Accelerating Video Carving from Unallocated Space

Hari Kalva[†], Anish Parikh[*], and Avinash Srinivasan[‡]

[†]Florida Atlantic University, 777 Glades Road, Boca Raton, FL, USA 33431;

[*]Nirma University Institute of Technology, Ahmedabad, India;

[‡]George Mason University, Fairfax, VA, USA 22030

Abstract

Video carving has become an essential tool in digital forensics. Video carving enables recovery of deleted video files from hard disks. Processing data to extract videos is a computationally intensive task. In this paper we present two methods to accelerate video carving: a method to accelerate fragment extraction, and a method to accelerate combining of these fragments into video segments. Simulation results show that complexity of video fragment extraction can be reduced by as much as 75% with minimal impact on the videos recovered.

Keywords: Video carving, forensics, fragment detection, complexity reduction

Introduction

Video carving, the process of recovering video from unallocated disk space, has become an essential tool in digital forensics [1, 2, 3]. There has been some prior work on video carving including open source tools such as Defraser. Hard disk capacity has continued to increase and 1 TB disks are now common on consumer grade laptops and PCs. As the disk capacity grows, the complexity of video carving grows as the amount of unallocated space also grows. In this paper we present algorithms to accelerate video carving.

Related work

Defraser is a forensic analysis application that can be used to detect full and partial multimedia files in data streams [8]. It is typically used to find (and restore) complete or partial video files in data streams (for instance, unallocated disk space). Defraser offers many tools to analyze a given video segment. It separates all the headers of a given video. It displays the size of each Sequence, GOP, Picture and Slice. It also displays the type of each picture i.e. whether it is a I , B or a P frame. It offers an option of selecting keyframes in a video.

Our approach differs significantly from that used by Defraser. The first and foremost difference is that Defraser employs byte-by-byte scanning. This greatly increases the processing time. To speed up fragment detection, we have used a smart skipping approach to skip some of the slice data in various pictures. This eliminates the need for byte-by-byte analysis and thus helps in reducing processing time. While skipping byte processing accelerates video carving, it might lead to unidentified frames/fragments; this is a tradeoff that can be set according to the requirements of users. A user can decide the amount of data loss which is acceptable to him while at the same time avail a proportional decrease in the processing time required.

The second important distinction between Defraser and our approach is our use of probabilistic methods to recombine the fragments. The recombination of fragments is done on the basis of picture size and picture type. This greatly reduces the "false hits" issue which is a major drawback of Defraser.

Media Watermarking, Security, and Forensics 2013, edited by Adnan M. Alattar, Nasir D. Memon,
Chad D. Heitzenrater, Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 8665, 86650H
© 2013 SPIE-IS&T · CCC code: 0277-786X/13/$18 · doi: 10.1117/12.2005743

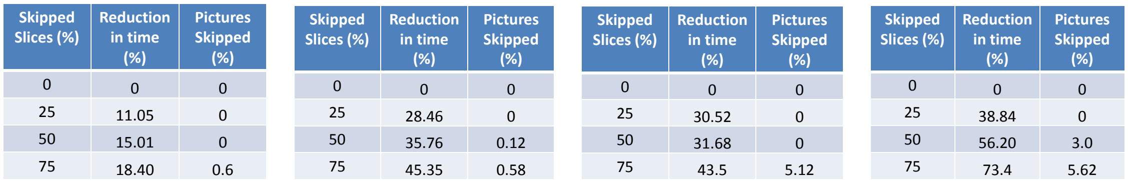
Video Carving Problem

In video carving, we extract video data from the disk to be analyzed and then further process it in a byte-by-byte fashion. Video data is a structured bitstream with the header bits defining the semantics of the bits that follow. Every byte of unallocated space has to be examined to identify video headers and construct video fragments. This is very time consuming if the size of the disk is large and many videos are stored on it. In the following discussion we present methods tested on MPEG-2 video, which can be easily extended to other video formats with minimal effort.

Fragment Detection

The first step of our approach is classifying the recovered data as video or non-video. If the data is non-video then we can totally ignore it. If it is video data, only then we further analyze it. Thus by not analyzing all of the recovered data we can save a significant amount of time.

To speed up fragment detection, we have used a smart skipping approach to skip some of the slice data in various pictures namely I pictures, P pictures and B pictures. This eliminates the need for byte-by-byte analysis and thus helps in reducing processing time. In this method, video headers are first identified using byte parsing. This step identifies video frames and sub-frames (slices). The size of the last parsed frame is used to estimate the size of the next frame and to determine the number of bytes to be skipped before repeating the byte parsing. While skipping byte processing accelerates video carving, it might lead to unidentified frames/fragments; this is a tradeoff that can be set according to the requirements of users. A user can decide the amount of data loss which is acceptable to him while at the same time avail a proportional decrease in the processing time required. Figure 1 shows the results of smart skipping. AS sh own in Figure 1.a, skipping in just Intra frames reduces the complexity by about 18% while missing the detection of 0.6 % of frames. I frames are a small portion of video (typically one every 15 frames) and complexity reduction is proportional. The gains in complexity with smart skipping applied to P, B, and all frames are shown in Figures 1.b, 1.c, and 1.d respectively. As the number of bytes skipped increases, processing time decreases, and this also results in some missed picture boundaries in a video. The results show that we can reduce the parsing time by almost 75% without a significant loss in frames detected.



Skipped Slices (%)	Reduction in time (%)	Pictures Skipped (%)
0	0	0
25	11.05	0
50	15.01	0
75	18.40	0.6

(a)

Skipped Slices (%)	Reduction in time (%)	Pictures Skipped (%)
0	0	0
25	28.46	0
50	35.76	0.12
75	45.35	0.58

(b)

Skipped Slices (%)	Reduction in time (%)	Pictures Skipped (%)
0	0	0
25	30.52	0
50	31.68	0
75	43.5	5.12

(c)

Skipped Slices (%)	Reduction in time (%)	Pictures Skipped (%)
0	0	0
25	38.84	0
50	56.20	3.0
75	73.4	5.62

(d)

Figure 1. Complexity reduction in fragment extraction

Fragment Recombination

Fragment recombination is essential for proper reconstruction of videos so that they can be used as evidence. Consider a scenario where there are N videos on a disk which are divided into M fragments (M>N). For a fragment of interest there are many possible fragments which can follow to it but there is only one right fragment. Identifying the correct fragment using exhaustive search method entails performing a continuity test (e.g., decodability using a standard decoder) for each pair of fragments in the set. This gives rise to a computational complexity of the order of N2. We have developed fast algorithms that reduce the number of fragment pairs to be evaluated for continuity.

Picture Type Probability

While the video coding standards do not prescribe any picture order, coding practices and compression requirements have resulted in specific coding patterns for picture types. Based on an analysis of a set of DVD videos, we have developed a model for the probabilities of picture types succeeding a picture of a given type. Figure 2 shows the probability of the next picture type given the current picture type. This will enable us to reduce the number of fragment pairs evaluated by examining the picture types of the frames at the end of one fragment and the beginning of another fragment. This also helps in eliminating false possibilities which may be the case in making a selection from seemingly similar fragments. One more advantage of this approach is that even if one or more fragments of a video are missing we can still reconstruct the video without much distortion

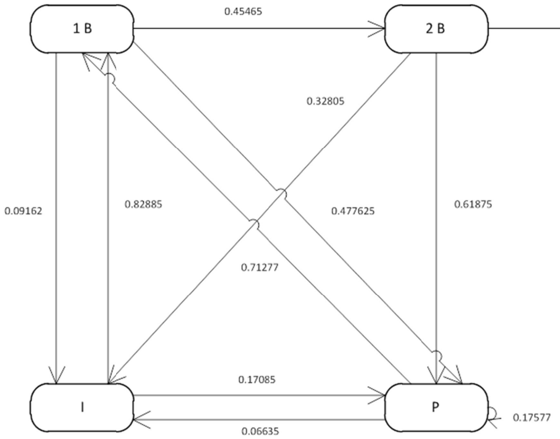


Figure 2. Picture type probabilities for typical video sequences

Picture Size Probability

A second approach towards reducing the number of fragment pairs evaluated is based on the average slice sizes of different picture types. This is based on the hypothesis that similar content will lead to similar picture sizes in compressed videos (except for fragments at shot boundaries). In this approach we limit evaluation to fragment pairs that have similar frame sizes at the fragment boundaries. Thus by combining frame type and frame size models, we can significantly reduce the fragment pairs to be evaluated for an overall reduction in video carving complexity.

Table 1: Complexity reduction in fragment recombination

Fragment #	Candidates based on picture type	Candidates based on picture type and average slice sizes
1	14	6
2	14	7
3	14	6
4	14	6
5	14	6
6	13	8
7	13	6
8	14	7
9	14	7

Table 1 shows the results of complexity reduction in fragment recombination. A data set with 35 video fragments from four video sequences was used in this evaluation. With brute force, we will have to evaluate all fragment pairs for continuity. With picture type probabilities, we can reduce the fragment pairs to be evaluated to 14 from the original 34. When we add frame size probabilities, the fragment pair candidates are reduced to around 7. This amounts to about 80% reduction in complexity compared to brute force method.

Conclusions and Future Work

We presented a new approach to accelerating video carving by exploiting the structure of video. MPEg-2 video was used in the experiments. Results show that the complexity of fragment detection and fragment combination cab be significantly reduced by employing the proposed methods. These methods can be adopted to other video formats for similar gains in complexity reduction.

As an extension of this work, we will employ higher level color analysis to further distinguish between several seemingly possible fragments in order to avoid making mistakes while reconstructing the video. This may be the case when there are a large number of videos on a disk and thus the set of possible fragments to choose from is very large.

References

[1] “Data Carving Concepts” SANS Institute InfoSec Reading Room. Last accessed on July 10th, 2012.

[2] A. Pal and N. Memon, "Automated reassembly of file fragmented images using greedy algorithms" in IEEE Transactions on Image processing, February 2006, pp 385-393.

[3] S. Garfinkel, “Carving contiguous and fragmented files with fast object validation,” in Proc. 2007 Digital Forensics Research Workshop (DFRWS), Pitts- burgh, PA, Aug. 2007, pp. 4S:2–12.

[4] A. Pal, K. Shanmugasundaram, and N. Memon, “Reassembling image fragments,” in Proc. ICASSP, Hong Kong, Apr. 2003, vol. 4, pp. IV–732-5.

[5] A. Pal, T. Sencar and N. Memon, "Detecting File Fragmentation Point Using Sequential Hypothesis Testing", Digital Investigations, Fall 2008.

[6] G. G. Richard, III and V. Roussev, “Scalpel: A frugal, high performance file carver,” in Proc. 2005 Digital Forensics Research Workshop (DFRWS), New Orleans, LA, Aug. 2005.

[7] M. McDaniel and M. Heydari, “Content based file type detection algorithms,” in Proc. 36th Annu. Hawaii Int. Conf. System Sciences (HICSS’03)—Track 9, IEEE Computer Society, Washington, D.C., 2003, p. 332.1

[8] Defraser, http://defraser.sourceforge.net/, Last accessed on December 2012.