DOI 10.5162/ettc2020/5.3

Airborne FTI Camera Performance

Russ Moore[1], Bruno Marchevsky[2]

1: Curtiss-Wright, 15 Terry Drive, Newtown, PA, 18940, USA
2: Z3 Technology, 100 N. 8th St, Suite 250 Lincoln, NE, 68508-1369, USA

Abstract: Flight test instrumentation (FTI) have long sought to benefit from the bandwidth, scalability, and low cost of Ethernet networks and Internet Protocol (IP) based systems. Merging video and imaging cameras in the next generation FTI networks will require new methods and protocols to enhance data movement and recording and for time coherency of data. This paper will address the system and performance of the video encoding and decoding components, and various network performance issues, and the vision for deploying camera and FTI equipment in the same Ethernet network.

Keywords: IP cameras, H.264 AVC and H.265 HEVC, FTI displays, telemetry

1. Introduction

An internet-protocol camera, commonly referred to as an IP camera, is a digital video camera much like a high-definition (HD) camera or even a webcam. It transmits and receives data over a network or utilizing internet protocols to communicate video and audio information. Unlike an ordinary webcam it is a standalone unit with its own IP address that requires nothing more than a network connection in order to transfer images. The IP camera connects to a network in exactly the same way as any other standard network device such as a laptop or printer or even FTI Data Acquisition System (DAS) devices.

There is increasing demand for high-quality HD video for airborne applications such as flight test instrumentation (FTI). Ideally, such new camera solutions can reduce the weight and difficulty of installing wiring, and enable data to be coherently combined with image data. Ethernet cameras can address these needs with built-in compression and IP data format encoding with multiple output video streams. Additionally, as Ethernet-based networks have become an attractive choice for FTI applications, we see increased requirements for integrating Ethernet-based cameras alongside FTI data acquisition equipment, network recorders, and telemetry systems as this removes duplication of wiring and devices.

Using an Ethernet camera that supports onboard compression enables video compression to be removed from the DAS, or it can eliminate a dedicated unit. The camera can be connected via an Ethernet switch directly into the system, like any other data acquisition unit. Even better, because there is no need for dedicated hardware compression, SWaP is minimized and installation wiring greatly simplified.

The images captured by an IP camera may be viewed from anywhere in the network, whether via pc, laptop or video display, telemetered to remote displays or dedicated video analysis equipment and network recorders. In many cases, as well as being able to view video footage and listen to audio streaming, the camera may also be controlled remotely.

The key considerations when designing and operating the FTI devices on an aircraft or vehicle are image quality, bandwidth and latency. This paper investigates and evaluates these factors.

2. The IP video ecosystem components for FTI

The key considerations when designing and operating the FTI devices on an aircraft or vehicle are image quality, bandwidth and latency. This paper investigates and evaluates these factors.

The internet was not originally designed to deliver video and synchronized audio. HTTP live streaming protocols emerged as a way to deliver video/audio files over the internet by breaking the stream into a sequence of small http-based video/audio chunks. By using http transactions, live streaming can transverse firewalls and use standard web servers to deliver video streams and scale when many cameras are streaming to end-points.

The IP cameras used for FTI missions need to be designed for rugged environment usage with aircraft power converters able to withstand short power outages and aircraft power levels, ruggedized enclosures able to withstand high vibration and shock environments as well as mitigation for EMI and EMC effects.

The base ecosystem consists of devices that provide data as use-cases, such as

- Acquire (data, video)
- Record
- Display
- Telemetering
- Processing

These are all supported by the network “fabric” which provides data movement and communication activities, as shown in Figure 1.

DOI 10.5162/ettc2020/5.3

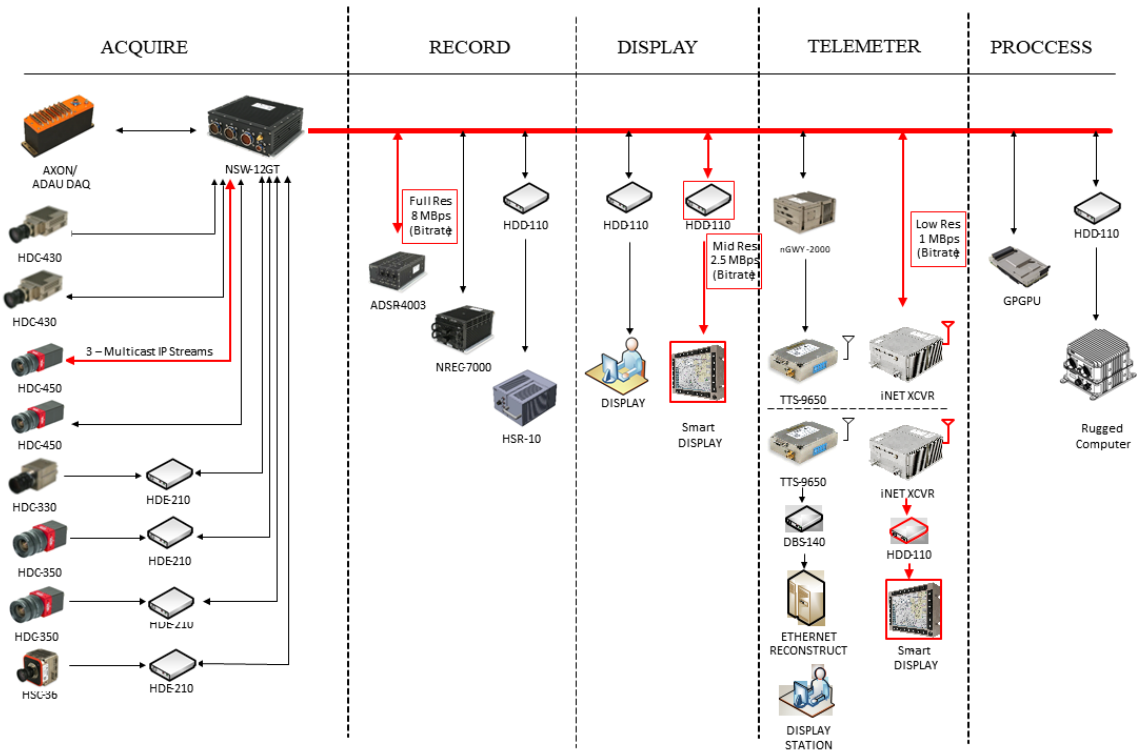

Figure 1: A network fabric supports all the use cases for the video data

3. Data acquisition in IP cameras for FTI Applications

Capturing and recording data from cameras in flight test applications is just as important as capturing video. Data is made available in multiple formats and transmitted in multiple interfaces, but the most common formats by far are carried over a standard serial port (RS232, RS485 or RS422) or as embedded data packets in the blanking time of SDI video. Single, slow events are many times represented by contact closures or their digital equivalent, a "GPIO" (general purpose input output).

Modern IP cameras tailored for flight test can acquire these data items embedded in the video stream, from a serial connection or from a GPIO. Because the data collection and the video capture of the camera are integrated and being performed in the video capture device, the IP camera system is able to closely correlate the time of arrival of these events with the time of each frame of the video, resulting in a more tightly integrated system with better performance and lower cost than a similar system built up from multiple boxes.

4. Synchronization of the Video and Data Streams

Compressing the video dramatically reduces its bandwidth requirements and is a necessary step for storing the video stream, or enabling its transmission over a communication method which provides less bandwidth than the full uncompressed video (e.g. an IP network). Algorithms such as JPEG, H.264 and H.265 among others "encode" the images or video according to the rules of that standard to reduce the number of bits required to represent the image.

Video compression is a complex operation which can sometimes introduce large amounts of jitter. In order for the receiver of the stream to be able to pace the frames correctly into the video compressor, the encoder includes a time stamp with each video frame which identifies the exact time when that frame is to be displayed. This same mechanism is used to time stamp the data acquired prior to bundling it with the video content for transmission. The decoder at the receiving side is then able to put together the sequence of events with their correct timing.

4.1. Synchronizing the Video, Audio and Data Streams with real Time

As mentioned, compressed video from the IP camera naturally carries timing information. It is possible, and often desirable, to correlate the time stamps from the video frames or metadata packets with a real-time clock. In order to do so, some IP cameras with IP-encoders, can use the time protocol IEEE-1588 PTP to synchronize the video acquisition with the time in the real world. Once the relationship is established the video and data time stamps can be related to the world clock.

Depending on the streaming protocol desired, the timestamps themselves can be made to carry real-time, as opposed to a time relative of when streaming started or power was turned on, or additional metadata. This can be added to the stream making the correlation between the camera hardware time and the world time reference.

The accuracy of the relationship depends on how the real-time is acquired and how good the time reference is. For

example, using PTP is inherently more precise than using the network time protocol (NTP). But neither would be accurate if the network server that provides the time reference to the IP camera itself has the wrong time or is set up incorrectly.

In practice time stamps can be correlated to precisions between 100's of nanoseconds and tens of milliseconds, which is perfectly adequate for the vast majority of applications.

4.2 Synchronization of streams between multiple IP cameras

It is possible (and fairly common) to synchronize and correlate in real-time the data output of multiple IP cameras, metadata streams and all.

The above mechanisms of providing a method to relate one camera's timestamps to a clock reference on the outside world also provides the way for multiple streams to relate to each other, provided that all IP cameras involved are synchronized to the same time standard reference. Once that reference is established, it becomes possible to collect time-correlated data from multiple IP cameras in multiple locations connected by just the IP network. Streams synchronized this way can be recorded for future use or displayed in real time, or both, all along retaining the timing relationship between the various video frames and data items.

5. Quality metrics and considerations for IP video and telemetry

5.1 Video image quality vs network bandwidth

Understanding the basics about network bandwidth availability and demands is critical to planning, designing and deploying a network-based video systems with IP cameras in FTI. To determine bandwidth availability, you first need to determine what locations you are communicating between.

The most important factor in determining how much bandwidth is available is whether or not you are sending video within a vehicle, or small space, or telemetering the video from an aircraft to ground locations. Within a small space such as a vehicle, the bandwidth availability could be large, say 70 Mbps to 700 Mbps. Sending video off-vehicle to ground stations or in a space application use, the bandwidth could be quite small, e.g. 250 kbps to 4 or 5Mbps.

5.2 Image quality vs compression loss

Video compression is necessary so that the amount of information created fits the allowed amount of capacity in the channel. Video information is highly compressible because it contains lots and lots of redundant, repetitive or not very important information and encoding can therefore eliminate the need to transmit or store such information. The encoding process is a collection of methods that seek to remove the "extra" information step by step, in an assembly line kind of way.

One example on how compression is achieved is the "color space" used to transmit the visual information. We all know that red, green and blue are the primary colors because they match the color receptor cells in our eyes. The light sensors in our retinas are much more sensitive to the overall light levels than to color, so the luminance information of an image is more important for our brain than the color information.

This causes the coloring book effect – a child can splash color in a coloring book, and as long as the black and white countour lines are visible our brains see a good picture. This effect is applied by video compression, where the algorithm is allowed to make the color information "fuzzier" in order to lower its bandwidth requirement.

To take advantage of this effect, the captured image which has red, green and blue components or RGB, are converted to a color space called YUV. The "Y" stands for luminance, or brightness, while U and V carry the color. Pretty much all image compression schemes use the coloring book effect and just discard a lot of the color information. H.264 and H.265 for example, discard a whopping 75% of the color information.

If we started with, let's say, four pixels, each with R, G and B values, H.264/H.265 will convert to the YUV space and keep four values of Y (that is the black and white, important part) and keep only one value of U and V. So instead of carrying twelve pieces of data (RGB x4) now it carries only six (YYYY UV). This alone is already a 50% reduction in the amount of information. The side effect is that for all intents and purposes these four pixels are painted with the same color at the receiver, when the YUV data is converted back to the RGB color space.

The interesting point to make from the above example is that information was lost. In the color space conversion the loss of information is perceptually minimal – the nature of our vision is such that we don't really see the negative effects of losing that data (color blurriness). The data loss, however unimportant, is real.

Other obvious ways to remove redundancy is by recognizing that video is by nature repetitive. Why transmit pixels – even in YUV space – if these same pixels were already transmitted before? So for example, if the video stream is that of a clock on a wall where nothing else moves, why keeping re-transmitting pixels of the wall around the clock? Instead, it should be sufficient to just send the pixels that changed when the arms of the clock moved.

Indeed, motion video compressors do just that. By storing the previous image on a local buffer, the encoder can use the stored frame as a reference to compare the new image with and only encode the parts that are different. Modern encoders go a lot further than just requiring images to be different. H.264/H.265 have methods to detect that objects moved within the frame, and instead of sending all the pixels of the affected area it can just send the information that "this object moved from here to there".

While all these schemes can achieve impressive compression ratios, there is only so much truly redundant information. At some point it becomes necessary to

remove visual information from the picture in order to achieve the amount of compression required to fit in the channel. In the clock example above, if we only transmit the pixels that have the clock hands we could lose the change in shading on the wall itself due to the change of illumination as the Sun goes up and down.

Perfect fidelity would require us to transmit all the pixels all the time. "Good enough" requires us to update these pixels only every so often as the motion is quite slow. The important thing is that with encoding the fading brightness of the wall in the evening will be in steps, as pixels are refreshed by the encoder. We can choose to update these pixels anywhere from very often to almost never and in the process create a trade-off between more bits to create a better image quality, or fewer bits to fit in a "smaller" channel or record longer durations.

5.3 Choosing the best compression parameters

Video compression is a complex process with many different "tools" looking for possible reduction in bits. Each of these steps/tools are minutely specified by the encoder standard, and many have parameters that can be adjusted at the encoder, just as above we can imagine a control parameter which would specify "how much different is different enough" to trigger transmitting that pixel. Tools exist that determine how big some area of pixels should be for subsequent operations, how far should the encoder look for objects that moved, how to look for camera movement (whole image shifting), how many and which frames to retain to use as reference for future comparison, how to pack the bits resulting of the encoding, etc. There are lots of different buttons and levers that affect the trade-off of quality vs. amount of bits.

In a compressed video system, such as the IP camera, the quality of the image is directly related to the amount of bits used to transmit it. More recent encoders are "better" because they are more efficient in the sense that they can create images that we perceive at the same quality level as other encoders, only using less bits to do so. This is accomplished by better, deeper analysis of the video given to the encoder, which of course tends to be computationally more intensive. As an example, H.265 coding is 25% to 50% more efficient than H.264, meaning that it requires 25% to 50% less bits than H.264 to achieve the same level of image quality. On the flip side the H.265 encoder uses about 10x the computing power required by H.264.

The grand takeaways are that it is better to use H.265 encoding technology if is available, as it is going to provide better image quality given the amount of bits that can be had. Also, experiment with encoding parameters. They are so complex and there are so many interactions between them that it is best to just play around and get familiar with them. The trade-offs can be significant and many times the trade-offs are not intuitive.

For example it is common that for smaller bit rates a smaller resolution stream looks better than one with higher resolution. For example at 250 Kbps a video stream at 320x240, 15fps will usually look better than a 720p, 60 fps stream (unless there is no motion).

When in doubt, ask the IP camera vendor for help. The IP camera designers will be familiar with the behavior of their camera (they are all different) in many different applications, and will be able to provide at least pointers for optimal configuration.

6. Video latency

Latency is defined as the amount of time between the instant an image frame is captured by a camera and the instant that frame is displayed. Low latency is a design goal for any system where there is real-time interaction with the video content is required; such as pilot, crew video interaction or through robotic and computer image processing applications.

There are several stages of processing required to make the pixels captured by a camera visible on a video display. The delays contributed by each of these processing steps—as well as the time required for transmitting the compressed video stream—together produce the total delay, which is sometimes called end-to-end latency.

The following aims to define and explain the basics of video latency, and discuss how one of the biggest impacts in reducing latency comes from choosing the right video encoding.

6.1 Acceptable latency

There are several stages of processing required to make the pixels captured by a camera visible on a video display. The delays contributed by each of these processing steps, as well as the time required for transmitting the compressed video stream, together produce the total delay.

But the biggest contributors to video latency are the processing stages that require temporal storage of data, i.e., short-term buffering in some form of memory.

There is no universal absolute measure of video latency. Instead, what is considered acceptable varies by use-case, or application, of the video. When the application is for human interaction, humans can react to stimulus input at roughly 5 Hz (human reaction times are noted in Table 1).

Table 1: Human / Operator Average Reaction Time

Stimulus	**Ave. Time**
Video / Visual	250 ms
Audio / Sound	170 ms
Touch / Pressure	150 ms

Video latency under 200-250 ms is generally acceptable while under 175 ms is considered low latency. In an application where a machine will interact with video stimulus, latency requirements are usually much lower; under 50 ms and many times under 30 ms.

6.2 Examining latency within systems

From camera to network to telemetry to video displays and recorders, most design goals look to achieve the

optimum balance of encoding speed, network transmission speed, telemetry bandwidth and video quality.

A downside to achieving low latency is sacrificing video image quality. Image sensor speed or image frames per second can affect latency since smaller frames-per-second add to latency. Testing also has shown that using the real-time protocol (RTP) and the real-time streaming protocol (RTSP) lower overhead time and helps lower latency performance. GOP size (group of pictures i.e. how many images are bundled together as intra and progressive coded pictures) is also important to latency; larger GOP size improves latency.

There are typically five main sources of latency in an Ethernet network IP camera video system:

- Image sensor/source
- Video encoder and compression
- Network and/or telemetry transport
- Video decoder
- Video display / renderer

The image source latency can come from image sensor data architecture and sensor data-clock and data serializer-deserializer, raw (Color Bayer Pattern) data processing and data to RBG conversion and, finally, color detect and segmentation. This is shown in Figure 2.

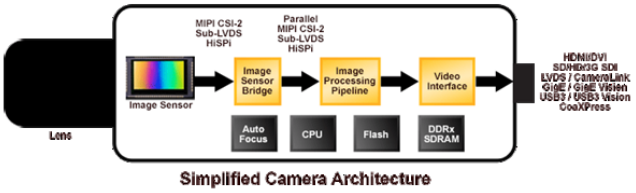

Figure 2: Simplified camera architecture

The video encoder and compression engine introduces latency as the encoder frame capture and process engine can scale, filter and translate image data. Many times a technique to lower latency will be to disable the upscaling process and to set image sensor resolution appropriately. **Fehler! Verweisquelle konnte nicht gefunden werden.** below provides latency times from an FTI IP camera using various compression techniques.

Table 2: Codec and latency table

Codec	Resolution	Latency
H265	1080p60	37-70 ms
H264	1080p60	35-60 ms
MJPEG	1080p60	37-50 ms
H265	720p60	30-50 ms
H264	720p60	30-50 ms
MJPEG	720p60	29-45 ms

Latency from network transport architecture illustrates how many times that latency is dependent on the network transport type. In testing it is shown that using RTP and RTSP protocols allows most network architectures to achieve the lowest possible latency results. This is primarily due to the low overhead of the protocol.

Networks can be as simple as star configured Ethernet or more complicated with Ethernet, PCM conversion, RF telemetry radio communications, PCM-to-Ethernet reconstruction, etc. Network throughput issues many times cause buffer data to stack up on the encoder and even momentary degradations can cause large amounts of added latency to video.

Next, decoder latency issues can be the greatest source of video latency. Here, timing is the issue. The decoder needs to tie the encoder video time to the display timing. The buffer level scheme should ensure the display never runs out of buffer to ensure smooth video playback. Hardware, FPGA-based, decoders are better at synchronizing the encoder-display timing. **Fehler! Verweisquelle konnte nicht gefunden werden.** shows encoder and decoder effects on frames and decoder network cache.

Table 3: Encoder - decoder effects

Encoder FPS	Decoder Network Cache	Frames in Decoder Buffer (For Smooth Playback)
60	80 ms	5
30	180 ms	5+
15	400 ms	6
7	900 ms	6+
5	1200 ms	6+

Finally, the latency at the video display or renderer needs to be considered. The display latency can vary widely with display design architectures and display settings and especially higher resolution settings can add latency. PC and gaming settings can provide lowest latencies. Measured display latency from 33 ms to 120 ms are typical. See figure 5 below for measured and theoretical latency.

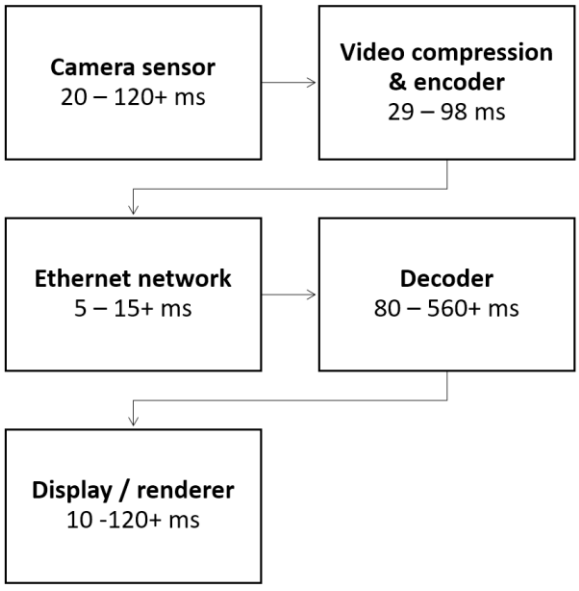

Figure 3: Measured and theoretical video latency

7. Conclusions

In the FTI implementation of IP camera networks, bandwidth dedicated to video is critical. This includes the information communicated from acquisition to recording, visual display and further processing.

The CODEC used greatly influences bandwidth and video quality. As stated earlier, the grand takeaways are that it is better to use H.265 encoding technology if is available, as it is going to provide better image quality given the amount of bitrate. At the video display, decoding functions tightly tied to display operations will also enhance latency timing.

Constructing the FTI network "fabric", i.e. managed network switches, is key. Elements such as IEEE-1588 PTP time grandmasters, and video and data acquisition equipment able to time synchronize and time-stamp the acquisition of data, and simultaneously send it to consumptions point, such as recorders and displays, will provide time coherency for data reduction activities.

Finally, while there are several elements that all contribute to latency in a video system, including image source, encoder, the network fabric and the display, the most impact is often attributed to the decoder. This latency can best be reduced by using hardware such as an FPGA decoder that is tightly linked to the display.

7. Glossary

DAS: Data acquisition system
EMC: Electromagnetic compatibility
EMI: Electro magnetic interference
FPGA: Field programmable gate array
FTI: Flight test instrumentation
GOP: group of pictures
GPIO: general purpose input output
HD: High definition
IP: Internet Protocol
NTP: network time protocol
PCM: Pulse code modulation
PTP: Precision time protocol
RF: Radio frequency
RGB: Red green blue
RTP: real-time protocol
RTSP: real-time streaming protocol
SDI: Serial digital interface
YUV: Luminance-Bandwidth-Chrominance