SCALABLE TRANSMISSION AND DECODING OF SPACE PACKETS FOR REMOTE SATELLITE IMAGE BROWSING

Han Oh[1], Jae Young Chang[1], Jin-Hyung Kim[2] and Hae-Jin Choi[1]
[1]Satellite Operation and Application Center, [2]Future and Converging Technology Research Division
Korea Aerospace Research Institute
Daejeon, Republic of Korea, 34133
ohhan@kari.re.kr

ABSTRACT

The Consultative Committee for Space Data System (CCSDS) defines a standard for the data compression algorithm applied to the image data from payload instruments. In the CCSDS image data compression (CCSDS-IDC) standard, an image is encoded using a three-level two-dimensional Discrete Wavelet Transform (DWT) and the Bit-Plane Encoder (BPE). Compared to the JPEG2000 standard, the compression performance is reported to be similar at high bit rates with much lower complexity. However, its lack of highly scalable features supported by JPEG2000 hinders smooth browsing of the high-resolution satellite images. In this work, a method of quickly accessing a region of interest in a high-resolution satellite image is introduced. It utilizes parallel processing and the structure of the space packets which contain the strip-based codestream. This method is particularly effective for remote satellite image browsing and the study demonstrates its performance using KOMPSAT (Korea Multi-Purpose Satellite)-3A images.

INTRODUCTION

As satellite technologies have become more sophisticated, the resolution of satellite images has increased significantly. Those massive satellite images are typically compressed on-board for efficient use of memory space and transmission channel bandwidth. The CCSDS-IDC standard [1] decomposes the image into several frequency bands and orientations using a three-level two-dimensional DWT, and the resulting wavelet coefficients are efficiently encoded by the BPE. Compared to the JPEG2000 standard [2] which exploits a similar DWT, the functionalities supported by the CCSDS-IDC standard are limited, but the computational complexity is appropriate for implementation on a resource-constrained on-board processing system, and the compression performance is reported to be similar at high bit rates [3, 4, 5]. Because of this low computation complexity and high compression performance, various satellites including KOMPSAT-3 and 3A employ the CCSDS-IDC standard.

Recent high-resolution satellite images are much larger than the maximum resolution of a display device. These images are typically viewed interactively through the user's zooming and panning requests. However, the CCSDS-IDC standard is not a straightforward way to view the

high-resolution satellite images quickly, because it lacks the scalable decoding feature supported by JPEG2000. This paper introduces a method of implementing a scalable CCSDS-IDC decoder which is useful for remote image browsing. This decoder relies on a parallel processing technique and the structure of the codestream.

CCSDS IMAGE DATA COMPRESSION STANDARD

The CCSDS-IDC standard supports both lossy and lossless compression. Lossless compression uses the 9/7 integer wavelet transform and lossy compression employs the 9/7 floating wavelet transform. A three-level Wavelet transform is applied to the image, and a total of 8 subbands are created. Prior to the bit-plane coding, the wavelet coefficients needed to reconstruct an 8×8 area in the spatial domain are grouped into one *block*. As shown in Figure 1, one block consists of 1 DC coefficient in the LL_3 subband and 63 AC coefficients extracted from the remaining subbands at the corresponding position. Consecutive blocks of 16 in the raster scan direction are grouped into a *gaggle*, which is a basic unit of entropy coding. Each set of gaggles is called a *segment*. Each segment is independently encoded, so other segments are not affected when data loss or corruption occurs. The number of blocks (or gaggles) constituting a segment is determined by compression performance, error robustness, and available memory.

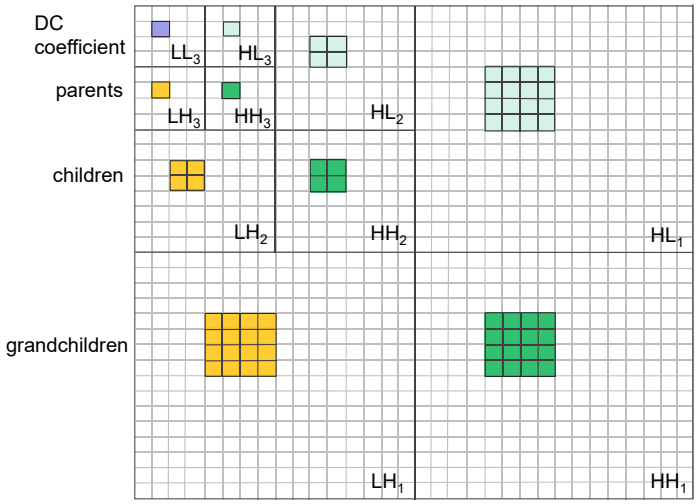

Figure 1: A single block is composed of 64 wavelet coefficients.

STRUCTURE OF KOMPSAT-3A SPACE PACKETS

As shown in Figure 2, compressed satellite images are delivered to the ground in the CADU (Channel Access Data Unit) format. The CADU consists of a synchronization mark, a transfer frame, and a Reed-Solomon (RS) channel code. A fixed-size source packet is constructed by grouping several pieces of transfer frame data except headers. The APID (Application Process Identification) in the header of the source packet is used to identify the type of channel (panchromatic, multi-spectral,

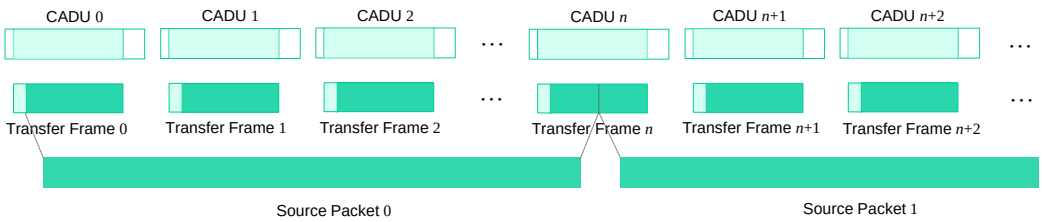

Figure 2: Construction of source packets.

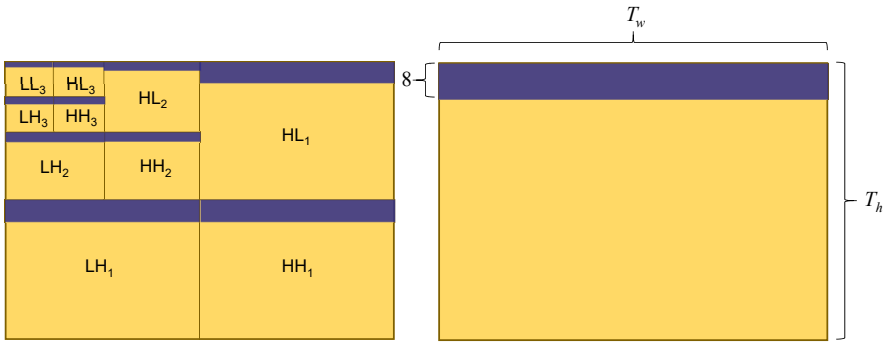

Figure 3: A segment is reconstructed as $T_w \times 8$ image. (a) Segment in the wavelet domain and (b) the corresponding spatial area in the spatial domain.

and so on).

The data portion of the source packet contains one segment of CCSDS-IDC codestream. Push-broom sensors equipped in the satellite, such as KOMPSAT-3A, have strip-based segments. As shown in Figure 3, one segment consists of blocks that represent $Tw \times 8$ pixels in the image. These N segments create a tile image of size $T_w \times T_h$ (or $T_w \times 8N$). The size of the tile image is also fixed, and the number of tile images increases as the capture time increases.

SCALABLE DECODING OF SPACE PACKETS

Figure 4 shows the channel configuration of a panchromatic (PAN) image. Since the PAN image has a very high resolution, it consists of eight channels with neighboring sensors, and each channel image is composed of several tile images whose number increases in proportion to the capture time. The tile images are independently encoded, so they can be processed in parallel. In the CCSDS-IDC standard, a segment is the minimum unit of parallel processing and spatial random access. In this work, however, a tile image which consists of N segments is used as the basic unit of parallelization and spatial random access for ease of management and implementation.

The DWT can inherently create images with various resolutions. A 3-level DWT in the CCSDS-IDC standard creates a total of four resolutions. As shown in Figure 5, the LL_3 image, which is a set of DC coefficients, produces the lowest resolution image by itself. An LL_2 image is synthesized with LL_3, HL_3, LH_3, and HH_3 images and has four times higher resolution than the LL3 image. The LL_2 image, again with HL_2, LH_2, and HH_2 images, produces an LL_1 image with a four

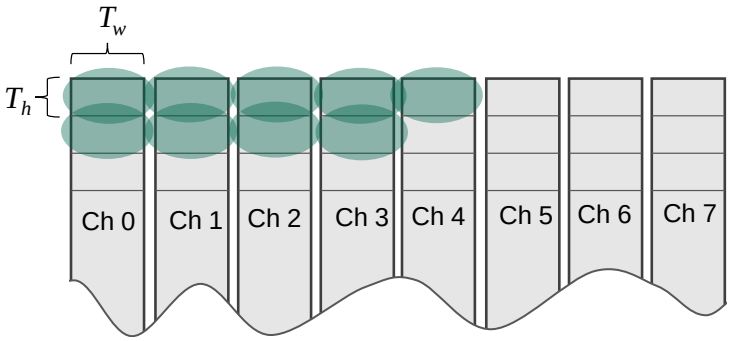

Figure 4: Panchromatic channels.

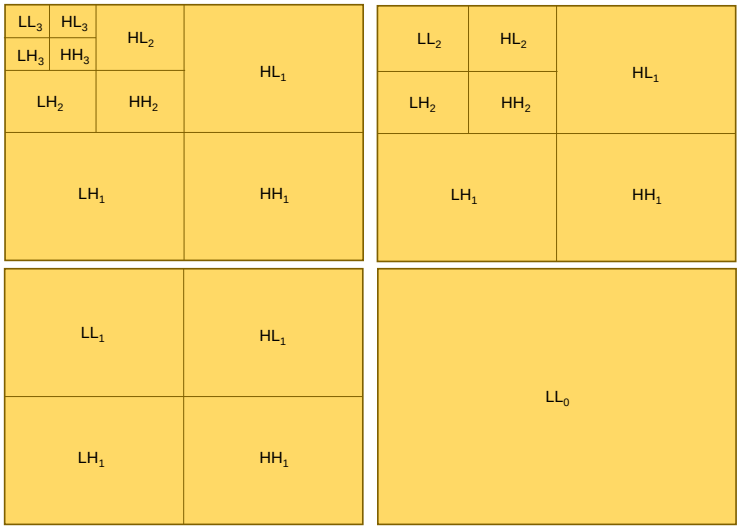

Figure 5: Wavelet image synthesis.

times higher resolution. In addition, an LL_0 image which has full resolution can be synthesized by combining LL_1, HL_1, LH_1, and HH_1 images. However, unlike the JPEG2000 codestream, the CCSDS-IDC code stream does not include a marker that distinguishes each subband. In the codestream for one segment, the DC components (LL_3) for entire blocks are placed in the front, and the AC components for each block are interleaved afterward. The LL_3 image corresponding to the overview image can be constructed by decoding the front part of the codestream. However, in order to construct higher resolution images (LL_2, LL_1, LL_0), all of the AC components of all blocks in the segment must be decoded.

Table 1 shows the decoding performance of the decoder (designated KARI), which utilizes tile image-based parallelization and resolution scalability, as proposed in the paper. For comparison, a reference decoder for the CCSDS-IDC standard [6] was used. The reference decoder does not exploit the inherent feature of the DWT and decompresses sequentially without parallel processing. The performance shown in Table 1 was measured on a system with an Intel i9-7980XE CPU with 18 cores and 64GB of memory, and a total of 36 threads. Each thread was allocated to process one tile image at a time.

resolution	KARI	[6]
LL_3 (overview image)	0.275	N/A
LL_2	5.619	N/A
LL_1	5.943	N/A
LL_0 (full resolution)	8.188	239.480

Table 1: Decoding time of the proposed decoder (seconds).

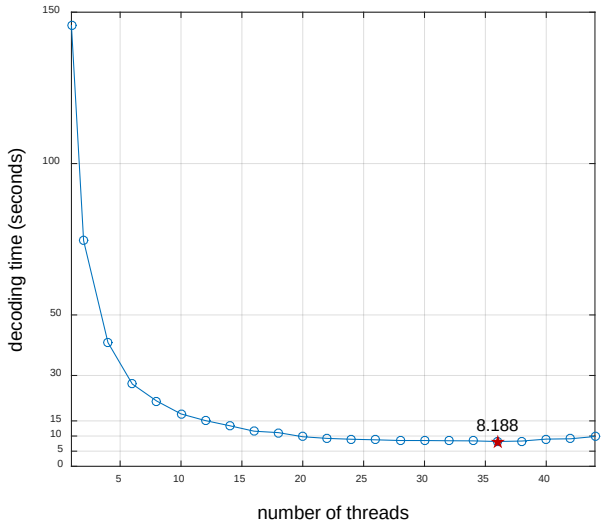

Figure 6: Decoding time of the LL_0 image by the number of threads (seconds).

The PAN image for the test consisted of 720 tile images taken from KOMPSAT-3A, and the size is approximately $24,000 \times 90,000$. The decoding performance was measured as the time taken to decompress from the source packet and store it as a TIFF or raw file on the hard disk. The experimental results show that the lowest resolution LL_3 image (about 3,000 pixels in width and 11,500 pixels in length) took 0.275 seconds, and the maximum resolution, LL_0 image, was 8.188 seconds, which is 29.25 times faster than the reference decoder of 239.480 seconds. The average decoding time per tile image (approximately 3,000 pixels in width, 1,000 pixels in height) of the reference decoder for the full resolution was 0.33 seconds, while the decoder presented in the paper was 0.011 seconds on average. It is worth noting that parallel processing is very effective for speedup.

Figure 6 shows the decoding time of the LL_0 image by the number of threads. As the number of threads to be processed in parallel increases, the processing time exponentially decreases. The best result was obtained using threads of 36, which is the maximum number of native threads the 18-core CPU supports was used. Using more than 36 threads did not help to improve the performance.

The high-resolution PAN image can be effectively displayed without decoding all the codestream if resolution scalability and spatial scalability are appropriately applied. An overview image is first displayed, and higher resolution images are then sequentially displayed at the request of the user. For the same display extent, as the resolution increases, the number of tile images to be decoded decreases. Figure 7 shows an example. For an LL_3 image, 64 tile images must be

(a) (b) (c) (d)

Figure 7: Example of Spatial and resolution scalability. (a) LL_3 image reconstructed from 64 tiles, (b) LL_2 image reconstructed from 16 tiles, (c) LL_1 image reconstructed from 4 tiles and (d) LL_0 image reconstructed from one tile.

decoded, but the LL_0 image can be displayed from only one decoded tile image.

Since high-resolution satellite images are viewed at various resolutions and locations, a user is likely re-visit an area which has been previously processed. Because of the limited memory capacity, it is difficult to keep all processed tiles in memory. Therefore, the proposed decoder applies a cache model. Tile images far from the tile images currently being displayed are first stored in the hard disk cache memory and removed from the memory. Then, when a display request is received for that area, the tile image is not decoded again but is loaded from the cache memory to reduce decoding time.

REMOTE IMAGE BROWSING

The scalable decoder described above is very effective for viewing images received at remote locations. Figure 8 shows a block diagram of an interactive CCSDS-IDC remote image browsing system, similar to the JPIP(JPEG2000 Interactive Protocol)-based remote browsing system [7]. The client connects to the server via two TCP channels. The first channel is bidirectional and used by the client to request the server to perform an operation or to receive the results of the operation from the server. The second channel is unidirectional and is used by the server to transmit the requested data to the client. To be specific, operations such as a connection request, satellite image list request, specified satellite image size request, region/resolution of interest request, and transmission stop request are communicated through the first operation channel, and the compressed data for the requested region/resolution of interest are transmitted in the second data channel. Since the data channel is dedicated for data transmission, no separate ACK is required. Also, as shown in Figure 8 (b), a unique session ID is allocated to a connected client so that a plurality of clients can connect to the server at the same time.

Figure 9 compares the time taken for each operation to view a high-resolution image remotely over the network protocol and to view the image from a local file. In both cases, the display regions and resolutions requested by the client are the same as the following: the display size is 1980×1080, which corresponds to full HD (High Definition). An overview (LL_3) image is first displayed and the image is panned 300 pixels to the right (denoted as PR) and then 300 pixels down (denoted as PD). Zoom-in is performed around the center of the last screen (denoted as LL_2). Then

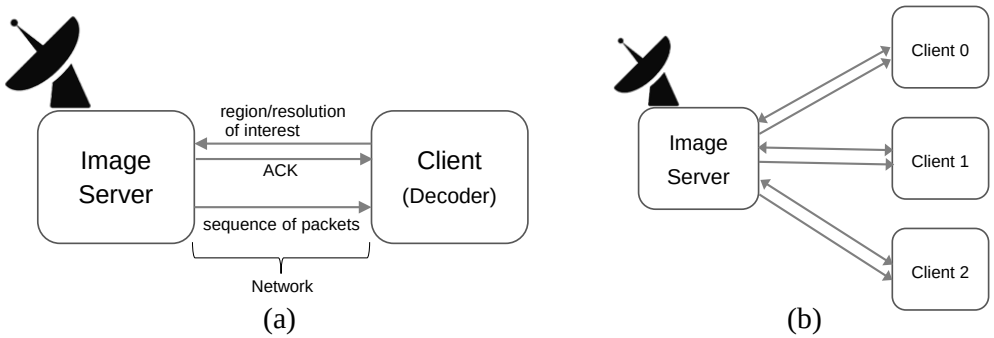

Figure 8: Client-server architecture for remote image browsing. (a) Client-sever interaction. (b) $1 : N$ client connection to the server.

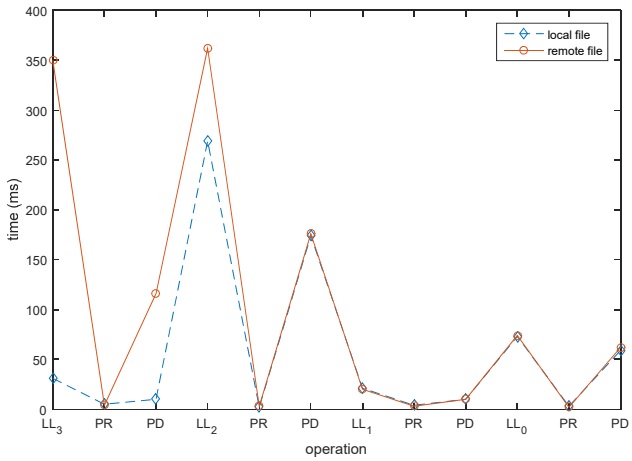

Figure 9: Operation time for each browsing operation.

it moves 300 pixels to the right, and 300 pixels down again. This process is repeated up to the full resolution LL0.

The network bandwidth is limited to 50Mbps. While viewing the 12 scenes, the longest operation, displaying LL_2, took 269ms for the local file, while it took 362ms for remote browsing. The difference in processing time between the local file and the remote browsing can be interpreted as the time required to transfer the data (approximately 100 ms). The reason that displaying LL_2 takes a long time is because the number of tile images it covers is not small and the CCSDS-IDC standard requires all AC coefficients be decoded, unlike JPEG2000, which only needs LL_3, HL_3, LH_3, and HH_3 to reconstruct LL_2.

Nevertheless, because the proposed decoder only processes the necessary area in parallel, the processing time is fast and the amount of data transmitted is not significant. Compared to the total size of a local file, 734.3MB, the size of data transferred while displaying all 12 scenes was 19.3MB. From this fact, it can be seen that the proposed decoder is effective for remotely viewing satellite images.

CONCLUSIONS

The CCSDS-IDC standard is designed to effectively compress images in a resource-constrained on-board computing system, which limits its ability to quickly decode regions of interest in high-resolution satellite images. To address this limitation, this paper proposes a method of more quickly viewing regions of interest with desired resolutions by decoding tile images in parallel utilizing the DWT and the structure of the codestream. In addition, this paper defines an interactive network protocol that facilitates the efficient transmission of remote images. Although parallel processing of tile images provides high performance at low resolutions, where several tile images appear at the same time on one screen, the performance improvement offered by parallel processing is reduced at high resolutions. More granular parallel processing by segments is possible, but the trade-off between ease of implementation and management and actual decoding performance should be considered. This is an on-going topic.

REFERENCES

[1] Consultive Committee for Space Data Systems (CCSDS), "Image data compression 122.0-b-1," Nov. 2005.

[2] D. S. Taubman and M. W. Marcellin, *JPEG2000: image compression fundamentals, standards, and practice*. Boston: Kluwer Academic Publishers, 2002.

[3] Consultive Committee for Space Data Systems (CCSDS), "Image data compression 120.1-g-2," Feb. 2015.

[4] I. Blanes, E. Maglid, and J. S.-Sagristá, "A tutorial on image compression for optical space imaging systems," *IEEE Geoscience and Remote Sensing Magazine*, vol. 2, pp. 8–26, Sep. 2014.

[5] G. Yu, T. Vladimirova, and M. N. Sweeting, "Image compression systems on board satellites," *Acta Astronautica*, vol. 64, pp. 988–1005, May. 2009.

[6] *CCSDS Image Data Compression Implementation*. (Available on-line at http://hyperspectral.unl.edu/).

[7] D. Taubman, "Remote browsing of JPEG2000 images," in *Proc. Int'l Conf. on Image Processing*, Feb. 2002.