Frame-Based Recovery of Corrupted Video Files Using Video Codec Specifications

Gi-Hyun Na, Kyu-Sun Shim, Ki-Woong Moon, Seong G. Kong, *Senior Member, IEEE*, Eun-Soo Kim, and Joong Lee

***Abstract*— In digital forensics, recovery of a damaged or altered video file plays a crucial role in searching for evidences to resolve a criminal case. This paper presents a frame-based recovery technique of a corrupted video file using the specifications of a codec used to encode the video data. A video frame is the minimum meaningful unit of video data. Many existing approaches attempt to recover a video file using file structure rather than frame structure. In case a target video file is severely fragmented or even has a portion of video overwritten by other video content, however, video file recovery of existing approaches may fail. The proposed approach addresses how to extract video frames from a portion of video to be restored as well as how to connect extracted video frames together according to the codec specifications. Experiment results show that the proposed technique successfully restores fragmented video files regardless of the amount of fragmentations. For a corrupted video file containing overwritten segments, the proposed technique can recover most of the video content in non-overwritten segments of the video file.**

***Index Terms*— Video file restoration, frame-based recovery, video file specifications, corrupted video data.**

I. Introduction

RECENTLY, a large amount of video contents have been produced in line with wide spread of surveillance cameras and mobile devices with built-in cameras, digital video recorders, and automobile black boxes. Recovery of corrupted or damaged video files has played a crucial role in role in digital forensics [1]–[3]. In criminal investigations, video data recorded on storage media often provide an important evidence of a case. As an effort to search for video data recorded about criminal, video data restoration and video file carving has been actively studied [4]–[6].

Most existing video data restoration techniques attempt to restore the source data using meta-information recorded in the header of a file system [7], [8]. The meta-information of filesystem contains file information such as file name, time of modification, physical location, link, etc. When the operator deletes a file, the corresponding file information in the meta-information of filesystem is updated as deleted although the video contents physically remain in the medium. Even though a video content exists in the media, it is challenging to recover the video data if the relevant meta-information is removed or altered. Conventional file restoration techniques find the meta-information of the deleted files to search for physical locations containing the actual file contents. However, the file cannot be restored if not all the file links are connected. Since a video file typically has a large volume of the data, it is highly likely to be fragmented although the meta-information remains in the file header [9], [10], [11]. When part of the file was overwritten, restoration of a video file with meta-information only may not be successful in most situations [12]. To tackle these problems, various techniques have been proposed by which if the file *start markers* and *end markers* are discovered based on the file signature, relevant data are collected to restore the video data [12]–[15].

Signature-based file restoration techniques search for the *start marker* (header) and the *end marker* (footer) to find a valid connection of the regions containing the header and the footer [16]. To increase the accuracy of the connection of the header and the footer regions, they used other information such as maximum size, embedded length recorded in the header. The analysis of the signature may offer a low success rate in video file restoration, when there are many file fragments and when some of them are overwritten. Especially, in the case a portion of a video file is overwritten, restoration of the video data using the file unit can be almost impossible because validation of restored file is failed by partially overwritten of restored file [17].

In this view point, we extend technique from conventional signature-based file restoration technique.[1] This paper proposes a technique to restore the video data on a frame-by-frame basis from its corrupted versions where the video data has been significantly fragmented or partly overwritten in the storage media. A video data consists of a sequence of video frames as the minimum meaningful unit of video file. The proposed method identifies, collects, and connects isolated video frames using the video codec specifications from non-overwritten portions of the video data to restore a corrupted video file.

Manuscript received March 18, 2013; revised August 8, 2013; accepted September 27, 2013. Date of publication October 17, 2013; date of current version December 17, 2013. This work was supported in part by the Midlong-Term Research Program through the National Forensic Service funded by the Ministry of Security and Public Administration under Grant NFS-NF-2013-10, and in part by the National Research Foundation of Korea grant funded by the Korea government under Grant 2012-0009224. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jean-Baptiste Thibault.

G.-H. Na, K.-S. Shim, K.-W. Moon, and J. Lee are with the Forensic Medicine Division, National Forensic Service, Seoul 158-707, South Korea (e-mail: ghna282@korea.kr; bluesks@korea.kr; kwmoon@korea.kr; ljfirst@korea.kr).

S. G. Kong is with the Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122 USA (e-mail: skong@temple.edu).

E.-S. Kim is with the Department of Electronic Engineering, Kwangwoon University, Seoul 139-701, South Korea (e-mail: eskim@kw.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2013.2285625

[1] This paper focuses on the restoration of video data since the restoration of a damaged video file from a surveillance camera. It is important in forensic investigation to recover criminal scenes to obtain the evidences from damaged video files.

The proposed technique restores the video data in a frame unit, not in a file unit. This is a simple, yet powerful video data restoration method that can recover a portion of the file even when a complete restoration of the file is not possible. The proposed frame-based video data restoration scheme can restore the video regardless of a filesystem. This approach can restore a video data from fragmented data stored on a corrupted or damaged video file. Since large size multimedia file tend to have a large amount of fragments [13], a file-based restoration technique may not be successful. File-based restoration of conventional methods is extremely difficult if the physical locations of all fragmented data are unknown or a part of file is overwritten.

The proposed method restores a corrupted or damaged video file using each video frame, the minimum unit of video file, using the index data on the disk area. In the region to restore, we extract the part of the data that can possibly be frame to do decoding. Then we collect the frames that can be connected after decoding to restore the video data. When a large amount of fragments exist and even when a part of file is overwritten, we can collect and connect remaining video frame to restore a video data. The technique consists of extraction phase and connection phase of relevant video frames. The extraction phase uses the video codec specifications to extract a set of video frames from the storage media. In the connection phase, the restored video frames are used to group and connect relevant video frames using the specifications of the video file used.

The main contribution of this paper is a video data restoration technique on the level of video frame, which is a minimum meaningful unit of a video file. Frame-based video file restoration enables recovery of a partially overwritten as well as fragmented video files by extracting and connecting the video frames into a video file. Experiment results show that frame-based restoration can recover most video contents stored in the medium using the codecs MPEG-4 Visual [18] and H.264 [19]. Experiment results with the video files randomly fragmented and/or overwritten reveal that the proposed frame-based video restoration scheme could recover the damaged video files. We can also show the proposed method could restore over 80% the fragmented video files regardless of the number of fragments when the overwriting level of file set the 50%. And We also tested this technique on the video files containing overwritten video segments. In this case, the proposed method could recover 90% of video data which is not overwritten video segments when the number of fragmentation is 10. However, conventional file-based video restoration techniques show limitations to restore video files containing overwritten video segments since all the video data are often required to be recovered to play the video contents in an application.

II. Previous Work

Recovery of damaged or corrupted video files obtained from a crime scene or a disaster site has provided a key evidence to resolve the cause. Conventional techniques for video file restoration use the meta-information of the file system to recover a video file stored in a storage medium such as a hard drive or a memory card [8]. The file system meta-information contains the information such as the address and the link of a video file that can be used for file restoration. Carrier [8] proposes a file restoration tool based on the file system, which was implemented in a software toolkit, *The Sleuth Kit* [20]. This program is based on the information from the file and directory structure of a storage filesystem. Video file restoration may not be possible with such techniques, however, when the file system meta-information is not available. Thus, attempts have been made to restore the video data from video contents, rather than the meta-information of a file system. This paper also presents a technique to restore damaged or corrupted video files irrespective of a file system.

The signature-based video restoration technique proposes *File Carver* [16] to address this problem. This method creates a database of the file header (beginning mark of file) and footer (the end mark of file), and define a set of rules for a specific file type. Signature-based file recovery techniques do not require file system information, which can be applied to a video file with no meta-information because of file system change and reformatting of a storage medium. Signature-based file recovery techniques identify the fragments from the byte-sequence (or magic bytes) containing file header or footer. Scalpel [16] does not rely on a file system to restore a video file. This technique requires an indexing step to find the file header and footer from a whole disc as well as a restoration step to recover indexed header and footer. We do note use file system metadata to restore the data between the header and footer to a file. This method is limited to the cases when the files are unfragmented. This method does not recover partially overwritten video files.

Garfinkel [13] utilizes additional information stored in the file to extend the idea to signature-based restoration techniques. For some files, file header may contain the information of file size or length. When the file footer does not exist, they can use this information to extract a file. A video file can be restored using Bifragment Gap Carving [13]. This method find a combination of the region containing the header and the footer to test if a video sample is valid. This computes the difference between the two data regions and check if the difference passes the predefined validation procedure. This procedure repeats until the gap passes the validation test. However, this method can only be applied to a video file with two fragments and this technique has limitation when the gap between the two file fragments is large.

SmartCarving technique was proposed to restore a file without being restricted by the number of fragments [12]. This technique, if it identifies the occurrence of fragmentation, combines the permutations of the fragment components and searches for the order of the fragments. They technique consists of three steps: preprocessing, collation, and reassembly. In the preprocessing step, they collect the called block part, which was not allocated to a file, using the file system information to reduce the size of the data to analyze. The collation step categorizes the collected blocks in the preprocessing step according to a file format. The reassembly step determines fragmented parts and merges them into a file. In [15], they extended SmartCarving to apply to multimedia

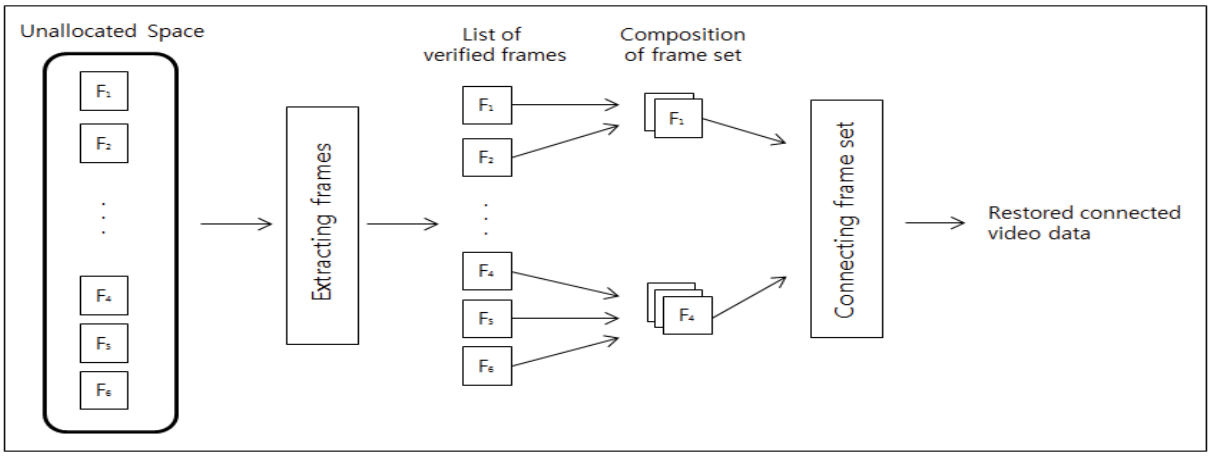


Fig. 1. Processing steps of the proposed frame-based video file restoration technique.

files. In the reassembly step, they increased the restoration rate of multimedia file by assigning a weight to each fragment using the decoded frame difference. However, the method presented in [15], which is also a file-based approach, has a limitation to restore a video file when a part of video file is overwritten.

In addition, graph theoretical carving was proposed by which the k-vertex disjoint graph is created to piece together fragments [14]. This technique proposed various greedy heuristic restoration techniques with which to use the matching technique and search for the sector/block order. The weight of all the fragment pairs should be calculated in advance, however, which is costly.

Most of previous technique bases its file restoration on a file unit, however, so only when a whole file is restored can the video be obtained. In general, the signature-based file carving techniques mentioned above consist of the following three steps [2].

1) **Identification Phase:** To identify a video fragment in a storage medium and to connect it to the previous fragment.
2) **Validation Phase:** To validate if all connected video fragments successfully form a playable video file.
3) **Validate by Human Expert:** To sort out false positive video segments by human expert.

The validation step checks if a restored video file is a playable video file. Conventional file-based video restoration techniques may fail to validate a restored video when a part of video is overwritten [17]. On the other hand, the proposed frame-based method carry out video restoration frame by frame, and is therefore applicable to restoration of partially overwritten video file.

III. Video File Restoration Using Video Codec Specifications

Video frame of a stored video file depends on the video codec used to encode the video file. And the video file that is encoded by codec also stored the decoding header information in start or end of video file. So that, the proposed method restore the video file using combination of frame data and decoding header information. The proposed technique applies to MPEC-4 Visual [18] and H.264 [19] video coding schemes, two popular video coding standards widely used in CCTVs, mobile devices, and automobile black boxes. For recover damaged or corrupted video, the proposed technique consists of two phases, extraction and connection as shown figure 1.

- **Extraction Phase:** The data are extracted based on video frame from the unallocated space, as extracted from the storage medium for restoration. The start code signature of video frame is searched for without considering the file system and the file composition. The frames are extracted based on the start code signature, the extracted frame data are verified through the decoder, and it is determined if the data are frames.
- **Connection Phase:** The codec and file specifications are used to connect the frames verified in previous phases. Based on the extracted frame sets, the length information of each frame recorded in the files is used to connect frame sets that are restored into a connected picture.

Figure 1 shows an overall process of the proposed file restoration technique. In extraction phase, we extract frame data, F1, F2, F4, F5, and F6, which have a start code signature of frame from the unallocated space, the region of a video file to recover, containing the deleted video files and verify if the decoded frame is a normal frame data. Verified frames form a frame set, which will be connected as far as it can go in the stage of connecting frame set. When the video file is fragmented, we restore a video file by connecting fragmented pieces of data. In case of a partially overwritten file, not overwritten parts are connected to create a connected video. In this manner, the proposed method finds meaningful data in the video file using the codec and convert into file structure after connecting them.

A. Extraction of Video Frames

A video file consists of a sequence of video frames, and each video frame is encoded into a binary data using a codec

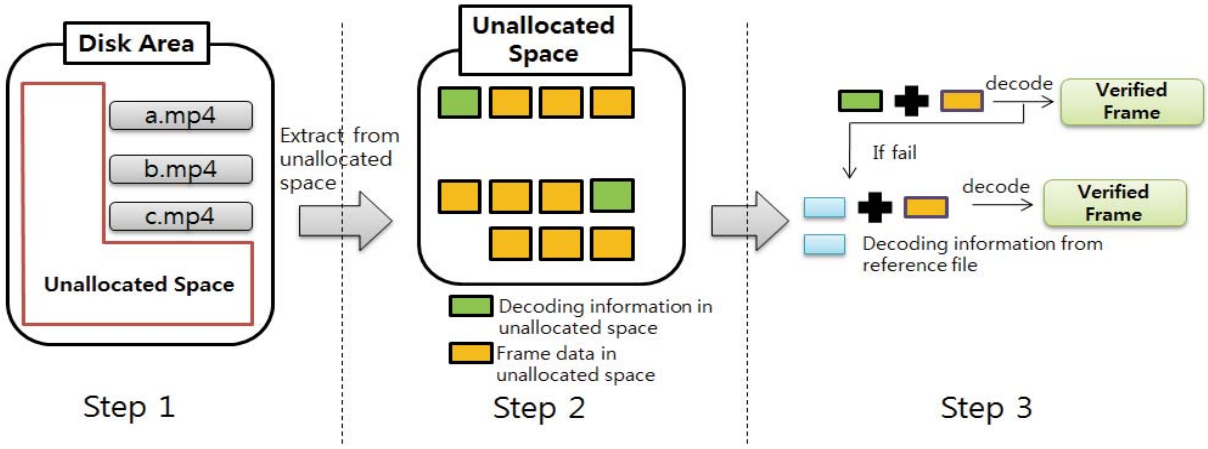


Fig. 2. The procedure of video file restoration using video file format specifications.

for data compression purpose. A codec inserts identifiers into each video frame to identify. The proposed method verifies if the data is a frame using the identifier characterized by a codec used in video encoding.

Figure 2 shows the steps for extracting the verified frame data from a storage medium. Step 1 is to extract an unallocated space using file system meta-information. Because deleted file data could be stored in unallocated space. It is possible to reduce the amount of data which frame has to be analyzed [15]. In practice, the popular forensic tools such as *Encase* [21] and *WinHex* [22] are used to extract unallocated space from storage medium.

In Step 2, we extract the signature of the frame data from unallocated space extracted in Step 1. In figure 2, yellow rectangles indicate the frame data with signature. If the frame data is found in the unallocated space, we verify them by decoder. For verifying frame data, the decoding header is attached in front of frame data. So the proposed method also extract the signature of decoding header information from unallocated space. We search for decoding header marked in green in figure 2. In general, decoding header is usually recored in the playback information and can be overwritten. In this case, if the decoding header encoded in the same manner as the file to be restored is found in an unallocated space, we can restore the video file. Even though the decoding header is not found, we can restore the video file using the decoding header of the reference file.[2] Reference video file refers to a video file encoded in the same codec as the video to be restored. In this paper, two most popular codecs, MPEG-4 Visual and H.264 are tested. Sections III-A1 and III-A2 describe how to find the signature of frame data and decoding header encoded in MPEG-4 Visual and H.264.

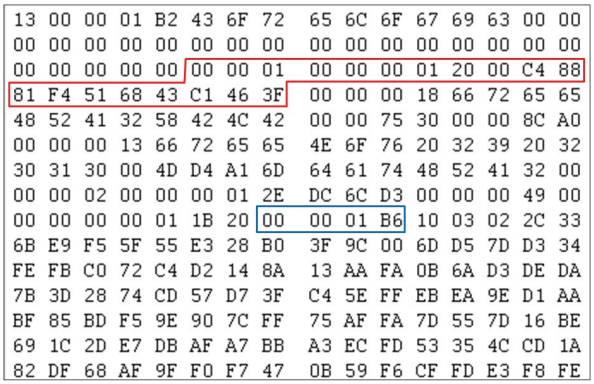
In Step 3, we verify the frame data extracted by combining frame data and decoding header using the signature of each codec. The frame data, which cannot be decoded, are combined with the decoding header information of the reference video file to re-verify the decoding.

[2]If the video data is encoded in the same format of codec, resolution and other options, the decoding header remains the same.

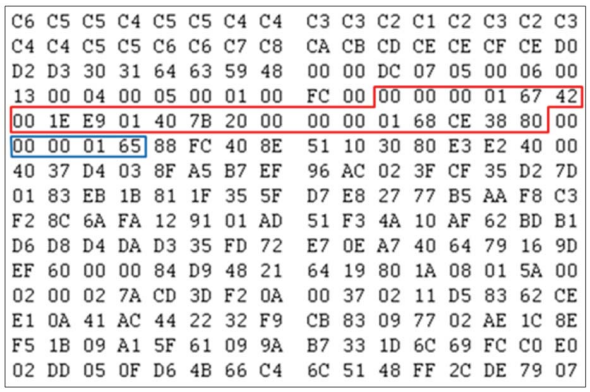
1) Mpeg-4 Visual Frame Extraction: MPEG-4 Visual is specified as a part of the MPEG-4 ISO/IEC standards 14496-2 [18]. The MPEG-4 Visual code begins with a start code signature (*0x000001*), and the next 1-byte indicates the type of the data that follows. For example, a code 0xB6 denotes the video frame. Then the start code of MPEG-4 Visual frame becomes *0x000001B6*. In order to decode a data into a video frame, the decoding header information is needed. The decoding header involves the start code, followed by the *video_object_layer_start_code* 0x20-0x2F. And the portion that starts with *0x00000120-0x0000012F* indicates the decoding header information. The frame data can be verified by decoding the frame data attaching the decoding header to front of them.

Figure 3(a) shows the MPEG-4 Visual decoding information and frame data contained in the actual unallocated space. For example, the red and blue boxes denote respectively decoding header information and the signature of video frame, MPEG-4 Visual video frames after *0x000001B6*. The decoding header information signature (*0x00000120-0x0000012F*) is extracted from the unallocated space. After that, we search for the frame data information signature (*0x000001B6*). These two pieces of information confirms that the video was encoded by an MPEG-4 Visual codec. If frame data is verified by the MPEG-4 Visual decoder, the decoder returns the size of the frame. The returned size will be used to connect frame.

2) H.264 Frame Extraction: H.264 is developed by ITU-T and ISO/IEC and standardized as ISO/IEC 14496-10 [19]. The data encoded in H.264 Codec is recorded in *Network Abstraction Layer* (NAL) unit. In H.264, there are a type with a Start Code (*0x00000001* or *0x000001*) in front of a NAL unit and a type with the size of the data following NAL stream in front of NAL. In this paper, H.264_Start refers to the first type, H.264_Length denotes the second type. A NAL unit enables to contain *Sequence Parameter Set* (SPS), *Picture Parameter Set* (PPS), and frame data (or Slice in H.264 Standard). SPS and PPS record the information necessary for decoding, such as the video profile/level, resolution, bit depth, and entropy coding mode, whereas the slices record the compressed frame data.



(a)



(b)

Fig. 3. Example video data encoded using (a) MPEG-4 Visual and (b) H.264.

H.264 requires SPS, PPS, and frame data (slices) for decoding. Figure 3(b) shows SPS, PPS, frame data (slice) encoded by H.264_Start in an unallocated space. The red box indicates decoding header information consisting of SPS and PPS, if a 1-byte code following a start code (*0x00000001*) is 0x67, or PPS if it is 0x68. In H.264 standard, if the last 5 bits in the first byte of a NAL unit following the start code or length information is 7, then it denotes SPS (or PPS if 8). Frame data (Slices) are classified into *Instantaneous Decoder Refresh* (IDR) frames and P/B frames. An IDR frame, as an independent frame, can also express pictures, whereas a P/B frame can express only pictures when it has a reference frame similar to that of the IDR frame. In H.264 standard, if the last 5 bits of the first byte of a NAL unit is 5, then it denotes IDR frame (or P/B frame if 1). In figure 3(b), the data in the blue box denotes IDR frame data because the code 0x65 comes after the start code.

H.264_Start finds the start code (*0x00000001*) in an unallocated space in the same manner as in MPEG-4 Visual, checks the last 5 bits after the start code. If the last 5 bits are 7, then SPS (PPS if 8, IDR frame if 5, and P/B frame if 1)

In H.264_Length, there exists the data length of a NAL unit in place of start code (0x00000001) in H.264_Start. If the NAL unit is SPS or PPS, there exists a 2-byte length information, a 4-byte length information in the frame data (Slices). In H.264_Length, we use the length information to extract SPS, PPS, frame data (slices), and combine them to verify through H.264 decoder.

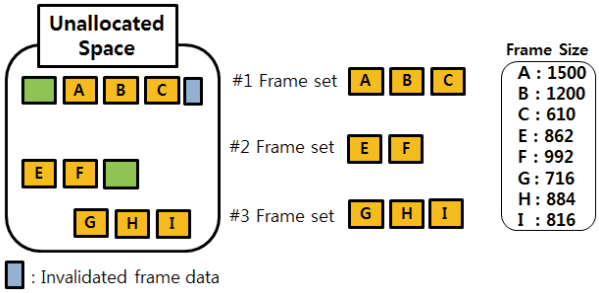


Fig. 4. Composition of Frame Set.

B. Connection of the Extracted Frames

The proposed technique, based on the verified frame data, forms frame set with physical locations of frame being continuous. The frame set compose verified frame in order before and after the relevant frame. The size information of each frame recored in meta-information of the files with the stored video data are used to connect the frame sets. By connecting frame sets, the fragmented video frames can also be connected and restored.

Figure 4 illustrates an example of the composition of frame set using the verified frame. The yellow rectangle denotes verified frame through decoder, and blue rectangle denotes a frame data with a start signature but not verified by the decoder. It can be consider that the data is not frame but contain start signature of frame data, fortunately. If this data is frame, it can be considered as either fragmented or partially overwritten data. When invalidated frame data occurs in the first frame set, or the physical offset between frame data is long like between second frame and third frame, a frame set determined. In this manner, all the verified frame in Section III-A composes the frame set. In figure 4, first frame set consists of the frame A, B, C and second frame set is frame E, F. The other frames are included third frame set. The frame size in right side results in decoding when verifying the frame. If the frame data is verified, the size of frame returns completely by decoder. And the verified frames are formed the frame set and it can be connected comparing size of frame data contained in frame set and the size information of each frame *(STSZ box)* which is contained in video files.

The proposed technique connects frame sets using the one of the meta-information of video file. The connection phase uses the file meta-information based on the restored frame sets and restores the data into connected video. The video file meta-information includes the offset location, size, and other information for each frame. This paper only uses the size information of frame in a video file. For a MPEG-4 file, the size information of each frame is recorded in *Sample-to-Size* (STSZ) box. STSZ box is also found in an unallocated space in Step 1 of Section III-A. The first four bytes of the signature starting with *stsz* denote the size of the STSZ box. The most important thing is that this size-information (STSZ box) volume is not big, so it is less likely than frame data to be fragmented or overwritten by other data. However, STSZ box often comes at the end of an MPEG-4 file. In this case, STSZ

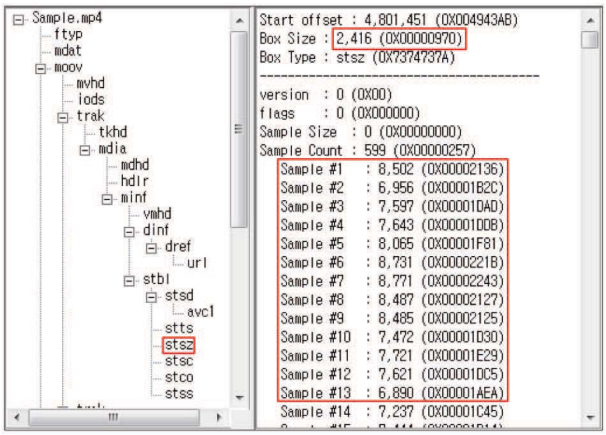


Fig. 5. *STSZ* box data in an MPEG-4 file.

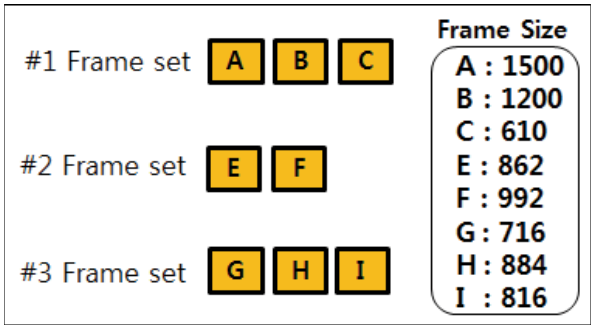
box can also be fragmented regardless of the volume size, which makes it difficult to find all the STSZ boxes. Therefore, the proposed method connects the video frames without STSZ box.

In order to use STSZ box for connecting frame sets, we tested more than 10,000 frames recorded in STSZ from 20 video files. In approximately 60% of the data, there existed the frames of the same length. It was only 0.1 % that two consecutive frames were found in the other part. And there was no more than 3 consecutive frames found. We can conclude that if more than 3 consecutive frames exist, it is the frame set consisting of the file that contains the corresponding STSZ.

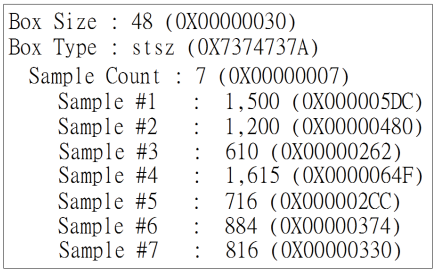
If the size information (STSZ Box) is discovered from the unallocated space, the STSZ box and the size obtained in the restoration process are compared to connect the video frames. if not discovered, the proposed technique also suggest the method to connect the video frames. After connecting the video frames, the proposed method reconstruct playable MPEG-4 file (.mp4) from the connected video frames.

Figure 5 shows example of *Sample-to-Size (STSZ)* box that have information on the sizes of all the frames in MPEG-4 files. An MPEG-4 file stores basic information in the *moov* field and the codec information, time stamp, offset, and size in the *stbl* field. The proposed method uses *stsz* field containing frame size information to connect frame set described above. The *STSZ* box, similar to the left square box, is recorded under *moov-trak-mdia-minf-stbl* in the MPEG-4 file structure. Appearing on the top right red square box, the *STSZ* box accounts for 2,416 bytes of the sample MPEG-4 file with 4,989,500 bytes. The size of this *STSZ* box is smaller than that of video data (about 25% of each frame in example of figure 5) , which makes it less likely for the video frame to be overwritten and fragmented, so it is compared with the frame data size obtained in the process, which enables the connection of the restored frames.

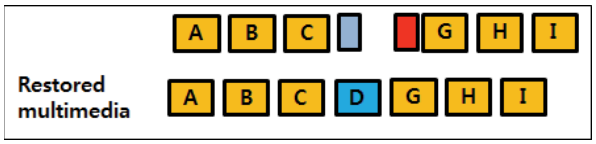
Figure 6 illustrate the example of connecting frame sets when the *STSZ* box is found in the unallocated space. Figure 6(a) is the formed frame set, and we compare the frame sizes generated by decoder and *STSZ* box from unallocated space described in figure 6(b). As mentioned above, *STSZ* box has the each frame size in file to recover, we can recover the



(a)



(b)



(c)

Fig. 6. Example restoring multimedia using *STSZ* box. (a) The Formed Frame set. (b) Size Index (*STSZ* Box) in Unallocated Space. (c) Restored multimedia.

video file by connecting frame set each other like figure 6(c). The frame sizes of the frame A, B, C match with the sizes of sample #1, #2, #3 in figure 6(b) and the frame size of the frame G, H, I match with the size of sample #5, #6, #7. So that we can assume that the frames can connect from sample #1 to sample #7 except sample #4. In this situation, we illustrate the connected frame like top layer in figure 6(c). This figure does not include sample #4. However, we can infer a sample #4 from the rear part of the first frame set (blue rectangle in figure 6(c)) and the front reaming area in cluster of the frame G (red rectangle in figure 6(c)). It is because that the decoder does not verify the frame data in blue rectangle in figure 4 through its data start with start code of frame. So that, we attempt to connect the first frame set and the third frame set by combination of blue rectangle combines and the red rectangle. Generally, file systems allow the specification of a fixed record length called cluster or block which is used for all write. From motivated it, blue rectangle and red rectangle are expended as long as exact size sample #4 from *STSZ* box. If the result of combination is verified by the decoder, the all the frames in *STSZ* box are connected perfectly like the bottom of figure 6(c). If not, we guess that the cluster located sample #4 is overwritten. So that we can restore two connected video as the first frame set and the third frame set.

We propose the method to connect frame set in case *STSZ* box is not found in the unallocated space as in figure 4.

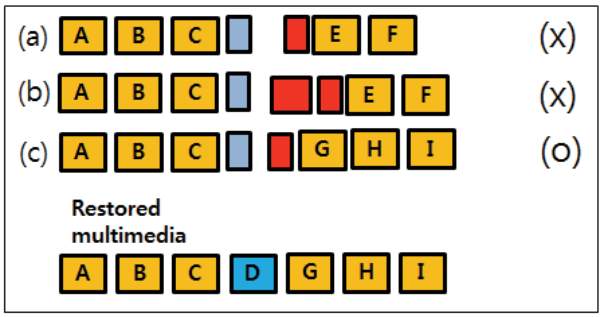


Fig. 7. Connection of the Verified Frame without a Size Index (*STSZ* Box).

We found the connected frame set that every frame set is 1:1 matching after that verified by decoder. In figure 7(a), the first frame set and the second frame set match each other according to cluster size. The blue rectangle and the red rectangle have same meaning as the blue rectangle and the red rectangle in figure 6. If the combination of blue rectangle and red rectangle is verified by decoder, the matching process stops. If not, we extend the blue rectangle or red rectangle as long as cluster like in figure 7(b), and verified it. By setting a threshold, unlimited cluster expansion can be prevented. After extension, the verification by the decoder does not pass, the first frame set combine with next frame set. And the combination repeats this process until the verification success like in figure 7(c). We restored the connected multimedia without *STSZ* box. However, this process takes time since all frame set is being matched one by one. Despite time complexity of $O(n^2)$, this method focuses to restore connected multimedia which is possible to record crime scene.

IV. Experiment Results

To evaluate the performance of the proposed technique, we tested for three kinds of video files encoded with MPEG-4 Visual, H.264_Start, and H.264_Length codecs, respectively. Experiments were carried out with different amount of data fragmentation and overwriting. For each codec, 20 video files (.mp4) were fragmented into 0–20 pieces in any size, and 0–90% of each video file was overwritten. First, to evaluate the fragmentation impact, with the overwriting level set at 50%, the number of fragmentations was changed. Second, to evaluate the overwriting impact, with the number of fragmentations set at 10, the overwriting level was changed.

Table 1 summarizes average file size, the number of intraframe (denoted IDR frame in H.264 standard) and the total frame count of the 20 video files encoded using three different types of codec. The file size and the number of frames were largest with H.264_Start, and the standard deviations by item varied significantly. Also, with both MPEG-4 Visual and H.264_Length, the 20 samples had a consistent picture file size, number of intraframes, and total number of frames. MPEG-4 Visual and H.264 codecs has 18 and 14 interframes following an intraframe, respectively. In terms of the ratio of file size and the number of frames, MPEG-4 Visual and H.264_Length differ by 3 times while file sizes differ by two times. MPEG-4 Visual has low compression ratio and each frame size is big.

To evaluate the performance of the proposed technique, the restoration ratio was evaluated by following equation.

$$\text{Ratio}(\%) = 100 * \frac{\textit{No. of Restored Video Frames}}{\textit{No. of Total Video Frames}} \quad (1)$$

The number of restored frames is the number of frames extracted from the storage medium using the proposed technique, and the number of the original video frames is the number of the original video frames that were used in the experiment. If all the frames of the original video were restored, the restoration ratio would be 100%; and if none was restored, the restoration ratio would be 0%. To check if the frame data were restored, the hash values of the restored video frames and the original video frames were compared through the MD5 function.

Conventional signature-based technique considers that a restored video file is validated if the file is playable in a native application program [15]. The proposed method selectively restores an non-overwritten part of a damaged or corrupted video file. To verify the performance of the proposed method, we conducted experiments to video files containing overwritten part and fragments. Since signature-based video restoration methods can not validate partially restored video files using native application or fast object validation [13].

The intraframe restoration ratio of the proposed technique is explained in figure 8. The intraframe restoration ratio is very important to restore video file in our proposed method. Because, if an intraframe is overwritten or dropped, the efficiency of the restoration is drastically reduced. When the intraframe is overwritten, the following interframe could not recover. On the contrary, when the intraframe is fragmented, the proposed method could connect the fragmented intraframe as shown figure 9. However, the proposed method could recover the 90% of intraframes with the exception of overwritten in figure 8. We also evaluates the restoration ratio of the technique not only for intraframes but also for all continuous picture frames for the MPEG-4 Visual codecs as figure 9. In figure 9, the recovery ration of continuous frames is higher than that of intraframes.

Figure 8(a) shows that the file recovery rate decreases as the number of fragmentation slightly increases. When the amount of overwrite was set to 50%, the recovery ratio was almost consistently 50%. For a high video compression ratio, there exist fewer intraframes than interframes, rarely leading to fragmentation of the intraframes. Restoration rate slightly decreases as the number of fragmentations increases. Video file can be restored using the intraframe unit, which is the minimum meaningful unit regardless of the codec. In case of MPEG-4 Visual codec, however, the restoration rate declines more than the cases of H.264_Start and H.264_Length as the number of fragmentations increases. This is because MPEG-4 Visual has a lower compression ratio than H.264_Start and H.264_Length, which are encoded with H.264 specifications that make the frame larger and slightly boost the probability of fragmentation and overwriting.

Figure 8(b) shows the impact of the amount of overwriting on the restoration rate of the proposed technique. As the amount of overwrite increases, the recovery rate decreases.

TABLE I
AVERAGE FILE SIZE AND NUMBER OF FRAMES IN THE TEST DATA

Codec	File size(MB)	Number of IntraFrames	Total number of Frames
MPEG-4 Visual	6.44	29.50	533.10
H.264_Start	39.4	271.35	4,055.40
H.264_Start	39.4	271.35	4,055.40

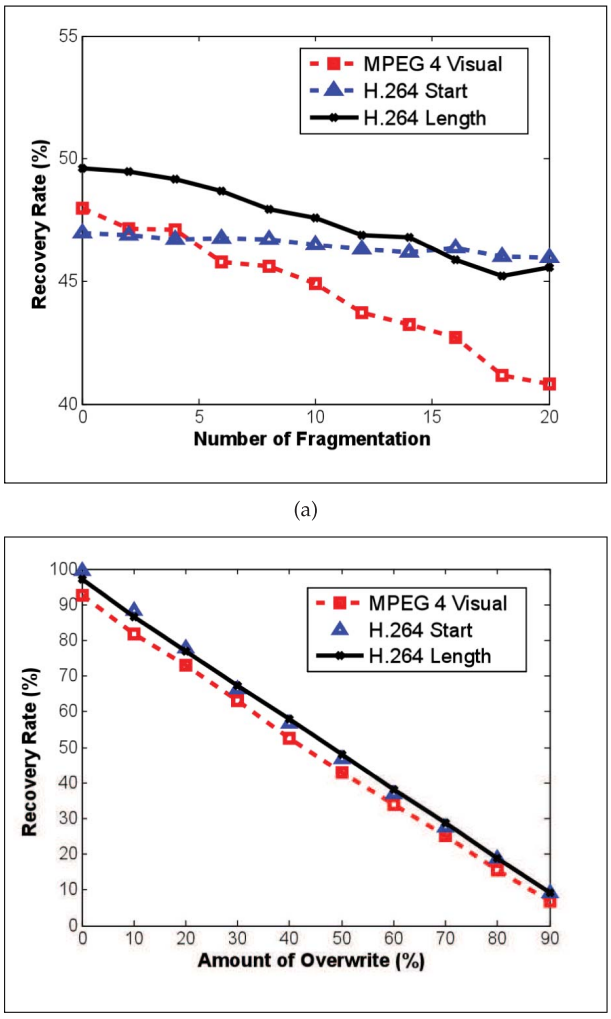


Fig. 8. Intraframe recovery performance. (a) Varing Fragmentation. (b) Varing Overwriting.

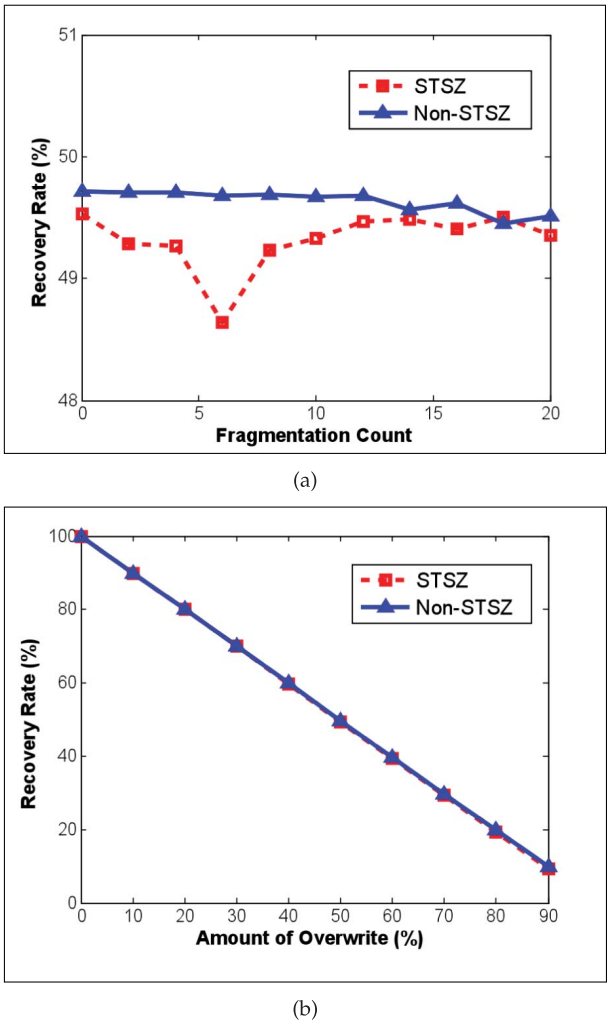


Fig. 9. Performance of the Frame Connection Strategy. (a) Varing Fragmentation. (b) Varing Overwriting.

When the target video data were overwritten, they disappeared, which made their restoration impossible. Non-overwritten portions of the video file can be recovered. Existing technique cannot restore the video file if part of the file data are overwritten, but the proposed technique could restore intraframes in portions, except where the file composition is broken or overwritten. MPEG-4 Visual offers a lower restoration rate than H.264_Start and H.264_Length because of lower compression ratio since its intraframe is generally bigger. The performance goes down slightly in the testing and evaluation because the number of fragmentations was set at 10.

Figure 9 shows the restoration rate when the interframes were connected to the intraframes and when connected frames was restored. In case of fragmented video files, the fragments were connected; and for overwritten files, non-overwritten portions were connected and the results were evaluated. The restoration rate was not affected by the type of the codec, so the experiment was limited to MPEG-4 Visual. Figure 9(a) shows the restoration rate according to the number of fragmentations. The fragmentation did not significantly affect the existence and the non-existence of *STSZ*. In both cases, when the overwriting ratio was close to 50%, the restoration ratio was close to 50%. This is because in the case of the existence of *STSZ*, the sizes of the frame, as verified in the frames through the decoders, i.e., the actual frame sizes according to the file specifications, were compared with those in the continuously recorded *STSZ*. When the sizes of the frames in the two portions continued to match three or more times, they were regarded as continuous frames, which enabled the connection of most of the frames, as verified with the decoders. Since frames, as verified by decoders, are connected by frame

set, the frames in the frame set and the fragmented frames between the frame sets are all connected, which enables the restoration of most frames into connected video. In the *non-STSZ* case, the fragmented frames between the frame groups are matched with all the frame groups and verified with decoders. If the frame sets with which the fragmented frames match, and the frame data, are verified with decoders, the two frame groups will be connected. The fragmented frames will be connected and the continuous video that connect the frames will be restored.

Figure 9(b) shows the effect on the frame connection according to the overwriting degree. The greater the overwriting degree was, the lower the degree of restoration was; but in the non-overwritten portions, most of the frames were connected and restored. This is because the size of the non-overwritten data can be identified with decoders and they can be connected by comparing them with *STSZ*, thereby enabling the connection of most of the frames. Also in the *non-STSZ* case, all the portions that were verified as fragmented with the decoders were matched with all the frame sets that were verified and discovered with the decoders, which enabled the connection and restoration of the frames into whole video even though they were fragmented or had been overwritten. In the cases in figure 9(a) and (b) that showed the existence of *STSZ*, only when the *STSZ* frame size information continued to match the sizes of the frames in the frame groups three times or more was it regarded as a frame. For the 20 videos, the sizes of more than 10,000 frames that were recorded in the *STSZ* were compared, which revealed that 60% of the frames had the same sizes, and that 0.1% of them had two frame sizes that were continuously duplicated. When more than three frame sizes continued, however, more than three continuous frame sizes were not equally discovered in other locations, which make them unique. In *non-STSZ* case, the time complexity was not considered an only the restoration ratio was experimented on and evaluated. When all the frame groups were matched 1:1, the time complexity was $O(n^2)$ and n was the number of frame groups. Since in a storage medium, many files are fragmented and overwritten, the *non-STSZ* case has the weakness of a long actual processing time. From the forensic viewpoint, obtaining the evidence from the videotaped incident is more important than processing time.

V. Conclusion

This paper presents a video restoration technique for fragmented and partially overwritten video files. The proposed technique guarantees the integrity of the restored frames because video files have the minimum number of frames to offer evidence. Large-size video files are often fragmented and overwritten. Many existing file-based techniques could not restore partially overwritten video files. Unlike most existing methods that use file format or file system meta-information, the proposed technique restores the data according to the minimum meaningful frame unit. Therefore, the proposed method restores almost frames in damaged or corrupted video files without being affected by the number of fragmentations. Especially, the proposed technique can restore the frames of the non-overwritten portions in partially overwritten files. We evaluated the intraframe restoration rate, as well as the restoration rate of the connected frames in the cases with the existence and non-existence of *STSZ* and with the video file recording the frame size.

Experiment results show that most of the frames were restored. The proposed frame-based file recovery technique increases restoration ratio. From the experiment for fragmented video files, 40 $\sim$ 50% of the data was recovered from a corrupted video with 50% overwriting regardless of the amount of fragmentation. For the experiments with overwritten video files, the portion of the video file not overwritten was recovered regardless of fragmentation.

References

[1] K. Medaris and R. Mislan. (2008). *Expert: Digital Evidence Just as Important as DNA in Solving Crimes* [Online] Available: http://news.uns.purdue.edu/x/2008a/080425T-MislanPhones.html

[2] R. Poisel and S. Tjoa, "Forensics investigations of multimedia data: A review of the state-of-the-art," in *Proc. 6th Int. Conf. IT Security Incident Manag. IT Forensics*, May 2011, pp. 48–61.

[3] H. T. Sencar and N. Memon, "Overview of state-of-the-art in digital image forensics," *Algorithms, Archit. Inf. Syst. Security*, vol. 3, pp. 325–348, Nov. 2008.

[4] L. Huston, R. Sukthankar, J. Campbell, and P. Pillai, "Forensic video reconstruction," in *Proc. ACM 2nd Int. Workshop Video Surveill. Sensor Netw.*, 2004, pp. 20–28.

[5] A. B. Lewis, "Reconstructing compressed photo and video data," Comput. Lab., Univ. Cambridge, Cambridge, U.K., Tech. Rep. 813, 2012.

[6] R. Poisel and S. Tjoa, "Roadmap to approaches for carving of fragmented multimedia files," in *Proc. 6th Int. Conf. ARES*, Aug. 2011, pp. 752–757.

[7] L. Aronson and J. Van Den Bos, "Towards an engineering approach to file carver construction," in *Proc. IEEE 35th Annu. OMPSACW*, Jul. 2011, pp. 368–373.

[8] B. Carrier, *File System Forensic Analysis*, vol. 3. Boston, MA, USA: Addison-Wesley, 2005.

[9] D. Billard and R. Hauri, "Making sense of unstructured flash-memory dumps," in *Proc. ACM Symp. Appl. Comput.*, 2010, pp. 1579–1583.

[10] G. Richard, III, V. Roussev, and L. Marziale, "In-place file carving," in *Advances in Digital Forensics III*. New York, NY, USA: Springer-Verlag, Jan. 2007, pp. 217–230.

[11] V. L. L. Thing, T.-W. Chua, and M.-L. Cheong, "Design of a digital forensics evidence reconstruction system for complex and obscure fragmented file carving," in *Proc. 7th Int. Conf. CIS*, Dec. 2011, pp. 793–797.

[12] A. Pal and N. Memon, "The evolution of file carving," *IEEE Signal Process. Mag.*, vol. 26, no. 2, pp. 59–71, Mar. 2009.

[13] S. L. Garfinkel, "Carving contiguous and fragmented files with fast object validation," *Digit. Invest.*, vol. 4, pp. 2–12, Sep. 2007.

[14] N. Memon and A. Pal, "Automated reassembly of file fragmented images using greedy algorithms," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 385–393, Feb. 2006.

[15] R. Poisel, S. Tjoa, and P. Tavolato, "Advanced file carving approaches for multimedia files," *J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl.*, vol. 2, no. 4, pp. 42–58, 2011.

[16] G. G. Richard and V. Roussev, "Scalpel: A frugal, high performance file carver," in *Proc. DFRWS*, 2005, pp. 1–10.

[17] T. Laurenson, "Performance analysis of file carving tools," in *Security and Privacy Protection in Information Processing Systems*. New York, NY, USA: Springer-Verlag, 2013, pp. 419–433.

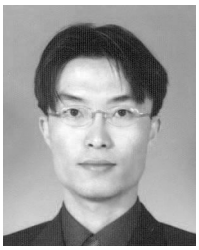
[18] *Information Technology—Coding of Audio-Visual Objects—Part 2: Visual*, ISO/IEC Standard 14496-2:2004, 2004.

[19] *Information Technology—Coding of Audio-Visual Objects—Part 10: Advanced Video Coding*, ISO/IEC Standard 14496-10:2009, 2009.

[20] B. Carrier. (2005). *The Sleuth Kit* [Online]. Available: http://www.sleuthkit.org/sleuthkit/

[21] (2001). *Encase* [Online]. Available: http://www.guidance-software.com

[22] (2004). *Winhex* [Online]. Available: http://www.x-ways.net/winhex/index-m.html



Gi-Hyun Na received the B.S. and M.S. degrees in electronic engineering from Kyungpook National University, Korea, in 1998 and 2000, respectively, and the Ph.D. degree in electronic engineering, Kwangwoon University, Korea, and he is a Researcher of the Digital Forensic Team, National Forensic Service, Korea. His research interests include image file recovery, reassembly of video data, mobile forensic, digital forensic, and 3D display.

Kyu-Sun Shim received the B.S. and M.S. degrees in computer science and engineering from Korea University, Korea, in 2009 and 2011, respectively. He is currently a Researcher of the Digital Forensic Team, National Forensic Service, Korea. His research interests include image file recovery, reassembly of video data, mobile forensic, and digital forensic.

Ki-Woong Moon received the B.S. and M.S. degrees in electronic engineering from Kyungpook National University, Korea, in 1999 and 2001, respectively. He is currently a Researcher of the Digital Forensic Team, National Forensic Service, Korea. His research interests include image processing, video compression and image forensic.

Seong G. Kong (SM'03) received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Seoul, Korea, in 1982 and 1987, respectively, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 1991. He was an Assistant Professor from 1992 to 1995, and an Associate Professor from 1996 to 2000 with the Department of Electrical Engineering, Soongsil University, Seoul. He served as a Chair of the Department from 1998 to 2000. From 2000 to 2001, he was with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, as a Visiting Scholar. He joined the University of Tennessee, Knoxville, TN, USA, as an Associate Professor with the Electrical and Computer Engineering Department, from 2002 to 2007. Currently, he is an Associate Professor and Graduate Program Director with the Electrical and Computer Engineering Department, Temple University, Philadelphia, PA, USA. His research interests include pattern recognition, image processing, and intelligent systems. He was awarded the Best Paper Award from the International Conference on Pattern Recognition in 2004, the Honorable Mention Paper Award from the American Society of Agricultural and Biological Engineers and the Professional Development Award from the University of Tennessee in 2005. In 2007 and 2008, he received the Most Cited Paper Award from the journal *Computer Vision and Image Understanding*. His professional services include Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS, Guest Editor of a special issue of the *International Journal of Control, Automation, and Systems*, and program committee member of various international conferences.

Eun-Soo Kim is a Professor of the Department of Electronics Engineering, Kwangwoon University, Korea, and the Director of the 3D Display Research Center and HoloDigilog Human Media Research Center, HoloDigilog. In 1984, he received the Ph.D. degree in electronics from Yonsei University, Seoul, Korea. He was a Visiting Professor with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA, USA. He served as the President of the Society of 3D Broadcasting and Imaging and the Korean Information and Communications Society from 2000 to 2011. He has served as the Editor-in-Chief of *3D Research* since 2010 and the General Chair of the international meeting Collaborative Conference on 3D and Materials Research since 2011. His recent research interests include 3-D imaging and display, free-space holographic virtual reality, and digital holographic microscopy.

Joong Lee received the B.S., M.S., and Ph.D. degrees in chemical engineering from Kwangwoon University, Korea, in 1994, 1999, and 2004, respectively, and the B.S. degree in computer engineering from Korea Open University, Korea, in 2010. He was with the Electrical and Computer Engineering Department, Temple University, Philadelphia, PA, USA, as a Visiting Scholar. He is currently a Chief of the Document and Image Division, National Forensic Service, Korea. His research interests include question document, image processing, mobile forensic, and digital forensic.