Hindawi
Journal of Electrical and Computer Engineering
Volume 2018, Article ID 3839104, 10 pages
https://doi.org/10.1155/2018/3839104

Research Article

A Retrieval Optimized Surveillance Video Storage System for Campus Application Scenarios

Shengcheng Ma,[1] **Xin Chen**,[1] **Zhuo Li**,[2] **and Yingjie Yang**[1]

[1]*School of Computer Science, Beijing Information Science and Technology University, Beijing, China*
[2]*Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, School of Computer Science, Beijing Information Science and Technology University, Beijing, China*

Correspondence should be addressed to Xin Chen; chenxin@bistu.edu.cn

Received 6 November 2017; Revised 30 January 2018; Accepted 19 February 2018; Published 8 April 2018

Academic Editor: Attila Kertesz

This paper investigates and analyzes the characteristics of video data and puts forward a campus surveillance video storage system with the university campus as the specific application environment. Aiming at the challenge that the content-based video retrieval response time is too long, the key-frame index subsystem is designed. The key frame of the video can reflect the main content of the video. Extracted from the video, key frames are associated with the metadata information to establish the storage index. The key-frame index is used in lookup operations while querying. This method can greatly reduce the amount of video data reading and effectively improves the query's efficiency. From the above, we model the storage system by a stochastic Petri net (SPN) and verify the promotion of query performance by quantitative analysis.

1. Introduction

With the promotion of the smart city, the smart campus, and other projects, the demands of the video surveillance system deployment have become more fine-grained and multipoint. The development tendency of the video surveillance system is moving towards "digital, networked, high-definition, and intelligent" [1]. The number of monitoring devices is increasing, the quality of video is continually improving, and the duration time of the video retention is extending. All these changes increase the amount of data produced by the video surveillance system rapidly. How to effectively organize these mass monitoring video data and how to quickly locate the relevant video in the postverification are the important requirements of surveillance video storage system.

With the development of distributed file systems and cloud storage [2], a large number of surveillance video storage systems are based on the IP-SAN technologies [3]. These technologies ensure system scalability, load balancing, high availability, data backup, and recovery, but these common storage technologies only design storage systems focusing on the writing efficiency of storage. Previous designs concern the ability to write multiple video data concurrently as fast as possible, but the optimal design of video content query is lacking. Some effective video organization methods [4] are proposed, but there is little consideration about the storage of large amounts of video data. However, the campus environment surveillance video data has its unique application characteristics. The surveillance video data is a write-based data. There is no spike or trough in the generation of data. The writing of the monitored video data is sequential. Video data is streaming media data with large file size; compared to the random writing of small size file, the speed of video data writing is fast, because there is not too much OPEN/CLOSE operation. As a kind of evidential data, there is little need for modification after writing. The content of data changes regularly with the school schedule. Query operation is relatively little, but the workload is quite large when it happens. It needs to read and match the mass data which have been stored. Sometimes it needs to intervene in manual operation and the time consumption is huge.

In view of the above summary of campus surveillance video data read and write characteristics, as well as organizational storage problems, we propose a surveillance video storage system CSVS (campus surveillance video storage) for campus applications.

The main contributions of this paper can be summarized as follows:

(1) We use a mature distributed file system to address the video data writing problem, the space scalable problems, data backup, and recovery issues.

(2) We put forward a video key-frame extracted function according to the school schedule time on the impact of video data. Design an index subsystem that combines video metadata with video key frames. This index system can greatly reduce the amount of data read when retrieving. And it can improve the retrieval efficiency and reduce the workload of manual intervention.

(3) We implement the prototype of the CSVS system. The system is modeled by stochastic Petri net that can help us to analyze the efficiency of the real environment in long-time running. According to performance analysis and evaluation, it proves that the number of queries fulfilled by key-frame index is 5 times that of the manual search in the same period.

The rest of this paper is organized as follows. Section 2 summarizes related works of video data storage system. We present a campus surveillance video storage system in Section 3. In particular, we illustrate the architecture of this system and explain how we solve the problem regarding scalability and security, and the organization of data and key frame extraction method are presented in this part. In Section 4, we introduce the experimental environment and conduct the performance evaluation. We conclude this paper in Section 5.

2. Related Work

Because of the development of security system, the surveillance video storage technology has been developed more maturely. According to the characteristics of surveillance video data and SAN storage technology, researchers have proposed a video surveillance storage system based on IP-SAN [5]. Each video frame is stored in a fixed-size data area. And the memory caching technology for video metadata improves search efficiency. But this cache technology is only suitable for data sets with relatively small amount of data. In addition, the memory is a volatile storage device, so it is difficult to guarantee the data recovery after the failure. In article [6], a video data cache VDB (video data buffer) is designed based on the image group GOP, but this design is optimized only for writing. When the cache data area is full, the data will be written to disk and no longer live in the cache, so it is not helpful to the video content retrieval. To speed up writing, a cache write storage system with IO polling mechanism is proposed [7]. Each video stream corresponds to a thread written to disk. When the buffer is full, the thread triggers the write mechanism and flushes all the cache data into the disk. It converts random writing to sequential writing, thereby improving data writing efficiency. However, in the support of data retrieval, only based on the time to optimize the search, it does not support the search based on video content.

A high-performance disk array called ripple-raid for continuous data storage is proposed by [8]. Surveillance video data is a kind of continuous data, so the design features of the program can improve the video data writing efficiency. Their updating strategy and incremental generation of checksum data function not only improve the performance of data writing but also improve the energy efficiency of the system. A surveillance video storage system called THNVR based on SAN is proposed in [9]; this system uses the SQLite database to store metadata information of video. It saves nonstructured surveillance video data with fixed length files. Metadata and video data are indexed separately to improve storage and indexing performance. Fixed-length file can avoid the generation of disk fragmentation, but SQLite only supports relational data and has no high availability considerations. It is not suitable for large data storage.

Le et al. [10] propose a scheme for using SMR (shingled magnetic recording) disks in RAID arrays to accelerate the speed of storage system. Compared to traditional disk write, SMR technology can enlarge the density of data and is suited for saving the log-structure data which is like the surveillance video. A new block I/O scheduling scheme called BID (bulk I/O dispatch) is designed in [11]. By organizing the order of block I/O requests to be served, the BID scheme changes random I/Os into sequential I/Os. This operation can save CPU wait time, so the performance is promoted. Because this scheduler is especially suited for the MapReduce kind of applications and the surveillance videos are almost sequential I/Os, it is not very suitable for video storage system.

In [12], the authors propose a distributed video recording system based on IaaS; the Hadoop HDFS is used as the storage file system, and MapReduce is proposed to analyze the video data. However, with the increasing amount of video data, there is a problem of bottleneck in the metadata center when high concurrent retrieval happened. To solve the problem of high concurrent retrieving, mass storage, and so on, Cao's team proposes a high-performance distributed storage system DVSS [1]; it uses multiple storage nodes to support system linear expansion and to solve the problem of large capacity. It uses the Redis database as metadata index to improve the efficiency of high concurrent retrieval. But its GOP-based video frame operation mode is only optimized for writing. It is not considering the function based on video content retrieval.

3. Design and Implementation of CSVS

The development of video surveillance system, with the digital and network trend, has transformed from the original analog signal transmission, through the digital signal transmission, to the network digital transmission in the present. The DVR (disk video recorder) as the representative of the digital surveillance system is gradually replaced by NVR (network video recorder). DVR combines video control with video storage to make the system more integrated and more applicable, but it can only store data on the disk of the local computer, which limits the size of the system data. NVR has the function of receiving IPC (IP camera) data, video codec, storage, real-time display, and so on. It can also forward the stored video data to other storage systems through the network [13] (Figure 1 shows an example).

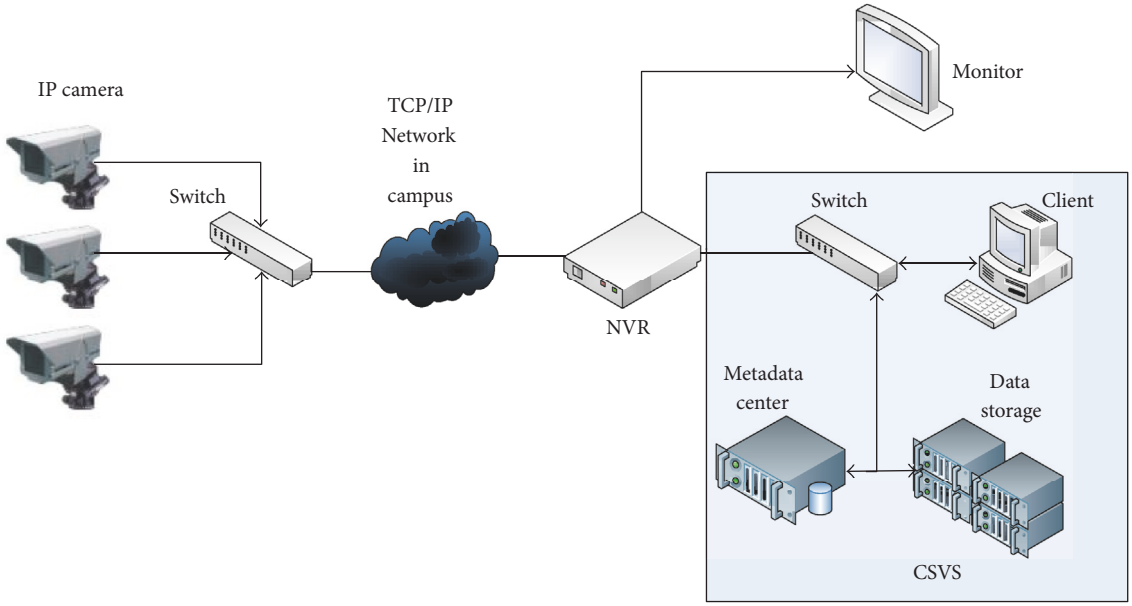

Figure 1: Video surveillance system deployment and the position of CSVS system.

The CSVS (campus surveillance video storage) system proposed in this paper is located at the back end of NVR. It provides massive video storage for monitoring system, and it also provides fast and accurate content-based video retrieval.

The implementation goals of CSVS system are as follows: scalability, where the system needs to support massive data storage and storage space scalability, security, where it is required to support data backup, storage, and lost data recovery, Fast query, where metadata information and video data are separately stored in the system, and the index of video frames is generated asynchronously. The query function based on video content is supported.

3.1. CSVS System Architecture. The CSVS system consists of three parts: the client, the metadata cluster, and the data storage cluster. The metadata cluster includes the video key-frame index center. System structure is shown in Figure 2.

The client of the system is responsible for the initiation of the tasks and the collection of operational results. The client provides internal and external APIs and video data import and export functions based on metadata and video content querys and other operations. All of these operations are launched by the client.

The metadata cluster is responsible for storing metadata information that describes the video sent by the client. The picture data generated by the key-frame extraction task are associated with the metadata information. It also provides search query function to the client. The cluster achieves load balance in order to make all the servers in the cluster undertake the tasks together.

The data storage cluster is responsible for reading and writing of video data. In the form of storage volumes, data replications are stored on different server nodes to ensure backup and achieve automatic recovery when data is abnormal. The entire cluster consists of multiple storage volumes, and the storage volume is composed of multiple storage service nodes. When a service node fails, the others still guarantee service. Linear extensions of space can be achieved by increasing the storage volumes.

3.2. System Scalability and Data Security. The metadata cluster is implemented based on MongoDB database, and the metadata information of the video is organized into Bson format for storage [14]. Because metadata information is mainly text, it occupies very little storage space compared to video data, so the main expansion of pressure is in the data storage cluster.

The data storage cluster is implemented based on the GlusterFS distributed file system [15]. According to this application scenario of surveillance video, we select specific features to serve our system. We use the replica function to automatically backup the data and the strip ribbon function to improve the writing performance. The strip ribbon function, which is like the RAID0 technique, makes different disks write different parts at the same time, so it accelerates the speed of writing.

3.3. Organization Form of Data. In order to provide a fast and accurate retrieval function, we save the metadata and video data in separated way. We save metadata to the Mongodb database cluster in key-value form.

The metadata field includes the video ID, the video name, the shooting position, the start time, the end time, the video duration, the video file size, the video storage path, the key-frame flag, and the key-frame storage path, as shown in Figure 3.

The ID field is the unique identifier of the metadata in the database. Video_name is the file name of the video data. Position is the position information where the video data is recorded; Start_time is the recording start time of the video

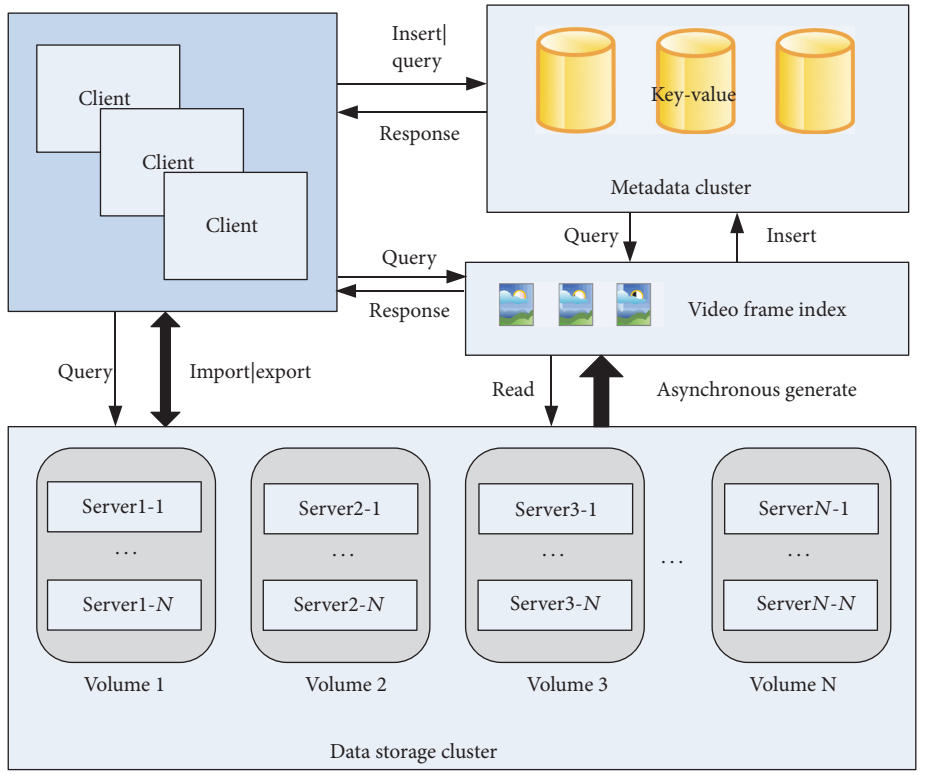

Figure 2: Architecture of CSVS.

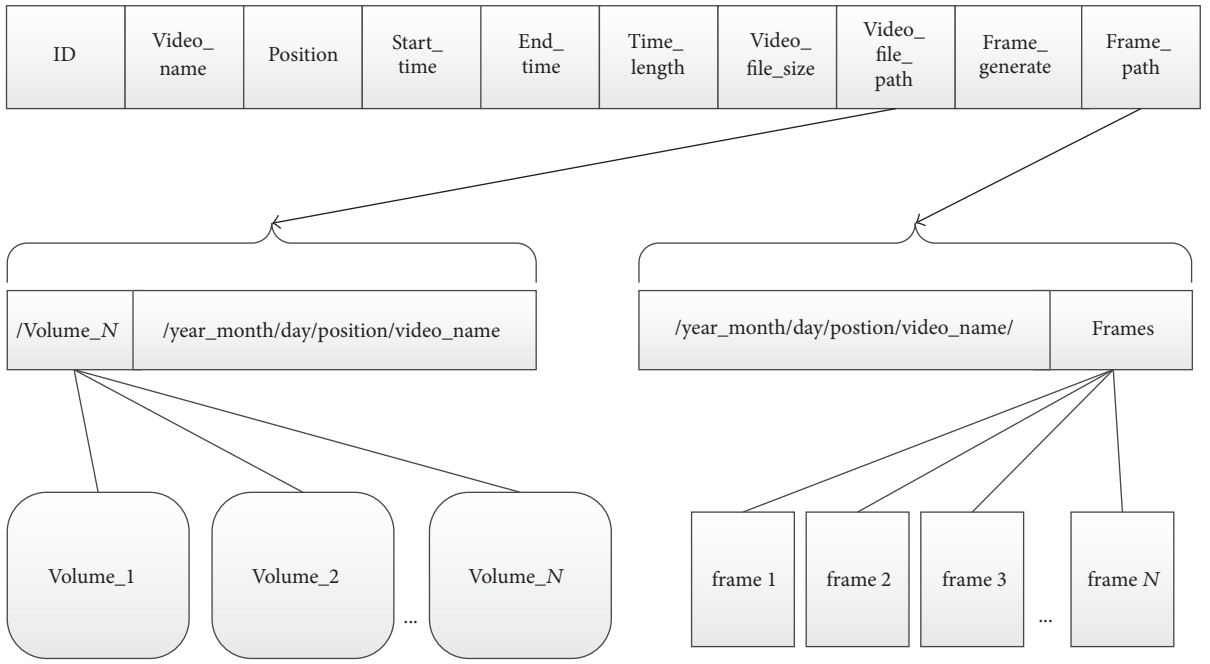

Figure 3: Metadata fields.

data. End_time is the shooting end time of the video data. Time_length is the duration of the video data which is equal to the shooting end time minus the shooting start time. Video_file_size is the file size of the video data. Video_file_path field saves the video file storage path. It has two parts: the first part is directory of storage path which represents the number of the storage volumes in the data cluster, and the second part is the directory of the video file to be stored in.

The Frame_generate field is used to identify whether a key-frame group has been generated. The Frame_path

field holds the storage path of the key-frame group of the video. This path stores the picture files generated by the asynchronous key frame extraction task, and it provides the index for video based on content retrieval.

3.4. Key-Frame Extraction of Surveillance Video. Surveillance video is different from other videos. The video shooting angle is stabilized. So the video background does not change too much. According to this feature, the difference between the frames in the video is compared with the way of calculating the histogram distance, and the video key frame is extracted [16]. Because the campus has fixed life schedule, the video data content is also regular.

Before class time and at school canteen dinner time, video content is the most abundant, but during the class and night hours, there is no change in the content of the video. If we retrieve video content through the human eye, it will inevitably increase unnecessary workload. If we do match content in each frame of the video by the machines, there is also a huge amount of computation. To solve this problem, combined with the above characteristics, we calculate the histogram difference as the rule to extract the video key frame [17] and build the content index of video.

In the operation of key-frame extraction, the formula for calculating histogram distance [18] has four methods: CORREL, Chi-Square, Bhattacharyya, and INTERSECT.

The CORREL method is defined as

$$d_{\text{correl}}(H_1, H_2) = \frac{\sum_i H_1'(i) \cdot H_2'(i)}{\sqrt{\sum_i H_1'(i) \cdot H_2'(i)}}, \quad (1)$$

where H_1 and H_2 represent two histograms and $H_1'(i)$ is defined as follows:

$$H_k'(i) = H_k(i) - \left(\frac{1}{n}\right)\left(\sum_j H_k(j)\right), \quad (2)$$

where n is the number of the bins.

The Chi-Square formula is as follows:

$$d_{\text{chi-square}}(H_1, H_2) = \sum_i \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i)}. \quad (3)$$

Bhattacharyya is defined as

$$d_{\text{Bhattacharyya}}(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sum_i H_1(i) \cdot H_2(i)}}. \quad (4)$$

At last, the INTERSECT method has the following formula:

$$d_{\text{intersect}}(H_1, H_2) = \sum_i \min(H_1(i), H_2(i)). \quad (5)$$

Our strategy for extracting key frames is to use as few frames as possible to summarize the complete video content. In order to improve the computational efficiency, we use the INTERSECT method, which is the minimum and fastest histogram distance calculation method.

```
INPUT: VideoPath, OutputPath, Threshold;
OUTPUT: Frames;
(1)  ret = access(VideoPath);
(2)  if ret == False then
(3)    exiterror_code;
(4)  end if
(5)  ret = access(OutputPath);
(6)  if ret == False then
(7)    mkdir(OutputPath);
(8)  end if
(9)  cap = captureFromFile(VideoPath);
(10) frame = QueryFrame(cap);
(11) baseHistogram = CalcHist(Frames);
(12) while frame = QueryFrame(cap)! = NULL do
(13)   curHistogram = CalcHist(Frames);
(14)   HistogramError = CompareHist
(15)  (baseHistogram, curHistogram,
(16)   COMP_INTERSECT);
(17)   curHistogram = CalcHist(Frames);
(18)   if instogramError < Threshold then
(19)     SaveImage(Frames);
(20)     CopyHist(baseHistogram, curHistogram);
(21)   end if
(22)   ReleaseHist(baseHistogram);
(23)   ReleaseHist(curHistogram);
(24)   ReleaseCap(cap);
(25) end while
```

Algorithm 1: Key-frame extraction.

For implementation of key-frame extraction algorithm description, see Algorithm 1.

The threshold is a floating point variable between 0 and 1. The closer the threshold is to 1, the more key-frames are written. Key-frame naming is a long integer + extension format. The long integer is the sequence number of the frames in the video. A frame sequence number can be used to locate the time point in which the frame appears in the video.

$$\text{Time} = \frac{\text{num}}{\text{frame\_rate}} + 1. \quad (6)$$

For example, if the duration of a video is 1000 seconds and 25 frames per second (frame_rate), the video is composed of 25000 frames. If the key-frame to be queried is the 1024th frame (num), the frame can be matched in the 41 seconds of the video according to (6).

3.5. Writing and Retrieving Data

3.5.1. Data Writing. The writing of the video data is initiated by the client. The client acquires the video data from the NVR. It generates the metadata information based on the video data and starts the writing operation.

The writing procedure is as follows:

(i) Get video data from NVR.

(ii) Read the video data file, parse out the video file, the video camera position, video start time, and end time,

calculate the duration of the video, parse out the video file size, generate a video file path, set the flag as 0 which means key-frames extraction is needed, and generate a key-frame storage path.

(iii) Insert data into the metadatabase.

(iv) Write the video data according to the storage path and continue processing the next video data.

(v) Start the asynchronous key-frame extraction task and check which key-frame flag is 0.

(vi) Extract the key frames and write them into storage path and set the flag to 1; writing operation is finished.

3.5.2. Retrieval of Video Data. The retrieval of video data is divided into ordinary retrieval and video content retrieval. Ordinary retrieval only needs to operate on the metadata database. The retrieval of video content is based on the key-frame index.

Ordinary Retrieval Process

(i) According to the search conditions, such as start time, end time, duration, shooting point, or the combination of these conditions, a query can be initiated to the metadatabase by a lookup operation.

(ii) The metadatabase returns all the data entries that match the conditions.

(iii) Parse the video storage path from all of the data entries.

(iv) Read video file directly based on the video storage path.

Video Content Retrieval Process

(i) Search in database based on metadata conditions (the same process as the ordinary retrieval process).

(ii) The metadatabase returns all the data entries that match the conditions.

(iii) Parse the key frame storage paths from the returned items one by one.

(iv) Compare the video content to key-frame group saved in storage path.

(v) Return the closest key frame, parse the video storage path of the frame, and calculate the time appearing in the video.

(vi) Locate the video based on the storage path and the time when it appears.

4. Experiment and Performance Evaluation

4.1. Experimental Environment. CSVS system relies on campus video surveillance system. We use 11 Hikvision DS-2CD5026XYD HD cameras and 1 Hikvision DS-8632N-I8 network video recorder in our system. We use 5 virtual machines to build the cluster of metadata and storage center. Each machine has Intel Xeon 2.5 GHz Core∗4, CPU, 12 GB memory, and 100 GB hard disk.

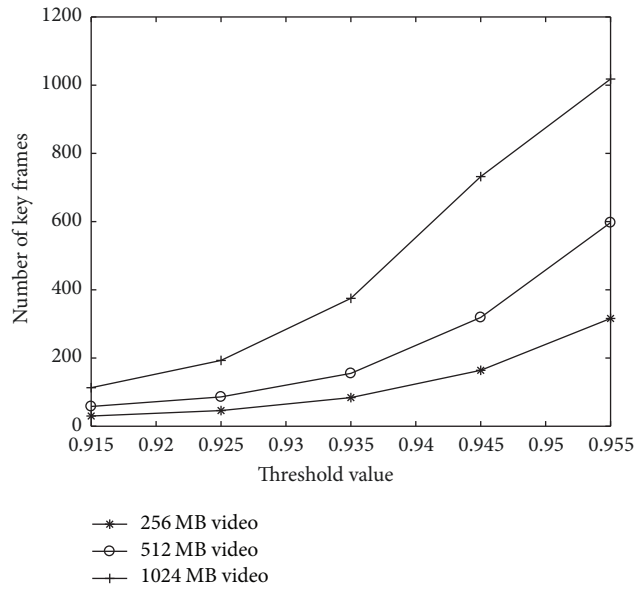

Figure 4: Threshold and number of key frames.

4.2. Threshold and Key Frames. According to our key-frame extracting algorithm, the threshold parameter decides the quantity of key frames generated from video. The content of each frame in video changes slowly with time. For calculating the extent of difference between each of the frames, we use histogram distance to measure them. The threshold is the criterion to identify which histogram distance is large enough to represent the frame that needs to be saved. The bigger threshold is, the more frames will be saved.

In Figure 4, we use three different size videos to verify the relationship between threshold and number of key frames. The test video sizes are 256 MB, 512 MB, and 1024 MB. We choose five numerical values of threshold from 0.915 to 0.955 to generate the key frames. The result shows that number of key frames will increase when threshold value gets bigger. Shown as 1024 MB video file, the number of key frames increases from 113 to 1018. A large number of key frames make the index more accurate but take up more storage space. So it is very useful to tune the threshold value to weigh the storage space and the index efficiency.

4.3. Efficiency of Content Retrieval. We have done some exercises to check the working efficiency of key-frame index function. The retrieval time is a criterion to compare our method with the GOP index method without key-frame index. We have obtained the result shown in Figure 5.

We save a group of video files in our CSVS storage system and DVSS system. These files' size is from 64 MB to 2048 MB. We choose a frame of video as target to retrieve in these two systems. The difference value of time cost between two methods will become bigger and bigger when the video files' size is increasing. The retrieval time of common method will rise sharply. By contrast, the key-frame index method will lift the time cost line gently.

Table 1: System comparison.

Exist/our work	Storage model	Fault tolerance	Scalability	Retrieval speed	Content retrieval
THNVR	SQLite + FS	Poor	Poor	Low	No
DSFS	CSM	SPOF	Poor	Low	No
DVSS	Redis + logical volume	Strong	Strong	High	No
CSVS	Key-frame + GlusterFS	Strong	Strong	High	Yes

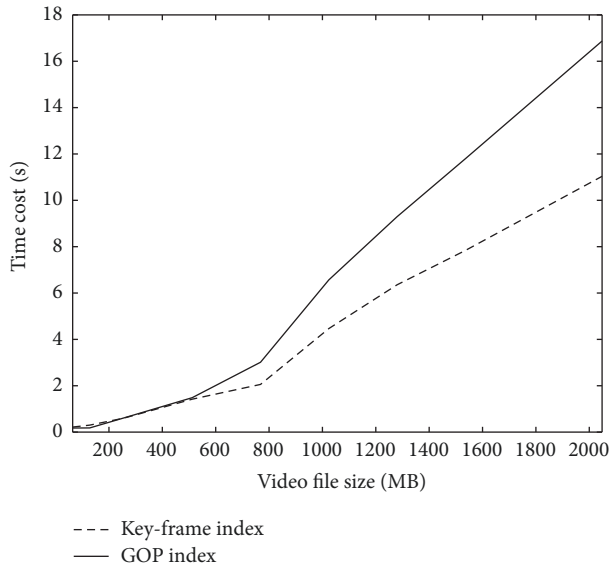

Figure 5: Comparison of retrieval time.

4.4. Comparison with Relative Work. Considering the function of content retrieval, our storage system has advantages compared with other existing systems. The statistical function results are shown in Table 1.

4.5. Performance Evaluation. Our surveillance video storage system is built in campus and operated by our team. It runs well but the workload is low. How to evaluate the performance in long-time runs if the workload is increasing?

To do the performance evaluation, a transient stochastic state classes method is proposed [19]. The authors provide an approach to continuous time transient analysis. Transitive closure of transitions identifies a transient stochastic graph. It is convenient to map a transient stochastic tree to do the classes analysis. The approach is applicable for any Generalized Semi-Markov Process. It is also suitable for performance evaluation in real-time systems. While this method is complex, Balsamo and his colleagues provide a powerful, general, and rigorous route to product forms in large stochastic models [20]. They present a building block concept composed by a group of logically related places and transitions. The state probability of building block equals the sum of the places' state probability in the group. This technique can effectively avoid the problem of the state space explosion. If the model is complex and difficult to solve, we can adopt this method to address the state space explosion problem. Our purpose is to obtain the stationary state probability. Therefore we adopt ordinary stochastic Petri nets (SPNs) to model and evaluate the performance.

Table 2: Meaning of transition.

Transition	Meaning
$T0$	Search from metadata center
$T1$	Parse result to get path
$T2$	Retrieve data by path
$T3$	Match key frames
$T4$	Retrieve data by key frames
$T5$	Return video by storage path
$T6$	Return video and time point by key frames

Stochastic Petri nets is a powerful tool for system performance evaluation [21–23]. In this paper, the basic theory of stochastic Petri nets is applied to model and evaluate performance of storage systems. According to [24], we assume that the firing rates of transitions are independent random variables with (negative) exponential distributions and represent the frequency of the data process for each function in our system. The isomorphic relationship between the stochastic Petri net and the Markov chain is used to calculate the stationary state probability, and the performance evaluation of query efficiency in storage system is provided.

According to the retrieval process in CSVS system, the stochastic Petri nets model is established by using PIPE (Platform Independent Petri Net Editor) tool [25]. As shown in Figure 6, it consists of 8 places, $P0$, $P1$, $P2$, $P3$, $P4$, $P5$, $P6$, and $P7$, and 7 transitions, $T0$, $T1$, $T2$, $T3$, $T4$, $T5$, and $T6$.

$P0$ represents query conditions based on metadata, $P1$ represents the query result based on the video content, $P2$ represents the query result returned by the metadata, $P3$ represents the video storage path, $P5$ represents the key-frame storage path, $P4$ represents the time of the key-frame in the video, $P6$ represents the ordinary retrieval and getting the results, and $P7$ represents the video content retrieval by matching key frame group and finding the results.

Table 2 shows the meaning of each transition in the CSVS system.

The reciprocal of firing rate is service time. We set $\tau = \{\tau_0, \tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6\}$ as service time. We compare every cost time of data processing functions in CSVS which represent the service time of transitions. Then we get the ratio of them, $T0$, $T1$, $T4$, $T5$ and $T6$ cost one unit of time to process the same amount of data. Retrieving data by path (i.e., $T2$) will cost ten units of time. Matching key frames (i.e., $T3$) will cost two units of time. Above all, $\tau = \{1, 1, 10, 2, 1, 1, 1\}$. All the

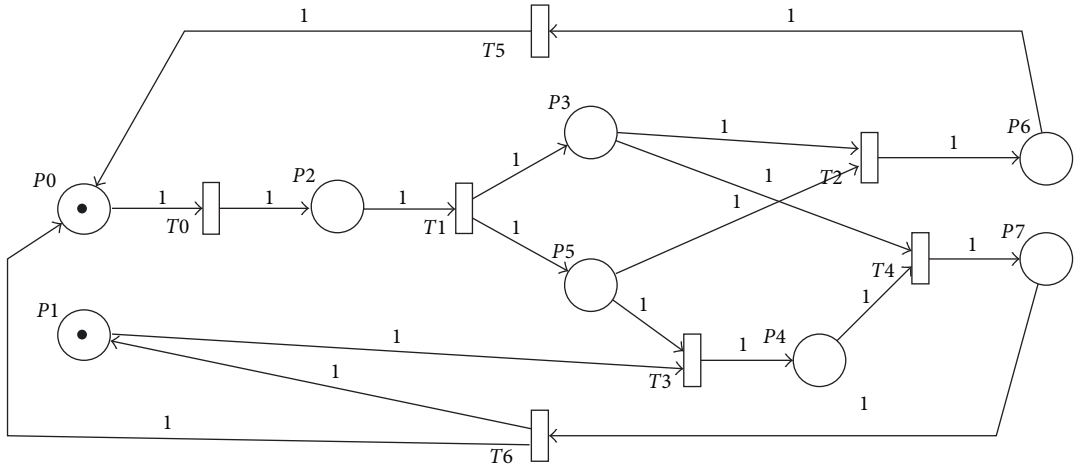

Figure 6: Retrieval process in CSVS system modeling by PIPE tool.

data of unit time cost are obtained from our prototype system. Then $\lambda = 1/\tau$; firing rate is $\lambda = \{10, 10, 1, 5, 10, 10, 10\}$.

According to the running time of the function, which corresponds to the transition in our retrieval subsystem, we set the firing rate $\lambda = \{10, 10, 1, 5, 10, 10, 10\}$ for the transitions $T0$, $T1$, $T2$, $T3$, $T4$, $T5$, and $T6$ as the time stochastic parameters subject to exponential distribution. Assuming an initial marking of one token in place $P0$ and $P1$ and no tokens in the rest of places, we can obtain a state matrix:

$$\begin{pmatrix} & P0 & P1 & P2 & P3 & P4 & P5 & P6 & P7 \\ M0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ M1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ M2 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ M3 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ M4 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ M5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (7)$$

Let Q be the transition probability matrix, and $X = (x_0, x_1, x_2, x_3, x_4, x_5)$ is the stationary state probability of the above $M0$, $M1$, $M2$, $M3$, $M4$, and $M5$. According to the Markov processes, there are linear equations as follows:

$$\begin{aligned} XQ &= 0 \\ \sum_i x_i &= 1, \quad 0 \leqslant i \leqslant 5. \end{aligned} \quad (8)$$

Solve the equations to obtain the stationary state marking probability [26]:

$$P[M0] = 0.18182,\ P[M1] = 0.18182.$$
$$P[M2] = 0.30303,\ P[M3] = 0.15152.$$
$$P[M4] = 0.0303,\ P[M5] = 0.15152.$$

At the same time, we can deduce the stationary state probability of each place:

$$P[M(P0) = 1] = P[M0] = 0.18182.$$
$$P[M(P1) = 1] = P[M0] + P[M1] + P[M2] + P[M4] = 0.69697.$$
$$P[M(P2) = 1] = P[M1] = 0.18182.$$
$$P[M(P3) = 1] = P[M2] + P[M3] = 0.45455.$$
$$P[M(P4) = 1] = P[M3] = 0.15152.$$
$$P[M(P5) = 1] = P[M2] = 0.30303.$$
$$P[M(P6) = 1] = P[M4] = 0.0303.$$
$$P[M(P7) = 1] = P[M5] = 0.15152.$$

We focus on $P[M(P6) = 1]$ and $P[M(P7) = 1]$; $P6$ represents ordinary retrieval, and $P7$ represents video content retrieval, and $P[M(P7) = 1]$ is almost equal to $5 * P[M(P6) = 1]$. From the analysis, the retrieval hit count of video content is five times that of ordinary retrieval during the same length of time.

Through the simulation and performance evaluation of the stochastic Petri net, it can be concluded that the CSVS system based on the key-frame index function can effectively improve the efficiency of video retrieval.

5. Conclusions

We propose a video storage system (CSVS) to cope with the problem of the low efficiency of video content retrieval in video storage system; a surveillance video storage system (CSVS) for campus applications is proposed in this paper. While achieving storage space scalability and high data availability, the key-frame extraction technology is applied to the storage index function of CSVS system. In the content-based retrieval operation, we find the video through the key frame and locate the time in seconds when the frame appeared in the video. The CSVS system is modeled and evaluated by stochastic Petri nets. According to the characteristics of stochastic Petri nets and Markov process isomorphism, the stationary probability of the marking is obtained. The probability provides the evidence for analyzing the efficiency

of system running. In the light of quantitative analysis above, we can find that the CSVS system with key-frame index can improve the efficiency of query based on video content.

In the follow-up works of CSVS system, the method of key-frame extraction will be studied continually. We will improve the algorithm based on histogram distance calculation and find a more accurate key-frame extraction technique. We also plan to optimize the algorithm for the matching process of the key-frame querying. And we need to further improve the efficiency of retrieval based on video content.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the following: the National Natural Science Foundation of China (nos. 61370065 and 61502040), National Key Technology Research and Development Program of the Ministry of Science and Technology of China (no. 2015BAK12B03-03), and Beijing Municipal Program for Excellent Teacher Promotion (no. PXM2017_014224.000028).

References

[1] S. D. Cao, Y. Hua, D. Feng, Y. Y. Sun, and P. F. Zuo, "High-Performance distributed storage system for large-scale high-definition video data," *Ruan Jian Xue Bao/Journal of Software*, vol. 28, no. 8, 2017, in Chinese.

[2] Z. Zhao, X. Cui, and H. Zhang, "Cloud storage technology in video surveillance," *Advanced Materials Research*, vol. 532-533, pp. 1334–1338, 2012.

[3] W. Zheng and X. J. Zhang, "Research and Application of IP-SAN," *Journal of Electrical and Computer Engineering*, 2003.

[4] J. Calic and E. Izquierdo, "Efficient key-frame extraction and video analysis," in *Proceedings of the International Conference on Information Technology: Coding and Computing, ITCC 2002*, pp. 28–33, USA, April 2002.

[5] J. He, *Study on key technique in video surveillance storage system based on IP-SAN*, Shanghai, Shanghai Jiaotong University, 2011, in Chinese.

[6] JX. Tang, *The software design of storage subsystem for network video surveillance system*, Zhejiang University, Zhejiang, 2013, in Chinese.

[7] M. Jiang, Z. Y. Niu, and S. P. Zhang, "Design and implementation of video surveillance storage system," *Computer Engineering and Design*, vol. 35, no. 12, pp. 4195–4201, 2014 (Chinese).

[8] Z. Z. Sun, Q. X. Zhang, Y. A. Tan, and Y. Z. Li, "Ripple-RAID: A high-performance and energy-efficient RAID for continuous data storage," *Ruan Jian Xue Bao/Journal of Software*, vol. 26, no. 7, pp. 1824–1839, 2015 (Chinese).

[9] J. Y. Wu, Y. Gu, D. P. Ju, and D. S. Wang, "THNVR: Distributed large-scale surveillance video storage system," *Computer Engineering and Applications*, vol. 45, no. 31, pp. 56–59, 2009 (Chinese).

[10] Q. M. Le, A. Amer, and J. Holliday, "SMR Disks for Mass Storage Systems," in *Proceedings of the IEEE 23rd International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, MASCOTS 2015*, pp. 228–231, USA, October 2015.

[11] P. Mishra, M. Mishra, and A. K. Somani, "Bulk I/O storage management for big data applications," in *Proceedings of the 24th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS 2016*, pp. 412–417, UK, September 2016.

[12] C.-F. Lin, M.-C. Leu, S.-M. Yuan, and C.-T. Tsai, "A framework for scalable cloud video recorder system in surveillance environment," in *Proceedings of the 2012 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC)*, pp. 655–660, IEEE, Fukuoka, September 2012.

[13] H. E. Xiao-Feng, *Analysis and Application of Network Video Recorder (NVR) Storage Technology*, China Science & Technology Information, 2011.

[14] Y. Gu, X. Wang, S. Shen et al., "Analysis of data storage mechanism in NoSQL database MongoDB," in *Proceedings of the 2nd IEEE International Conference on Consumer Electronics - Taiwan, ICCE-TW 2015*, pp. 70-71, June 2015.

[15] P. Zhou, F. Dong, Z. Xu, J. Zhang, R. Xiong, and J. Luo, "ECStor: A Flexible Enterprise-Oriented Cloud Storage System Based on GlusterFS," in *Proceedings of the 4th International Conference on Advanced Cloud and Big Data, CBD 2016*, pp. 13–18, China, August 2016.

[16] G. Liu and J. Zhao, "Key frame extraction from MPEG video stream," in *Proceedings of the 3rd International Symposium on Information Processing, ISIP 2010*, pp. 423–427, China, November 2010.

[17] Y. Yang, F. Dadgostar, C. Sanderson, and B. C. Lovell, "Summarisation of surveillance videos by key-frame selection," in *Proceedings of the 2011 5th ACM/IEEE International Conference on Distributed Smart Cameras, ICDSC 2011*, Belgium, August 2011.

[18] R. Bradski G and A. Kaehler, *Learning OpenCV*, OReilly Media, 2014.

[19] A. Horváth, M. Paolieri, L. Ridi, and E. Vicario, "Transient analysis of non-Markovian models using stochastic state classes," *Performance Evaluation*, vol. 69, no. 7-8, pp. 315–335, 2012.

[20] S. Balsamo, P. G. Harrison, and A. Marin, "Methodological construction of product-form stochastic Petri nets for performance evaluation," *The Journal of Systems and Software*, vol. 85, no. 7, pp. 1520–1539, 2012.

[21] M. K. Molloy, "Performance Analysis Using Stochastic Petri Nets," *IEEE Transactions on Computers*, vol. C-31, no. 9, pp. 913–917, 1982.

[22] A. Marin, S. Balsamo, and P. G. Harrison, "Analysis of stochastic Petri nets with signals," *Performance Evaluation*, vol. 69, no. 11, pp. 551–572, 2012.

[23] S. Distefano, F. Longo, and M. Scarpa, "Marking dependency in non-Markovian stochastic Petri nets," *Performance Evaluation*, vol. 110, pp. 22–47, 2017.

[24] W. M. Zuberek, "Performance evaluation using unbounded timed Petri nets," in *Proceedings of the Third International Workshop on Petri Nets and Performance Models (PNPM89)*, pp. 180–186, 1989.

[25] P. Bonet, C. Llado M, R. Puigjaner et al., *PIPE v2.5: a Petri Net Tool for Performance Modeling*, 2007.

[26] C. Lin, *Stochastic Petri Nets and System Performance Evaluation*, Tsinghua University Press, Beijing, China, 2nd edition, 2005.