ABSTRACT

Conventional mosaicing techniques convert a video from frame-based representation to scene-based representation, but they usually lack dynamic information so that their mosaic is not complete. In this paper, we present a novel method to detect moving objects in the video sequences, then add them into the static background mosaic to represent the scene completely. This novel algorithm separates static and dynamic information in a video sequence, builds the background mosaic from static part and reconstructs moving objects on the static mosaic. We have implemented our techniques and the experimental results demonstrate the effectiveness of our approach.

1. INTRODUCTION

Mosaicing techniques represent video sequences in scene-based format and remove the redundant information. Usually, video sequences have a large amount of information redundancy because there is a large overlap of information across consecutive frames. If we “stitch” the frames from the same scene together and remove the overlapped part, we get a large panoramic image which represents the original scene without any redundant information.

Traditional mosaicing techniques mostly deal with static scene, but we argue that a complete mosaic should also include the dynamic part. In this paper, we present an algorithm for building dynamic video mosaic. It can not only represent the background completely, but also reconstruct the moving objects.

This algorithm has many kinds of applications, for example, video resolution enhancement, panoramic video representation, mosaic based video coding schema, shake artifact removal, etc.

The rest of this paper is organized as follows. Section 2 describes the related work. Our algorithm is presented in Section 3. We apply the algorithm on real video and get pleasing results, as given in Section 4. In Section 5, we give conclusion and discuss the future work.

2. RELATED WORK

We will explore two areas of previous research in this section: static mosaic and motion detection.

Panorama mosaic as a field of research came into existence at the beginning of 1990’s. The present techniques can be roughly divided into two categories: hardware-based [12] and software-based, and the latter is our focus.

In an early work, Anandan et al. [1] give the traditional way to find correspondence and make motion estimation: minimization of sum-of-squared-difference (SSD), which forms the basis of many present techniques. Later, various kinds of mosaicing techniques are presented. Among them, Bhosle et al. [2] describe a method using geometric hashing algorithm to reduce time complexity. Gonzalez et al. [5] provide a novel method in mosaic construction to deal with looping problem, through distributing the accumulated error of positions of all images in the mosaic. All these methods work in pixel spatial domain, while Jones et al. [11] propose a method in compressed domain, which can compute the camera motion directly from the motion vector encoded in a MPEG video stream.

However, these techniques all deal with static scenes. Irani et al. first propose the concept of “dynamic mosaic” in [8]. This mosaic is updated every time with the current input image, but without layer separation or object detection. Later, Bhosle et al. [3] present a method, which does background extraction, then removes the moving objects from the final mosaic. The result is for static background only. In [4], Davis provides another algorithm, which detects the moving objects and places them in one single position along their trajectory. The result is for background with static objects. In [9], an improved algorithm is presented, which can reconstruct the complete trajectory of the moving objects on the mosaic. However, since only the trajectory is displayed, we still don’t know the exact shape of objects.
Compared with these techniques, our work not only provides the whole background, but also displays objects’ trajectory as well as objects themselves. Therefore, we build a more complete mosaic than those of existing techniques.

3. ALGORITHM

In this section, we describe our algorithm, which can be divided into four steps: static mosaic construction, coarse frame registration, moving objects detection, and moving objects reconstruction.

3.1. Static Mosaic Construction

The technique for static mosaic construction has been explored by many researchers, therefore, in this step, we just follow the work done by others.

To get the latest information from the video, we build dynamic mosaic [8], which increases every time when a new frame is added. If the scene is static, we can calculate the sum-of-square difference (SSD) of the pixel values to measure the match quality, as shown in equation (1).

\[ E = \sum (I(r) - J(r))^2 \]  

where \( I \) and \( J \) are intensity images and \( r \) denotes pixel positions.

After we get the correspondence information, we use 2D 8-parameter perspective motion model [13] to build the mosaic.

Since we deal with dynamic scenes, we use color selection method in image integration, otherwise, the color combination method will introduce blurring in moving parts.

3.2. Coarse Frame Registration

Now we get a complete mosaic consisting of all input frames, and a list of motion matrices for each input frame. In this section, we will describe how to register each input frame on the mosaic coarsely (without any detection or separation).

To do image integration in mosaic construction, we use backward homographies which map the mosaic to the input frames. Since the mosaic we built is a dynamic one updated backward homographies which map the mosaic to the input image 2, named the source image, and \( H_{BS} \) denotes the backward matrix which maps the mosaic to the input image 2, named destination image. \( n \) denotes the number of video frames.

Every final reference matrix maps one of the corresponding input images to the final mosaic. We can use these matrices to do the coarse frame registration. After this coarse registration, we have a series of mosaic images, and on each of them one input frame has been registered. These images represent a complete background, on which dynamic frames move following the trajectory of camera motion.

3.3. Moving Objects Detection

After the above two preparation steps, now we discuss the important part to detect the moving objects and perform layer separation. Motion detection can be generalized to change detection, and changes can be detected at different levels: pixel, edge, or region. To make the detection simple and efficient, the method we present here is a combination of the three.

Firstly, we compute the difference picture between coarse registered mosaics using equation (3).

\[ DP_{jk}(x, y) = \begin{cases} 1 & \text{if} \quad |F(x, y, j) - F(x, y, k)| > \tau \\ 0 & \text{otherwise} \end{cases} \]  

where \( H_{FR} \) denotes the final reference matrix, \( H_{BS} \) denotes the backward matrix which maps the mosaic to the input image 1, named the source image, and \( H_{BD} \) denotes the backward matrix which maps the mosaic to the input image 2, named destination image. \( n \) denotes the number of video frames.

The result is a binary image indicating the changed parts between images. However, one assumption of this simple method is that the camera is static. If there is camera motion and the whole scene is moving, the difference picture
is almost useless. Actually, our preparation steps aim to remove the camera motion by mosaicing the background, and make the moving objects simply detectable.

What we actually want to detect is motion, but difference picture shows the changes which can also be caused by other factors in addition to motion, such as illumination changes and misregistration. To detect the motion more accurately, we choose to compute the optical flow [6] as another measurement for motion detection. After we get the optical flow $u(x, y), v(x, y)$ of every pixel, we do a block-based detection above the pixel-based difference picture. The motion value of each block is calculated as equation (4).

$$
\text{M}_{\text{block}} = \sum_{S_{\text{block}}} \sqrt{u_{(x,y)}^2 + v_{(x,y)}^2} \quad (4)
$$

where $\text{M}_{\text{block}}$ indicates the motion property of the block, $S_{\text{block}}$ means this computation is done within the size of one block, and $\text{DP}(x, y) = 1$ means we only include the pixels in changed parts.

If the $\text{M}_{\text{block}}$ value of one block exceeds a certain threshold, this block is primarily detected as moving. To make the detection result better, besides using pixel-block combined method, we still make some improvements according to the general properties of video sequences.

One of most obvious properties we can use is motion constancy. In [7], motion constancy is an assumption which assumes that motion remains uniform in the analyzed sequence. According to this, one block will be considered noise unless there is at least one moving block in its neighborhood in the next frame.

The other property we can utilize is camera focus. Generally speaking, if there are moving objects in the shoot scope, videographers follow these objects and place them in the center part to attract more attention from viewers. Therefore, if the detected moving blocks are very close to the edge of the frames, we can assume that they are noise blocks and discard them safely.

The whole procedure in this step is shown in Algorithm 1.

3.4. Moving Objects Reconstruction

Till now, we have got the complete static mosaic, the estimated reference matrix, and the moving blocks in each input frame. Using these information, we build static background without moving objects and reconstruct those objects on the final mosaic.

Firstly, we separate the detected moving objects from the background. We do pixel compensation instead of simply resetting pixel value to zero to avoid black holes in the image. This method is to replace the pixel value in the moving parts in one input image with the average value of all the corresponding pixels from background parts in other input images.

Secondly, we construct the mosaic again as we have done in our first step, but this time with static background parts only.

Thirdly, we reconstruct the detected moving blocks. The reconstruction is based on pixel although the detection results are based on block. This is because our block-based detection doesn’t follow the boundaries of objects accurately. If we place the whole blocks, there will be a visible box around the moving objects. Therefore, we only select suitable pixels in the moving blocks to replace the background pixels based on the difference between their intensity values.

4. EXPERIMENTAL RESULTS

In this section, we use some video examples to evaluate our approach. We show the results in our homepage and only give the analysis and discussion here.

All these experiment results can be accessed in our website given by URL: http://www.comp.nus.edu.sg/~shenhui/project.

In this site, we provide input video clip, static mosaic, segmented objects, and output video mosaic for several examples.

The examples can be summarized using Table 1 (The test platform is Pentium IV 1.6G with Windows 2000. All examples have 10 frames.).

From Table 1, we find that there are quite a few false alarms in coarse detection, because it is easily influenced by the mosaic alignment. However, our two improvement methods can eliminate most of the noise and smoothen the results.

Furthermore, we should point out that the scenes in our

<table>
<thead>
<tr>
<th>input</th>
<th>Intensity Image 1: $J$, Intensity Image 2: $K$</th>
</tr>
</thead>
<tbody>
<tr>
<td>output</td>
<td>Moving Blocks: $MI_{B1}, MI_{B2}, \ldots, MI_{Bn}$</td>
</tr>
</tbody>
</table>

**Procedure:**
1. Calculate the difference picture between $J$ and $K$
   $$DP_{JK} = \text{Difference}(J, K).$$
2. Calculate the optical flow between $J$ and $K$
   $$(u, v) = \text{OpticalFlow}(J, K).$$
3. Create a moving image and divide the image into blocks
   $$MI = (MI_{B1}, MI_{B2}, \ldots, MI_{Bn}).$$
4. Calculate the motion value for each block. If this value exceeds a threshold, we primarily detect this block as moving
   $$M_{\text{block}} = M(u, v, DP).$$
5. Discard a moving block if its motion is not constant.
6. Discard a moving block if it is close to the edge.

**Algorithm 1:** Moving Objects Detection
### Table 1. Quantitative Evaluation of Examples

<table>
<thead>
<tr>
<th>Property</th>
<th>Ex 1</th>
<th>Ex 2</th>
<th>Ex 3</th>
<th>Ex 4</th>
<th>Ex 5a</th>
<th>Ex 5b</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Size</td>
<td>320 x 240</td>
<td>320 x 240</td>
<td>320 x 240</td>
<td>320 x 240</td>
<td>320 x 240</td>
<td>320 x 240</td>
</tr>
<tr>
<td>Block Size</td>
<td>15 x 15</td>
<td>20 x 20</td>
<td>20 x 20</td>
<td>15 x 15</td>
<td>20 x 20</td>
<td>25 x 25</td>
</tr>
<tr>
<td>Time (ms)</td>
<td>15163</td>
<td>14671</td>
<td>18456</td>
<td>15152</td>
<td>17164</td>
<td>17796</td>
</tr>
<tr>
<td>Coarse Blocks</td>
<td>275</td>
<td>152</td>
<td>384</td>
<td>482</td>
<td>405</td>
<td></td>
</tr>
<tr>
<td>Refined Blocks</td>
<td>109</td>
<td>75</td>
<td>83</td>
<td>82</td>
<td>119</td>
<td>100</td>
</tr>
</tbody>
</table>

examples are similar so that our algorithm seems not to be applicable to various situations. Actually, this constraint is caused by our quite simple mosaic construction method. It requires the scene to be almost planar and without motion parallax. Meanwhile, the objects in the scene should have great contrast with the background. This assumption is reasonable because otherwise, it is even difficult for human beings to detect the objects, let alone computer vision techniques.

Therefore, a better mosaicing technique for more complex scenes can help to generalize our algorithm. Also, we can utilize more parameters to detect the moving objects because we find the coarse detection still has a long way to go. For example, there are some noise blocks in several experiments, and these noise blocks cause problems again in later steps.

## 5. CONCLUSION

In this paper, we have implemented a novel system to separate the moving parts from the static parts in video sequences, then reconstruct the moving parts on the background mosaic, which is built from the static parts. The most important contribution of our work is that our new algorithm is able to retain all the information in dynamic scenes, especially the information of moving parts, which is usually ignored in the past work. However, our work is only a preliminary step and much more future research needs to be done, for example, the illumination change problem, more accurate motion detection, the mosaic extension between different shots, etc. Actually, our work can be viewed as a starting point for a novel representation technique which can completely decompose video sequences into some descriptors and reassemble them in a new format, according to users’ needs or application requirements.

## 6. REFERENCES


