2016 2nd International Conference on Next Generation Computing Technologies (NGCT-2016)
Dehradun, India 14-16 October 2016

Storage Optimization of Video Surveillance from CCTV Camera

Shikhar Arora
University of Petroleum and Energy Studies,Dehradun, India
hi.shikhararora@gmail.com

Karan Bhatia
University of Petroleum and Energy Studies, Dehradun, India
karan_bhatia98@yahoo.com

Amit V
Assistant Professor,
University of Petroleum and Energy Studies, Dehradun, India
amit.verma@ddn.upes.ac.in

***Abstract* — Closed-circuit television (CCTV) or video surveillance is the most useful technology mostly used in the field of security purposes. CCTVs can be found at many places ranging from public to private places. One of the most challenging problem in installing the CCTV cameras at large scale is storage space occupied by the footage. Footage is mostly stored in secondary storage devices like hard disk drives. So, to reduce the storage space, compression techniques are applied. The proposed idea is to remove the adjacent redundant frames. We are proposing a method to optimize the storage space occupied by the CCTV footage by deleting the redundant frames by comparing the adjacent frames using MSE (Mean Squared Error) between the adjacent frames of the video clip. The approach will optimize the storage maintaining the information as well as quality of the video clip.**

***Keywords : CCTV, redundant, MSE*.**

I. INTRODUCTION

CCTV also known as video surveillance, uses video cameras to transmit a signal to a specific place or to a limited set of display devices like monitors. It is different from broadcast television as the signal is not openly transmitted, it may employ P2P or point to point, point to multipoint, or mesh wireless links. Though almost all video cameras fit this definition, the term is most often applied to those used for surveillance in areas that may need monitoring such as ATMs, airports, banks, casinos, military installations, convenience stores and even homes.

In industrial plants, CCTV equipment may be used to observe parts of a process from a surveillance room or central control room, for example when the environment is not suitable for humans. CCTV systems may operate continuously or only as required to monitor a particular event. More recently, decentralized IP cameras, some equipped with megapixel sensors, support recording directly to NAS or network-attached storage devices, or internal flash for completely stand-alone operation. Surveillance of the public using CCTV is particularly common in many areas around the world. In recent years, the use of body worn video cameras has been introduced as a new form of surveillance.

CCTV cameras are basic requirement for security of any place, but it requires high maintenance cost of upgrading memory after a certain period of time. The optimization of video size by removing redundant frames reults in same information stored in less disk space. A video at its lowest quality generated from 1 CCTV camera, which is generally recorded at 352X240 resolution and 30 fps occupies 10 GBs of storage in 1 day and there are 245 million CCTV cameras in the world right now running 24X7 capturing every second of year. And storage for data is huge problem, some cameras even auto delete all the footage after certain time period.

II. RELATED WORKS

According to survey done by IHS Technologies [3] on Video Surveillance and Storage in September of 2014, there are 245 million video surveillance cameras installed globally in 2014. Out of which 71% are from Asia. And a low quality video on a CCTV camera is generally recorded at 352 X 240 resolution and 30 fps which occupies 1TB of storage in 138 days. In a Technical Review Paper submitted by Seagate [2] in 2012, it shows that in compression techniques done to store the CCTV footage usually results in loss of quality and does not affect much on size of video. So, they suggested to use Backup plus option by Seagate. A Combined DWT-DCT approach to perform Video compression base of Frame Redundancy published in International Journal of Advanced Research in Computer Science and Software Engineering : ISSN: 2277 128X.[1] In recent researches done on this topic, suggest to use the cloud storage for each CCTV footage. But this requires fast internet connection in every CCTV cameras. So, it is not feasible in coming years.

III. EXISTING SYSTEMS

There is no physical product in this area, but there is heavy research going on in this topic since 2012.

Seagate published a technical research paper “Video Surveillance Storage: How Much Is Enough?” in 2012, it shows that in compression techniques done to store the CCTV footage usually results in loss of quality and does not affect much on size of video. So, they suggested to use Backup plus option by Seagate.

And other research done includes primary and secondary storage systems for CCTV. Primary includes the storage device connected to camera and secondary is used for backup which is server or cloud. Secondary can be more than one. Data from primary is compressed and stored in secondary and after that primary is cleaned and indexed to secondary storage. In this data is not lost as most of the CCTV cameras are used for security purposes, so loss in data is not something that should happen.

In recent researches done on this topic, they suggested to use the cloud storage for each CCTV footage. But this requires fast internet connection in every CCTV cameras. So, it is not feasible in India in coming years.

IV. Proposed Method

The main objective is to propose a method to optimize the size on disk of video clip obtained from a CCTV camera. In this work, Python is being used as platform for implementing various image processing and computer vision techniques. Firstly, frames are extracted from a CCTV footage and MSE(Mean Square Error) is used to compare adjacent frames of the video and then the remaining frames are combined at the same frame rate to get the final compressed video. Now, take a look at each module in depth in next sections.

V. Methodology

The project follows an Incremental Development Methodology. Multiple development cycles would be taking place here, making the life cycle a "multi-waterfall" cycle. Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases.

The main advantage of this kind methodology is that one would be able to use the software or rather the program during its build phase and could improve the design as well as alter the source code during that stage without the need of performing each step individually.

This model is more flexible – less costly to change scope and requirements and it is easier to test and debug during a smaller iteration.

VI. Modules

The project consists of 3 modules :

A. Extracting frames from video

B. Comparing adjacent frames of video

1) *Comparing MSE of adjacent frames*
2) *Deleting the frames having MSE < Threshold*

C. Combining the remaining non-similar frames into video

Flow Chart

Fig. 1. **Flow Chart**

Data Flow Diagram

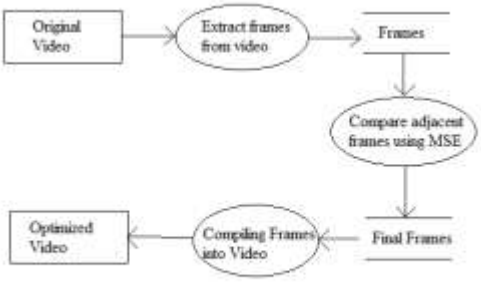

Fig. 2. **Data Flow Diagram**

VII. Proposed Algorithm

Step 1: Input CCTV footage
Step 2: Extract frames from video
Step 3: Calculate the MSE value for key frame and test frame
Step 4: Set key frame = 1st frame and test frame = 2nd frame
Step 5: if MSE >= Threshold, then
key frame = test frame
test frame = key frame + 1
save the key frame
else
test frame = test frame + 1
Step 6: if there are more frames
Repeat Step 5
else
Combine the resulted frame to get optimized video with same fps
Delete original video and frames of both videos

VIII. Analysis Of Each Module

Module 1 : Extracting frames from Video

To extract frame from a video, video is paused at every frame using cv2.waitKey() and that frame is saved as an image cv2.imwrite()

```
1   import cv2
2
3   vc=cv2.VideoCapture('D:\\PROJECT\\
    test.mp4')
4   c=1
5
6   if vc.isOpened():
7       rval , frame = vc.read()
8
9   else:
10      rval=False
11      print("error")
12
13  while rval:
14      rval, frame = vc.read()
15  cv2.imwrite("D:\\frames\\" +
    str(c) + '.jpg',frame)
16  c = c + 1
17  cv2.waitKey(1)
18
19  vc.release()
```

Module 2 : Comparing adjacent frames of video

First MSE of each frame is computed and then then the threshold is judged through recursive testing for the footage in the environment and the threshold for the video we used comes out to be 5.

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

where,
m : no. of horizontal pixels (width of frame)
n : no. of vertical pixels (length of frame)
I, K : frame
I(i, j) : pixel intensity value at the co-ordinate (i,j)

If MSE = 0, then the frames are 100% same and if MSE = 2, the difference between frames are not visible to human eye. And in our video there is a timestamp, so after many test cases, MSE turn out to be 5.

Threshold is judged through recursive testing for the footage and the threshold for the video we used comes out to be 5. And then the frame is taken as a key and MSE is calculated of that frame and its adjacent frame named test frame, if the MSE is less than 5, then the test frame is discarded and if MSE is greater than 5 then the test frame is saved.

We have also used other Algorithms like SSIM(Structural Similarity), Histogram, Wavelet Transform but MSE(Mean Square Error) is the most precise and feasible. So, MSE is used in the project to compare frames.

```
def mse(imageA, imageB):
    err=np.sum((imageA.astype("float") -
       imageB.astype("float"))**2)
    err /=
       float(imageA.shape[0]*imageA.shape[1])
    return err
```

MSE Function in Python

```
z = 1
k = 1
n = 2
nf = 29168

for i in range(0, nf-1):
#while True:
    a=cv2.imread("frames\\"+str(k)+".jpg",0)
    b=cv2.imread("frames\\"+str(n)+".jpg",0)

    m = compare_images(a,b)
    if m >= 5 :
     cv2.imwrite("D:\\final_output_mse_5\\"+
       str(z) +".jpg" , a)
     z = z+1
     print(k,n,m)
     k = n
     n = k+1

    else :
     print(k,n,m)
     n = n+1
```

Module 3 : Combining the remaining non-similar frames into video

VideoWriter() class in python is used to capture a video, process it frame-by-frame and constructors and methods to save that video. To read frame cv2.imread() is used. Output Video is in greyscale as all the computations are done using greyscale as pixel to noise ratio is lowest in greyscale. And features of images are clearly visible and easy to distinguish in greyscale. After the optimized video is generated from resulted frames the input video and the processing data that is frames of both videos (input and optimized) are deleted.

IX. Experimental Results

This work is implemented using Python 2.7.11. The objective of this work is to optimize the storage space occupied by CCTV footage based on redundancy of adjacent frames.

In this work, the MP4 file has been set as input video as most of the CCTV footage is recorded in .mp4 format. The proposed work is beneficial in places with less density and fewer movement. A video having maximum frame rate will give the better compression ratio.

The minimum hardware requires CPU to be clocked at 1.8 – 2.3 GHz Intel® Core™ i3 processor with atleast 4GB DDR3 RAM and with HDD having more than 5 GB free for processing 500MB of video. These 5 GB will be cleared after the video is optimized.

We performed this work on input test video of duration 20 minutes, size of 110 MB, having resolution 640 X 480 and 25 frames per second.

After compression techniques are applied, the output video obtained is of duration 7 minutes, size of 76 MB, having resolution 640 X 480 and 25 frames per second.

The compression in duration of video is 65% and in size is 30.91%

X. Conclusion and Future Works

An algorithm is design for storage optimization especially for Closed-Circuit Television (CCTV) as storage is a real challenge with increasing market demand for the third eye at almost every specific place in the city.

With the same motive an algorithm is given the paper which is giving satisfactory results by reducing average space of any partially dense area footage by nearly 31%. Still problem is open to welcome better results.

- This algorithm can be implemented in a CCTV camera, which includes no connectivity to internet.
- Our method works great with low dense areas, a better method to go with high density can be done.
- Face recognition, edge detection techniques like Sobel etc. can be used to increase the efficiency.
- Images can be converted to blurred to get better pixels' values.
- Arduino and Raspberry pie can be used for the implementation of these algorithms into CCTV camera.

References

[1] Jasmeet Kaur, ACombined DWT-DCT approach to perform Video compression base of Frame Redundancy Volume 2, Issue 9, September 2012 International Journal of Advanced Research in Computer Science and Software Engineering, pp-325-332

[2] Seagate, Video Surveillance Storage: How Much is Enough? White Paper, February 2012
http://www.seagate.com/files/staticfiles/docs/pdf/whitepaper/video-surv-storage-tp571-3-1202-us.pdf

[3] Video Surveillance and Storage Survey by IHS Technology. September 2014
https://www.emc.com/collateral/analyst-reports/ihs-video-surveillance-storage-intersection-it-physical-security.pdf

[4] Xiaosong Wang, Xinyuan Huang, Hui Fu A Color-texture Segmentation Method to Extract Tree Image in Complex Scene, International Conference on Machine Vision and Human-machine Interface, 2010, pp 621-625.

[5] T.Chiang and Y.-Q. Zhang, A new rate control scheme using quadratic rate distortion model, IEEE Trans. Circuits Syst. Video Technololgy, Feb.1997