How to make traffic surveillance video credible and authentic

Adam Stančić
Ultra d.o.o.
Research and Development Department
Gajeva 2, 47000 Karlovac, Croatia
adam.stancic@gmail.com

Ivan Grgurević
Faculty of Transport and Traffic Sciences
University of Zagreb
Vukelićeva 4, 10000 Zagreb
ivan.grgurevic@fpz.hr

Abstract. *The assurance of credibility and authenticity transforms the surveillance video record into a trustworthy source of data and information. Without these features the surveillance video can be interpreted only as a visual description of the situation at the observed location. This provides the possibility of research and utilization of the surveillance video record to all subjects who are involved in traffic surveillance and security.*

Keywords. Surveillance video, credibility, authenticity, cryptography, data manipulation

1 Introduction

The digital surveillance video system should be considered as one of the basic parts of Intelligent Transport Systems (ITS). Modern surveillance system is focused on the issues of surveillance and autonomous detection of disturbances in traffic, transport or logistic system. Surveillance video properties like credibility, authenticity, integrity, confidentiality and authorization rarely represent the point of interest inside traffic and transportation science community. Modern digital video surveillance cameras store the data in digital form in the various desired formats according to their final purpose. With today's technology it is easy to process the digital video and to obtain any desired effect with amazing precision and reality of the presentation which makes it suitable for the film art, but its drawback is very low level of authenticity. For this reason the traffic expert can use video as a source of data and information with great reserve. How can one be sure that nobody has manipulated the digital video and that what is displayed is precisely what the author of the recording wanted to display?

The implementation of the cryptographic protection can assure credibility and authenticity of the stored surveillance video data. Traffic surveillance video can be presented as a set of video frames. Each video frame is a still image which can be cryptographically processed i.e. digitally signed. All cryptographically processed data are stored in the database and can be presented, compared and searched easily in the desirable form. This paper will present the process of surveillance video manipulation and comparison between original and manipulated videos. The verification method is based on the comparison between original and manipulated video frame hash value. The best way to secure data is to digitally sign each surveillance camera video frame. Hash value calculation is just one step below the digital signing process (because digital signature is the encrypted hash value). The implementation of the digitally signed surveillance video should be considered as upgrade of the intelligent transport systems. The captured video is already stored on hard drives and the entire procedure could be accomplished inside the surveillance center. Today's level of development of the information and telecommunication system and application support allows fast processing of large amounts of data.

1.1 Aim of research

The aim of research is to demonstrate the possibility of detecting manipulation over the captured traffic surveillance video. The detection method must be efficient regardless of the amount of manipulation – even one pixel change in entire surveillance video must be detected. Moreover, the detection procedure must point to the (one or more) video frames whose content has been manipulated. The tools used must not be some expensive and professional application.

The aim is to guide any user who has a personal computer and access to the Internet on how to perform this procedure. This is the main reason why they are used as freeware, shareware and trial applications.

2 Surveillance video credibility

As mentioned previously in the text the traffic surveillance video can be used as a rich source of data and information for traffic experts. The digital store format of the surveillance video makes it very suitable for manipulation. That is the reason (possibly the main one) why in some cases surveillance video cannot be used as reliable evidence. According to the legislative point of view, any kind of the digitally stored data can be credible and authentic only if they have a digital signature. Surveillance video files require a large amount of memory and may not be unsuitable for digital signing. In some cases only smaller parts of videos are interested or even one frame. So, there are some questions that should be answered in this paper:

- How to make the digital surveillance video credible and authentic?
- How to detect manipulation over surveillance video?
- How to ensure credibility and authenticity of single video frames?
- How to detect a set of manipulated frames?

Possible methods of ensuring the surveillance video credibility and authenticity are presented in the previous work [1]. These methods strongly rely on cryptography. Each video frame could be observed as a single still image. Furthermore, each image (of the surveillance video) is digital document which could be digitally signed. All digital signatures can be stored in the database.

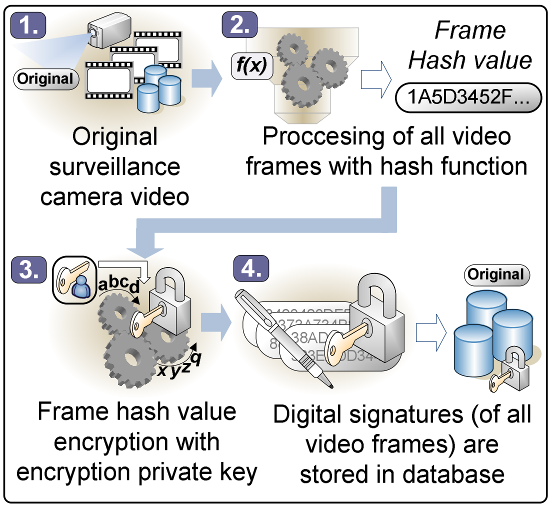

1.
Original
Original surveillance camera video
2.
f(x)
Frame Hash value
1A5D3452F...
Proccesing of all video frames with hash function
3.
abcd
xyzq
Frame hash value encryption with encryption private key
4.
Digital signatures (of all video frames) are stored in database
Original

Figure 1: Digital signing of the video frames

The process of verification of the traffic surveillance video is very similar to the process of digital signing of the video frames. The difference is the comparison process between two databases – database with original video frames hash values and database with copy of the video frames.

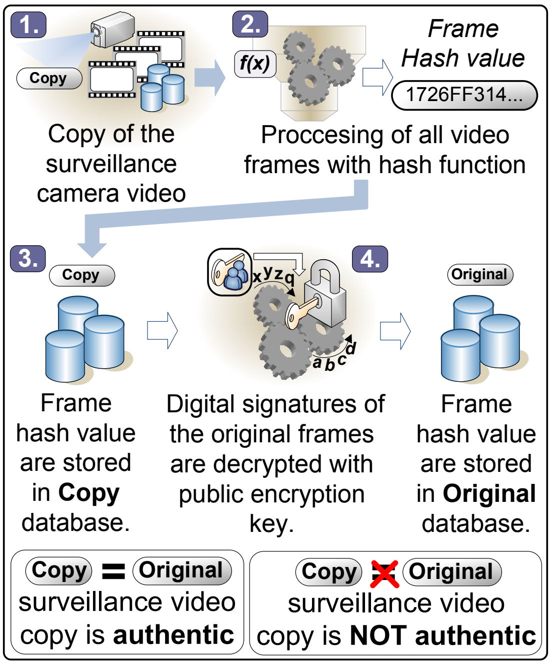

1.
Copy
Copy of the surveillance camera video
2.
f(x)
Frame Hash value
1726FF314...
Proccesing of all video frames with hash function
3.
Copy
Frame hash value are stored in Copy database.
xyzq
abcd
4.
Digital signatures of the original frames are decrypted with public encryption key.
Original
Frame hash value are stored in Original database.
Copy = Original
surveillance video copy is authentic
Copy ✗ Original
surveillance video copy is NOT authentic

Figure 2: Verification of the credibility and authenticity for copied surveillance video

Surveillance center pair of cryptographic keys and digital certificate is published by a certification authority (CA). Interested parties or systems can additionally check the keys and certificate credibility at CA [2]. Figure 3 displays such a situation:

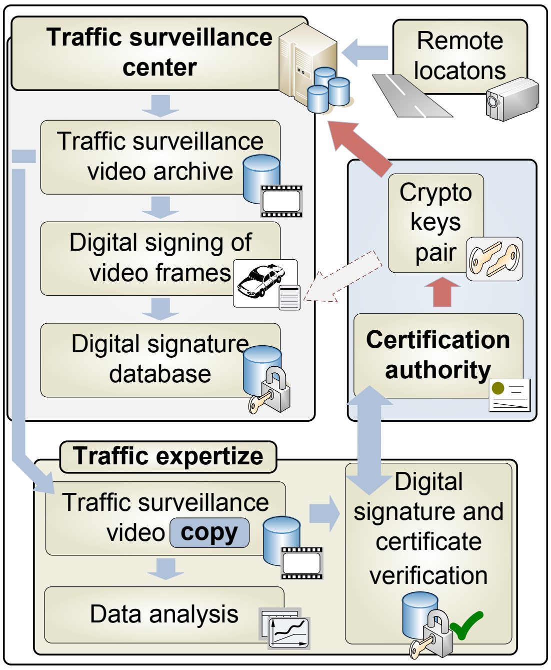

Traffic surveillance center
Remote locatons
Traffic surveillance video archive
Digital signing of video frames
Digital signature database
Crypto keys pair
Certification authority
Traffic expertize
Traffic surveillance video copy
Data analysis
Digital signature and certificate verification

Figure 3: Credibility and authenticity verification

3 Detection of manipulation over traffic surveillance video

The example of detection of manipulation over traffic surveillance video will use freeware, shareware and trial programs. User with the basic knowledge about Microsoft Windows or Linux operative system and (actual) personal computer should run this example. The example is divided into several steps:

- Surveillance camera video capturing;
- Surveillance camera video reproduction;
- Video frames extraction;
- Modification of the original video frames;
- Modified video frames embedding;
- Modified video frames extraction;
- Hash values calculations;
- Database creation and data importing;
- Hash values based table comparison;
- Query result analysis;
- Visual comparison between frames.

3.1 Example of video manipulation

The captured traffic surveillance video is stored at the surveillance video center. All video materials are stored in the digital format which is suitable for manipulation.

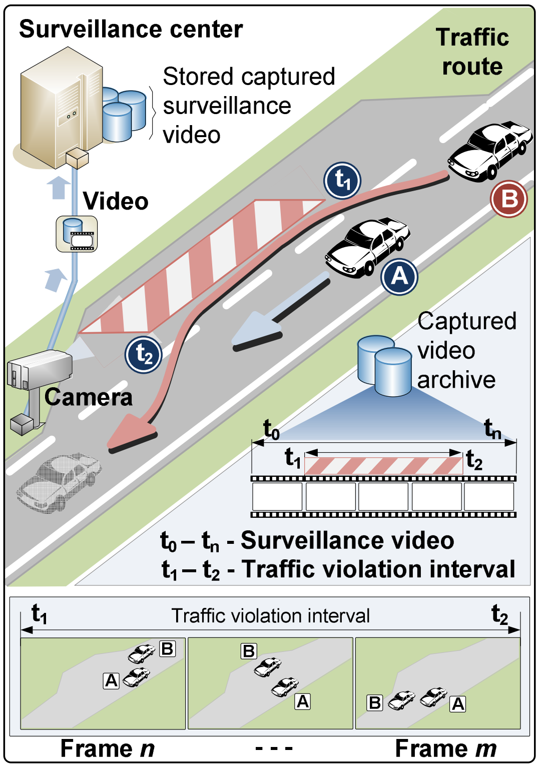


Figure 4: Surveillance video capturing

Figure 4 demonstrates a possible situation. A car (described as *B*) tries to overtake another car in front of it (described as *A*). Thus, car *B* must accelerate and bypass car *A* – to accomplish this goal car *B* must commit possibly two traffic violations! First (possible) violation is speeding and second (obvious) violation is overtaking car *A* from the right side. Car *B* did not violate the traffic laws all the time – it was only a few seconds in violation (described as *Traffic violation interval* $t_1 – t_2$). These few seconds can be observed as a few surveillance video frames (described as frame interval *n* - *m*). Let us suppose that someone tries to manipulate the captured surveillance video. For example, it would be sufficient to manipulate only a few seconds of the captured video. Such short manipulation will be very hard to notice by the visual method. Next example will show a similar situation: original surveillance video was manipulated for a very short time. The main questions are: how to detect the manipulation and at what moment was the video manipulated?

Surveillance camera video capturing > Surveillance video camera system is set on a highway. The camera is oriented towards the highway. These are the basic facts about the captured surveillance video:

Table 1: Captured surveillance video properties

General properties – original video	
File name	Survideo.mpg
Size	232.906 KB ~ 227,5 MB
Length	00:03:35 = 215 sec
Demuxer	MPEG* PS demuxer
MD5**	14ad91948d21fa4949f7bb66cab030d0
Video properties – original video	
Resolution	720 x 480 pixel
Asp. ratio	1.3333
Format	0x10000002
Compres.	Lower Field First VBR
Bitrate	8.000 kbps
Frames/sec	29.970 ~ 30
Codec	ffmpeg2 [mpeg2video]

*- MPEG - *Moving Picture Experts Group* [3]
**- MD5 - *Message-Digest algorithm 5* [4]

Surveillance camera video reproduction >
Application *SMPlayer*[1] (freeware version: 0.6.7 - SVN r2831) and is used for the purpose of playing and retrieving basic information about the video. *SMPlayer* is GUI (*Graphic User Interface*) for another command-line component - *MPlayer* (version: SVN r28311).

[1] SMPlayer © 2006-2009 Ricardo Villalba [http://smplayer.sourceforge.net/]

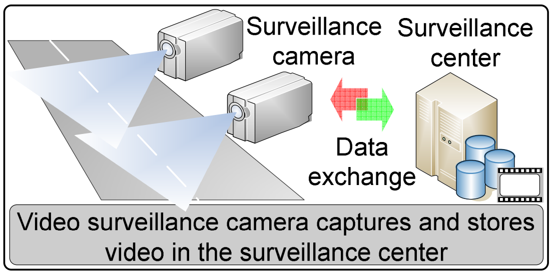

Surveillance camera
Surveillance center
Data exchange
Video surveillance camera captures and stores video in the surveillance center

Figure 5: Captured video storing and archiving

Video frames extraction > The next step is extraction of the video frames from the captured video. Every video frame will be stored in the form of a still image in *jpeg* (*Joint Photographic Experts Group*) format [5].

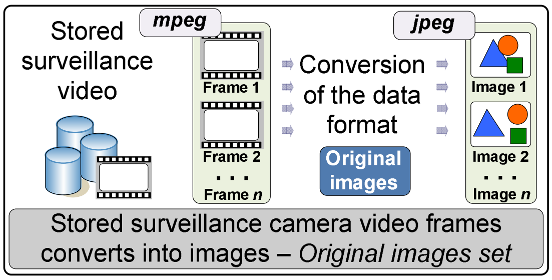

Stored surveillance video
mpeg
Frame 1
Frame 2
. . .
Frame n
Conversion of the data format
Original images
jpeg
Image 1
Image 2
. . .
Image n
Stored surveillance camera video frames converts into images – Original images set

Figure 6: Video frames extraction

For this operation the application *MPlayer* will be used. The input in the command-line (or console) should be:

```
mplayer Survideo.mpg -vo jpeg
-nosound
```

This means that the *MPlayer* should play the captured video, but using as output still images which are saved (in *jpeg* format) on the hard drive. Switch – nosound releases CPU (*Central Processing Unit*) resources from the audio reproduction. Every second of the captured video has 30 video frames; hence, from the entire captured video application a total of 6,468 frames is extracted. It is recommended that the frames should be extracted into a separate directory i.e. *Original_frames*.

Table 2. Extracted video frames properties

First frame name	00000001.jpg
Last frame name	00006468.jpg
Total number of frames	6468
Duration of one frame	1/30 second
Duration of all frames	215,60 seconds
First frame timeline posit.	0,00 sec. of video
Last frame timeline posit.	215,56 sec. of video
Compression	JPEG
Frame size	720 x 480 pixel
Frame resolution (DPI)	72 x 64 pixel
Max. number of colors	24-bit - 16,7 mil
Total file size (all frames)	144.654.303,00 bytes
Mean file size	22.364,61 bytes

Modification of the original video frames > The next step is modification of the extracted video frames. For this purpose 60 frames are modified with the image manipulation program - GIMP[2] (freeware version 2.6.6.). Sixty frames equal 2 seconds of surveillance video (60 frames / 30 frames per second). Sixty modified frames among 6,468 frames are 0.9276% of all the extracted surveillance video frames. The modification of a few surveillance video frames can be obvious, but in the video file there is a large number of frames. It is very hard to detect such a small change by visual inspection. One of the aims of the presented method is how to find the location (frame number or video play time) where manipulation begins and where it ends.

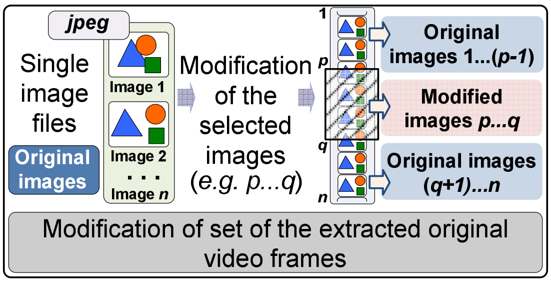

jpeg
Single image files
Original images
Image 1
Image 2
. . .
Image n
Modification of the selected images (e.g. p...q)
1
p
q
n
Original images 1...(p-1)
Modified images p...q
Original images (q+1)...n
Modification of set of the extracted original video frames

Figure 7: Video frames modification

Manipulation of a surveillance video file can be detected easily because the file size or MD5 hash value are changed. A more complicated problem is what kind and where the surveillance video has been processed? Visual inspection for such a short video (215 seconds) may represent an arduous job.

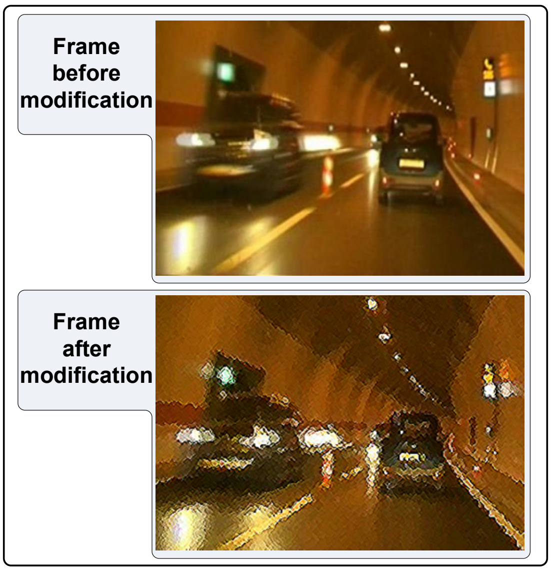


Figure 8: Manipulation over frame

[2] GIMP © 1995-2008 S. Kimball, P. Mattis and the GIMP Development Team [http://www.gimp.org/]

Surveillance video frame modification will be performed by using *Noise* → *Slur*[3] filter which is standard part of the GIMP application. *Slur* filter parameters are set to the following values: *Random speed* = 10, *Randomization* (%) = 50 and *Repeat* = 15. Example in Figure 8 depicts a single frame manipulation. Strong manipulation over frame i.e. the image becomes completely dark, can be visually detected, but partial manipulation is very hard to notice. That is, hiding of the registration plate can be accomplished by a minor frame manipulation. After modification of all 60 consecutive video frames (saved in jpeg image format) it is time for next stage in this example.

Modified video frames embedding > Embedding of the modified video frames inside the original surveillance video requires video editing application: *Ulead VideoStudio*[4] (trial version 9.00). In video edit part of application, the entire captured surveillance video is reproduced as a chain of video frames. Each frame is 1/30 second long, so that 30 consecutive frames in chain are one second of the video.

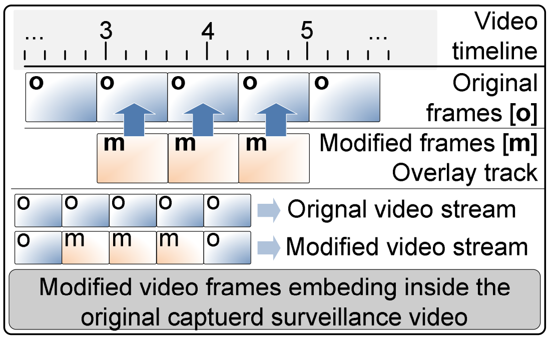


Figure 9: Modified video frames embedding

Modified and original frames are equally long and the previously modified frames must be inserted at the same place as the original video frames. This "position" on the time line represents the moment when the modified frames will overlay the original frames (and deceive the viewer that they are original frames). Accurate frame embedding operation can be very hard in some video editing application. The GUI application uses the mouse drag-and-drop methods which can be sometimes very inconvenient. Depending on the computer speed, available memory, inserted image size and video properties, the embedding operation may take some time. For this example the embedding time was only 110 seconds. Modified surveillance video will be saved in the same format as the original surveillance video. Table 3 presents the inserted modified video frames properties.

Table 3: Inserted modified video frames properties

First frame name	00000750.jpg
Last frame name	00000810.jpg
Total number of frames	60
Duration of one frame	1/30 second
Duration of all frames	2 seconds
First frame timeline position	25,000 sec. of video
Last frame timeline position	27,000 sec. of video
Compression	JPEG
Frame size	720 x 480 pixel
Frame resolution (DPI)	72 x 64 pixel
Max. number of colors	24-bit - 16,7 mil
Total file size (all frames)	2.703.592,00 bytes
Mean file size	45.059,866 bytes

Table 4 shows the basic facts about the modified captured surveillance video.

Table 4: Modified surveillance video properties

General properties – modified video	
File name	Survideo_modified.mpg
Size	232.858 KB ~ 227,5 MB
Length	00:03:35 = 215 sec
Demuxer	MPEG PS demuxer
MD5	b91b12f9930ff036a47fba37f7f40212
Video properties – modified video	
Resolution	720 x 480 pixel
Asp. ratio	1.3333
Format	0x10000002
Compress.	Lower Field First VBR
Bitrate	8.000 kbps
Frames/sec	29.970 ~ 30
Codec	ffmpeg2 [mpeg2video]

After embedding the manipulated frames in the surveillance video and processing, the video editing application saves the modified surveillance video file. The differences between the original and the modified video are presented in Table 5:

Table 5: Differences between orig. and modif. video

Original surveillance video	
Size	232.906 KB
MD5	14ad91948d21fa4949f7bb66cab030d0
Modified surveillance video	
Size	232.858 KB
MD5	b91b12f9930ff036a47fba37f7f40212

Differences between the original and the modified surveillance video file is size 48 KB (original is larger) and MD5 hash value. It is obvious that the original video file has been modified but there is no indication about the kind of manipulation, where and how it has been modified.

Modified video frames extraction > The next step is the extraction of the video frames from the previously modified surveillance video. Every video frame will be stored in the form of a still image in *jpeg* format.

[3] GIMP - Slur filter [http://docs.gimp.org/en/plug-in-randomize-slur.html]

[4] Ulead VideoStudio [http://www.ulead.com/products/runme.htm#video]

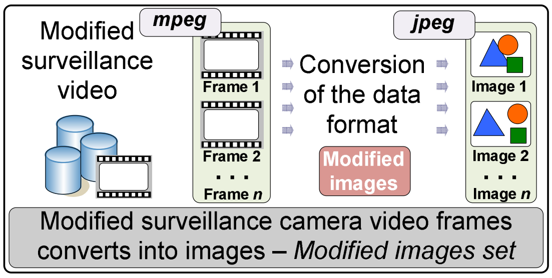

Modified surveillance video
mpeg
Frame 1
Frame 2
. . .
Frame n
Conversion of the data format
Modified images
jpeg
Image 1
Image 2
. . .
Image n
Modified surveillance camera video frames converts into images – Modified images set

Figure 10: Modified video frames extraction

For this operation the application *MPlayer* will be used. The input in the command-line (or console) should be:

```
mplayer Survideo_modified -vo jpeg
-nosound
```

Every second of the modified surveillance video has 30 video frames; hence, from the entire captured video the application extracts a total 6,468 frames. It is recommended to extract the frames into a separate directory i.e. *Modified_frames*.

Hash values calculations > Hash MD5 value calculation from a large number of files requires another application: *HashMyFiles*[5] (freeware version 1.43). The application calculates hash values (MD5, SHA-1, and CRC) and extracts information from the file properties for each file in the selected directory.

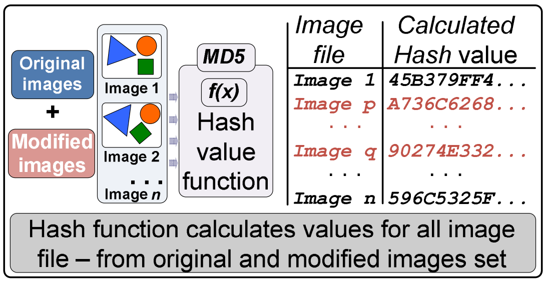

Original images
+
Modified images
Image 1
Image 2
. . .
Image n
MD5
f(x)
Hash value function
Hash function calculates values for all image file – from original and modified images set

Image file	Calculated Hash value
Image 1	45B379FF4...
Image p	A736C6268...
. . .	. . .
Image q	90274E332...
. . .	. . .
Image n	596C5325F...

Figure 11: Hash values calculations

In these example directories *Original_frames* and *Modified_frames* must be selected. All the calculated and extracted information can be saved in tab delimited text file. Tab delimited text file is appropriate for data export in SQL database. Hash values from both directories *Original_frames* and *Modified_frames* are saved in a separate tab delimited text file.

Database creation and data importing > All hash values (and other information if needed) are stored in SQL database. For this example *MySQL*[6] (freeware version *Windows Essentials* (x86) 5.1.34) database server is used. The application *SQL Manager 2007 Lite for MySQL*[7] (freeware version 4.4.2.1) is used as the client tool for *MySQL* server database.

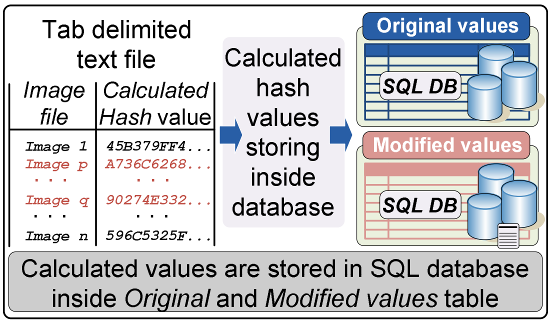
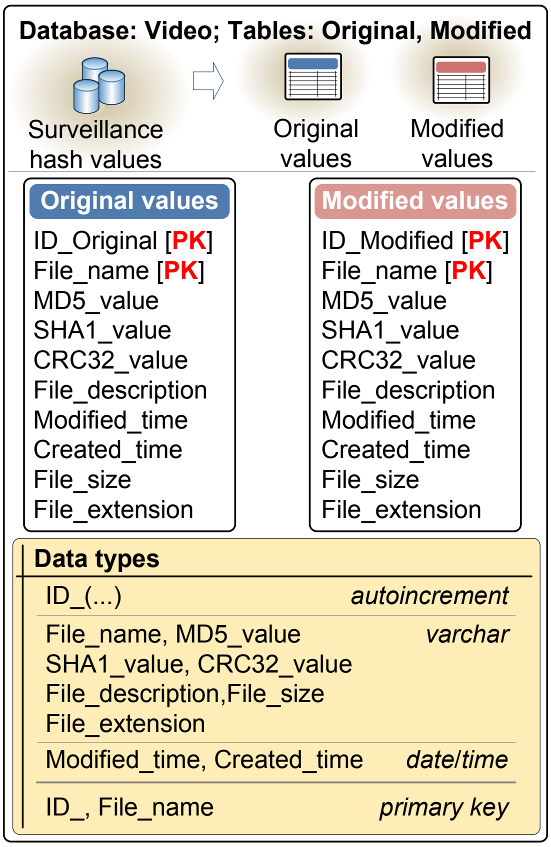

Tab delimited text file
Calculated hash values storing inside database
Original values
SQL DB
Modified values
SQL DB
Calculated values are stored in SQL database inside Original and Modified values table

Image file	Calculated Hash value
Image 1	45B379FF4...
Image p	A736C6268...
. . .	. . .
Image q	90274E332...
. . .	. . .
Image n	596C5325F...

Figure 12: Hash values database importation

After user login on the database server, the first task is the creation of the database named *Video*, and two database tables: *Original_values* and *Modified_values*.



Original values	Modified values
ID_Original [PK]	ID_Modified [PK]
File_name [PK]	File_name [PK]
MD5_value	MD5_value
SHA1_value	SHA1_value
CRC32_value	CRC32_value
File_description	File_description
Modified_time	Modified_time
Created_time	Created_time
File_size	File_size
File_extension	File_extension

Data types	
ID_(...)	*autoincrement*
File_name, MD5_value SHA1_value, CRC32_value File_description,File_size File_extension	*varchar*
Modified_time, Created_time	*date/time*
ID_, File_name	*primary key*

Figure 13: Surveillance video database structure

Table metadata are shown in Figure 13. In the table *Original_values* and *Modified_values* corresponding tab delimited text files are imported. (*Original_frames.txt* file is imported in *Original_values* database table and *Modified_frames.txt* file is imported in *Modified_values* database table).

[5] © 2007 - 2009 Nir Sofer [http://www.nirsoft.net]

[6] © 1995-2008 MySQL AB, 2008-2009 Sun Microsystems, Inc. [http://www.mysql.com/]

[7] © 1999-2008 EMS Database Management Solutions [http://www.sqlmanager.net/]

Hash values based table comparison > After data insertion it is time to compare the original and the modified data. The comparison is based on the difference between video frames MD5 hash values in original and modified tables. If the MD5 values are equal in both tables then there is no manipulation detected over the video frames and vice versa – the difference in MD5 hash values indicates manipulation over video frames.

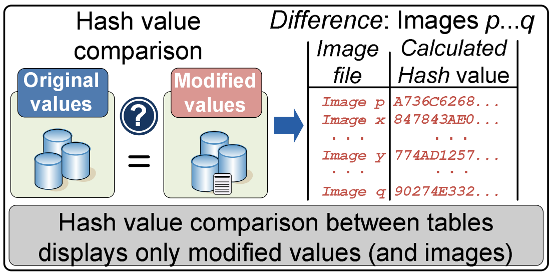


Figure 14: Table comparison

The SQL (*Structured Query Language*) which searches for the video frames with different MD5 hash value may look like this:

```
SELECT O.ID_Original, M.ID_Modified,
O.File_name, M.File_name, O.MD5_value,
M.MD5_value, O.SHA1_value,
M.SHA1_value, O.CRC32_value,
M.CRC32_value, O.File_description,
M.File_description, O.Modified_time,
M.Modified_time, O.Created_time,
M.Created_time, O.File_size,
M.File_size, O.File_extension,
M.File_extension
FROM Original_values O
INNER JOIN Modified_values M ON
O.File_name = M.File_name
WHERE O.MD5_value <> M.MD5_value
```

After a few seconds of processing the *MySQL* client application displays a table with the result. The result table consists of all the values from the tables of *Original_values* and *Modified_values*. Similar columns from both tables are aligned in such a way as to ease the reading and query result analysis.

Table 6: Query result for video frames properties

First frame name	00000750.jpg
Last frame name	00000827.jpg
Total number of frames	77
Duration of one frame	1/30 second
Duration of all frames	2,56 seconds
First frame timeline position	25,000 sec. of video
Last frame timeline position	27,566 sec. of video
Compression	JPEG
Frame size	720 x 480 pixel
Frame resolution (DPI)	72 x 64 pixel
Max. number of colors	24-bit - 16,7 mil
Total file size (all frames)	3.042.232,00 bytes
Mean file size	39.509,50 bytes

Query result analysis > The query results have slight deflection from what is expected. In the previous steps there are 60 modified video frames, but after extraction and comparison, the database reports 77 frames with modified MD5 value. The first detected modified frame name is 00000750.jpg - which is exactly the first modified frame embedded in the surveillance video. The last detected modified frame name is 00000827.jpg, but this is not what was expected! Based on the last modified video frame name, it was expected that frame 00000810.jpg should be the last one with different MD5 hash value. Modified frames from 00000811.jpg to 00000827.jpg are out of (the expected) range. A possible reason is the MPEG storage format.Video compression formats (like MPEG) store only the changes which occur from one video frame to the next video frame. In some situations the amount of changes between consecutive frames is very low so that the compression technique reduces the amount of data that must be stored in the video file. If the scene changes drastically then the complete frame image is stored in the video data stream. The next frame from this (completely created) frame stores only the change between two of them. The frame in which a complete image is stored is the so-called key frame. It is possible that MD5 hash values are changed between two consecutive key frames.

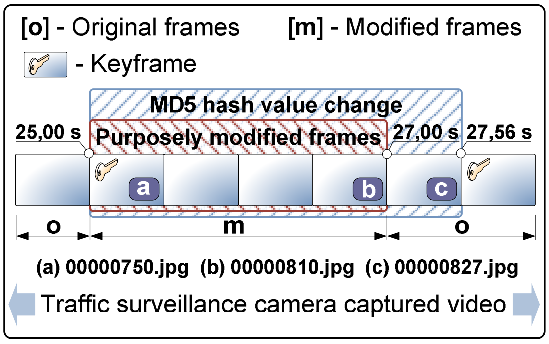

[o] - Original frames
[m] - Modified frames
- Keyframe
MD5 hash value change
Purposely modified frames
25,00 s
27,00 s
27,56 s
a
b
c
o
m
o
(a) 00000750.jpg (b) 00000810.jpg (c) 00000827.jpg
Traffic surveillance camera captured video

Figure 15: Modified MD5 hash values position

Visual comparison between pictures > Additional method of comparison and detection of the manipulated video frames can be the usage of visual inspection application.

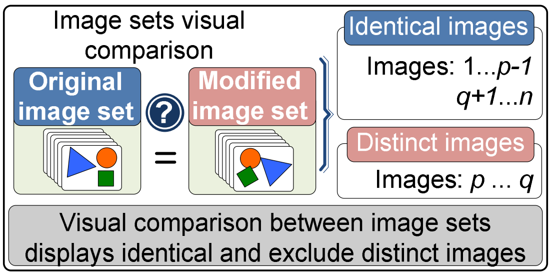


Image sets visual comparison
Original image set
?
=
Modified image set
Identical images
Images: 1...p-1
q+1...n
Distinct images
Images: p ... q
Visual comparison between image sets displays identical and exclude distinct images

Figure 16: Visual comparison applications

This sort of application is primarily used for help in maintaining large photo galleries. Duplicate pictures in photo galleries should be avoided and this is a task for this sort of application. The approach is diametrically opposed – this application will display duplicates, not differences between surveillance video frame sets.

4. Conclusions

In the example of detecting manipulation over traffic surveillance video, freeware, shareware and trial applications are used. All the chosen applications, except video editing, can be executed on today's most popular operative systems: *Linux* and *Microsoft Windows.* The aim is to provide an experimental method for the exploration of traffic surveillance video credibility and authenticity. The example is divided into several consecutive sections. The aim is to find an answer to two questions: is there a way to detect manipulation over surveillance video and how to point out which part of video stream has been modified. The result of the experiment gives the answer that it is possible to prove manipulation and detect video frames set where the change (from the original video) has occurred.

Acknowledgments

[1] **SMPlayer**, application documentation, application available at `http://www. smplayer.sourceforge.net/` Accessed: 27th April 2009.

[2] **MPlayer**, application documentation, web documentation available at `http://www.mplayerhq.hu/DOCS/HTML-single/en/MPlayer.html` Accessed: 27th April 2009.

[3] **GIMP - Slur filter**, application documentation, web documentation available at `http://docs.gimp.org/en/plug-in-randomize-slur.html` Accessed: 27th April 2009.

[4] **Ulead VideoStudio**, application documentation, application available at `http://www.ulead.com/products/runme.htm#video` Accessed: 27th April 2009.

[5] **HashMyFiles**, application documentation, application available at `http://www.nirsoft.net/utils/hash_my_files.html` Accessed: 27th April 2009.

[6] **MySQL server**, application documentation, web documentation available at `http://dev.mysql.com/docs/` Accessed: 27th April 2009.

[7] **EMS SQL Management Studio for MySQL server**, application documentation, web documentation available at `http://www.sqlmanager.net/en/products/studio/mysql/documentation` Accessed: 27th April 2009.

References

[1] Grgurević I., Stančić A., Škorput P.: **Credibility and authenticity of digitally signed videos in traffic**, Promet - Traffic&Transportation, Scientific Journal on Traffic and Transportation Research, (0353-5320) 20, Zagreb, Croatia, 2008, pp. 405-414

[2] Narodne Novine, službeni list RH (*Official Gazette*) No. 10/2002: **Zakon o elektroničkom potpisu**, available at `http://narodne-novine.nn.hr/default.aspx` Accessed: 27th April 2009.

[3] Electronics & Communication Engineering Journal, Tudor P.N.: **MPEG-2 video compression**, available at `http://www.bbc.co.uk/rd/pubs/papers/paper_14/paper_14.shmtl` Accessed: 27th April 2009.

[4] Network Working Group – Request for Comments: 1321: ***The MD5 Message-Digest Algorithm,*** available at `http://tools.ietf.org/html/rfc1321` Accessed: 27th April 2009.

[5] World Wide Web Consortium – Graphics on the Web, Hamilton E.: **JPEG file interchange format version 1.02**, available at `http://www.w3.org/Graphics/JPEG/jfif3.pdf` Accessed: 27th April 2009.