Using ASTERIX CAT-240 for Radar Video Distribution – Practical Considerations from Deployed Applications

Dr David G Johnson and Mr Richard Warren
Cambridge Pixel Ltd,
Cambridge, UK
dave@cambridgepixel.com

Abstract

This paper describes the practical use of ASTERIX CAT-240 messages for the distribution of radar video in naval command and control systems. The standard, which has emerged from the European Air Traffic Control community, offers a method of harmonising the exchange of radar video from a sensor or server in to multiple display clienst. This paper describes the background to the standard and explains some of the practical challenges for deployed applications in naval command and control applications.

INTRODUCTION

The presentation of a primary radar picture on a command and control display requires that the original radar video from the sensor is acquired and then scan converted to create the radar image, along with graphical components for map and symbols. The acquisition process involves interfacing to the video provided by the radar. Although modern radars employ all-digital processing stages, a common interfacing standard is based on an analogue video signal, accompanied by a range-zero reference trigger and azimuth signals (for example synchro or ACP/ARP pulses that indicate incrementing rotation). The analogue video interface was the only option when the radar processing was based on linear amplification of analogue signals, but when radars moved to digital processing the provision of an analogue output, which requires an otherwise unnecessary digital to analogue conversion, may still be the easiest way of ensuring compatibility of the radar signals with existing displays.

An analogue output video must be re-digitised for the purpose of displaying the radar picture on a modern command and control display. Therefore a modern radar and display system may incorporate a digital-to-analogue conversion in the radar processing followed by an analogue-to-digital stage in the display computer. As well as the potential for noise introduction and signal quality loss, the unnecessary creation and re-sampling of the analogue video adds cost and complexity to the system design. Clearly a preferred option is to avoid the analogue stages and maintain all-digital processing through to a network output based on standard IP protocols.

Although there have been no technical barriers to the specification of a standard for the network-based distribution of radar video, there have been a number of factors that have hindered progress. As the market for radar interface products is small and specialised, a formal standardisation process could not be justified. For radar manufacturers themselves there is an interest in maintaining proprietary standards, to ensure their radars are used with their own display or processing software. From a commercial perspective it may not be desirable to permit third-party access to the radar data. This is the situation with many lower cost radars, for example as used in commercial shipping applications. The situation makes it difficult to consider these radars for specialised applications, where otherwise the specification of the radar sensor would make it a candidate. With the inability to interface to the proprietary digital network, the network interface is unusable, so where the radars still provide the old legacy analogue interface this provides the only interfacing option.

A STANDARD FOR NETWORK DISTRIBUTION OF RADAR

A standard for radar video distribution has emerged in recent years and is becoming increasingly adopted by forward-looking radar companies offering open-standards solutions. The solution is based on ASTERIX, which is a set of standards originally developed for message and data exchange in European air traffic control applications. The ASTERIX standards identify a collection of message types, called categories or CAT.

For example, CAT-48 messages define the format for the exchange of track reports that define target positions. The standard that covers radar video distribution is CAT-240.

There is nothing specific in the standard that ties it to air traffic control and the family of standards for plots, tracks and video are now routinely used for naval command and control, security and vessel traffic radar applications.

ASTERIX CAT-240

After its specification as a standard in 2009, ASTERIX CAT-240 has been adopted by a number of radar manufacturers as their preferred network video standard. For example, the SharpEye series of solid state radars from Kelvin Hughes – see Figure 1- have an option for CAT-240 output. This permits the radar to be used with standard display applications that accept this format, for example a Radar Visualisation Application shown in Figure 2.

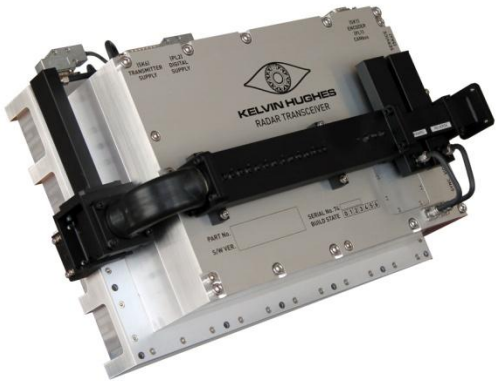

Figure 1 - The solid-state SharpEye series of radars from Kelvin Hughes combine exceptional performance with open-systems interfaces based on ASTERIX standards

In this situation a standard Windows-based software application can interface directly to the radar to receive the video with no analogue interfacing. The same application can be used to control the radar, meaning that a complete radar control and display application can be provided based on open standards and running on a low-cost laptop, or similar, platform.

The CAT-240 standard is a way of exchanging radar video between a source of data and one or more destinations. The data is provided in polar-format, which is a sequence of radar returns at changing angles, each comprising a set of samples representing the radar video at increasing range. For example, if the radar's range is digitised into 4,000 samples for each pulse and there are 1,000 pulses per second (prf = 1kHz) then the CAT-240 data stream represents a set of 1,000 messages per second, where each message encodes 4,000 samples of data. The receiving equipment is responsible for receiving this data and processing it for display, for example, with a scan converter that converts the polar-format video into a PPI (plan position indicator) image that can be shown with maps and overlays – see Figure 2 for an example.

COMPRESSED OR UNCOMPRESSED VIDEO

The CAT-240 standard defines each radar return as a combination of a header and a data block. The header provides information such as the angle represented by the block, the range of the radar video, time of day etc.

Within the data block of the CAT-240 message the video can be compressed or uncompressed. For the compressed option the ASTERIX standard does not specify the type of compression to be used. In principle this permits a proprietary compression scheme to be used to encode the data in the ASTERIX wrapper. In practice however, existing implementations of CAT-240 use either uncompressed data or else data compressed using a well-known compression scheme, such as ZLIB.

The obvious benefit in compressing video is to reduce the network bandwidth for distribution. Indeed, compression can offer a typical bandwidth reduction of 10 to 1 in the case of processed video. A minor consideration in the use of compressed video is that resources are needed for compression and decompression. On the decompression side, where there is typically significant computing resource available associated with the display application, there is generally no problem with the decompression process. On the compression side, the resources may be more limited, especially if the radar processing is implemented in an FPGA, for example.

Another, more significant, consideration with the use of compressed data is that whereas the bandwidth of the resulting video will generally be smaller than the uncompressed video, the bandwidth will be variable, with less compression where there is high clutter (sea, ground and weather, for example). The benefit of using uncompressed video is that the overall system performance, especially the network loadings, can be tested and can be safely relied on to remain constant. Any subsequent change in the input data, for example from additional clutter, will have no affect on the network load or data processing. Conversely, a

compression process will achieve an output data rate that is dependent on the nature of the input data, with higher levels of clutter and noise reducing the available compression. It means that a system that appears to work correctly in one condition could experience a system overload with consequent loss of data when the operating environment changes. Great care must be taken to test the system in a worse case condition.

PACKETISATION OF ASTERIX CAT-240

With or without compression, the size of a CAT-240 packet comprising header and data is likely to be several Kbytes long, perhaps 10s of Kbytes. Consideration must then be given to the delivery of such a large packet through standard IP protocols. The ASTERIX CAT-240 standard does not define the way that the packet is delivered, leaving this to the underlying transport protocol. In the common distribution method of UDP, a large packet size needs to be broken into smaller units each less that the packet size for the UDP protocol (called the maximum transmission unit or MTU). Although the transport process will, in theory, handle the delivery of large packets of data by automatically breaking them into smaller units and reassembling them on the other side, a problem exists in the IPv4 protocol which means the process would not work as expected in a practical implementation.

FRAGMENTATION OF LARGE UDP PACKETS

The ID field in the IP header is 16-bits, which means that ID values are re-used every 65536 packets. At high data rates required for radar distribution, this ID field will wrap around quite frequently. For example, for a 2KHz PRF, the ID field will wrap around roughly every 32 seconds. A problem can occur when one of the fragments is lost for some reason on the network. For example, consider that a big packet is split into 4 fragments, but only 3 arrive correctly. When this happens, the receiving operating system stores the 3 fragments in a "packet reassembly buffer", waiting for the 4th fragment to arrive. The 4th fragment is lost for some reason, so it never arrives, but approximately 32 seconds later the ID field wraps around and a new set of fragments is sent with the same ID field as before. The receiving system thinks that the first fragment of the new group matches the missing one from 32 seconds earlier (in the example above) and so it reassembles the 3 buffered fragments with this new one and makes a big packet from them. This has two problems. The reassembled packet is corrupt (because it has three fragments from one packet, and a fourth from a different packet). Also, it leaves three fragments in the packet reassembly buffer again, waiting for their 4th. After another 32 seconds, the ID field wraps around again and it all starts again. A single missing fragment can therefore cause a self-propagating effect, with three fragments added to the packet reassembly buffer every time, waiting for the 4th one which seems to appear 32 seconds later. Whenever any fragment is lost on the network, another of these self-propagating instances is started, so the packet reassembly buffer can grow and grow, never sorting itself out. The result is that a small packet loss, potentially in itself not noticeable, cascades into a major data loss and hence loss of radar data at the receiving equipment.

Fragments are discarded if they are not completed within the "packet reassembly timeout". On Windows, this is set at 60 and hence PRFs that cause the ID value to wraparound faster than that are susceptible to this problem. On Linux it is configurable, although a balance must be made between the requirements of a lower timeout value for radar distribution, and a higher value that might be necessary for other networking operations.

In summary, UDP fragmentation can be problematic at high data rates, where the ID field is re-used faster than the packet reassembly timeout of the operating system. Gradually more and more corrupt messages will be received and also the loading will increase because the reassembly buffer will be growing.

The solution to this problem is to make sure that fragmentation cannot happen with high data rates. If IP jumbo frames are supported in the networking equipment, the sender can send large packets because fragmentation should not occur. But, if jumbo frames are not supported, the sender must only send packets that are smaller than the MTU. Practically, this can be difficult with ASTERIX CAT-240 for a couple of reasons. Firstly, the specification isn't entirely clear on how large returns should be divided into smaller sub-packets, so it is important to know that the sender and receiver are compatible. It would be possible for a sender to take a view on packet division that is not compatible with the receiving equipment. Secondly, if compression is enabled then it really only makes sense in CAT-240 for each sub-packet to be compressed individually, rather than compressing the whole spoke and then chopping that up into sub-messages. Until it is known how much a spoke will compress, it is not known whether the compressed size will be less than the MTU or not. If it is more than the MTU, how many packets should it be divided into to guarantee that each sub-packet will be less than the MTU once compressed? Unless care is taken, it could be possible to perform lots

of compressions of the same data until the individual packets are small enough.

In summary, although the ASTERIX CAT-240 standard defines the core data structures for the distribution of radar video data, careful consideration needs to be given to a practical implementation. Cambridge Pixel's SPx software library implements a set of library modules that encapsulate the CAT-240 standard and handle the packetisation and compression. By using the same library software on the sending and receiving side, the data is correctly distributed. This approach has been used with commercially available radars, such as the Kelvin Hughes SharpEye.

GENERATING ASTERIX CAT-240 DATA FOR SIMULATION

Testing a server or client application with real-time data can be a challenge. Where test patterns are used in place of real radar signals, misleading performance results can be obtained, especially if compression is enabled. An idealised test pattern that compresses very well might give a very low network bandwidth, meaning that the system appears to behave correctly. When a real radar signal is connected with noise and clutter the compression will reduce and suddenly the data rates increase.

System testing with simulated radar video, which includes noise, clutter, terrain and targets, is highly desirable. Cambridge Pixel's SPx Simulator – see Figure 3 - is a software product that generates real-time video, tracks and navigation data for representative system testing. The software can be used to define complex movement scenarios of targets and moving radars, or alternatively it can synchronise to an existing simulator that defines target positions. The software generates ASTERIX CAT-240 video, which comprises targets, terrain and simulated noise, and supports either uncompressed or ZLIB compressed video. This allows representative data to be presented onto the network to test system performance. In addition to video, the software generates ASTERIX track data and optionally navigation data to report a changing radar position.

SUMMARY

The use of ASTERIX CAT-240 for the network distribution of radar video provides a standards-based solution that promises to improve interoperability of radar equipment. Where radar manufacturers are interested in promoting open standards interfacing, rather than developing closed proprietary solutions, the CAT-240 standard can be used effectively, provided various practical considerations relating to compression and packet fragmentation, are considered. The use of a toolkit of software modules that can generate and receive the CAT-240 standard, as provided by Cambridge Pixel, is a convenient and cost-effective way to include standards support into new products.

BIO DATA OF AUTHOR(S)

Dr David Johnson graduated from the University of Hull with a BSc and Phd in Electronic Engineering. As a software engineer and engineering manager, he has worked in industry developing image and radar processing solutions for over 20 years. Dr Johnson started Cambridge Pixel in 2007 to develop cost-effective software solutions for radar processing applications.

Richard Warren is a graduate of Oxford University's school of maths and computing, and has been the lead software engineer on a range of radar processing projects through 15 years in industry. Mr Warren currently leads Cambridge Pixel's software engineering team in the development of radar tracking and display solutions for Windows and Linux platforms.

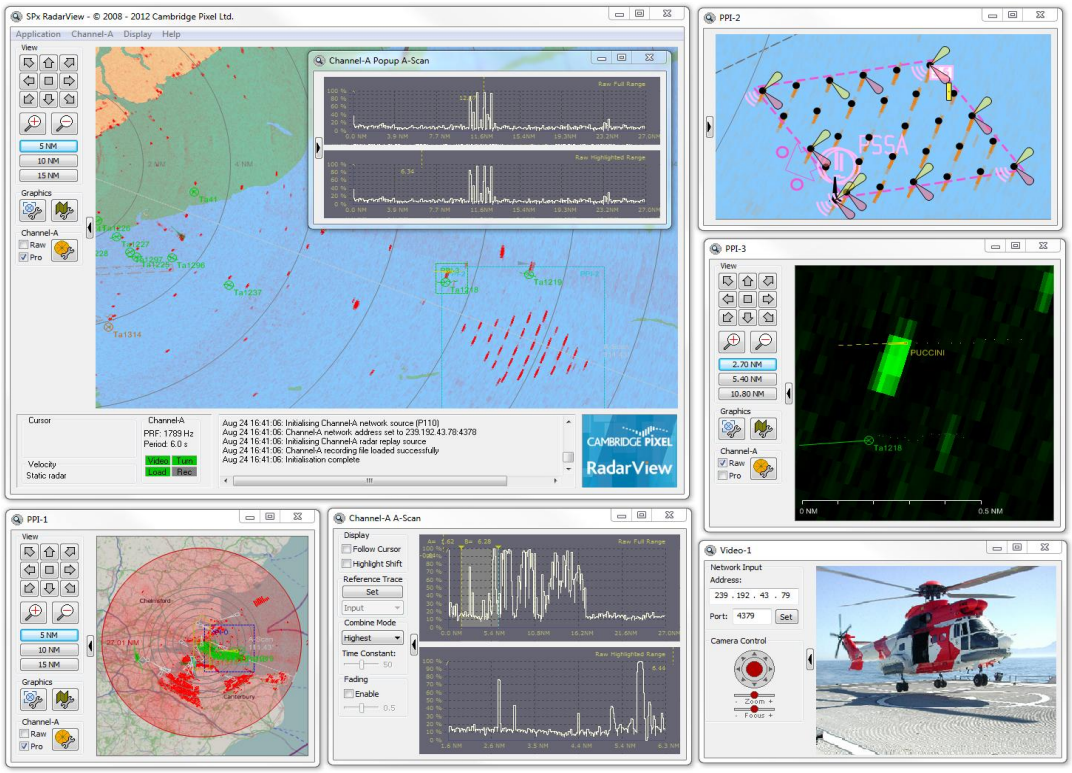

Figure 2 - Display of radar video received from a Kelvin Hughes SharpEye radar using ASTERIX CAT-240 network video

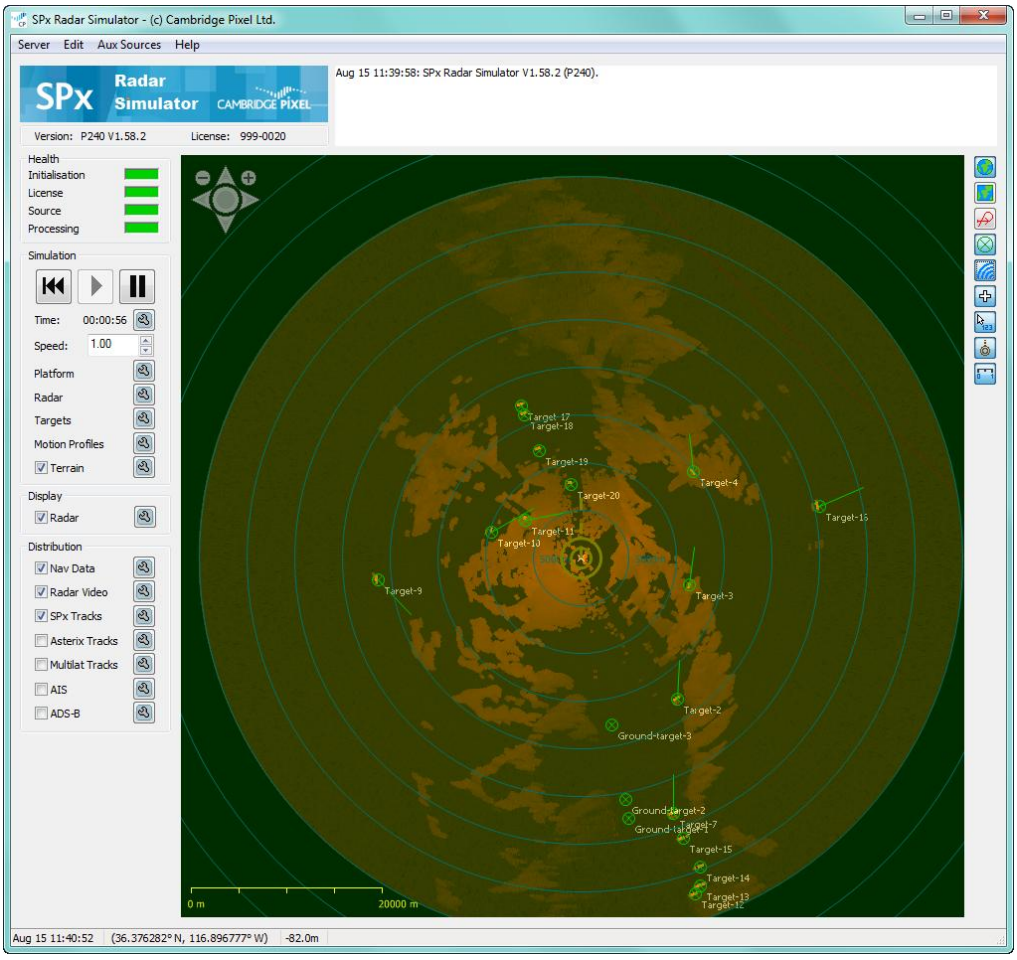

Figure 3 - Cambridge Pixel's SPx Simulator generates simulated ASTERIX CAT-240 video along with terrain, targets and representative system noise.