An Indoor Surveillance System with High-Rate Video Compression and Rotational Monitoring Capabilities

Ping-Chung Chen (陳秉中) and Wen-Hsiang Tsai (蔡文祥)
Department of Computer and Information Science,
National Chiao-Tung University, Hsinchu, Taiwan, R.O.C.
Tel: 886-3-5712121 Ext. 56650, e-mail: gis90558@cis.nctu.edu.tw

Abstract

An indoor surveillance system with high-rate video compression and rotational monitoring capabilities is proposed. The proposed surveillance system includes two parts, one being a client site, and the other a sever site. At the client site, the system records the background scene and extracts foreground objects from captured frames by image comparison. In the proposed rotational surveillance system, in order to identify the rotational background scene correctly, an image sequence of the background scene is first captured by allowing the camera to rotate through a circle. A technique of image matching is then employed to find a suitable background image for eliminating the background region in the current frame. At the sever site, the system superimposes the foreground object video on the pre-saved background images to construct a complete surveillance video for display. Finally, good experimental results show the feasibility of the proposed techniques.

1. Introduction

Recently, there are many studies on vision-based surveillance systems [1-5]. In a surveillance system, a property of the surveillance video is that most scenes are changeless because the camera is static during image taking. Taking advantages of this property, we can perform content-based video compression just like MPEG-4 by discarding the changeless scene and compressing just the foreground objects. Besides the most popular static monitor, it is also desired to investigate the rotational surveillance system in this study. Because the rotary movement of the camera causes changes in each image frame, the system will have to spend more bits to compress the surveillance video. Although the scene in the video is changed continuously, we know that the monitored scope is not changed actually. If we can find out some way to discard the rotational background, we will again need to process only the foreground objects, and the goal of high-rate compression in a rotational surveillance system can be achieved as well.

To identify which region in one frame is part of a foreground object, it has been proposed in the past to construct a 3D monitored scene as the reference model and compare the current captured scene with the model for background identification and removal. However, the involved 3D techniques are complicated and unsuitable for use in general surveillance systems. Therefore, it is desirable in this study to develop some techniques that only use 2D image analysis methods to implement real-time monitoring and video compression capabilities.

2. Main Procedures

Before the monitoring stage, we perform the operations of the learning stage. The background images of the monitored scene should be stored into the database for identifying the foreground object in the monitoring stage. The next step is for the client to

start capturing surveillance video frames and extracting foreground objects by background elimination. Background images in the database are used as reference frames for background removal. Each extracted foreground object is then circumscribed into a rectangular sub-image, and the sub-images that belong to an identical object are grouped into an object video for content-based compression. The client then transmits the compressed object video to the sever site. The server receives and decompresses the object video and combines its frames with the background images to reconstruct a complete surveillance video for inspection at the sever site. On the other hand, if the surveillance system is rotational, the rotational scene frames are saved in the learning stage, and we use some image correspondence methods to identify foreground objects in the monitoring stage.

Generally speaking, for environment surveillance applications, the compression ratio yielded by the proposed system is higher than that of a general block-based video compression method (like MPEG-1) and not lower than that of a common content-based technique (like MPEG-4). A brief flowchart of the proposed system is illustrated in Fig. 1.

3. Techniques for High-Rate Compression of Surveillance Videos

The key idea of the proposed techniques is to process just the foreground object and remove the changeless background to achieve high-rate compression. Background learning is executed before the monitoring procedure is started in a surveillance system. With a static camera, background learning means that we may store only one background image into the database. For this purpose, we must ensure that no foreground object exists in the background image to be saved.

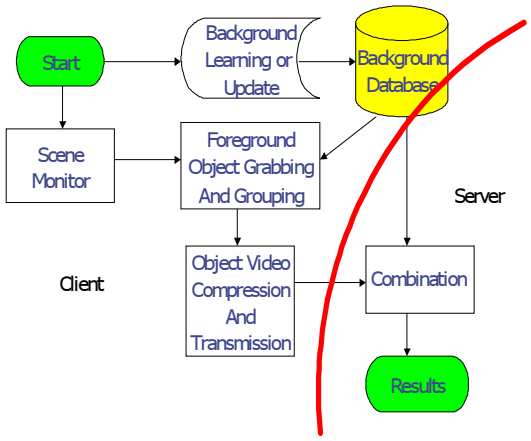

Fig. 1 Flowchart of the proposed system.

3.1. Foreground Object Grabbing

After the procedure of background learning and at the beginning of foreground object grabbing, we conduct image segmentation by a traditional method of frame differencing. After the previous steps, each pixel in the image frame is categorized into two groups: the foreground object part and the changeless background part. Because some foreground objects made by noise or coming from unsuitable thresholding are not physical objects that we care about, a method for noise elimination is desired. Besides, in the subsequent stages, the location and the size of each physical object are the important information that need be known. Therefore we adopt a fast algorithm of row-by-row connected-component labeling to connect adjoin objects and calculate their sizes and locations in the frame. The goal of noise elimination can then be accomplished by a process of connected component removal because tiny objects can be seen as non-actual objects.

3.2. Object Grouping for Integrated Compression

After the step of object grabbing, foreground objects are detected, and the regions of the changeless background part can be removed.

However, foreground objects are not suitable to be compressed. The shape of each object is arbitrary, and the block-based compressor only accepts rectangle images. This means that objects should be circumscribed into rectangular shapes before the compression operation can be started. After performing the process of foreground object grabbing, one physical object might be separated into several smaller object parts, called sub-objects. If two sub-objects are close to each other, it is reasonable to combine them together. Furthermore, because each object occupies a block region, block overlapping might happen. By this reason, if the distance between two objects is too small and if we do not merge them, we will waste the system resource to process repeated regions when these objects are compressed and transmitted. Because each object in one frame is regarded as a rectangular sub-image, and the locations, widths, and heights of these sub-images are known, we may calculate the minimum Euclidean distance between every pair of objects to determine the relation of objects. If two objects overlap each other, their minimum Euclidean distance d will be computed to be 0. Therefore, a decision rule for merging two objects is: if d is smaller than a threshold τ, then merge them. The new object resulting from merging is a block that includes the two original objects. Although this rule is not perfect to identify objects, the process is fast.

On the other hand, parts belonging to an identical physical object in consecutive frames must also be compressed together to increase the compression rate. Although we can understand which region is part of a foreground object, the relation of each object in different frames is unknown. Fortunately, the locations of these objects are available, and we can design some rules to determine coarsely the relationship among these objects by the distance between every pair of objects. Even though such rules to identify objects can be found and we can gather objects belonging to an identical physical object in each frame into a sub-image sequence of objects, yet the size of an identical object in each frame might be different and it is required that pictures in a sub-image sequence must be circumscribed into a fixed rectangle when they are compressed. To simulate content-based compression methods, we try to find the relation between every pair of objects in nearby frames in this study. If the minimum Euclidean distance d between two objects in consecutive frames is smaller than a threshold τ, the two objects are merged into an identical sub-image sequence. Other regions that do not belong to any object sub-image sequence are then removed. Removing useless regions can reduce the amount of data to achieve the goal of high-rate video compression.

3.3. Dynamic Background Update

Although the process of background learning can be used to grab objects correctly, the background is not always changeless. A situation might occur: a foreground object enters the monitored scope and keeps stationary. Then the region of this object will always be grabbed. In order to avoid this condition, this area should be regarded as part of the background. This means that a new background needs to be constructed, and the procedure of learning needs to be repeated. The operation of such re-learning is called *dynamic background update* in this study. Like the process of background learning, there must be no moving object in the monitored scope during the period of background update. Furthermore, the updated image of the new background will be sent to the server site, so the process of update will increase the load of the communication channel. Therefore, the procedure of

background update should not be executed if the background scene does not change *obviously*. We conduct object grabbing in one captured frame before background update. If some objects appear, it means that the scene might be changed, and the operation of background update must be performed. In order to ensure that no active object appears in the monitored scope when we update the new background, we must capture a new frame and compare this frame with the previous captured frame. If the two frames are obviously different, this means the scene is not stable, and the operation of background update will *not* be executed. The following steps are performed at fixed intervals.

Step 1: Perform the process of object grabbing by comparing the original background image I_1 with the current captured frame I_2.

Step 1.1: If no object is in I_2, the operation of background update is not necessary.

Step 1.2: If some objects are detected in I_2, save I_2 to analyze the background next.

Step 2: If I_2 includes objects, capture another frame I_3 a little time later. Perform object grabbing in I_3 by comparing I_3 with I_2.

Step 2.1: If some objects are still in I_3, then the new background image is not available, and the operation of background update is not executed.

Step 2.2: If no change between I_3 and I_2 is detected, the operation of background update is performed. Regard I_3 as the new background image, use it to replace the original background picture I_1, and sent it to the database of the server site immediately.

4. Proposed Techniques for Rotational Surveillance

Methods proposed for video compression for rotational surveillance systems are more difficult to design because the background in a surveillance video sequence changes continuously, and most commercial rotational platforms are not usually designed to possess digital processing units to grab the positions of cameras. In order to know the relative positions of cameras and scenes, many people tried to construct 3D models in the learning stage and perform camera calibration. However, this is neither flexible nor friendly for common users. It is more desirable to use 2D techniques to solve the problems of camera location. If we can estimate the position of a rotational camera at any moment, then the corresponding background of a certain position can be found. We use the technique of image matching to solve these problems. However, the complexity of image matching is high, so a method for system speed-up is necessary.

4.1. Analysis of Rotational Monitoring

In a rotational surveillance system, there is a mechanical turning point on its rotational platform. The camera on the platform will turn back when a pinpoint of the platform contacts the turning point. We can store a background scene sequence by allowing the camera to rotate through a circle in a rotational surveillance system to replace the idea of storing just one background image in a static system. And the techniques of image matching can help knowing the position of a camera and find the most suitable background image with the current captured frame. If the rotational platform rotates regularly, the proposed method of image matching can be stable, and object grabbing and grouping can be performed effectively. Finally, the high-rate content-based video compression algorithm can be applied again in the rotational surveillance system.

4.2. Learning Stage of Rotational Monitoring

We capture an image sequence of the

background scene without any moving object by allowing the camera to rotate through a circle. In a static surveillance system, all we have to do is to capture one background image and save it. But in a rotational surveillance system, just recording the background image sequence is not enough because the information of these image frames is not sufficient for use in searching the most similar background image for the current frame, which is required in the monitoring stage. Although we can apply image matching to all background images in the database after capturing each frame, this way wastes too much time and system resource. We define in this study the *initial point* to be the image frame of the scene taken at the turning point, and the final point the frame that has the same property as the initial point with a one-round delay. By analyzing which frame is the initial point or the final point, we can know whether the camera is at the turning point at a certain moment. With the help of this information, we can calculate the current location of the camera, and therefore can find the most suitable background easily. By this way, we also can find the position of the camera without employing a camera calibration procedure which is required in other 3D scene analysis works. Although it is very simple for human beings to analyze all of the background images to look for the initial or final point by eyes, yet sometimes to access the database in the surveillance system is not very easy. So we propose a method to find the initial point automatically in this study. If we compute the correlation coefficient of every two consecutive frames, we will find that the similarity of the two consecutive frames taken at the location of the turning point is higher than those at other positions. A reason for this to be true is that the movement between frames becomes smaller at the turning point due to the slowdown of the rotational platform at the moment of rotation reversing, and this phenomenon makes the frames captured at this point to be very similar.

4.3. Monitoring Stage of Rotational Monitoring

We grab the foreground object by comparing a captured frame with the background frame in the monitoring stage. Therefore, the most important task of this stage is to find the best-fitting background image of each captured frame. After the leaning stage, background images are stored, and the information of the initial point and the final point are known. Consequently the monitoring stage is started. We then analyze each frame captured by the rotational camera but do not perform object grabbing before finding the frame of the initial point. The reason why object grabbing is postponed is that the position of the camera needs to be estimated. If we know the probable position of the camera, the probable region that is monitored by the surveillance system can be found easily. Then we can take less background samples to perform image matching and improve the system performance. Since we know that the frames of the initial point and the final point are both taken at the turning point of the rotational platform, we can assume that the position of the camera is right on the turning point when the correlation between the current frame and the frame of the initial point is very high. After locating the position of the camera, the procedure of object processing can be performed. The speed of the rotational platform is known in advance and we can predict the moment at which the platform will reverse its rotational direction, so the system can prepare to find the frame of the final point and re-estimate the position of the camera again at that time.

In the monitoring stage, we find the most similar background image in the database for the

current frame. Because image matching consumes a lot of time, we need some methods to decrease the computational complexity. When the platform is rotational, the searching range in the database needs to be changed immediately. That means that we need to select a probable range for background image search in the database. If we look for the suitable background image only in this range instead of searching all images in the database, the computing time can be reduced significantly to achieve the goal of real-time processing. Knowing the frame number of the initial (or the final) point and the frame rate fps (frame per second) of the camera is useful to find the current probable scope. By understanding the frame number of the initial point and the final one, the position of the camera can be known when the camera arrives at the initial or the final point. After locating the camera at the initial point, we can shift the searching position to the probable range of background frames in the database after each frame is captured in the monitoring stage. After the probable search range is found out, we propose further to use a binary hierarchical matching algorithm to find the best-fitting background image in this scope. The algorithm is basically a coarse to fine matching process. In a probable range with n frames, the order of getting the background images in the database for binary hierarchical matching is as follows, and the decision rule of the best background image fitting is based on the use of the correlation coefficient γ between the current captured frame and the current accessed background frame in the database: if γ is larger than a threshold δ, then the searched frame is determined to be the best. This matching method helps reducing the frequency of comparison.

Step 1: Get the central frame F_c of the probable range.

Step 2: Get the two end frames of the probable range, which are $n/2$ frames away from F_c, and select from them the one, say F_H, with the higher correlation with the input capture frame. Let R denote the search scope which is set to be $n/2$ frames.

Step 3: Get the two frames in the probable range, which are $R/2$ frames away from F_H, and select from them the one, say F_H, with the higher correlation with the input capture frame. If the computed correlation γ of F_H is larger than a threshold δ, then take F_H to be the desired best fit background image; otherwise, set the new search scope R as a half of the old one, i.e., set (new R) = (old R)/2, and continue.

Step 4: Repeat Step 3 until the search scope decreases to be a pre-selected limit or until a certain number of iterations is reached. At that time, take the F_H with the largest correlation γ as the desired most suitable background, and stop.

When some objects appear in the captured frame, the correlation coefficient γ cannot always be larger than threshold δ because the monitored scene is changed by these objects. If we cannot find the best-fitting background image after too many iterations, we determine the frame with the maximum correlation coefficient in all of the frames that have been searched before is the most suitable background image, as is done in Step 4 in the above algorithm. By this way, we can avoid the situation that the searching cannot be finished. The time complexity of this method is O(log n), and the time complexity of the usual way of searching all the background frames in the probable range is O(n). Obviously, the employed binary hierarchical matching method is effective to improve the

efficiency of the rotational surveillance system.

4.4. Dynamic Background Update for Rotational Monitoring

Like the same processes in a static surveillance system, we must ensure that there is no moving foreground object in the monitored scope during the interval of background update. Since the data size of a sequence of background images around a circle (360 degrees) is too large, we do not perform background update if the background scene is changeless. An algorithm of background update in a rotational surveillance system is described as follows:

Step 1: Record the positions where objects appear in the monitoring stage.

Step 2: Gather statistics on the appearance frequency of objects before background update.

Step 2.1: If the frequency is very low or very high, cancel the operation of background update.

Step 2.2: If the frequency is not very low or high, cluster the positions where objects appear. If the number of positions in a cluster is too small, discard it.

Step 3: Count the number of clusters. If this number is too big, cancel the operation of background update.

Step 4: Save the images at positions of the clusters and regard them as temporary background images.

Step 5: Capture frames at the positions of the clusters. Perform object grabbing in these frames by comparing them with the temporary background images.

Step 5.1: If no object appears in all the positions of a certain cluster, perform background update in the regions of this cluster by taking the temporary background images of the regions as the background images.

Step 5.2: If some objects appear in the positions of a cluster, discard the regions of these positions.

5. Client-Server Interaction for Centralized Monitoring and Environment Image Display

After the image sequences with foreground objects have been compressed into several compressed object videos at the client site, they need to be displayed at the server site; however, videos that only include foreground objects cannot be regarded as complete surveillance videos. Fortunately, background images have already been stored in the database of the server site, and combining foreground objects with the background can create the complete video. In order to display the scene correctly, the client computer should send not only the video of objects but also information about the current video. The server computer will then know how to display the video suitably by using these data. Besides these operations, the server computer needs to understand how to produce the rotational effect, just like the actual rotational scene in a rotational surveillance system.

Finally, the video should appear to be like never being processed, so we need to conduct boundary smoothing when the video with foreground objects is superimposed on the background images. The illumination of the background picture and the decompressed frame with objects may be different, so combination of them may create seams. If we ignore this effect, the results of integrating the background image and the foreground object will be unsatisfactory because the blocky effect is obvious. In order to handle the illumination difference in

background images and videos with objects, we need to change the illumination of background images to accommodate the new environment. If the illumination in the surveillance video is low, then the illumination of the background image needs to be decreased. The method used to deal with the illumination of the two frames is called *color calibration*. Since the illumination is the most important factor that we need to care for in the procedure of color calibration, the YUV model can be used here again because the illumination (Y) component of the YUV model can be processed separately and independently. And now we can get the new YUV values and the corresponding RGB values of each pixel in the background image by modifying their Y values by a method of color calibration, as described by (5.1) and (5.2) below:

$$Y_{new} = K_y \times Y_{background} \tag{5.1}$$

$$K_Y = \frac{\sum_{i=1}^{n \times m} Y_{monitored\_scene}}{\sum_{i=1}^{n \times m} Y_{background}} \tag{5.2}$$

Where m is the width of the frames, and n is the height of the frames. In the above formula, K_y is defined to be the average of the illumination of the monitored scene divided by the average of the illumination of the background image. The global illuminations in the current monitored scene are calculated in the monitoring stage at the client site when objects appear in the monitored scope.

The result of combining the foreground and the background after color calibration may be not perfect. We use the alpha-blending technique to smooth the seams by margining the sub-image into the background image. The method of alpha-blending is a procedure of combining a translucent foreground color with a background color. By this way, a new blended color is produced.

6. Experimental results

In this study, the results of compressed videos are displayed in the sever site. The server computer creates complete surveillance videos by combining foreground object videos with suitable background images. Some frames in foreground videos are shown in Fig. 2, and Fig. 3 is the resulting video of combining the foreground object with the background scene in a rotational surveillance system.

Fig. 2 Some frames in foreground videos in a rotational surveillance system.

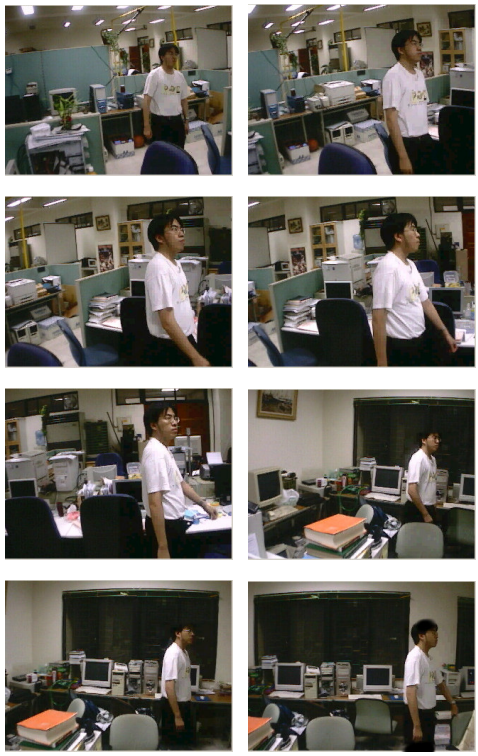

Fig. 3 Some resulting videos in a rotational surveillance system.

7. Conclusions

In this study, static and rotational surveillance systems with the capability of high-rate video compression have been proposed. For the proposed system, effective methods for surveillance video analysis based on the concept of content-based video compression have been proposed, which can be employed to reduce the amount of the transmitted video data with better qualities The methods essentially try to remove the changeless background scene and merge close foreground objects together to increase the compression ratio.

REFERENCES

[1] Tomas Sikora, "MPEG digital videoW-coding standards," *IEEE Signal Processing Magazine*, Vol. 14, No. 5, September 1997.

[2] Tomas Sikora, "The MPEG-4 video standard verification model," *IEEE Transaction on Circuits and Systems for Video Technology*, Vol. 7, No. 1, February 1997.

[3] Touradj Ebrahimi, "MPEG-4 video verification model: A video encoding/decoding algorithm based on content representation," *Signal Processing: Image Communication 9*, 1997.

[4] R. Yalluri, K. Oehler, T. Bannon, J. Courtney, A. Das, and J. Liao, "A robust, scalable, object-based video compression technique for very low bit-rate coding," *IEEE Transaction on Circuits and Systems for Video Technology*, Vol. 7, No. 1, February 1997.

[5] Isabel Martins and Luis Corte-Real, "A video coder using 3-D model based background for video surveillance application," *Faculdade de Engenharia da Universidade do Porto/INESC*, 1998.