Consistent 3D Background Model Estimation from Multi-Viewpoint Videos

Iraklis Tsekourakis
Stevens Institute Of Technology
itsekour@stevens.edu

Philippos Mordohai
Stevens Institute Of Technology
Philippos.Mordohai@stevens.edu

Abstract

We present an approach for estimating the 3D background model of a scene from a collection of synchronized videos. Unlike previous work, our method is fully automatic, does not require empty frames depicting just the background, and makes very mild assumptions about the foreground. The constraint on the cameras is that they should have sufficiently narrow baselines to enable multi-view stereo matching. Using the images and primarily the depth maps as inputs, our algorithm detects potential background pixels to generate initial per-camera background models, which are then fused to form the final, consistent 3D background model. We show results on diverse video sequences captured using different camera configurations. Despite the challenges posed by the input videos, in which some parts of the background are always occluded in the images, we are able to extract accurate models of the background that are effective in foreground segmentation. This would have been impossible using conventional background subtraction methods that operate on the frames of each camera separately. Moreover, fusion makes the per-camera background models consistent.

1. Introduction

While undeniable progress has been made in the past few years in multi-view 3D reconstruction [32, 36, 41, 17, 33], the emphasis remains on achieving high accuracy, photo-realism and unprecedented size in reconstructed models of *static* scenes. A much smaller branch of multi-view stereo addresses the modeling of dynamic scenes, in which people, animals, robots or other visible surfaces move. We will refer to such scenes as *dynamic* and the process of estimating the 3D shape of all visible surfaces in them at each time instant as *dynamic 3D reconstruction*. We expect dynamic 3D reconstruction to rise in popularity given its wide range of exciting and commercially attractive applications that include free-viewpoint video; 3D TV and movies; video-games with user-controlled viewpoint; markerless motion capture; biomechanical analysis of human motion for athletes or patients; and dynamic augmented reality.

Promising results have been obtained by methods that employ full 3D representations of the world [39, 5, 28, 1, 16, 34, 11, 3, 29], but the models are typically of a single foreground person or surface. The single exception that is capable of handling multiple objects, which have been segmented from the background, is the work of Cagniart et al. [3]. In all cases, the scenes' stationary parts have been removed as a pre-processing step and have never been re-introduced in the final spatiotemporal model. This is justified in part since the background in many cases is uniformly colored backdrop whose only purpose is to be easily segmented and deleted.

One of the next breakthroughs for dynamic reconstruction methods is the capability to model more interesting scenes in which one or more actors interact with objects and furniture, open doors and, in general, perform complex activities. An important step in this direction is an approach for extracting consistent models of the background from the video, ideally without requiring "empty" frames of the background only. These models will serve two purposes: they will ensure that the stationary parts of the scene remain consistent through time improving the viewing experience; and they will significantly aid in the segmentation of the dynamic foreground, improving the accuracy of foreground reconstruction. Larsen et al. [22] and Guillemaut and Hilton [14] have argued that temporal inconsistencies are more easily noticed, and also more disturbing, on stationary or slowly moving parts of the scene. Taking this argument a step further, it is more important in terms of subjective visual quality to assign constant depth values to stationary parts of the scene, even if these depths are slightly wrong, than to generate dynamic 3D models in which there is drift or flickering on the stationary parts.

Our objective here is to make a contribution in this direction. Our approach is capable of fully automatic extraction of a 3D model of the stationary background from a collection of multi-view video sequences. We use the term *back-*

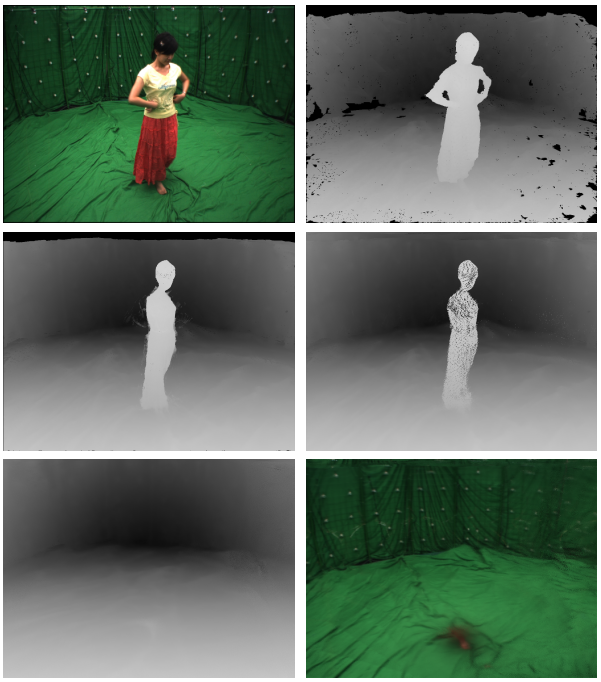

Figure 1. First row: input image and depth map, obtained using the software of [17], from the Redksirt data set [25]. Second row: estimated background depth model from Single View initialization, and Fusion. Third row: estimated background depth and color model using the proposed Iterative Fusion method with field of view constraints.

ground for parts of the scene that remain stationary, regardless of whether they are in front or behind of dynamic elements. The background model we compute from the videos comprises both depth and color information. It is initialized separately for each camera and consistency is enforced in a second stage in which partial background models are fused. Throughout, we assume that the cameras are calibrated, stationary and synchronized. While it is possible to lift these assumptions in future work, it is far from straightforward. Precise radiometric calibration, however, is not required. Figure 1 shows an input frame from one of the 20 cameras that recorded the Redskirt data set of Liu et al. [25], as well as the depth map we estimated for it using the software of Jancosek and Pajdla [17]. The second row displays the background depth models extracted from the intermediate steps of our algorithm. The last row shows our final background model estimate from the viewpoint of the same camera.

An important aspect of our approach is that it does not require empty frames without any foreground. In fact, there are parts of the background that remain occluded in every single frame of the input sequences (Fig. 2). We are still able to infer the correct background even for these regions. Two groups [23, 14] have recently published algorithms that share this property. The factor that distinguishes our work from that of Lee et al. [23] is that we adopt stereo matching to generate 3D shape estimates, while they take a silhouette-based approach. Moreover, Lee et al. require the foreground to be entirely visible in all cameras throughout the sequence. The approach of Guillemaut and Hilton [14] is also related to ours, but it is not fully automatic; the user must specify the number of layers and draw a trimap partitioning one frame into definite foreground, definite background and unknown regions.

We present results on publicly available multi-view data published by Zitnick et al [44] and Liu et al. [25]. Since no ground truth is available for the background of these scenes, we validate our results by testing their usefulness in segmenting the foreground in some of the input frames. We also demonstrate background model synthesis for a camera that had been excluded from the estimation.

2. Related Work

The ability to segment the background is critical for 3D reconstruction of static scenes that rely on silhouettes entirely [26, 9, 43] or partially [15, 10]. In practice, a background removal technique [35, 8, 37] is applied separately on the frames of each camera using images with no foreground to learn a color model which is used for segmenting the silhouettes. This approach to foreground-background segmentation is clearly suboptimal since it does not benefit from the availability of observations of the same surfaces from multiple viewpoints.

Methods that enforce cross-camera consistency in foreground segmentation include the one of Campbell et al. [4] that divides the volume around the foreground expected location into a voxel grid and uses a graph cut to solve for the voxels occupied by the foreground object. The latter must be at the center of the images to enable automatic initialization. Variational approaches with similar objectives have also been proposed by Kolev et al. [19] and Reinbacher et al. [31]. The former is semi-automatic, operates on voxels and generates a 3D model of the foreground. The latter can be fully automatic, under the assumptions of [23], that is if the foreground is entirely visible in all cameras, and by operating on pixels obtains more accurate boundaries. Djelouah et al. [6, 7] proposed an approach that links multiple views via an MRF and is able to handle videos as input, and not just individual frames. These methods achieve spatially consistent segmentation but do not estimate depths for the background. A different cosegmentation approach that bears some similarity to ours is that of Kowdle et al. [21]. Its objective is to segment an object in a set of images leveraging appearance and stereo cues. The latter are used to generate appearance models for reconstructed planer patches. This approach is able to reconstruct the background as well, but it is limited to rigid foreground objects.

Some of the single-camera background subtraction

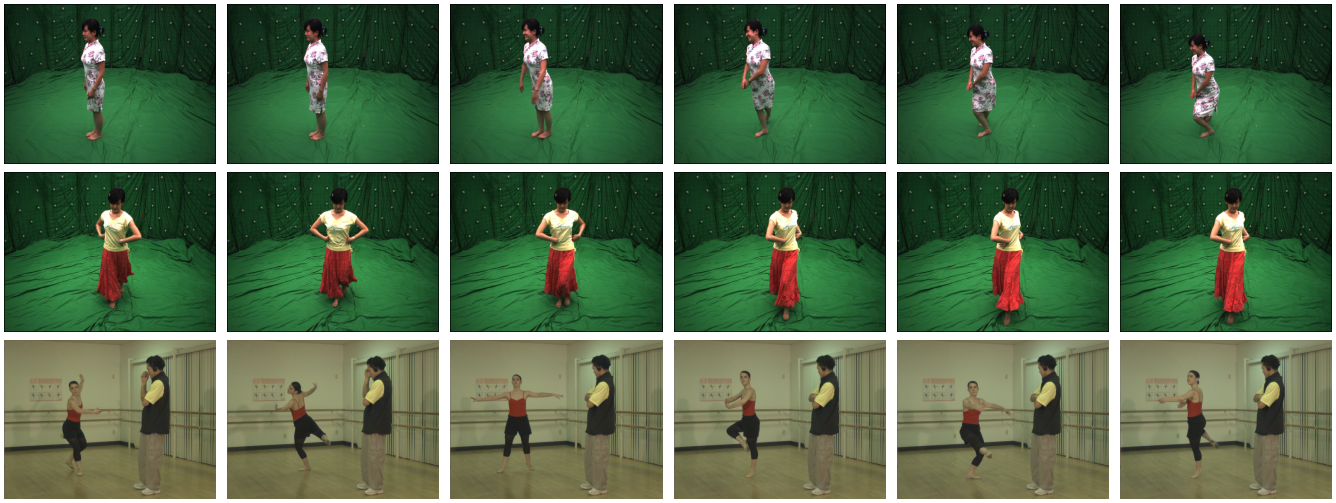

Figure 2. First row: Cheongsam data set [25]. One image is displayed every 5 frames to cover the length of the 30-frame video. Second row: Redskirt data set. One image is displayed every 3 frames of the 20-frame video. Third row: Ballet sequence [44]. One image is displayed every 20 frames of the 100-frame video. The dancer is usually at the center of the image, occluding part of the background. The observer in the ballet videos does not move his legs throughout. He just slightly rocks back and forth.

methods do not enforce temporal consistency since they were designed to operate on a single image. However, as mentioned before, temporal inconsistency in static parts of the scene leads to salient and disturbing artifacts. This was addressed in the context of video-conferencing by algorithms that segment stereoscopic video into layers [13, 20] utilizing both depth and color information. The intended application, however, is background replacement and not free viewpoint video.

We now turn our attention to dynamic reconstruction methods that do not discard the background, but treat it as a set of regular surfaces and attempt to reconstruct it. As mentioned before, there is no method that uses a 3D world-based representation that reconstructs the background. The following methods are all viewpoint-based, that is they estimate disparity and motion for the pixels of a reference frame. In this setting, processing both static and dynamic surfaces is more natural, but the resulting models do not allow viewpoint changes. An approach that seeks spatial (multi-view) and temporal correspondences for pixels that is of interest to us is that of Larsen et al. [22]. It estimates a rough background model per camera, which is used to stabilize pixels that are likely to belong to stationary surfaces. Such pixels can be detected based on color similarity to the background and low confidence for a different depth. Yang et al. [42] detect pixels on moving objects based on optical flow and appearance information and then reconstruct static and dynamic elements of the scene by modeling it as a collection of rigid bodies. Also relevant to our work are variational methods that enforce temporal consistency either by making predictions about the depth or disparity of the next frame [12, 24], by using a Kalman filter per point [30] or by approximating the scene as a collection of rigidly moving planar patches [40].

3. Problem Statement

The objective of our research is to estimate background models comprising depth and color information per pixel, given as input multi-view video sequences captured by calibrated, stationary, synchronized cameras. (Throughout, we use the term *view* to indicate a viewpoint, i.e. a specific camera, and *frame* to denote an image taken at a different time.) Our approach is fully automatic and does not require blank frames for initialization. It makes essentially no assumptions about the foreground, which does not have to be fully visible [23] or centered in the images [4], but it should be somewhat distinguishable from the background. The first processing step is the depth maps estimation for each frame of every camera. After depth maps have been estimated, our approach no longer depends on color information.

Depth estimation is treated as pre-processing in this paper. We use the CMPMVS software of Jancosek and Pajdla [17] for the data of Liu et al. [25] and the provided depth maps for the data of Zitnick et al. [44]. After computing the 3D model for each time instant using CMPMVS, we project it onto all cameras to generate the corresponding depth map. Starting from the images and depth maps, we compute a background model for each camera separately (Section 4). Because in all these sequences there are certain background parts that are always occluded by dancers, these single-view background models are *noisy and incomplete*. To correct these issues and to enforce consistency across views, we fuse the background models of multiple cameras with respect to the viewpoint of each camera in the set. To this end, we modify the depth map fusion approach

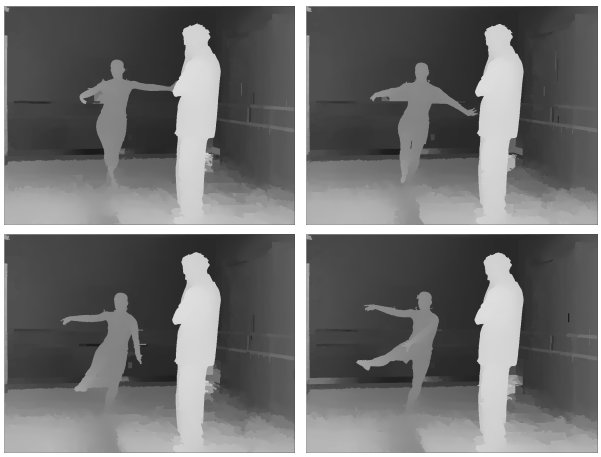

Figure 3. Noisy depths from the ballet video. Notice the artifacts due to segmentation failures, as well as how the depth of the floor fluctuates from frame to frame.

of Merrell et al. [27] which is presented in Section 5.

Since there is no ground truth for our problem, we validate whether the estimated background models are useful and consistent. We chose two tasks for this validation: foreground segmentation based on depth inputs and background synthesis for a novel view. Results are shown in Section 6.

4. Single-view Background Initialization

In this section we present the first step of our algorithm, single-view background initialization, which extends the method of Larsen et al. [22]. Here, we aim to automatically create an initial background model for each camera separately, using information from the specific camera only. (Images from other cameras are used to estimate depth maps, but are not directly used as inputs here.) The inputs to this stage are the color image sequence $\mathrm{I}_{i,f}$ and the depth map sequence $\mathrm{D}_{i,f}$, where f denotes the frame number, and i is the camera number. We use all images and depth maps for a given camera, that is i is fixed while f spans all time instants when frames were captured.

The challenge is that the foreground occludes parts of the background throughout the video. Figure 2 shows frames of one camera from each of the three data sets we use [25, 44]. For background pixels that are sometimes occluded and sometimes visible, we would like to be able to extract their depth and appearance. In order to achieve this, we apply k-means clustering on the colored depth maps. For each pixel (x, y), we form F 4-tuples, where F is the total number of frames, containing the color and depth information in all frames.

$$S(i_o, x_o, y_o) = \{I^{(r)}_{i,f}(x,y), I^{(g)}_{i,f}(x,y), I^{(b)}_{i,f}(x,y), D_{i,f}(x,y) \\ | x = x_o, y = y_o, i = i_o, f \in \{1, F\}\} \quad (1)$$

where $I^{(r|g|b)}_{i,f}(x,y)$ denotes the red, green or blue channel of pixel (x, y) in image i. For each pixel we collect F 4-tuples and apply k-means to them, seeking the cluster that is further away from the camera. Due to the differences in length among the video sequences, we set the number of clusters k for k-means as a function of the number of available frames. Specifically, we use $k = \lfloor \frac{F}{15} \rfloor$ throughout the paper. The centroid of the cluster that is further away from the camera is selected as the background color and depth representation for pixel (x, y).

Additional challenges include the short duration of the two data sets from [25] (20 and 30 frames) and the noisy depth maps for the data set of [44], which can be seen in Fig. 3. These cause errors in the single-view results, which are removed after multi-view fusion.

Results from this stage can be seen in Figs. 1, 4 and 5 labelled as Single View results. The artifacts are due to the background being always or almost always occluded in these pixels and motivate the need for multi-camera processing to correct them. The output of this stage is the input for the multi-view fusion algorithm described in the next section. As can be seen in Figs. 1, 4 and 5 the results are far from satisfactory due to large pieces of the foreground covering parts of the background that are never observed by the reference camera.

5. Multi-view Fusion

The inputs to this stage are a depth map and a color image per camera generated according to the previous section. Each camera in turn is used as a reference view for fusion, which is guided by depth. Color information is propagated and used to generate the background models' color component. Our goal of generating a fused background depth model from a number of single-view depth models is similar to that of the depth map fusion algorithm of Merrell et al. [27]. There is an important difference that makes modifications to that algorithm necessary: unlike [27] that seeks the most likely depth according to support and visibility constraint violations among the inputs, we have a bias for a consistent depth that is far from the reference camera. We begin by presenting the algorithm and explain the modifications in the process.

Initially, all depth maps are rendered onto the reference view resulting in the accumulation of multiple depth candidates on each pixel of the reference view. During the rendering process, two or more pixels from a depth map may project in the same pixel in the reference view. In this case, we always select the one with the larger depth favoring the background. We, then, consider each of these candidates one by one by first accumulating support from other depth candidates for the same pixel. We say that depth estimate d_i is supported by d_j if $|d_i - d_j| \leq \varepsilon d_{range}$, where d_{range} is the depth range of the scene and ε is 0.05 as in [27]. Support accumulation for a single depth candidate is the same

as in [27], but since we have no confidence values associated with the depths, each supporting depth increases the score of d_i by 1[1]. A more important difference is that *we allow all candidates to accumulate support* and not just one as in [27]. We update the depth of each candidate by taking the average of its initial depth and all depths that support it.

$$s_i = \sum I(|d_i - d_j| \leq \varepsilon d_{range})$$
$$d_i^f = \frac{1}{s_i} \sum d_j \text{ where } |d_i - d_j| \leq \varepsilon d_{range} \quad (2)$$

where $I()$ is an indictor function that returns 1 if its argument is true, s_i is the number of depth candidates supporting d_i including itself and d_i^f is the fused depth.

After support has been accumulated for all depth candidates and the depth values have been updated, we test whether they violate the free space of any of the input depth maps. Being in front of an observed background surface is an indication that a depth candidate is not correct or not part of the background. To test for free space violations we render all depth candidates from the reference view to all the other views. A free space violation occurs when a depth candidate appears in front of the background model of a non-reference camera. This is a conflict and it is penalized by subtracting 1 from the score s_i of d_i. To avoid penalizing depth candidates that essentially agree with a background model, a violation is only recorded when the depth candidate is in front by at least εd_{range}.

Unlike the approach of Merrell [27], we do not test for nor penalize occlusions. This is because occlusions, i.e. the candidate depth being occluded with respect to the reference view, are in some sense desirable for our application. We seek to complete the missing background layer behind remnants of the foreground in the reference view (see Figs. 1, 4 and 5), not to determine the most likely visible depth.

We finally select the depth candidate with the highest score for each pixel, keeping track of all other candidates that supported it. Each candidate in turn has indexes to the depth and color values included in its cluster in the view that generated it. We use the mean of all colors to generate the background color image.

Iterative fusion: The above fusion process removes unwanted artifacts from the foreground and improves the background models' consistency among the different cameras. There is no reason to believe, however, that it has converged. In fact, fewer and smaller artifacts may still remain, leaving us facing the same problem. We address it by repeating the fusion process using the outputs of the previous process as inputs. It turns out that this iterative process approximately converges after a few iterations. We use five iterations to generate the results in the next section.

[1]The fact that we do not require confidence maps for the fusion as in [27] allows us to use as inputs depth maps computed by any algorithm. Clearly, the quality of the inputs affects to a certain degree the quality of the output, but all stages of our algorithm are robust to large fractions of outliers.

6. Experimental Validation

In this section, we present and evaluate the background models created by our algorithm. We use the following multi-view video sequences made available by their authors. Cheongsam [25] is captured in a dome of diameter equal to 4.2 m by twenty cameras in a ring around the scene. Each video is 30 frames long, but one of them had to be dropped due to missing frames. Redskirt is captured in the same dome, but the videos are 20 frames long. The ballet data [44] are acquired by eight cameras forming a 30° arc, thus with much narrower baselines. The depth range in this scene is 7.6 m. For the first two sequences, we reconstructed 3D models for each frame separately using the CMPMVS software [17] and then projected the models onto the cameras to generate the input depth maps. All fusion-based results have been computed using three cameras on each side of the reference camera in the fusion process. Zitnick et al. [44] provide depth maps for the ballet data. We used the provided depth maps as inputs and used all eight cameras in each fusion process, since the baseline is narrow. (We have obtained results on the breakdancers data as well,

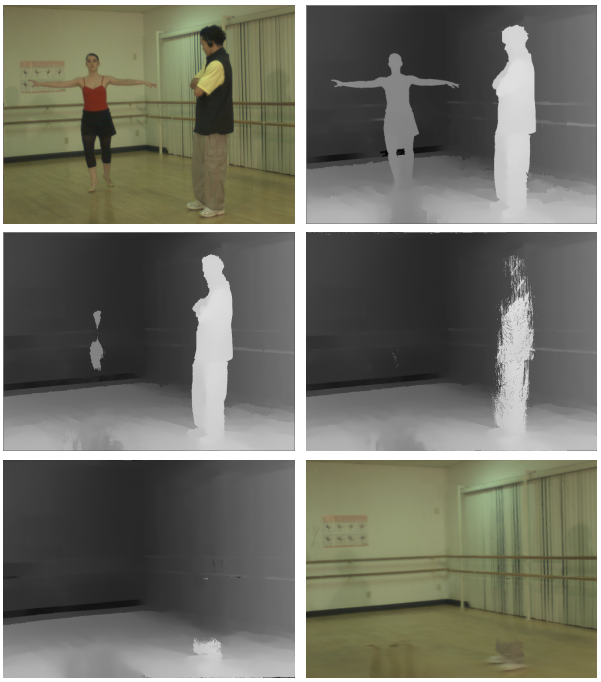

Figure 4. First row: input image and depth map from the ballet data set. Second row: estimated background depth model from Single View initialization, and Fusion. Third row: estimated background depth and color model using the Iterative Fusion method with free space violation (FSV) constraints.

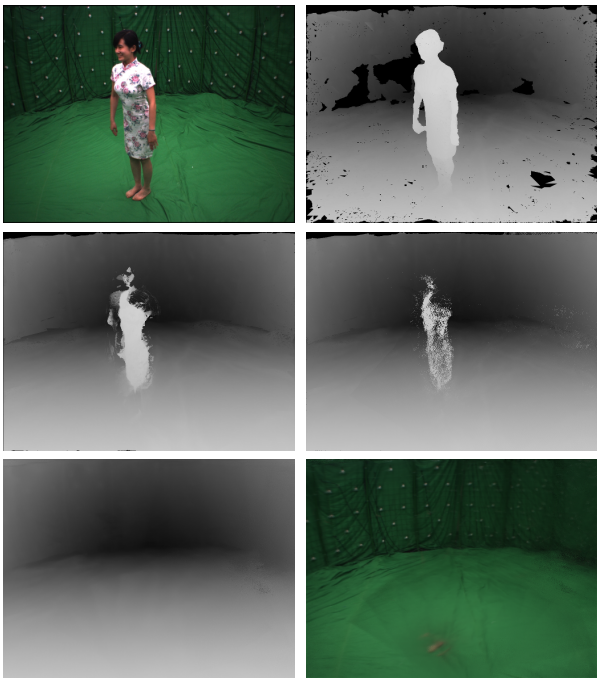

Figure 5. First row: input image and depth map, obtained using the software of [17], from the Cheongsam data set [25]. Second row: estimated background depth model from Single View initialization, and Fusion. Third row: estimated background depth and color model using Iterative Fusion with FSV constraints.

but do not show them here due to the unresolvable ambiguity between foreground and background for the spectators. They lean against the staircase bobbing their heads but do not move their torsos.)

The estimated models for all data sets are shown in Figs. 1, 4 and 5. All models were generated applying the iterative fusion method for 5 iterations with FSV constraints enabled. These background models are the *primary outputs our method.* Fig. 6 shows an example.

In the absence of ground truth, we first evaluate the background models by testing their effectiveness in foreground segmentation. As baselines, we use the two background subtraction methods provided by the OpenCV library. They are named MOG [18] and MOG2 [45, 46] and use mixtures of Gaussians to represent the background. They are applicable to our settings because they do not require empty frames to learn the background model. To assess the contribution of each aspect of our approach, we evaluated models generated by the Single View method of Section 4, the Fusion method with and without free space violations (FSV) and the Iterative Fusion method with FSV constraints. The inputs to this part for a camera i are the depth background model D_i^{bg} and the input depth map of the specific frame $\mathrm{D}_{i,f}$. Every pixel (x, y), is classified as background if the input depth value of the specific pixel $\mathrm{D}_{i,f}$(x,y) is in a range of 2 σ of the background model depth value D_i^{bg}(x,y). We use σ = 1.5 d_{range}

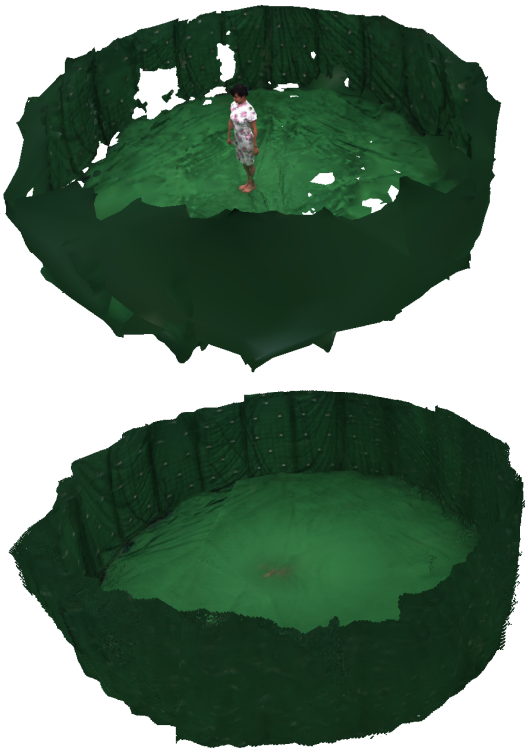

Figure 6. Top image: the output of CMPMVS for the first frame of all 20 cameras of Cheongsam. Bottom image: the estimated 3D model of the background using Iterative Fusion with FSV constraints.

throughout the paper.

To improve the global consistency of the results, we formulate foreground segmentation as a binary Markov Random Field (MRF) and use graph cuts to minimize an energy function comprising a data and a smoothness term. The data term is based on a Gaussian distribution of the distance of the input depth from the background model depth. We chose the value of the standard deviation to be equal to 4 disparity values. (We intentionally did not use color here since two of the datasets have green backgrounds.) The smoothness term follows a Potts model with edge weights set according to the strength of the intensity edges between neighboring pixels. We used the implementation provided by [38] based on [2] to minimize this energy.

In the first experiment we used 18 frames from camera 1 from the Cheongsam data for which segmentation ground truth was provided [25]. Foreground segmentations were computed using background models generated by all methods. Table 1 summarizes the accuracy of all methods tested. True positives (TP) represent foreground pixels that are segmented as foreground correctly, false positives (FP) are background pixels segmented as foreground, true negatives (TN) are background pixels that were segmented correctly and false negatives (FN) are foreground pixels that were segmented as background. The numbers on the ta-

	MOG	MOG2	Single View	Fusion	Fusion-FSV	It.Fusion-FSV	It.Fusion-FSV MRF
TP	12,262	37,564	19,617	25,727	39,127	45,178	45,959
FP	38,301	7,438	30,947	24,837	11,437	5,385	4,605
TN	731,862	718,897	733,663	733,784	734,288	734,318	734,532
FN	4,005	16,970	2,204	2,083	1,580	1,550	1,335
IOU	0.2247	0.6061	0.3718	0.4914	0.7504	0.8669	**0.8855**

Table 1. Accuracy of all methods on the Cheongsam data set using the confusion matrix and the Intersection over Union metric.

ble are pixels in each category averaged over the 18 frames. In order to account for the much larger number of background pixels compared to the foreground, the intersection over union (IOU) metric was used. It is defined as:

$$\text{IOU} = \frac{TP}{TP+FP+FN}$$

Iterative fusion with the FSV constraints outperforms the rest of the methods. MOG2 is more competitive than MOG1 and surpasses the single-camera method (Section 4) and regular fusion. This, however, is due to the green background which is very helpful to the appearance based methods. Removing the occlusion constraints from fusion results in fusion-FSV which dominates all previous methods. Iterating fusion-FSV leads to a significant improvement in TP and IOU, while applying MRF-based optimization further improves the results.

We also evaluated iterative fusion with FSV constraints and the background subtraction methods of OpenCV on the Redskirt and ballet data. The results can be seen in Table 2. The numbers in the table are pixels in each category averaged over four frames, which were provided as ground truth segmentation, in the Redskirt case and three frames for the ballet data. The frames for the latter were manually segmented by us due to the lack of ground truth. To visualize the foreground segmentation results we generated images such as those in the right column of Fig. 7 that show only the pixels that were detected as foreground. Both FP and FN appear as flaws in these masks.

In order to further validate the background models' accuracy, a view synthesis test was conducted. Twenty cameras were used to record the Cheongsam data. We have excluded camera 10 from all processing, including 3D reconstruction. We used the excluded view as a virtual reference view for fusion that does not contribute any depth candidates. Depths from the six neighboring cameras were fused and a background model was estimated for that view. Since the resulting model has some holes due to occlusion, an iterative median filter was applied to fill them. This model was then used to segment the foreground for 11 images taken by the excluded camera, for which ground truth segmentation was available. (This is a different camera than the one used for the results in Table 1.) The hypothesis is that if segmentation accuracy is not degraded, then the background model is consistent with this essentially virtual camera and can be used for novel view synthesis in free-viewpoint video. Figure 8 shows the color and depth components of the background model, an input image and the segmented foreground. The accuracy in this task was indistinguishable with the previous experiments, as shown in Table 3. The IOU falls from 0.8855 when images from camera 10 are included (Table 1) to 0.8657 when the background is synthesized for that view. The MRF in this case does not help as it leads to a reduction of IOU to 0.8460, which we do not consider significant.

	MOG	MOG2	It.Fusion-FSV	It.Fusion-FSV MRF
Redskirt				
TP	14,609	47,681	53,288	64,849
FP	47,790	14,718	9,112	7,034
TN	708,444	688,682	723,136	712,291
FN	15,558	35,350	897	2,255
IOU	0.1874	0.4878	0.8418	**0.8747**
ballet				
TP	9,342	33,051	103,704	102,252
FP	105,768	81,968	11,315	12,767
TN	665,677	644,155	649,733	666,732
FN	5,734	27,256	21,718	4,679
IOU	0.0773	0.2323	0.7584	**0.8542**

Table 2. Accuracy on the Redskirt and ballet data using the confusion matrix and the IOU metric. Notice how the single-view methods using only color modeling are substantially more effective on the data with the green background, but perform worse otherwise.

	TP	FP	TN	FN	IOU
Synth.	42,595	5,709	737,227	899	0.8657

Table 3. Novel view synthesis evaluation on Cheongsam using the iterative fusion method with FSV constraints

7. Conclusions

We have presented a fully automatic technique for multi-view background model estimation. Since our approach is guided by depth, it is robust to variations in illumination or the response functions of different cameras. Our method is flexible and can operate on any input depth maps. In the experiments presented, the data were collected in widely different setups. The data of Zitnick et al. [44] were acquired with all cameras on one side of the scene, while the data of Liu et al. [25] were collected with 20 cameras around the scene. Our method is successful in these settings, in which even a perfect monocular method would have failed. The method of Larsen et al. [22] would have also failed, as can be seen in Figs. 1, 4 and 5, where it is referred to as the Single View method.

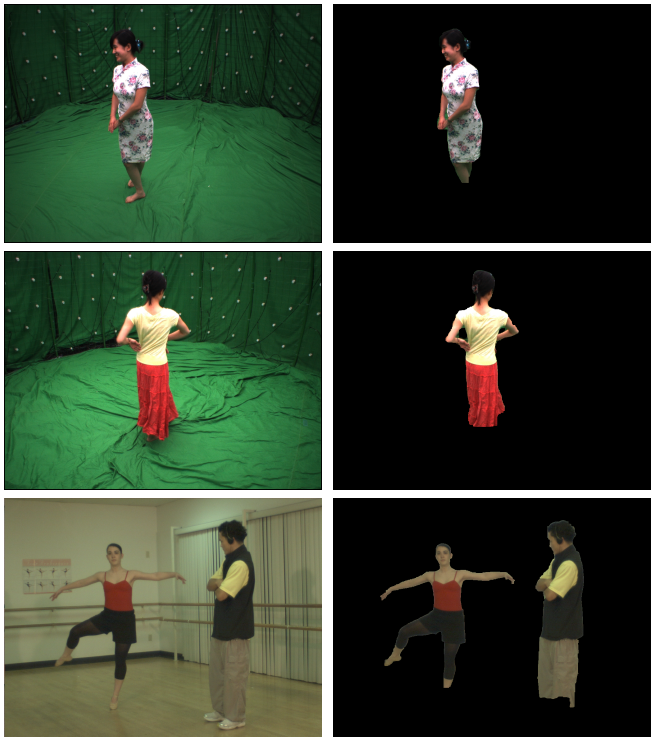

Figure 7. Left column: input images from all data sets. Right column: resulting foreground segmentation masks using the iterative fusion method with FSV constraints. The man's lower legs in ballet never move and are considered background by our algorithm.

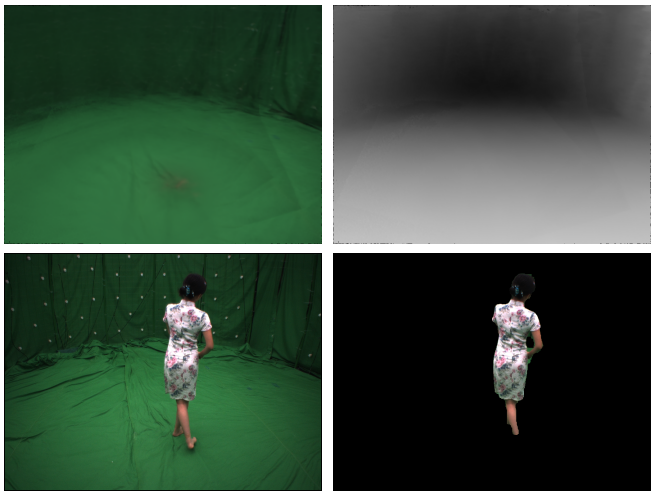

Figure 8. First row: synthesized background color and depth models. Second row: color input image and resulting foreground mask.

In order to achieve our goal, we presented modifications to the depth map fusion approach of [27] that are likely to be broadly applicable. We iterated the process and obtained progressively more consistent depth maps and we also did not commit to a single depth candidate per pixel, but instead tested all of them. The use of free-space constraints is specific to background modeling only and makes a significant difference as shown in 1. Fusion makes the per-camera background models consistent, enabling applications such as free-viewpoint video that require consistent background models as the virtual viewpoint shifts.

The strongest assumption we have made is that the camera configuration enables stereo-based depth estimation. This is also a requirement for our future work that will focus on reconstructing the dynamic elements of the scene. Having these background models is expected to provide a significant boost in accuracy when the final foreground depths are estimated. We anticipate increased accuracy near occlusion boundaries between static and dynamic surfaces as well as increased accuracy of foreground depth due to restrictions in the depth search range. Our future work will also investigate ways of robustly integrating color information more tightly in the computation. Finally, we plan to generate datasets with multiple static layers and enhance our approach to handle multiple background layers.

Acknowledgements This research has been supported in part by the National Science Foundation award #1217797.

References

[1] E. Aganj, J. P. Pons, F. Segonne, and R. Keriven. Spatio-temporal shape from silhouette using four-dimensional delaunay meshing. In *ICCV*, 2007. 1

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–1137, 2004. 6

[3] C. Cagniart, E. Boyer, and S. Ilic. Free-form mesh tracking: a patch-based approach. In *CVPR*, 2010. 1

[4] N. D. F. Campbell, G. Vogiatzis, C. Hernandez, and R. Cipolla. Automatic 3d object segmentation in multiple views using volumetric graph-cuts. *Image and Vision Computing*, 28(1):14–25, 2010. 2, 3

[5] R. L. Carceroni and K. N. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to non-rigid 3D motion, shape and reflectance. *IJCV*, 49(2-3):175–214, 2002. 1

[6] A. Djelouah, J.-S. Franco, E. Boyer, F. Le Clerc, and P. Pérez. Multi-view object segmentation in space and time. In *ICCV*, 2013. 2

[7] A. Djelouah, J.-S. Franco, E. Boyer, F. Le Clerc, and P. Perez. Sparse multi-view consistency for object segmentation. 2015. 2

[8] A. M. Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. In *ECCV*, pages II: 751–767, 2000. 2

[9] J. Franco and E. Boyer. Efficient polyhedral modeling from silhouettes. *PAMI*, 31(3):414–427, 2009. 2

[10] Y. Furukawa and J. Ponce. Carved visual hulls for image-based modeling. In *ECCV*, pages I: 564–577, 2006. 2

[11] Y. Furukawa and J. Ponce. Dense 3D motion capture from synchronized video streams. In *CVPR*, 2008. 1

[12] M. Gong. Real-time joint disparity and disparity flow estimation on programmable graphics hardware. *CVIU*, 113(1):90 – 100, 2009. 3

[13] G. G. Gordon, T. J. Darrell, M. Harville, and J. I. Woodfill. Background estimation and removal based on range and color. In *CVPR*, pages II: 459–464, 1999. 3

[14] J.-Y. Guillemaut and A. Hilton. Space-time joint multi-layer segmentation and depth estimation. In *Intrnational Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 440–447. IEEE, 2012. 1, 2

[15] C. Hernández Esteban and F. Schmitt. Silhouette and stereo fusion for 3D object modeling. *CVIU*, 96(3):367–392, 2004. 2

[16] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 2007. 1

[17] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR*, pages 3121–3128. IEEE, 2011. 1, 2, 3, 5, 6

[18] P. KaewTraKulPong and R. Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In *Video-Based Surveillance Systems*, pages 135–144. Springer, 2002. 6

[19] K. Kolev, T. Brox, and D. Cremers. Fast joint estimation of silhouettes and dense 3d geometry from multiple images. *PAMI*, 34(3):493–505, 2012. 2

[20] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Probabilistic fusion of stereo with color and contrast for bilayer segmentation. *PAMI*, 28(9):1480–1492, 2006. 3

[21] A. Kowdle, S. N. Sinha, and R. Szeliski. Multiple view object cosegmentation using appearance and stereo cues. In *ECCV*, pages 789–803, 2012. 2

[22] E. S. Larsen, P. Mordohai, M. Pollefeys, and H. Fuchs. Temporally consistent reconstruction from multiple video streams using enhanced belief propagation. In *ICCV*, 2007. 1, 3, 4, 7

[23] W. Lee, W. Woo, and E. Boyer. Silhouette segmentation in multiple views. *PAMI*, 33(7):1429–1441, 2011. 2, 3

[24] F. Liu and V. Philomin. Disparity estimation in stereo sequences using scene flow. In *British Machine Vision Conference*, 2009. 3

[25] Y. Liu, Q. Dai, and W. Xu. A point cloud based multi-view stereo algorithm for free-viewpoint video. *IEEE Trans. on Visualization and Computer Graphics*, 16(3):407–41, 2010. 2, 3, 4, 5, 6, 7

[26] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *ACM SIGGRAPH*, 2000. 2

[27] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J. M. Frahm, R. Yang, D. Nistér, and M. Pollefeys. Real-time visibility-based fusion of depth maps. In *ICCV*, 2007. 4, 5, 8

[28] J. Neumann and Y. Aloimonos. Spatio-temporal stereo using multi-resolution subdivision surfaces. *IJCV*, 47(1-3):181–193, 2002. 1

[29] T. Popham, A. Bhalerao, and R. Wilson. Multi-frame scene-flow estimation using a patch model and smooth motion prior. In *British Machine Vision Conference Workshop*, 2010. 1

[30] C. Rabe, T. Müller, A. Wedel, and U. Franke. Dense, robust, and accurate motion field estimation from stereo image sequences in real-time. In *ECCV*, pages IV: 582–595, 2010. 3

[31] C. Reinbacher, M. Rüther, and H. Bischof. Fast variational multi-view segmentation through backprojection of spatial constraints. *Image and Vision Computing*, 30(11):797–807, 2012. 2

[32] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, pages 519–528, 2006. 1

[33] Q. Shan, R. Adams, B. Curless, Y. Furukawa, and S. M. Seitz. The visual turing test for scene reconstruction. In *3DV*, pages 25–32, 2013. 1

[34] J. Starck and A. Hilton. Surface capture for performance-based animation. *IEEE Computer Graphics and Applications*, 27(3):21–31, 2007. 1

[35] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, volume 2. IEEE, 1999. 2

[36] C. Strecha, W. von Hansen, L. J. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *CVPR*, 2008. 1

[37] J. Sun, W. Zhang, X. Tang, and H.-Y. Shum. Background cut. In *ECCV*, pages 628–641, 2006. 2

[38] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, A. Agarwala, and C. Rother. A comparative study of energy minimization methods for markov random fields. In *ECCV*, pages 16–29, 2006. 6

[39] S. Vedula, S. Baker, P. Rander, R. T. Collins, and T. Kanade. Three-dimensional scene flow. In *ICCV*, pages 722–729, 1999. 1

[40] C. Vogel, S. Roth, and K. Schindler. View-consistent 3d scene flow estimation over multiple frames. In *ECCV*, pages 263–278, 2014. 3

[41] H. Vu, P. Labatut, J. P. Pons, and R. Keriven. High accuracy and visibility-consistent dense multi-view stereo. *PAMI*, 2011. 1

[42] W. Yang, G. Zhang, H. Bao, J. Kim, and H. Y. Lee. Consistent depth maps recovery from a trinocular video sequence. In *CVPR*, 2012. 3

[43] A. Zaharescu, E. Boyer, and R. P. Horaud. Topology-adaptive mesh deformation for surface evolution, morphing, and multi-view reconstruction. *PAMI*, 33(4):823–837, 2011. 2

[44] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. S. Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. on Graphics*, 23(3):600–608, 2004. 2, 3, 4, 5, 7

[45] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 28–31, 2004. 6

[46] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006. 6