An UHD video handling system using a scalable server over an IP network

Hiroyuki Kimiyama*

Network Innovation Laboratories,
Nippon Telegraph and Telephone Corporation,
Yokosuka-shi 239-0847, Kanagawa, Japan
Email: kimiyama.hiroyuki@lab.ntt.co.jp
*Corresponding author

Mitsuru Maruyama

Department of Information Network and Communications,
Kanagawa Institute of Technology,
Atsugi-shi, Kanagawa 243-0292, Japan
Email: maruyama@nw.kanagawa-it.ac.jp

Masayuki Kobayashi

PFU Business Forerunner Limited,
Yokohama-shi, Kanagawa 220-0012, Japan
Email: kobayashi.masay.pfu@jp.fujitsu.com

Masao Sakai

NTT IT Corporation,
Yokohama-shi, Kanagawa 231-0032, Japan
Email: sakai.masao@ntt-it.co.jp

Satria Mandala

School of Computing,
Telkom University,
Bandung 40257, Indonesia
Email: satriamandala@telkomuniversity.ac.id

Abstract: Ultra-high definition (UHD) videos, such as 4K and 8K, have four times higher resolution than current HDTV videos. Broadcasting stations want to deliver these videos with as little loss as possible. However, uncompressed UHD video handling systems, such as storage and transmission systems, on IP networks are not available. We researched and developed a scalable high-speed video server system to store and deliver HD or 4K videos without compression by adapting a PC cluster technology. To handle videos with higher resolution than

4K, we also developed a method that synchronises with multiple the servers and 4K video transmission systems. We achieved 54 Gbit/s delivery performance with single cluster server comprising 24 PCs. We also implemented the synchronisation method on the server and QG-70 video transmission system. As a result, we were able to demonstrate the world's first bi-directional, real-time, uncompressed 8K-video transmission system and the 8K video-on-demand system.

Keywords: ultra-high-definition video; scalable high-speed video server; 8K-video transmission system; 8K-video on-demand system.

Reference to this paper should be made as follows: Kimiyama, H., Maruyama, M., Kobayashi, M., Sakai, M. and Mandala, S. (2017) 'An UHD video handling system using a scalable server over an IP network', *Int. J. Advanced Media and Communication*, Vol. 7, No. 1, pp.1–19.

Biographical notes: Hiroyuki Kimiyama is a Senior Research Engineer of Nippon Telegraph and Telephone (NTT) Network Innovation Laboratories. He received his BE and ME from Tohoku University, Japan, in 1988 and 1990 and his PhD from the University of Electro-Communications in 2010. He joined the NTT Corporation in 1990. Currently, he is studying distributed and parallel processing technologies for ultra-high-quality video handling system and storage server system over IP networks and SDN. He is a Member of the Association for Computing Machinery (ACM), the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan, and the Information Processing Society (IPSJ) of Japan.

Mitsuru Maruyama received his BE, ME, and PhD in Computer Science from the University of Electro-Communications in 1993, 1985, and 1999. He joined NTT Laboratories in 1985 and engaged in research and development of an IP-based real-time video transmission and archiving system. He was appointed a Professor of Kanagawa Institute of Technology in Atsugi, Japan, in 2012. He is currently studying fast-protocol processing system architectures. He is a Member of the IEEE Communications Society.

Masayuki Kobayashi joined PFU Limited in 2010. He is developing video transmission systems such as QG-70 in the video and broadcasting business department.

Masao Sakai joined NTT IT Corporation in 2006. He is developing video delivery systems, video servers and contents management systems in the video system department.

Satria Mandala received his PhD in Computer Science from Universiti Teknologi Malaysia. Currently, he is an Assistant Professor at School of Computing, Telkom University, Indonesia. In addition, he is also a Visiting Senior Researcher at UTM-IRDA Digital Media Centre, Universiti Teknologi Malaysia. His research interests include issues related to multimedia networking, wireless network security, information retrieval, and biomedical sensors.

This paper is a revised and expanded version of a paper entitled 'Uncompressed 8K-video system using high-speed video server system over IP network' presented at *The 2015 Asia Pacific Conference on Multimedia and Broadcasting (APMediaCast 2015)*, Bail, Indonesia, 25 April, 2015.

1 Introduction

Many expect ultra high definition television (UHDTV), which has four or more times higher resolution than existing HDTV, to be the next generation television standard since it provides a more immersive viewing experience. In fact, SMPTE and ITU-R have already standardised UHDTV as the next television standard in SMPTE ST 2036-1 (2014) and ITU-R BT.2020-2 (2015). The London Olympic Games in 2012 saw the BBC and the NHK, Japan Broadcasting Corporation, delivering 8K video with compression to eight separate locations through existing research networks as experiment described by Sugawara (2013). During the recent 2014 FIFA World Cup producers shot the games with 4K and 8K cameras and distributed them to multiple countries shown in Studio Daily (2014).

Currently, broadcasters can easily use video encoders or decoders to get 4K video and relay it through a network. They can also easily set up video servers to store it for archiving and delivery to other broadcasting stations as 'materials' for video programs. At the same time, 10 Gbit/s Ethernet equipment and commercial service are now taking over from their 100 Gbit/s Ethernet counterparts. This makes it possible to configure 4K video and 8K video handling systems without compression over high-speed networks if transmission systems are available. However, 8K video transmission, storing, and delivering systems are not currently available despite the availability of 8K video equipment such as display, camera, and editing systems. Therefore, systems that can handle 8K video via network will be required for future broadcasts.

Transmitting 4K video without compression requires network bandwidth of at least 12 Gbit/s. In addition, handling uncompressed 4K videos requires PC clusters comprising multiple PCs because of the internal bandwidth limitation of a single general-purpose PC. Accordingly, we implemented uncompressed 4K/HD video server system using a cluster configuration method we had previously proposed. We evaluated maximum delivery performance of the server system using 24 PCs and 12 storages. We confirmed the server system could deliver up to 54-Gbit/s video streams simultaneously. This maximum performance is equivalent to four uncompressed 4K 60fps video streams or 36 uncompressed HD video streams. We also evaluated maximum performance of the server system configured with a more recent PC server whose storage and CPU performances were respectively five times and twice those of older PC servers. In so doing we confirmed the video server could deliver 12-Gbit/s video streams.

We have also developed a new method that enables synchronising multiple 4K video servers with 4K video transmission systems. We developed it in order to achieve real-time delivery of 8K dual-green formatted video (8K-DG video) signals, for which the required bandwidth is 23.8-Gbit/s. We implemented this method in the aforementioned our video server system. We also implemented this method in an actual uncompressed 4K video transmission system which is commercially available. In order to configure an 8K-DG video transmission system without compression, we combined two 4K video servers and four 4K video transmission systems specifically. We used the system to successfully transmit 8K-DG video without compression in real time and to deliver it on-demand bi-directionally between 400-km apart two locations, Osaka and Tokyo, through commercial 100 Gbit/s line.

The rest of the paper is organised as follows. In Section 2, we describe the method we propose to configure and implement a scalable high-speed server system. We also present performance evaluation results obtained for a video server configured with the method. In Section 3, we describe the synchronisation method we propose to configure an

uncompressed 8K video transferring, storing, and delivery system using 4K video servers and transmission systems. Section 4 shows the configuration of our experiment system and the results we got with it for 8K-DG video on-demand delivery and transmission over a 100 Gbit/s commercial network. In this Section we also show the results we got on the buffer size needed to absorb jitter. We conclude the paper with a summary of key points in the last section.

2 Scalable video server system

Broadcasting stations have started shooting 4K video rather than HDTV video and compressing and transmitting it through existing dedicated networks, e.g., satellite or microwave links. On the other hand, they can transmit UHD video without compressing it through an IP network rather than a dedicated network because IP network bandwidth increases rapidly and 100 Gbit/s bandwidth is available. Therefore, we researched configuration method for video server with scalability to deliver and store uncompressed video through an IP network.

2.1 Scalable architecture

As mentioned in Section 1, transmitting an uncompressed 4K video requires about 12 Gbit/s bandwidth. Because the internal data transferring speed of a single PC is not sufficient for transmitting multiple 4K video without compression, we adapted a video server with PC cluster architecture. Basically there are three such stream server architectures described in Kimiyama et al. (2015), which we show in Figures 1–3 and will briefly explain here.

2.1.1 Architecture no. 1

The first architecture, shown in Figure 1, was described by Akutsu et al. (2015). All server PCs (nodes) have their own storages to store video data, which are divided into fragments during the storing process as the figure shows, and deliver them to client system connected to same IP network. The client sends delivering requests of video data to all nodes. The nodes receive the requests and send fragmented videos one by one to the client via the network. The client assembles the received fragmented video data and plays them back.

This is the simplest architecture. However, the client must be responsible for avoiding collisions or congestion caused by video packets that multiple nodes send to the network switch simultaneously. This is because the clients are the only devices that can control video traffic with requests. Therefore, each client should observe the status of the whole network and servers. In other words, each client should adjust the timing in sending requests with the status information to servers in order to avoid multiple clients sending requests to the same server at the same time. If clients cannot adjust the timing, the server system configured by this architecture should accept a limited number of clients to avoid congestion. Therefore, we should prepare a larger number of nodes to satisfy the system performance requirements.

2.1.2 Architecture no. 2

The second architecture (Figure 2), described in He et al. (2003), is different than one in which a storage area network (SAN) connects storages to nodes. Each node is responsible for gathering and assembling video data in the storages via the SAN. Therefore, the client

can request video data from only one node. As a result, clients do not need to check the whether the network is congested or not between themselves and the nodes. However, read requests are sent arbitrarily from the nodes, which receive requests from the clients, to all storages when the multiple nodes receive requests from multiple clients. Storages read the video data which are requested from the nodes according to their own algorithms of scheduling, e.g., 'first in first out'. As a result, congestion will occur in the SAN. Thus, the video data from the storages may be delayed and playback on the client may be interrupted. Therefore, a server system configured by this architecture should accept only a limited number of clients to avoid congestion. As a result, we should prepare more nodes to satisfy system performance requirements.

Figure 1 Architecture no. 1: scalable video server system using PC-cluster

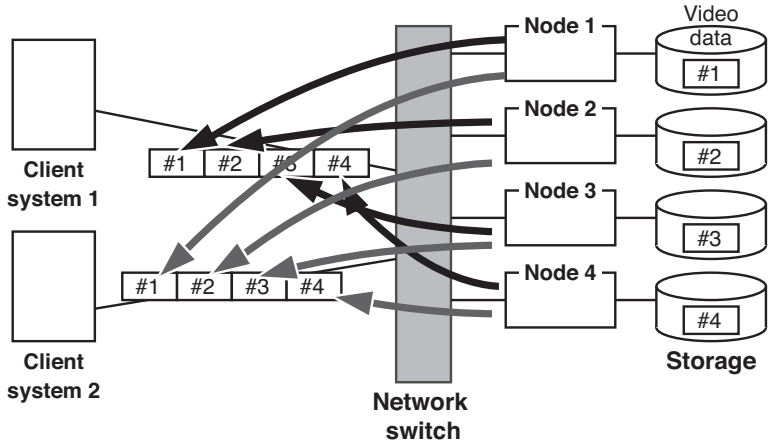
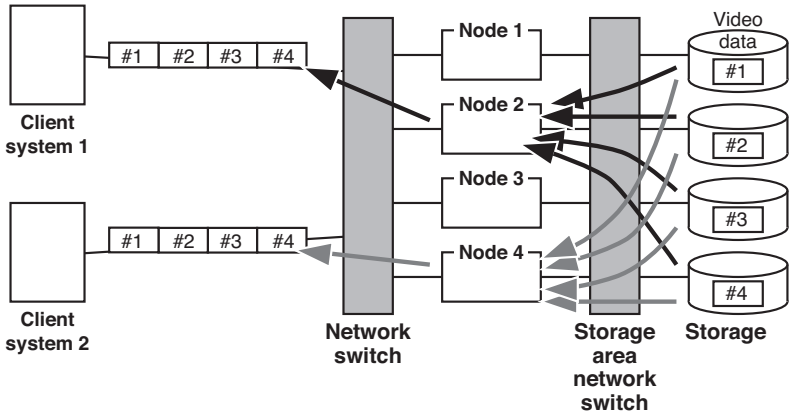


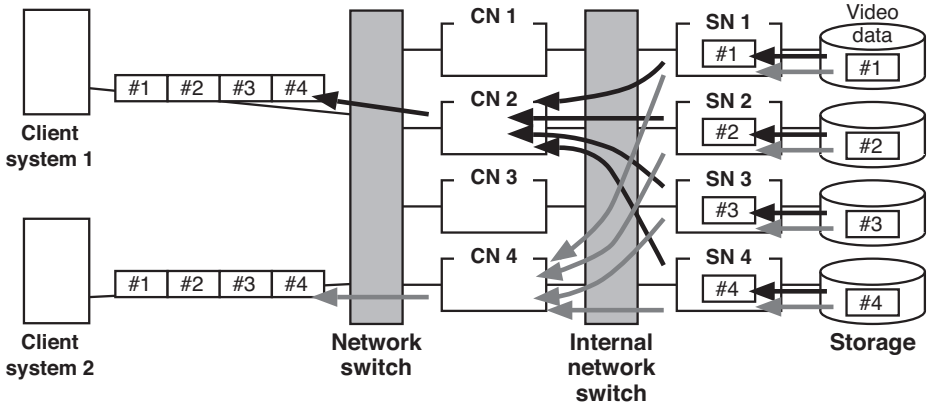
Figure 2 Architecture no. 2: scalable video server system using PC-cluster



2.1.3 Architecture no. 3

For the third architecture (Figure 3), proposed by Fukazawa et al. (1997) and also described in Islam et al. (2015), two server PC types are prepared: communication nodes (CNs) and storage nodes (SNs). The clients can send request for video data to CNs through IP network, the CNs relay the requests to all the SNs, and the SNs read video data from the storage which are connected separately. An internal network whose speed is generally much higher than that of Ethernet or a SAN, is introduced between the CNs and SNs. For example, a DDR InfiniBand has 20-Gbit/s bandwidth, which is twice as much as that for a 10 gigabit Ethernet used in IP networks. The client requests video data from one CN in the same way as with Architecture no. 2. The CN that receives the request is responsible for gathering and assembling video data from SNs via the internal network.

Figure 3 Architecture no. 3: scalable video server system using PC-cluster



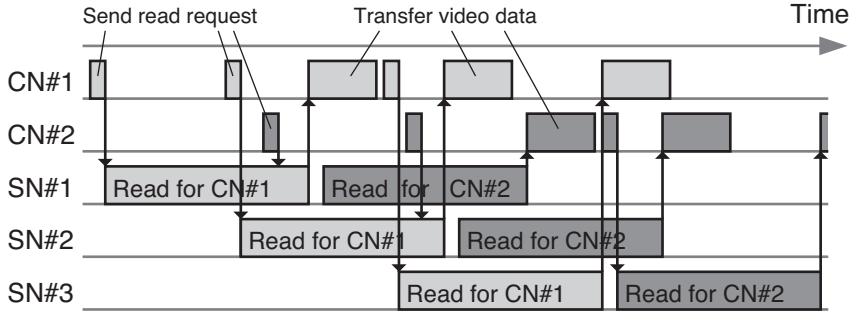
To configure the video server with this architecture we should prepare twice as many PCs as for the other two. However, we can control the order in which they read video data and transfer it to the CN on time by implementing an appropriate scheduling algorithm in the SN. We can also control the timing with which they send requests to prevent congestion caused by video data being sent from multiple SNs at the same time by implementing another scheduling algorithm in the CN. We show an example of a process time chart for CNs and SNs in which the above algorithms are implemented in Figure 4. As this figure shows, the scheduling algorithms prevent SNs from sending video data to the same CN at the same time in order to avoid congestion on the link between the CN and the internal network switch. Even if congestion were to occur on the internal network, video packets transferred from SN to CN would not be affected since the network has sufficiently higher bandwidth to transfer collided packets with very short delay. As a result, clients can receive video data from the CN and play back the video continuously.

Table 1 shows the results we got from comparing these architectures. Even though 'Architecture no. 3' requires twice the number of PCs as indicated in Kimiyama et al. (2015), we chose it to avoid collisions because the collisions interrupt real-time video transmission.

Table 1 Comparison between architecture nos. 1–3

	Architecture no. 1	*Architecture no. 2*	*Architecture no. 3*
Simplicity	◯	△	×
Resources	◯	△	×
Collision	×	×	◯

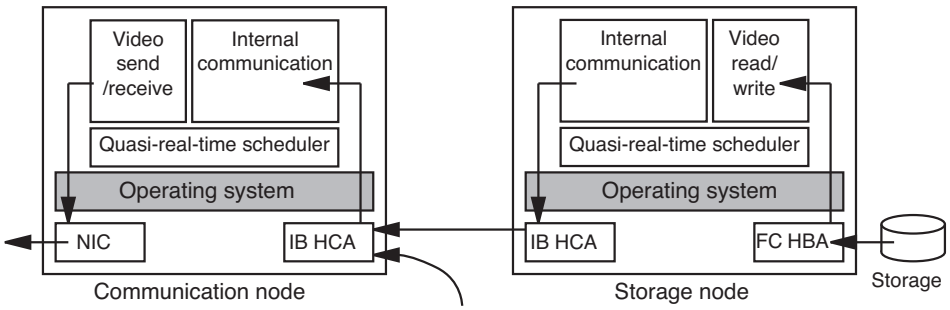
Figure 4 Example of process time chart of CNs and SNs for Architecture no. 3



2.2 Implementations

We designed software for the CNs and SNs in order to implement the aforementioned 'Architecture no. 3'; Figure 5 shows the software structure. We used the 'quasi real-time scheduler' module proposed in Kimiyama et al. (2004) in both types of nodes. This module can control 'video read/write', 'video send/receive', and 'internal communication' with real-time scheduling in the application level.

Figure 5 Software structures of 'Communication node' and 'Storage node'



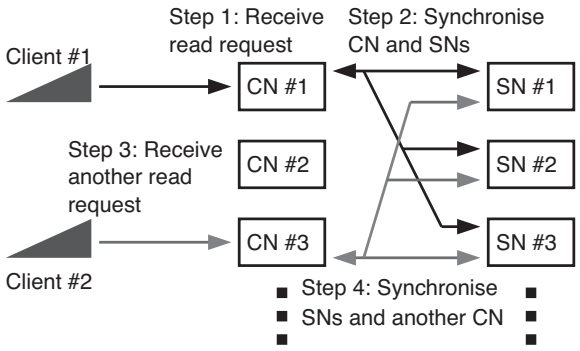
IB HCA: InfiniBand Host Channel Adapter
NIC: Network Interface Card
FC HBA: Fibre Channel Host Bus Adapter

The 'video send/receive' module in the CN generates multiple instances and each instance sends video streams to each client with a specified transfer rate. The 'video read/write'

module in the SN reads or writes video data from storage while keeping the order of requests in order to maintain QoS. The 'internal communication' module is incorporated in both CNs and SNs to transfer synchronised video data between them. This is the reason the internal clock of the PC server is not so accurate and each PC clock may be different from the others.

One of the CNs, the one that receives the first request from a client, synchronises with the SNs via an internal network with this 'internal communication' module. If another CN receives the next request, this CN synchronises with the SNs via the internal network in the same way. Finally, all CNs and SNs synchronise with each other as shown in Figure 6. In order to achieve synchronised transferring between CN and SN, we implemented the general flow control method described in Ogura et al. (2006) into this module.

Figure 6 Synchronisation method between CNs and SNs for Architecture no. 3



2.3 Scalability evaluation

2.3.1 Scalability for number of nodes

In order to evaluate the server's scalability by measuring maximum throughput, from 4 to 24 general PC servers of which specification are shown in Table 2 were used. Our experimental system for evaluation is shown in Figure 7. We prepared an external storage for each SN because we considered a large storage area was needed for storing the amount of uncompressed 4K video data, since 1 h of 4K video data requires 5.4 TByte storage capacity. Table 2 also shows the storage specifications we used. We used two 4-Gbit/s FibreChannel interfaces to connect the storage to the SNs.

Figure 8 plots the measured maximum delivering capability of the whole server system while the number of CNs and SNs vary from 2 to 12 each. The vertical axis shows the maximum delivering capability of the whole server system and the horizontal axis shows the number of SNs and CNs. As the figure shows, the maximum delivering capability increases to the number of CNs and SNs in proportion. We achieved the PC-cluster based scalable server system with 53.5-Gbit/s maximum delivering capability when we used both 12 CNs and 12 SNs.

An 11.88 Gbit/s delivering capability is required for one 4K video (59.94 frames/s) without compression in real time and a 1.485-Gbit/s capability is required for one HD video (29.97 frames/s). We confirmed our server system could simultaneously deliver four 4K video or 36 HD video without compression in real time.

Figure 7 Configuration of experimental system for performance scalability evaluation

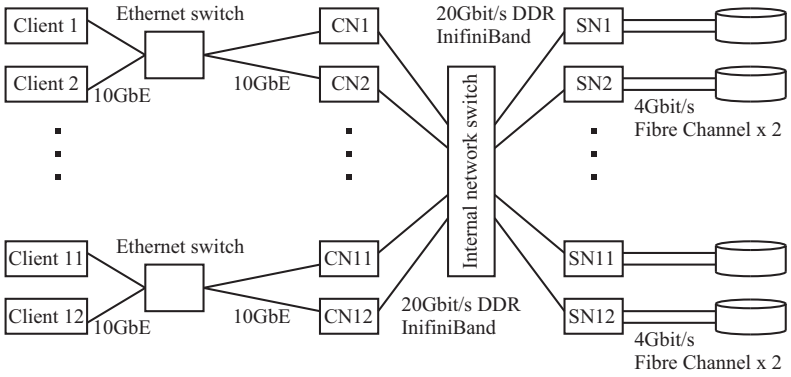


Figure 8 Relationship between measured maximum delivering capability and number of CNs and SNs

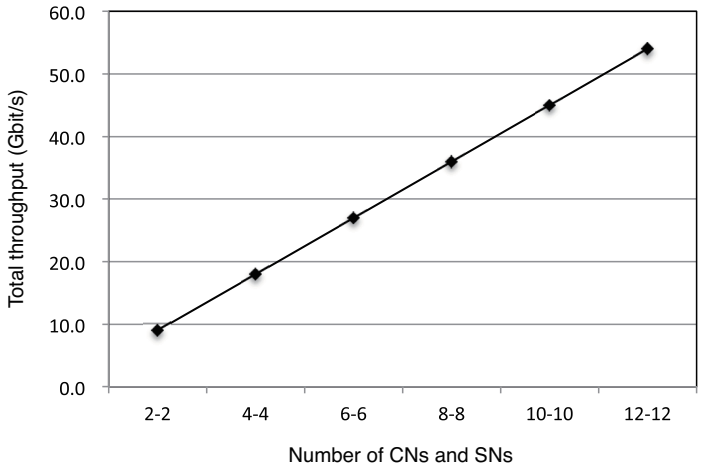


Table 2 Hardware and software specifications of CNs and SNs

Hardware	
CPU, chipset	Intel Quad Core Xeon 3.33 GHz (X5470) x 2, Intel 5000X
Memory	4 GByte
HCA	QLogic 7104-HCA-128LPX-DD
Storage (for SN)	InforTrend S16F-R1430
HBA (for SN)	QLogic QLE2462
NIC (for CN)	NTT-IT PRESTA 10G NIC
Software	
OS	Red Hat Enterprise Linux 4

2.3.2 Scalability for node performance

The ‘video send/receive’ module creates multiple instances for each client to communicate with each client as described in Section 2.2. Using a higher-performance PC would enable

us to configure a higher delivering capability server since it would make it possible to increase the number of instances. We measured the maximum delivering capability of a server configured from a very recent PC whose specifications are shown in Table 3. The PC had an internal RAID storage configured with 16 SSDs and an Adaptec RAID card connected to a CPU directly. We configured that the CPU could communicate with the RAID card through a PCI Express 2.0 x8 bus (32 Gbit/s maximum transfer capability) and the RAID card could communicate all SSDs directly through 6.0-Gbit/s Serial ATA buses. From the bus speed we estimated that the PC's storage performance was about five times that of older PCs. From a CPU benchmark site; Passmark Software (https://www.cpubenchmark.net/high_end_cpus.html), we also estimated it had twice the CPU performance of older PCs.

Table 3 Specifications of newer PC for 12-Gbit/s video server system

Hardware	
CPU, chipset	Intel Xeon E6-2620 (6 cores) x2, Intel C606
Memory	32 GByte
HBA	Adaptec RAID 71605
SSD	Intel 520 x16
NIC	Intel X520-DA2
Software	
OS	Cent OS 6

The CN and SN software programs are installed in this higher performance PC to evaluate scalability of our architecture for PC performance. We used internal loopback interface as internal communication network in order that these softwares communicated each other. We observed maximum throughput of the higher performance server was 11.88-Gbit/s during this evaluation. Accordingly the older PC's delivery performance is estimated at 2.23-Gbit/s per PC, the newer PC's performance is five times faster than the older one. Our proposed architecture makes it possible to configure scalable video server with both number of PCs and performance of PCs.

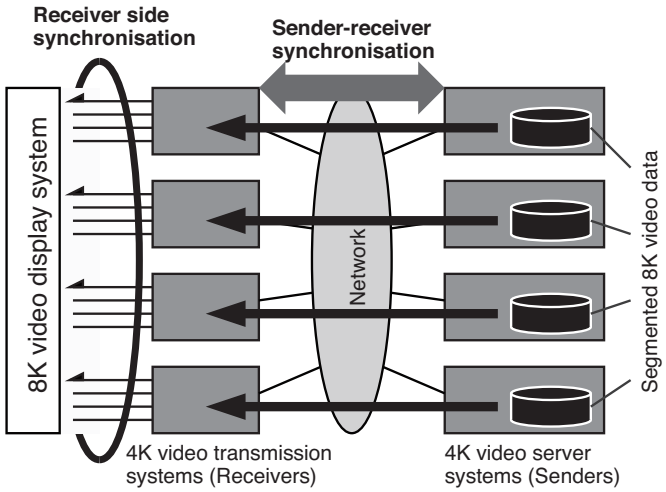
3 Uncompressed 8K video system

Many expect '8K' to be the video format of the future as it provides the highest resolution video in the world today. This means if we could develop a system that can handle 8K video without compressing it, it would be able to handle the resolution of any known video. Therefore, we designed 'dual green formatted 8K (8K-DG)' IP video server system without compression. The NHK Science and Technology Research Laboratories originally proposed this format, and cameras and monitors that can handle it have already been commercialised. They input or output geometrically divided 8K-DG video (59.94 frames/s) through 16 HD-SDI interfaces. We felt we should be able to handle 8K-DG video signals as synchronised 16 HDTVs or four 29.97-frames/s 4K (4K@29.97p) video on our video server system. Handling the signals in this manner would require a method enabling the server system to deliver and store 16 HD-SDI signals with synchronisation at the same time. However, since available devices could only handle up to four such signals simultaneously, we developed

a method with which our server system could deliver and store 16 HD-SDI signals at the same time.

Current 8K projectors or 8K LCDs require a strict synchronisation mechanism because their buffers are too small to absorb any differences in video signal phases being input through 16 HD-SDI interfaces. Furthermore, we should develop a method that would enable the server to tune the sending speed of 8K-video data to playback speed on display to avert underflow or overflow of video data in the buffer memory of the receiver device. Figure 9 shows the configuration of an 8K video server system we developed using 4K video server systems and transmission systems that can handle 4K video in real time. The figure also shows the synchronisation mechanism we propose for the system, which comprises two methods. The first method is 'receiver side synchronisation', which enables the receivers (i.e., the 4K video transmission systems) to synchronise with each other. The second is 'sender-receiver synchronisation', which enables the receivers to synchronise the senders (i.e., the 4K video server systems). We will describe the details of these mechanisms in the following subsections.

Figure 9 Configuration and synchronisation mechanisms of 8K video server system using 4K video servers



3.1 Synchronisation method

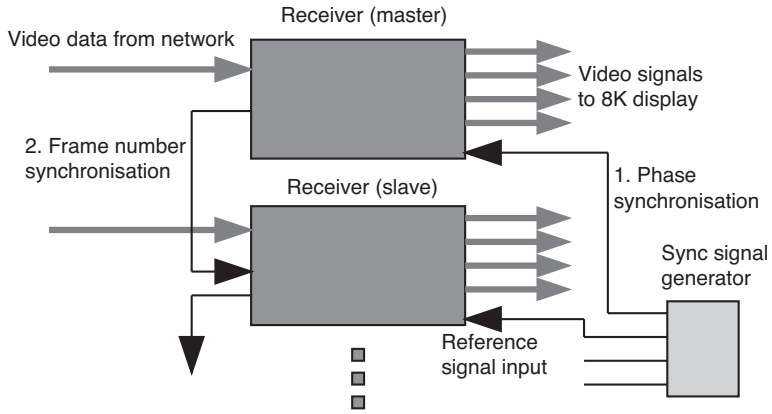
3.1.1 Receiver side synchronisation

We can explain method of synchronisation for 8K video system by separating it into two separate methods: 'frame number synchronisation' and 'phase synchronisation'. We used the first method to roughly synchronise the receivers and then the second to strictly synchronise them. Figure 10 shows the outline of both methods.

Our system's 8K video display requires all video input signals to synchronise with each other with the same frequencies and the same phases. 'Phase synchronisation' enables the receivers to adjust the phase and frequency of all input signals. In order to achieve this, we prepared a reference signal generator that inserts reference signals into the receivers.

Since most video receiver devices (e.g., video decoders) have a reference signal input to synchronise other video equipment (e.g., a video monitor or switcher) on the receiver side, we can implement this method easily without developing additional functions.

Figure 10 Outline of 'phase synchronisation' and 'frame number synchronisation' methods



However, we needed to guarantee that the same video frames would be output from all receivers to the 8K display at the same time. Since the reference signals do not include an information related frame number, the 'frame number synchronisation' mechanism was required to develop, which enables receivers to output the same video frames to the 8K display at the same time. In other words, we needed to develop a mechanism which the time code or the frame number could be shared among all receivers. For this the following two methods were considered.

- *Method 1*: A master device is elected from the receiver devices and broadcasts information related with the frame number to other receiver devices through the network.
- *Method 2*: A master device is elected from the receiver devices and informs a neighbouring receiver device about information related with the frame number. The receiver device that receives the information then relays it to the next neighbouring receiver.

The first method can be implemented more easily than the second one. With it, however, some receivers may not receive broadcast packets since broadcast packets are unreliable. This meant that we needed to use another method to deliver broadcast packets reliably. Therefore, we decided to implement the second method to share frame number information.

With it, however, video frame number information is not always included in video data. Accordingly, we also stored the frame number information with video data into the video server storage and inserted this information into video packets which were sent to receivers.

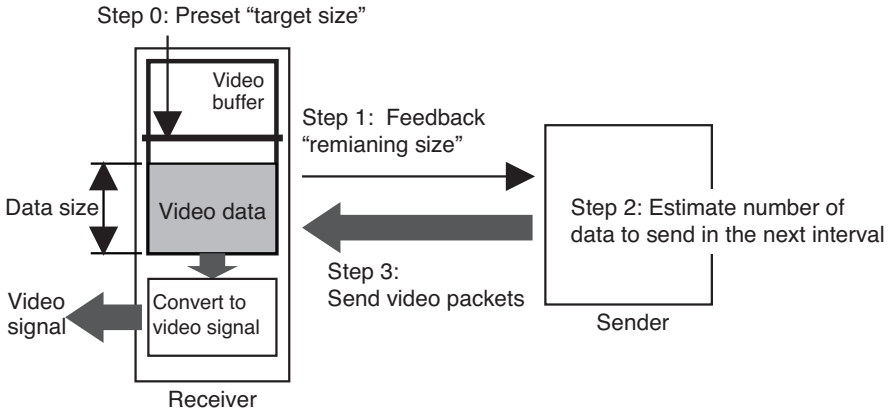
3.1.2 Sender-receiver synchronisation

All receivers needed to receive video data from the servers at the same speed as the output signal speed to the 8K video display. Although the servers, which are configured with a

general PC, can estimate output signal speed, this estimated speed is not accurate since PC quartz crystals are less accurate than those of video equipment. This means if a sender sends video data to the receiver with self-estimated speed, underflow or overflow occurs in the receiver. Accordingly, we developed a synchronisation method using feedback from the receiver (Figure 11). This method consists of four steps:

- *Step 0*: The 'target size' is determined from the amount of jitter in the network and pre-configured in the sender devices. After pre-configuring, the sender devices start to send video data with self-estimated speed.
- *Step 1*: The receiver devices return 'remaining size' as buffer status information at fixed intervals. The 'remaining size' includes size of data remaining in their buffers.
- *Step 2*: The sender devices estimate how many bytes of video data to send in the next interval from the returned 'remaining size' so that the senders can reduce the difference between the 'remaining size' and the 'target size' to zero. The senders record the 'remaining size' history as time series data and use the data to take network jitter into account.
- *Step 3*: The sender devices send the estimated amount of video data.
- *Step 4*: The senders and receivers repeat the procedures from Step 1 to Step 3.

Figure 11 'Sender-receiver synchronisation' method



4 Experiment

4.1 Implementation

We newly designed the protocol as a 'sender-receiver synchronisation' protocol to implement synchronisation method proposed in the previous section. We also added a function to exchange the capabilities and the status between the senders and the receivers to

prevent senders from sending high-speed video to the wrong receivers. It was necessary to do this because uncompressed 4K video would have too big impact on the network if the sender were to send to the wrong equipment or to the equipment which was not ready to receive video data. We implemented this protocol in our video server system (XMS) described in Section 2 and also in 4K video transmission system: QG-70 which is manufactured and provided by PFU Ltd.

Our XMS can convert stored video data in internal or external storage to multiple video streams and send them to multiple QG-70s in real time. It can also store video streams received from QG-70s into the storage as video files in real time. The QG-70 is capable of restoring the four original HD-SDI signals simultaneously with very short processing time, either from IP packets received from another QG-70 or from the XMS through a 10 Gbit/s Ethernet interface. It can also send video IP packets converted from up to four HD-SDI signals simultaneously to other QG-70s or XMSs.

We added two RS-232c interfaces to exchange information related frame number between two neighbour QG-70s. The QG-70's reference signal input was used as a function for phase synchronisation. The sender-receiver synchronisation protocol (described in the previous section) was implemented in the QG-70 and XMS.

4.2 Experimental configuration

In order to demonstrate that the proposed synchronisation method was valid, we configured two sets of experiment systems (Figure 12). The first was an '8K video transmission in real time' system configuring eight QG-70s, one 8K LCD, and one 8K camera system. The second was an '8K on-demand video' system configuring four QG-70s, two XMSs, and, one 8K LCD monitor. Both were used to transmit 8K live video from Tokyo to Osaka in real time. Both can share the same QG-70s because both are able to send and receive multiple video streams at the same time. We installed CN and SN software into a single PC server to configure the XMSs. The specifications of the PC was shown in Section 2. We configured the CN and SN software so that they could communicate with each other via a loopback interface without using an external network. We configured each XMS so as to store and deliver half of the uncompressed 8K video because we can configure the server with short response time.

For the 8K real-time transmission experiment, we placed the QG-70s and the 8K LCD in Osaka as the receiver equipment and the QG-70s and the 8K camera system in Tokyo as the sender equipment. We also placed two XMSs in Osaka for an 8K video-on-demand experiment. Table 4 lists the 8K video equipment used in these experiments. We configured a 100-Gbit/s network between 400-km apart two locations, Osaka and Tokyo with a commercial 100-Gbit/s Ethernet service and dark fibres. Two extra QG-70s were placed in both Tokyo and Osaka for ultra-high quality and low-latency video conferencing using uncompressed 4K@60p video to monitor another location. As a result, we configured the experimental systems to do three experiments: 8K video transmission in real time, 8K on-demand video delivery, and bi-directional 4K transmission in real time.

4.3 Experimental results

We carried out the three experiments on 5 and 7 February, 2014. From the results we confirmed that the 8K video, which was received from both the XMSs and the QG-70s through the 100-Gbit/s network, played back very smoothly. These results show

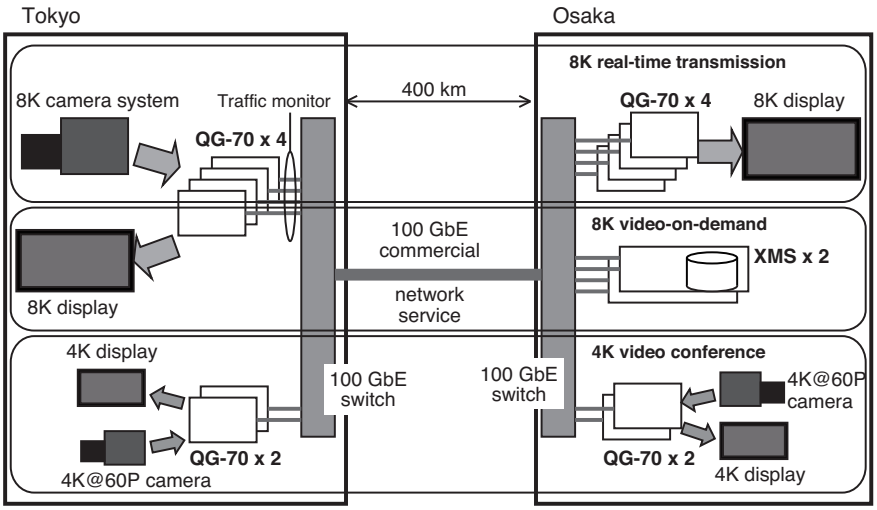
the proposed synchronisation method works correctly. They also confirmed that no packet loss and no noises or defects on the 8K monitors occurred during the experiments on either side.

Table 4 List of 8K video equipment for the experiment

Equipment	*Model*	*Vender*
8K monitor in Osaka	Prototype	Sharp Corp.
8K monitor in Tokyo	Prototype	KAIT
8K camera system	AH-4800 + AH-4802	Astrodesign

We set up a traffic monitor between the 100GbE network switch and the QG-70s in Tokyo (Figure 12) so that we could measure throughput and jitters of the video traffic. Figures 13 and 14 show time variability of measured throughput from QG-70s and XMSs respectively. We confirmed both throughputs were stable and the average value was 25.6 Gbit/s (comprising 1.8 Gbit/s for overhead of video packets and 23.8 Gbit/s for 8K-DG video).

Figure 12 Experimental system for 8K on-demand video and 8K video transmission system on 5 and 7 February, 2014



We also confirmed that the variation of the 8K server system throughput was larger than that of the 8K transmission system throughput. This is because most of the QG-70 functions, including the one for adjusting the throughput, were implemented with hardware. On the other hand, most of the XMS functions were implemented with software. Hardware can execute video sending processes with a shorter period than software in general. Therefore, the measured XMS throughput had more jitter than the measured QG-70 throughput.

Figure 13 Measured throughput from QG-70s

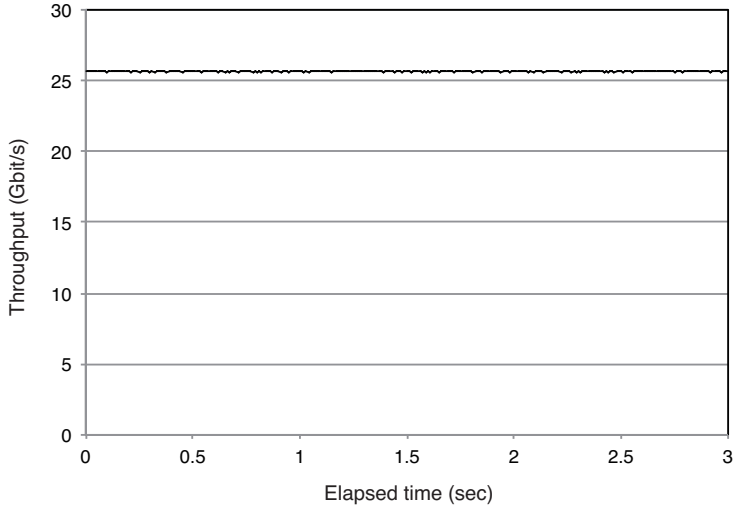
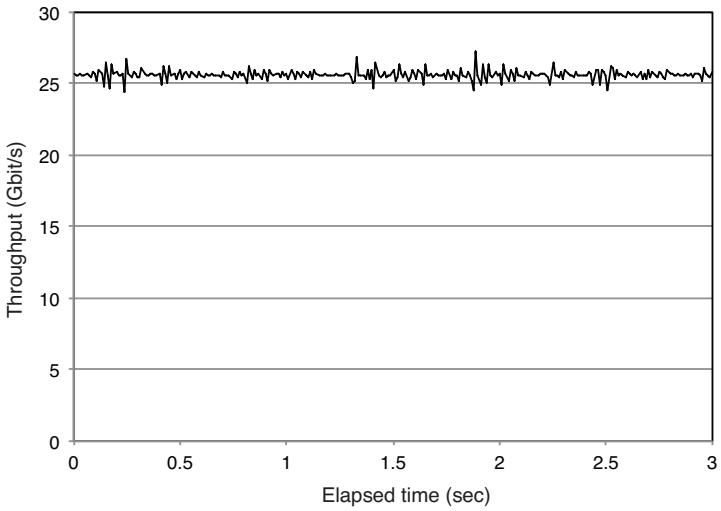


Figure 14 Measured throughput from XMSs



We evaluated the size of the video buffer memory in a receiver to absorb variance of video sending rate. The required memory size for a receiver affects how much memory is implemented in it. In other words, the size affects the receiver system cost. The number of video data bytes remaining in the receiver buffer memory is estimated by subtracting the number sent to the monitor from the number actually received from a sender. Therefore, we can get $B(T)$, the video data bytes remaining in the buffer memory at time T, by equation (1).

$$B(T) = \int_0^T v_{\mathrm{recv}}(t)\mathrm{d}t - v_0 \times T, \tag{1}$$

where $v_{\mathrm{recv}}(t)$ is actual video receiving rate at time t and v_0 is the video rate for playback on monitor. If $B(T)$ is positive for any T, this indicates overflow occurs in the receiver and $B(T)$ bytes remained in the buffer memory. If $B(T)$ is negative for any T, this indicates that underflow occurs in the receiver at time T and 'initial buffering' is required to avoid underflow in the client. The 'initial buffering' means that the receiver must store video data before sending it to the 8K video monitor until the 'initial buffer' is filled up with video data. Therefore, we can estimate required buffer size B_{total} by equation (2)

$$B_{\mathrm{total}} = \max(B(T)) - \min(B(T)). \tag{2}$$

To estimate the buffer size from the above results, we replaced integrals with sums because we observe the video rate at discrete time periodically. We used equation (3) to estimate $B(T)$ instead of equation (1).

$$B(T) = \sum_{i=1}^{N} v_{\mathrm{recv}}(t_i)\Delta t - v_0 \times T, \tag{3}$$

where Δt is the monitoring interval, T is $N \times \Delta t$, and t_i is $i \times \Delta t$.

We calculated B_{total} for both 'real-time transmission' and 'on-demand' system and show these results in Table 5. We verified that the small (5 Mbytes minimum) video buffer memory in the receiver could be used to absorb the video transfer rate variability.

Table 5 Estimated video buffer size (KBytes) for receiver device of video transmission and on-demand system

	Real-time transmission	*Video on-demand*
$\min(B(T))$	–62	–2369
$\max(B(T))$	58	2607
B_{total}	120	4977

To develop the world's first 8K video real-time handling system over 100-Gbit/s Ethernet without compression, we implemented the proposed synchronisation method in our scalable video server system (XMS) and QG-70 uncompressed 4K-video transmission system. We proved that the on-demand system could be configured with two XMS systems and four QG-70s and real-time transmission system also could be configured with eight QG-70s. We confirmed that measured 8K-DG video throughput from both XMSs and QG-70s was stable and that the average throughput was 23.8-Gbit/s without protocol overhead. This demonstrated that using our proposed synchronisation method could be configured an 8K system easily.

5 Conclusion

We proposed a way to configure and implement a high-speed and scalable video server system using PC-cluster. Actual PC clusters were used to evaluate the total delivery capability of a server system configured with the proposed method. We confirmed the maximum throughput of the server system was 53.5-Gbit/s when using 24 PCs and also

confirmed the system capability could be scaled up by increasing the number of PCs. Another video server system was also configured using the same software and one recent PC having five times the storage capability of an older PC. We confirmed this new video server could deliver video data at 12 Gbit/s. This corresponded to a fivefold increase in delivery capability per PC over older server systems. These results confirmed our proposed architecture was scalable in terms of both the number of PCs and PC performance.

To develop an 8K-DG video handling system without compression using existing 4K-video transmission systems and 4K video server, we proposed a new synchronisation method that enabled all receivers to output synchronised multiple 4K-video signals as 8K-DG video signals. We adopted two synchronisation procedures ('sender-receiver synchronisation' and 'receiver side synchronisation') and developed methods to implement them. To ascertain whether our proposed methods were valid, we implemented them in QG-70 4K-video transmission system and our proposed XMS server system. We placed QG-70s and XMSs in Tokyo and Osaka and connected the two cities (400-km apart) via a 100-Gbit/s Ethernet. We configured an 8K video transmission system in real time without compression using QG-70s implementing our synchronisation methods; we also configured an 8K on-demand video delivery system using XMSs and QG-70s.

Experiment results confirmed that by using our proposed methods we had successfully developed the world's first bi-directional, real-time, 8K-video transmission system and the first-ever 8K video-on-demand system without compression. We verified that the proposed synchronisation method provided smooth video transmission from an 8K video system by preparing a 5-Mbyte video buffer memory in the receiver. We also verified we can configure UHD (4K and 8K) video systems using the proposed server system.

Acknowledgements

The authors received generous cooperation during this verification test from the National Institute of Information and Communications Technology, the Nara Institute of Science and Technology, ASTRODESIGN, Inc., Sharp Corporation, Hokkaido Television Broadcasting Co., Ltd., and NTT Communications Corporation.

A portion of this verification test was supported by JSPS Research Grants no. 24800069 and no. 26330121.

References

Akutsu, H., Ueda, K. and Chiba, T. (2015) 'Reliability analysis of highly redundant distributed storage systems with dynamic refuging', *Proceedings of 2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, IEEE, 4–6 March, Turku, Finland, pp.261–268.

Amrane, A., Mellah, H., Aliradi, R. and Amghar, Y. (2014) 'Semantic indexing of multimedia content using textual and visual information', *Int. J. Advanced Media and Communication*, Inderscience Publishers, Vol. 5, No. 2–3, pp.182–194.

Fukazawa, K., Suzuki, H. and Sasaki, C. (1997) 'Distributed video server using a new striping technique', *Proceedings of SPIE 3228, Multimedia Networks: Security, Displays, Terminals, and Gateways, 147*, SPIE, 2–5 November, Dallas TX, USA, pp.147–157.

He, X., Beedanagari, P. and Zhou, D. (2003) 'Performance evaluation of distributed iSCSI RAID', *SNAPI '03 Proceedings of the International Workshop on Storage Network Architecture and Parallel IOs*, ACM, 27 September–1 October, New Orleans, MO, USA, pp.11–18.

Islam, N.S., Lu, X., Wasi-ur-Rahman, M., Shankar, D. and Panda, D.K. (2015) 'Triple-H: a hybrid approach to accelerate HDFS on HPC clusters with heterogeneous storage architecture', *Proceedings of 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, IEEE, 4–7 May, Shenzhen, China, pp.101–110.

International Telecommunication Union Radiocommunication Sector (2015) *Parameter Values for Ultra-High Definition Television Systems for Production and International Programme Exchange*, Recommendation ITU-R BT.2020-2.

Kimiyama, H., Shimizu, K., Kawano, T., Ogura, T. and Maruyama, M. (2004) 'Real-time processing method for ultra-high-speed streaming server based on PC Linux', *Proceedings of The 18th International Conference on Advanced Information Networking and Applications (AINA-2004)*, IEEE Computer Society, 29–31 March, Fukuoka, Japan, Vol. 2, pp.441–446.

Kimiyama, H., Maruyama, M., Kobayashi, M. and Sakai, M. (2015) 'Uncompressed 8K-video system using high-speed video server system over IP network', *Proceedings of 2015 Asia Pacific Conference on Multimedia and Broadcasting (APMediaCast 2015)*, IEEE, 23–25 April, Bali, Indonesia, pp.99–105.

Lee, J. P., Kim, S.H. and Park, Y.W. (2014) 'Temporal pattern recognition based interactive video-on-demand streaming technique', *Int. J. Advanced Media and Communication*, Inderscience Publishers, Vol. 5, Nos. 2–3, pp.140–164.

Ogura, T., Kimiyama, H., Kugimoto, T., Kawano, T., Shimizu, K. and Maruyama, M. (2006) 'Method of achieving an EDL-based video editing function on a PC-cluster distributed-RAID video stream server', *Proceedings of International Workshop on Advanced Image Technology (IWAIT 2006)*, The Institute of Electronics, Information and Communication Engineers (IEICE), 9–10 January, Naha, Japan, pp.139–143.

Shishikui, Y., Iguchi, K., Sakaida, S., Kazui, K. and Nakagawa, A. (2013), 'High-performance video codec for Super Hi-Vision', *Proceedings of the IEEE*, IEEE, Vol. 101, No. 1, pp.130–139.

Society of Motion Picture and Television Engineers, 2014 *Ultra High Definition Television - Image Parameter Values for Program Production*, SMPTE ST 2036-1.

Studio Daily (2014) *2014 FIFA World Cup Games Will Be Shot in 4K, but Not 3D – Studio Daily*, http://www.studiodaily.com/2014/06/2014-fifa-world-cup-games-will-be-shot-in-4k-but-not-3d/

Sugawara, M. (2013) 'Advanced in SUPER Hi-VISION in 2012 – public viewing at the London Olympics, and Standardization at the ITU-R', *New Breeze*, ITU Association of Japan, Vol. 25, No. 1, pp.4–9.

Website

Passmark Software, *PassMark Intel vs AMD CPU Benchmarks – High End*, https://www.cpubenchmark.net/high_end_cpus.html