ground is quite complex as Figure 2 shows. First, raw video is captured by the sensor and sent to a processing board. Video is then sent through a series of decoders to present that as actual video data to the system. This actual video data is then sent to the on-board hardware accelerated encoder. The video is then encoded to H.264.

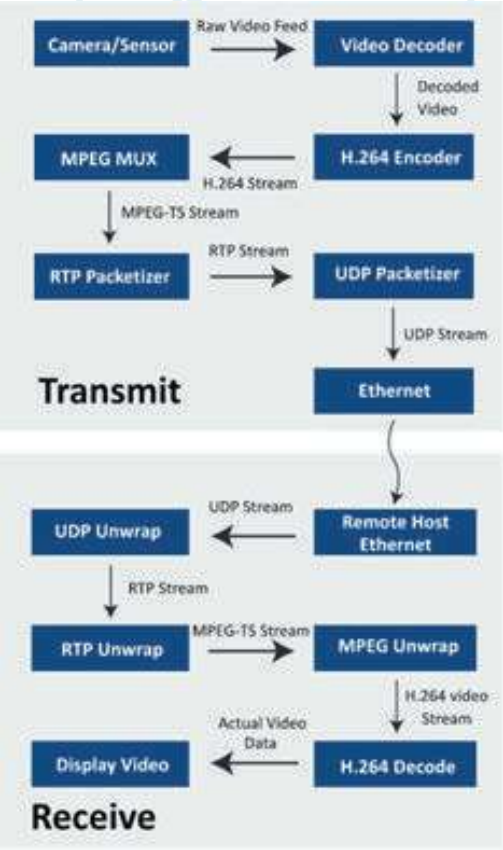


Figure 2: Block diagram shows the full pipeline of sending video remotely and then displaying at control.

The resulting compressed video is then muxed with audio and/or metadata streams typically into an MPEG2-TS container. MPEG2 Transport Stream is most widely used container for streaming formats because of its 188-byte encapsulation that allows for easy segmentation of data for live viewing attachment. It also supports most necessary program formats and is perfect for synchronization of KLV metadata typically found in aerial payloads. MPEG2-TS can then also be encapsulated into RTP (Real-Time Transport Protocol) packets. The RTP packets are then merged into UDP packets and sent out over Ethernet. Back at ground, the UDP packets are captured and the reverse procedure occurs (UDP --> RTP --> MPEG2-TS).

The MPEG2-TS stream must then be unwrapped, and the video is decoded into a presentable format. This now presentable pixel format is rendered to the screen to be viewed. All these steps are expected to be performed as fast as possible, preferably (but unrealistically) with no latency.

The main "time" costs in the above system are the time to encode, the time to relay back to ground, and the time to decode. As previously mentioned, the time to relay back to ground is solely dependent on the environment of the aircraft and the means to send back the data. The encoding time is manageable based on hardware and settings. Dedicated H.264 hardware acceleration is typically required to properly meet user expectations while also not stealing precious CPU cycles from other applications.

With a dedicated encoder, the time to encode and wrap into a streaming format is minimized. Encoding is only one part of the large video pipeline and many other parts of that pipeline typically cannot have their latency improved upon (camera sensor buffering, network intermittency/latency, etc.). Typically, the encoding pipeline is focused to be improved upon as much as possible. It is also important to note the video must eventually be decoded on the receiving side. This means the encoded stream must be extracted from the MPEG2-TS mux and decoded back to regular pixel data before being displayed. This is essentially reversing the whole encoding process already performed and in turn takes a similar amount of time to the encoding process. Like the encoding process, this too must be optimized as much as possible to ensure a timely display of remote video. Many of the latest GPUs (graphics processing units) have built-in decoders that display applications should utilize. The receivers also must render the decoded data to the screen. This means the GPU rendering performance/buffering and monitor latency must also be examined to ensure optimal implementation.

4. The Reality

Video needs to be sent back to ground in a timely fashion. Otherwise, the entire mission could be compromised. The reality of the situation is video being shown at ground will always be "late" when compared to what is actually happening. The goal is to limit how "late" the video is by as much as possible. If the video is too "late", it is ultimately useless, and decisions made back at ground are being made a faulty reality. The result of these decisions based on incorrect data can be devastating. A dedicated encoder onboard must then be optimized for the specific environment. As previously mentioned, the strict bandwidth limitations are a real challenge. With strict bandwidth limitations, the video will never look perfect. And if it is a high motion video, it may not even be close. Hence if the video isn't time sensitive (monitoring the video instead of controlling back at ground), certain optimizations can take place at the encoding layer which add extra time (in the magnitude of ms) but may improve better quality. Implementing higher profiles/levels may take a few extra milliseconds (5-10ms) but may be the necessary addition to make the video useable. Filters (like temporal motion filters, resizers, denoisers) can be applied pre-encode to make the encoding process simpler for the encoder. Again, any addition to the pipeline will always add some degree of extra time, but it is a matter of pro/con tradeoff between latency and quality.

5. The Tradeoff

Encoded video will always have the tradeoff between size, quality, power consumption and speed. The smaller the size, the worse the video will look. And the more processing that needs to be performed to improve this quality adds to the final latency and power. Further developments in the encoding field will continue to improve this situation, but as encoding algorithms progress in complexity, so does the computational difficulty. H.265 (also known as HEVC) is the successor to H.264 and was first truly deployed in 2014 but is just now starting to be more widely implemented. Whereas hardware technology over the course of the last 15+ years since the original release of H.264 has led it to a point where implementation is more available and simpler, many computers are still just starting to integrate H.265 acceleration. And H.265 requires hardware acceleration for real-time performance as currently deployed software solutions (x265 for example) are highly difficult on CPUs and even when using up the full power on an embedded CPU, it still can't keep up with high definition video in real time.

NVIDIA and other GPU manufacturers now include video codec acceleration built directly into their GPU ASICs to allow for readily available hardware accelerated encoding to be easily accessible when GPUs are being utilized. Intel also has started including an encoding core on their CPUs. But even with these options available, only the "encoding" part of the pipeline is performed. Video capture, delivery of video frames and retrieval of encode data, muxing, and streaming of data out to ethernet must all be performed on the host SBC CPU board. This again steals potentially critical CPU resources away from other applications. These systems may also not be useable in more power restrictive integrations.

6. Alleviations and Implementation

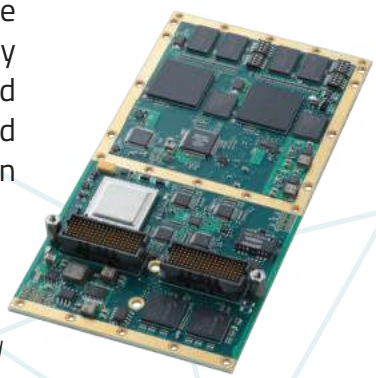
Based on implementation and application need, there is the possibility to implement a system which highly alleviates this tradeoff simply via brute force with multiple streams. If the encoding product has multiple dedicated encoders on board which can be customized individually and bandwidth availability permits, then both encoders can be utilized to meet all needs For example, one encoded stream can be set to encode at the full 30 frames per second at low quality ensuring every frame is sent timely and with no frame drop. The other encoded stream can be set to encode at a higher quality but at a lower framerate (5fps for example) and with more leniency for buffering. This way a video feed can be analyzed in real time with no frame loss for live use and the second higher quality stream can be recorded (locally) or referenced live if there is a sudden need for high visual fidelity.

Figure 3: The Tyton VS2X supports H.265 and H.264 encoding with 4x 3G SDI inputs and KLV metadata parsing.

EIZO Rugged Solutions offers a wide variety of products in different form factors that can be used to integrate encoding solutions. The Tyton VS2X (Figure 3), a box level encoding solutions are highly configurable with a dedicated 8 encoding cores each of which can be utilized by any video input. The Tyton line of products also introduce support for H.265 in addition to H.264 encoding, KLV metadata extraction and parsing, and a wide variety of additional preprocessing filters (resizers, text overlay, frame rate reducers, etc.).

A low power H.264 XMC solution called the Condor VC102x (Figure 4) has been deployed for almost a decade. All embedded NVIDIA GPUs available from EIZO include access to the NVENC core which has integrations with FFMPEG and the NVIDIA's own Video Codec SDK. EIZO provides a wide variety of sample applications demonstrating use of this GPU encoder.

Figure 4: The Condor VC100x XMC H.264 encoder



7. Conclusion

There will always be latency in sending video from aircraft back to ground. There is simply no way around this reality. The solution is to alleviate as many bottlenecks as possible in the pipeline prior to implementation. A dedicated hardware encoder is an absolute necessity to limit this delay. Only a dedicated customizable encoder optimized for high efficiency with low power consumption can be used in both manned and unmanned avionics streaming.

This article was originally published in COTS Journal's April 2014 Issue. It was then revised in October 2021 and re-released with updated content.