RELATIVE DEPTH LAYER EXTRACTION FOR MONOSCOPIC VIDEO BY USE OF MULTIDIMENSIONAL FILTER

Jing-Ying Chang, Chao-Chung Cheng, Shao-Yi Chien, and Liang-Gee Chen

DSP/IC Design Lab.,
Graduate Institute of Electronics Engineering and Department of Electrical Engineering,
National Taiwan University, Taipei, Taiwan
Email: {jychang, fury, sychien, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

This paper presents a relative depth layer extraction system for monoscopic video, using multi-line filters and a layer selection algorithm. Main ideas are to extract multiple linear trajectory signals from videos and to determine their relative depths using the concept of motion parallax. The proposed superficial line model used for detecting slow moving objects provides sufficient taps within few frames to reduce frame buffer, while the closest-hit line model used for detecting fast motion objects provides few enough taps to prevent blurring. To increase the correctness of layer map, three-level layer map co-decision is used to compensate low texture region defect.

1. INTRODUCTION

Depth maps of two dimensional (2D) image and video set are essential for image-based rendering (IBR) and three dimensional (3D) environment reconstruction. To feel the distance of an object, human understand the world's 3D structure by use of several perceptual cues [1]. But for 2D media, some important cues are missing so that we can not easily reconstruct depth maps. Fortunately, we still have the opportunity to find motion parallax information along the time axis in a video sequence. Motion parallax comes from the phenomenon that objects closer the camera should have higher speed in image plane when the camera pans.

For multi-view image sources, many depth map generation algorithms based on stereo disparity exist nowadays. But the algorithms for monoscopic video are still evolving. Reference [2], [3] propose a segment-based motion estimation (SBME) algorithm and then converting motion vector of each segmented piece into a depth value. This algorithm focuses on the case that a camera is panning across a static scene. For those objects with their own motion, it is hard to judge their depths. Moustakas et al. [4] use object-based bidirectional motion estimation and Bayesian occlusion handling to generate stereoscopic video. Disparity estimation is a good choice to extract depth map. By use of this method, two successive frames are treated as inputs of stereoscopic video. Similar to the situation of SBME, disparity estimation is suitable for static scene only. Meanwhile, these methods requires many pre- or post-processing steps like segmentation and occlusion handling. Another way to extract depth maps employs the characteristic of linear trajectory (LT) signal. Frequency planar filter and LT filter are two related filters to extract LT signals. Previously, they are used for object detection, object tracking, and image rejection [5][6]. Because these filters detect the object speed and moving direction during a sufficient time interval, we can consider the velocity factor as another form of motion parallax. This approach assumes the camera is panning at a constant velocity, but the scene need not be completely static. This method still holds as long as relative constant velocities of objects appearing on frames do not violate the assumption of motion parallax.

The system we propose here uses multiple LT filters with closest-hit and superficial line models to detect the trajectory of objects and different depth layers. The advantage of LT filters is that they can work fine without many pre- or post-processing works, and then they can be applied on more general cases. The proposed closest-hit and superficial line models used in multiple LT filters are designed correspondingly for fast and slow motion objects due to each case requires different degree of pixel correlation. After filtering, we combine filtered results from three videos with different frame sizes to determine the relative depth of an pixel. Our system focuses on two applications, stereo scenery museum from travel program and 3D space reconstruction for surveillance system. In most cases of these applications, panning is the main function of cameras. Many resulting sequences satisfy the assumption of motion parallax so that we apply trajectory filter to our system.

This paper is organized as follows. Section 2 describes the concept of LT signal and the relationship to non-accelerating rigid objects. In Section 3, we present the proposed algorithm, which includes two line models, depth map extraction, and layer generation. Section 4 shows subjective depth maps using different line models and the relative depth layer extraction result. Section 5 is dedicated to concluding remarks and future research.

2. LINEAR TRAJECTORY SIGNALS

$x(t_1, t_2, ..., t_M)$ is a multi-dimensional (MD) LT signal if there exists a direction $v \in t^M$ such that $\frac{\partial(t_1,t_2,...,t_M)}{\partial v} = 0$. In a video sequence, if one rigid body moves along certain direction with fixed speed, this kind of object can be considered as a 3D LT signal within 2D-spatial and time domain. From the view of true 3D LT signal, the signal value will not change along the trajectory direction. To slice this 3D signal with a group of parallel planes perpendicular to the direction, images shown in all planes will be identical. Furthermore, in Fig. 1, all parallel planes which their included angles between the vector and themselves are not 0 degree still slice the signal with same images shown on them except a translation. That is, if we take $\mathbf{t}_3$ as temporal axis and $\mathbf{t}_1$ and $\mathbf{t}_2$ as spatial axes, this signal is presented in a way similar to a video sequence.

For a 3D LT signal, there is no value change along the traveling direction. It means, in frequency domain, this kind of signal lies on

This work was supported by the National Science Council, R.O.C., under Contract NSC94-2215-E-002-029.

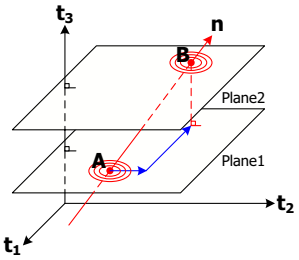


Fig. 1. For a LT signal with vector n and passing through point A and B, two parallel planes (P1 and P2) with normal vector t_3 slice the signal. The images shown in P1 and P2 are identical except a translation. This signal is presented in a way similar to video frames.

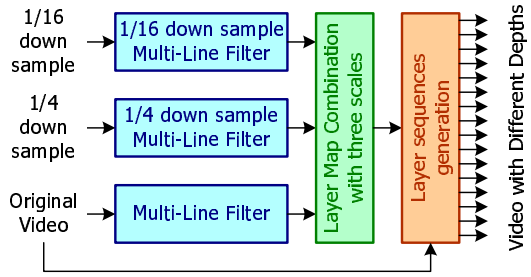


Fig. 2. System block diagram. The original video is resized and passed to the multi-line filter. After three level filtering, three layer maps are combined to remove noise. The combined result and original video are used in generating sequences for different depths.

a plane perpendicular to the traveling direction. To extract a rigid body with fixed moving speed from one video sequence, it can be done by filtering its frequency plane, which can be realized by two ways. The first one is to transform the video into frequency domain. After processed by frequency plane filter, the filtered result is then transferred back to spatial domain. Another way is directly to convolute the video and filter band in spatial domain. Because filtering one plane window in frequency domain is similar to averaging with a equal-weighting line in spatial domain, convolution is easier to realize and requires no frequency transformation [6].

3. ALGORITHM

To extract relative depth layers from a monoscopic video with a static scene, each layer can be seen as one rigid body when a camera is panning at a constant velocity. As described in Section 2, we can extract one layer by applying a line filter with known moving speed and direction. The system block diagram of relative depth layer generation is shown in Fig. 2. The original video is resized to $1/4$ and $1/16$ first. The generated sequences are then passed to the three-level multi-line filter. The reason to use three-level filter will be described in Section 3.2. The output layer maps are combined to remove noise. The final depth layer map and original video are used in generating sequences for different depths. The key component of this system is the multi-line filter. In Fig. 3, three core operations of multi-line filter are the line filter generation, filtering, and best layer candidate selection.

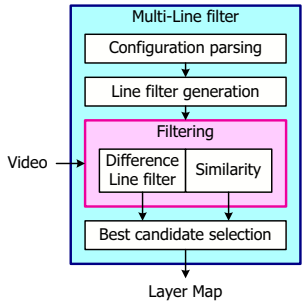


Fig. 3. Multi-line filter block diagram. "Difference line filter" checks the difference between filtered result and original signal. "Similarity" checks the unity of pixels within filter window.

3.1. Line Model

Since the line filter should be an average filter along the moving direction, a good filtered result depends on an appropriate line model to present the vector. Instead deriving the line from inverse transformation of a plane in frequency domain, we model the line in spatial domain.

Due to the sequence is composed of discrete signal, line width and weighting of each pixel should be taken care to make the discrete line look closer to the line in continuous form. Two traditional models of line are integer pixel model and overlapped pixel model, shown in Fig. 4, where each square represents a pixel, and the line represents the selected vector. The filter is then applied on the shaded squares. We can get clear filtered result when applying integer pixel model. However, due to few sample points caused by the fact that the line do not cross most centers of pixels, this model usually requires large window for more filter taps under the condition that motion correlation still holds. For example, we have to buffer more frames for slow motion object and buffer larger 2D spatial window for fast motion object. Besides, the most serious shortcoming of overlapped pixel model is that all equal weighting taps make the filtered result blur. This situation is more noticeable for fast motion vector because the line overlaps several neighboring pixels in one frame. Anti-aliasing line model is an extension of overlapped pixel model. According to distances from pixel centers to the line, each pixel has different weighting coefficients. This model alleviates blur result in overlapped pixel model, but in fast motion, blur defect still exists because of average from too many neighboring pixels.

By observing natural videos, two different line models, superficial line model and closest-hit line model, are proposed for slow motion and fast motion. Fast motion is defined when speed is faster than 1 pixel/frame, and slow motion is defined when speed is slower than 1 pixel/frame. If moving speed is not integer pixel/frame, colors of the corresponding area of one object in successive frames will have some difference. This situation can be seen as under sampling in spatial domain, and then one object in a frame may not be identical with itself in next frame. For example, Fig. 5 shows two relative regions of chair back shifting left in the slow motion video "panslow" in successive frames. Since the moving speed is not integer pixel/frame, to match the regions requires different weighting for neighboring pixels. The situation described above happens in both slow motion and fast motion. But for fast motion objects, the interpolation effect only happens between two or three neighboring pixels at most in one frame. This situation can be seen as under sampling in time domain. It is unsuitable to use the model which has

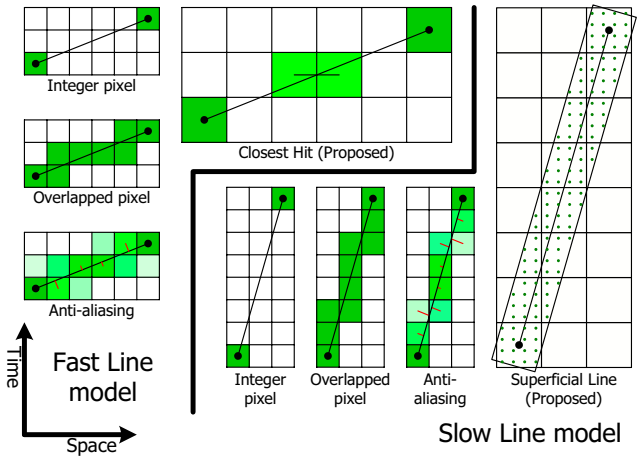
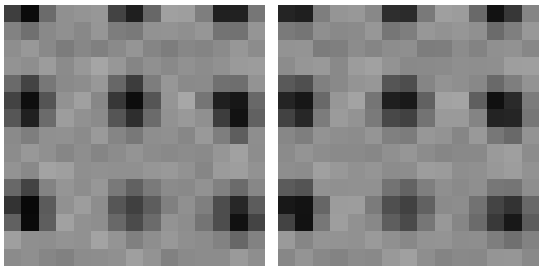


Fig. 4. Fast (upper left) and slow (lower right) motion line model.



(a) (b)

Fig. 5. Two relative regions of chair back shifting left in the slow motion video sequence "panslow" in successive frames.

several coefficients in a frame for fast motion objects. This is why we use different line models for fast and slow motion objects.

Closest-hit line model is adopted for fast motion object. For a 2D line (time axis and 1D spatial axis), the weighting coefficient of each pixel is defined as

$$W = \begin{cases} 1-d & 0 \le d \le 0.5 \\ 0 & \text{ohterwise} \end{cases}$$

d is the distance between the pixel center and the line along spatial direction. This model can make sure that no more than two coefficients in one frame. For slow motion object, we use superficial line model for filtering. To form a superficial line, one 1-pixel-wide line is drown on the filter window first. Next the weighting of each pixel is defined as the overlapped area between the pixel and the 1-pixel-wide line. We simplify the area calculation by using the hit rate of quarter pixels, like the dots in Fig. 4.

After generating two 2D line models for x- and y-direction speed, 3D line model is formed by multiplying this two lines, as shown in Fig. 6. This method provides fast and slow motion separation consideration for x- and y-direction, since most situations are the cases where cameras pans faster along horizontal direction and slower along vertical direction.

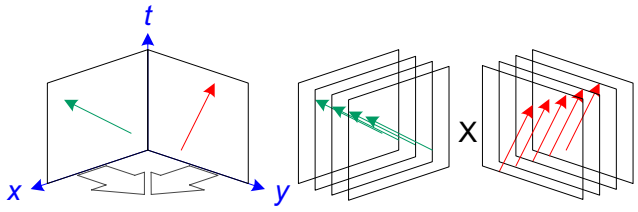


Fig. 6. Multiply x-t line and y-t line to generate 3D line

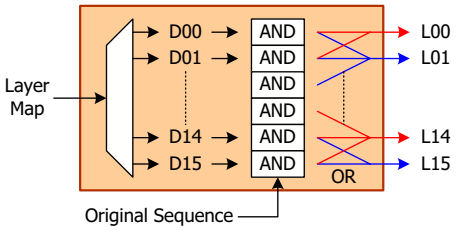


Fig. 7. Block diagram of layer sequences generation.

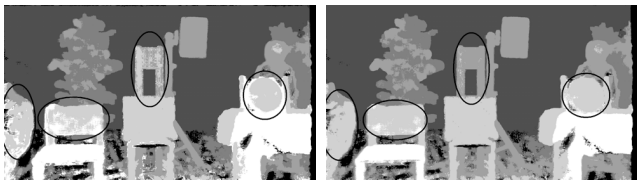
3.2. Filtering and Layer Extraction

Generated line models are applied to difference line filter and similarity block. Difference line filter is a combined operation of line filter and subtraction between filtered result and original frame. If the output of one difference line filter is more closer to zero, the vector is more possible to be the moving speed and direction of the object. In addition to examining the difference result, the overall similarity between pixels in filter window and the pixel at filter center is calculated for co-decision-making of best layer candidate. Higher percentage of similarity means the vector is a probable candidate.

These two filter operations give a difference map and a similarity percentage map for each frame. Before using them for co-decision-making, each map passes through an $N \times N$ averaging window to smooth layer decision result and eliminate peak noises. The value N is decided according to the frame size which appropriate correlation range is related to. At the co-decision step, values of the two maps are normalized to the same order. Finally, best layer candidate for one pixel is chosen according to the vector with the biggest sum of the two corresponding map values.

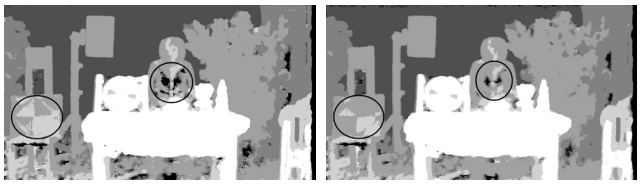
After generating three scale relative depth layers from multi-line filters, the resized versions are up-scaled back to original size. Sometimes, we will get best result from several vector models so that we can not tell which vector is true for this area. This situation can be alleviated by applying multi-line filter to three scale videos because down-sampled video also reduces the size of low texture area. Every pixel in the final layer map is decided to one of layers which the layer value at the position is the closest one to the average value of the three layers.

The final stage of our system is "layer sequences generation". Layer sequences are derived from the "AND" operation between original video and each layer. Since each object may exist across several depth layers, to make each object look more completed, three neighboring layer sequences are "OR" together as shown in Fig. 7. In this example, we generate 16 relative depths by non-linear quantizing the magnitude of velocity.



(a) Integer pixel model. (b) Superficial line model.

Fig. 8. Line model comparison.



(a) Original layer map. (b) Combined layer map.

Fig. 9. Three-level layer map comparison.

4. EXPERIMENTAL RESULTS

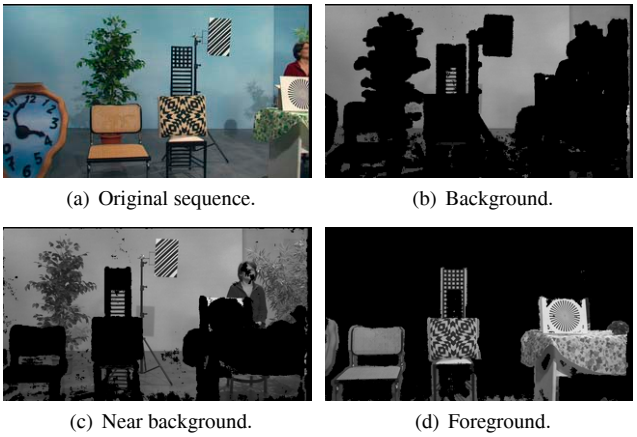
Fig. 8 shows the improvement of proposed line model. There are some broken layers on two chair backs and a discontinued layer around the circle (indicated by black rings) when using integer pixel model. This defect has been removed when using superficial line model. Furthermore, the advantage of three-level layer combination is shown in Fig. 9. Low texture on human body (the right ring) and discontinued layer on cushion surface (the left ring) are alleviated when we use lower-level layer map to compensate this low texture hole. Fig. 10 and Fig. 11, where we use slow line model and fast line model respectively, show the final relative depth layer results. In these results, some pieces belonging to the same object are classified into different layers and make it looks imperfect. This kind of situation happens mostly in two cases. One is that an object is not a LT signal. The other is that the size of low texture area is too wide. But we still can see the foreground and background layers in "panslow" and human layer in "night" are extracted clearly.

5. CONCLUSION

In this paper, the system of relative depth layer extraction for monoscopic video using multi-line filter is proposed. The superficial line model provides sufficient taps within few frames to reduce frame buffer, while the closest-hit line model provides few enough taps to prevent blurring. To increase the correctness of layer map, three-level layer map co-decision is used to compensate low texture region defect. This system can be extended to generate relative depth from compressed video. With motion vector distribution in a compressed video, we do not need to use general vector set to filter the video and then we may reduce computational complexity.

6. REFERENCES

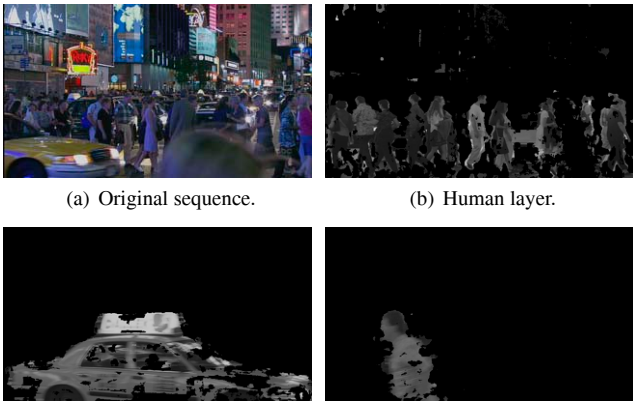
[1] Neil A. Dodgson, "Autostereoscopic 3d displays," *IEEE Computer*, vol. 38, no. 8, pp. 31–36, 2005.



(a) Original sequence. (b) Background.

(c) Near background. (d) Foreground.

Fig. 10. "Panslow" layer sequence.



(a) Original sequence. (b) Human layer.

(c) Foreground. (d) Foreground.

Fig. 11. "Night" layer sequence.

[2] Fabian Ernst, Piotr Wilinski, and Kees van Overveld, "Dense structure-from-motion: An approach based on segment matching," in *European Conference on Computer Vision*, 2002, pp. 217–231.

[3] Harm Peters et al., "Application specific instruction-set processor template for motion estimation in video applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 4, pp. 508–527, Apr. 2005.

[4] K. Moustakas, D. Tzovaras, and M. G. Strintzis, "Stereoscopic video generation based on efficient layered structure and motion estimation from a monoscopic image sequence," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 8, pp. 1065–1073, 2005.

[5] Boaz Porat and Benjamin Friedlander, "A frequency domain algorithm for multiframe detection and estimation of dim targets," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 398–401, 1990.

[6] Soo-Chang Pei, Wen-Yi Kuo, and Wan-Ting Huang, "Tracking moving objects in image sequences using 1-d trajectory filter," *IEEE Signal Processing Lett.*, vol. 13, pp. 13–16, 2006.