Background Modeling Using Depth Information

Yu-Lun Liu and Hsueh-Ming Hang
Department of Electronics Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C.
E-mail: alex04072000@hotmail.com, hmhang@mail.nctu.edu.tw

***Abstract*— This paper mainly focuses on creating a global background model of a video sequence using the depth maps together with the RGB pictures. The first key concept is the near objects block the scenes at the back. With the aid of depth information, we can identify the closer moving objects. Secondly, we develop a recursive algorithm that iterates between the depth map and color pictures. Comparing to the existing schemes, our proposed method can produce better quality background images and it improves the background depth map at the same time.**

I. INTRODUCTION

A general foreground detection using background modeling needs to overcome a few common problems: (1) illumination changes including changes gradually or suddenly, (2) slowly moving background objects such as movement of audience or the presence of dropping leaves in background, and (3) changing background.

In the previous studies, several approaches of constructing the background model have been proposed. Generally, people use the color and motion information, and the other characteristics of the background as features. Then, a model is developed for the background images. The most commonly used method is the Gaussian mixture model [1][2], the nonparametric model [3], the codebook background model [4], and others.

II. BACKGROUND MODELING USING DEPTH INFORMATION

A. Main concept and problems

3DV has an advantage that it contains depth information compared with the ordinary video sequences. Hence, there might be a way for us to model the background using simpler or faster algorithms based on the depth information. Typically, the stationary background has the largest depth value or the smallest disparity in a scene. Thus, we first collect the minimum disparity values of the whole scene over the entire sequence. We first try the time wise min filter applied to every pixel to form the initial background disparity map,

$$D_{min}(u,v) = \min_{1\le n\le N} D(u,v,n), \quad (1)$$

where D_{min} is the minimum background disparity map, N is the total number of frames of a video sequence, and D is the original disparity sequence. Fig. 1 shows the D_{min} map of Poznan_Street sequence.

At the first look, the overall disparity map seems to be acceptable. But with a close look, we can find some wrong disparity values, which are farther than they are supposed to be. We enlarge some of these areas in Fig. 2.

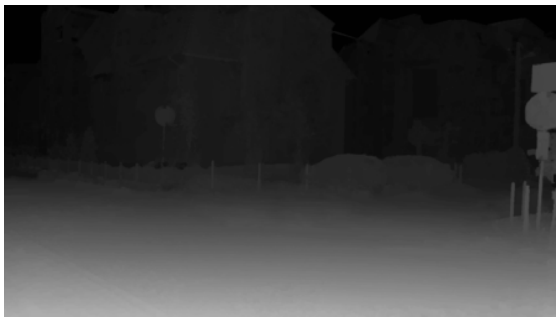


Fig. 1 D_{min} of the Poznan_Street sequence.

Fig. 2 Enlarged disparity errors.

These artifacts occur due to inaccurate depth estimation algorithm or active depth sensors, which contain noises in the disparity maps. Typically, the incorrect disparity pixels do not occur very often in the entire sequence. Hence, how to create a robust background texture image and depth map with reduced erroneous depth pixels is our goal.

B. Proposed depth-based background modeling algorithm

We develop an iterative algorithm which contains two elements: 1) extract the minimum disparity values, and 2) update the region of interest (ROI), which indicates the background region. The flow chart of the proposed algorithm is shown in Fig. 3.

The proposed algorithm mainly contains 4 steps: (1) construct an initial background color image using the depth information. (2) Create/update the ROI based on the initial/updated color background image. (3) Construct the background depth map based on the ROI. (4) Construct the color background image based on the ROI. The last 3 steps are repeated iteratively to produce the final and better results.

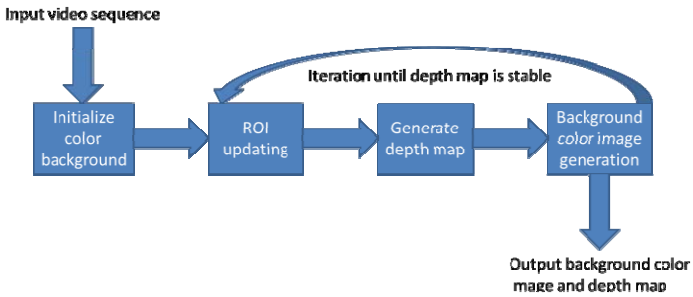


Fig. 3 Flow chart of the proposed depth-based background modeling algorithm.

1. *INITIALIZE COLOR BACKGROUND*

We make assumptions that: (1) the background pixels should have the relatively larger depth values, and (2) incorrect depth values do not appear often. We first construct the initial color background image by:

$$I_0(u,v) = \frac{\sum_{n=1}^{N}\left(I(u,v,n)\times\frac{1}{D(u,v,n)-D_{min}(u,v)+1}\right)}{\sum_{n=1}^{N}\frac{1}{D(u,v,n)-D_{min}(u,v)+1}}, \quad (2)$$

where I_0 is the initial background color image. We add 1 in the denominator of the weight to prevent the dividend from infinite. The basic concept of this equation is to use the color pixel having deep depth values to construct the background image. If the wrong depth pixel number is small, the result will be dominated by the majority. A sample of I_0 of the Poznan_Street sequence is shown in Fig. 4.

Fig. 4 I_0 of the Poznan_Street sequence.

The result above shows that weighting average according to reciprocal of difference between the current disparity and time-wise minimum disparity works reasonably well. However, there appears a ghost artifact of a car on the left portion of the background image. This is due to its long stay in the same position. Furthermore, in this process, we like to create a good disparity map that leads to a good color background image.

2. *ROI UPDATING*

In this step, we decide which pixels can be included into the background region. The initial ROI is set as a 3D matrix with all 1, and the dimension is (height of the image)*(width of the image)*(number of frame in the sequence). That is,

$$ROI(u,v,n) = 1, \ for\ all\ u,v,n. \quad (3)$$

Next, update ROI. The concept is to select the pixels that are close to the true background image. At the moment, the true background is approximated by the current mean value. Thus, for each frame, we compare every color pixel value that is close to I_0, and then update the ROI. More precisely, we use the sum of squared difference (SSD) to measure the similarity.

$$difference(u,v,n) = \sum_{R,G,B}\left(I(u,v,n) - I_0(u,v)\right)^2, \forall (u,v,n)\epsilon ROI. \quad (4)$$

Now, we need to find a threshold that is used to decide ROI pixels if the difference in (4) is greater than this threshold. We simply use the mean of the difference in the current ROI.

$$mean_{difference(u,v)} = \frac{\sum_{for\ all\ n\ where\ (u,v,n)in\ ROI} difference(u,v,n)}{\sum_{for\ all\ n\ where\ (u,v,n)in\ ROI} ROI(u,v,n)}, for\ all\ u,v. \quad (5)$$

We update the ROI indices using the following process.

$$ROI(u,v,n) = 0, \quad if\ difference(u,v,n) > mean_{difference(u,v)}, \quad \forall (u,v,n)\epsilon ROI. \quad (6)$$

Based on our notion of ROI, we only need to consider the pixels belonging to ROI. And by removing the pixels in ROI, we can eventually reach a stable result. Fig. 5 below shows the flow chart of this block.

Fig. 5 Flow chart of updating ROI.

3. *GENERATE DEPTH MAP*

After updating ROI (which are mostly the background pixels), the next stage is using the time-wise min filter to generate the background depth map in ROI,

$$D_{iteration}(u,v) = \min_{1\le n\le N,\,(u,v,n)\in ROI} D(u,v,n), \quad (7)$$

where $iteration$ denotes the iteration index, for example, D_1 stands for the first iteration.

4. *BACKGROUND COLOR IMAGE GENERATION*

In this stage, we use the depth map produced in the last stage to produce the weighted average of color images. The procedure of this stage is the same as the first stage, but D_{min} becomes the depth produced in the last stage, and only the pixels in the ROI are calculated.

$$I_{iteration}(u,v) = \frac{\Sigma_{for\ all\ n\ where\ (u,v,n)in\ ROI} I(u,v,n) \times \frac{1}{|D(u,v,n) - D_{iteration}(u,v)+1|}}{\Sigma_{for\ all\ n\ where\ (u,v,n)in\ ROI} \frac{1}{|D(u,v,n) - D_{iteration}(u,v)+1|}} \quad (8)$$

$, for\ all\ u, v.$

The above process is repeated iteratively until the depth map becomes stable.

III. POST PROCESSING

Some example of the resultant depth map produced by the proposed iterative process is enlarged and shown below in Fig. 6.

Fig. 6 Corrected depth maps and images and Post-processed depth maps: the left is D_{min}, the second is D_{12}, the third is D_{final}, and the right is I_{12}

As you can see, most of the wrong depth values are fixed. The road sign post in D_{min} was broken, but in D_{12}, it is clearly much improved using our proposed algorithm. However, there are still some wrong depth values on the round road sign board. Assuming that the depth values in most of the frames are correct, we apply a time-wise *mode filter* (most frequent value) for every pixel in ROI.

$$D_{final}(u,v) = \underset{1 \le n \le N}{\text{mode}}\, D(u,v,n),\ \forall (u,v,n) \epsilon ROI. \quad (9)$$

After applying the mode filter, the results are also shown in Fig. 7.

The depth values on the signboard and the antenna of the car are closer to what we would expect. The final depth map of this example is shown in Fig. 7.

IV. EXPERIMENTAL RESULTS

We apply our proposed algorithm to several MPEG test sequences and show the results below. The first row of an example is the selected images from a sequence, and the second row is their associated depth maps, and the third row is the background image and its depth map.

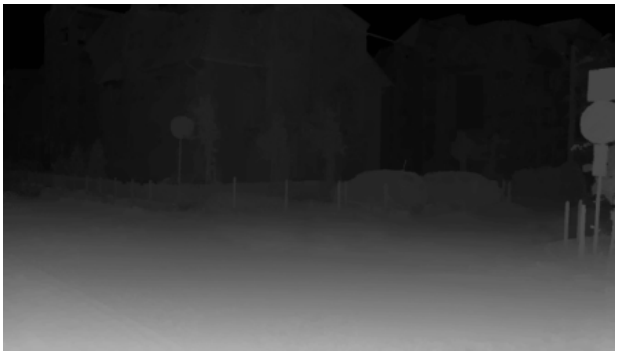
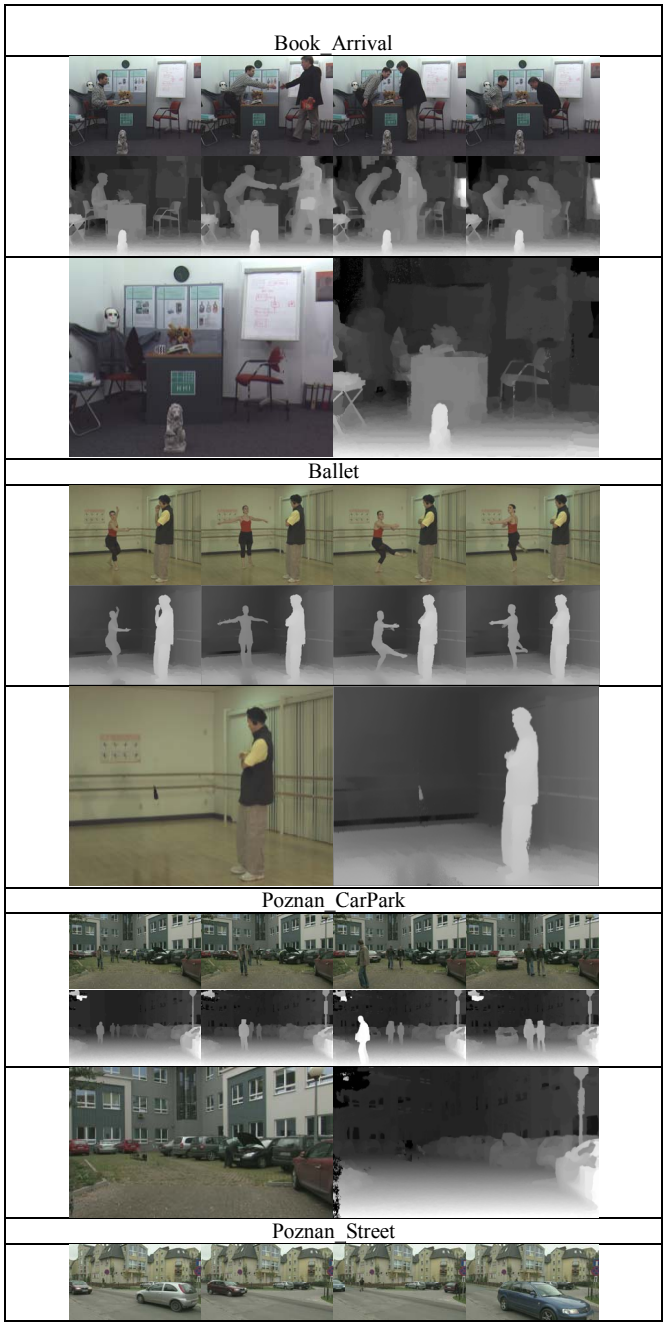


Fig. 7 D_{final} of the Poznan_Street sequence



We also compare the results of our proposed algorithm with some other background modeling methods [5]. With the help of the depth information and the proposed iterative algorithm, our background images are clearly superior.

Proposed method 	Static Frame Difference 
Weighted Moving Mean 	Adaptive Background Learning 
Fuzzy Sugeno Integral 	Fuzzy Choquet Integral 
Fuzzy Gaussian 	Simple Gaussian 
Gaussian Mixture Model of Zivkovic 	Gaussian Mixture Model of Laurence Bender 
VuMeter 	Adaptive SOM 
Fuzzy Adaptive SOM	

V. CONCLUSIONS

As the depth sensor becomes popular, in addition to the RGB video, we also have the depth information. Based on the collected depth maps, which may not be all correct at every pixel or every frame, we propose an iterative scheme that utilizes the depth information to produce a better background color image and its associated depth map as well. Our proposed algorithm has the following nice properties. (1) It iteratively removes wrong depth values to eliminate the influence of noisy depth map. (2) It is a non-parametric model; that is, we have no pre-assumptions on the probabilistic models or parametric models on the background images. (3) Compared with the conventional background modeling methods, our method typically produces a better background image. In addition, in this process, we also obtain a good quality background depth map, which is critical for virtual view synthesis applications. But clearly, this approach relies on the good quality of depth maps.

VI. ACKNOWLEDGEMENT

This work was supported in part by the NSC, Taiwan under Grant NSC 102-2221-E-009-123 and by the Aim for the Top University Project of National Chiao Tung University, Taiwan.

REFERENCES

[1] C. Stauffer and E. Grimson, “Adaptive Background Mixture Models for Real-time Tracking”, in *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pp. 246-252, 1999.

[2] D. S. Lee, J. J. Hull, and B. Erol, “A Bayesian Framework for Gaussian Mixture Background Modeling”, in *Proceedings of International Conference on Image Processing*, pp.973-976, 2003.

[3] A. Elgammal, D. Harwood, and L. Davis, “Non-parametric Model for Background Subtraction”, *Proc. of ICCV '99 FRAME-RATE Workshop*, 1999.

[4] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. S. Davis, “Real-Time Foreground-Background Segmentation using Codebook Model”, *Real-Time Imaging*, pp. 172-185, 2005.

[5] A. Sobral, “BGSLibrary: an openCV c++ background subtraction library,” *IX Workshop de Visão Computacional* (WVC 2013). Rio de Janeiro, Brazil, 2013, Software available at <http://code.google.com/p/bgslibrary/>.