Inferring $3D$ Volumetric Shape of Both Moving Objects and Static Background Observed by a Moving Camera

Chang Yuan and Gérard Medioni
Institute of Robotics and Intelligent Systems
University of Southern California, Los Angeles, CA 90089, USA
{cyuan, medioni}@usc.edu

Abstract

We present a novel approach to inferring $3D$ volumetric shape of both moving objects and static background from video sequences shot by a moving camera, with the assumption that the objects move rigidly on a ground plane. The $3D$ scene is divided into a set of volume elements, termed as voxels, organized in an adaptive octree structure. Each voxel is assigned a label at each time instant, either as empty, or belonging to background structure, or a moving object. The task of shape inference is then formulated as assigning each voxel a dynamic label which minimizes photo and motion variance between voxels and the original sequence. We propose a three-step voxel labeling method based on a robust photo-motion variance measure. First, a sparse set of surface points are utilized to initialize a subset of voxels. Then, a deterministic voxel coloring scheme carves away the voxels with large variance. Finally, the labeling results are refined by a Graph Cuts based optimization method to enforce global smoothness. Experimental results on both indoor and outdoor sequences demonstrate the effectiveness and robustness of our method.

1. Introduction

We study video sequences with one or more objects moving rigidly on a ground plane. There may exist static $3D$ structure in the scene as well. The video camera undergoes general $3D$ motion while imaging the scene by perspective projection. A typical example is that of an airborne camera following vehicles running on a road.

The goal of our approach is to infer the $3D$ dense shape of the scene from video sequences. The scene, including moving objects and static background, is divided as a set of $3D$ volume elements, called voxels, organized in an adaptive octree structure [16]. The shape inference task is then formulated as a voxel occupancy labeling problem, namely deciding whether each voxel is occupied by an object or not.

This work has a broad range of potential applications. For instance, it can be applied to improve aerial surveillance systems with the inference of 3D distance and motion of moving objects. It can also be used for image synthesis with the 3D volumetric shape of moving objects.

The reason for aiming at a voxel-based method rather than a surface-based one [11] is that it is very difficult to recover the $3D$ shape of an object's surface from a moving camera while the surface itself is also moving. Existing solutions (e.g. [8]) work only with an affine camera, but not a perspective camera in our case. In contrast, as the voxels remain static in the scene, their occupancy can be effectively determined by photo consistency tests [13, 18], regardless whether the reconstructed object is moving or not. Object surfaces can be then inferred from the voxel occupancy results, although at a coarse level.

The main difference between our approach and existing volumetric methods [13, 5, 15, 14, 19, 17] is that our method directly handles the case that moving objects are observed by a moving camera. As the moving objects occupy different voxels at different times, the label of a voxel may change over time. In other words, a dynamic label, instead of a constant one, is assigned to each voxel. Therefore, the dense $3D$ shape of background structure and moving objects is inferred in a uniform manner.

Another difference is that we do not have as much control of the scene (*e.g.* turning tables) as in the previous methods, as our real-world videos are captured from passive image sensors. The number of calibrated views (usually 8-10) is also smaller than those methods. Therefore, our results may not have the same quality level as those methods do. However, our approach can be applied to not only the classical area of photo-realistic image synthesis, but also to $3D$ object tracking and mensuration.

The overview of our approach is presented in Figure 1. A number of initial processes are required before the shape inference process. The first process [21] is to segment the original images into motion regions and static background. After removing the pixels that belong to the static back-

1-4244-1180-7/07/$25.00 ©2007 IEEE

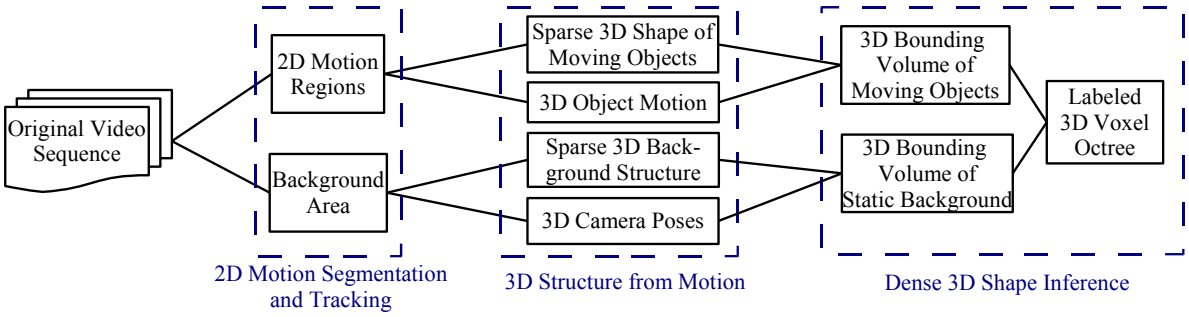


Figure 1. Overview of our approach.

ground, we identify a number of motion regions in each frame and tracked them across the sequence. The second one [20] is to build a sparse reconstruction of both the background and moving objects, namely camera poses and sparse sets of $3D$ points. Structure from Motion techniques [3, 10] are applied to reconstruct the background and each moving object individually. The 3D motion of moving objects is solved by assuming that the object motion trajectory is planar [20].

The volumetric shape inference process starts by determining the bounding volume of voxels. A slanted pyramid is established for each camera by projecting the image rectangle onto the ground plane. The bounding volume, which is the union of all the pyramids, is decomposed into an octree structure of voxels with different sizes. The voxels which correspond to image patches with more details, such as motion regions, are sub-divided into smaller voxels.

The voxel labeling process is based on a robust photo-motion variance measure defined between two voxels at two times. The measure combines both photometric and motion cues to determine if two voxels belong to the same surface patch of an object. The photo variance is defined as the normalized cross correlation between multiple oriented image patches projected from voxels. The motion variance evaluates how close the moved voxel center is to the other one.

The voxel labeling process is divided into three steps. The first one utilizes the sparse set of points solved in the initial processes. Since these points are located on the object surface, the voxels which contain these points are labeled as the corresponding object. The voxels which intersect with optical rays extending to these surface points are labeled as empty.

The second step is a deterministic voxel coloring process. Each voxel is iterated in the ascending order of its distance to the camera centers. At each voxel, the photo-motion variance is evaluated if the voxel belongs to every possible object. If the minimum variance is above a threshold, the voxel is carved away. Otherwise, the voxel is assigned the label of that object (static background is also treated as an object with zero motion). The photo-motion variance measure is evaluated between multiple pairs of frames, in order to remove possible occlusion effects. This deterministic step, however, does not impose any global constraints onto labeling results and relies on a pre-determined threshold.

The final step of the labeling process refines the labeling results by an energy minimization based method. A global energy function which integrates the photo-motion variances from all voxels is minimized by the Graph Cuts method [1]. The energy function also includes a smoothness cost which imposes smoothness constraint to the adjacent voxels in both temporal and spatial domain. The deterministic and graph cuts methods are repeated a few times with different variance thresholds, which reduces the chances of getting stuck at a local minimum.

The rest of the paper is organized as follows. The related work is reviewed in Section 2. Section 3 presents the initial processes of our approach. The volumetric decomposition and the three-step voxel labeling method are introduced in Section 4 and 5 respectively. The experimental results are shown in Section 6, followed by the conclusion and discussion in Section 7.

2. Related Work

There exists a large body of work on building volumetric models from multiple calibrated images. A survey and performance evaluation of recent approaches are presented in [12]. Only those methods that are most relevant to our paper are reviewed here. These methods are divided into two categories: deterministic methods and energy minimization based methods.

The Voxel Coloring method [13] is one of the first deterministic methods that combine image silhouette and color information to build $3D$ volumetric models. Since any voxel on a Lambertian surface corresponds to consistent image patterns, a photo consistency test is applied to every voxel: if the color variance is larger than a threshold, the voxel is labeled as empty; otherwise it is labeled as part of the object surface. A number of subsequent approaches, such as Space Carving [5] and Generalized Voxel Coloring [14], extend the original approach with more general cam-

era placements and more efficient labeling methods. These deterministic methods, however, face the difficulty of finding an appropriate threshold for carving the inconsistent voxels, which is inherently varying in different image regions. Furthermore, these methods does not impose global smoothness and may make conflicting decisions in the sequential carving process.

In contrast, energy minimization based methods do not have such problems. Instead of making a harsh decision for each voxel, a global energy function is defined to accumulate the variance of all voxels. This energy function is minimized by discrete optimization techniques, such as Graph Cuts [1]. [15] first introduced the Graph Cuts method to find the optimal visual hull. [19] and [17] extend the approach by utilizing the visual hull as a topological constraints over the scene and searching for an optimal $3D$ volumetric cut in a voxel based graph. Further constraints are imposed with known surface patches. Our method extends these approaches to handle dynamic scenes with moving objects.

To our knowledge, the only other paper on volumetric reconstruction of dynamic video scenes is [18]. In it, the space carving scheme is extended to carving pairs of voxels between two time instants, called hexels. A stereo pair of camera is used to capture the $3D$ scene, thus allowing non-rigid object motion and dense scene flow. In our case, however, we use only one camera and assume that the object motion is rigid. Furthermore, our approach is not limited to two frames, but also works on long sequences.

3. Initial Processes of the Approach

A number of initial processes are needed before inferring the dense $3D$ shape. The first process is to segment the moving objects from the static background and then track them along the video sequence [21]. $2D$ homographies between consecutive frames, which is induced by the ground plane, are robustly computed from the matched Harris corners [2]. Any pixels that are inconsistent with the inter-frame homographies are labeled as residual pixels, including motion regions and parallax pixels corresponding to static $3D$ structures. The parallax pixels are filtered out by imposing epipolar constraints [3] and parallax rigidity constraints [21]. The detected motion regions are then linked into different object trajectories across multiple frames by a spatio-temporal tracking method.

Consequently, the original video sequence is segmented into the static background (plane and parallax) and a number of 2D object trajectories. Video frames with large inter-frame camera motion are preferred for a reliable reconstruction. Therefore, a number of distinct frames with long baselines, called "key frames", are automatically selected from the whole sequence [10]. Hereafter, the reconstruction processes only deal with these key frames.

The second process applies Structure from Motion (SfM) techniques to these key frames to build a sparse $3D$ reconstruction of both the static background and moving objects [20]. Camera poses and the position of the ground plane are obtained by a plane-based self-calibration method [9]. Then the $3D$ points on static structures are computed by classical SfM techniques [3, 10]. The 3D shape of each moving object is solved by the same SfM techniques based on the relative motion between the object and the moving camera. The object scale and motion trajectory is recovered by enforcing an additional constraint that the object motion must be parallel to the ground plane [20].

The output of the two initial processes includes a set of calibrated images, in which a number of motion regions are identified and tracked. The sparse $3D$ shape of background structure and moving objects is estimated as well as the $3D$ object motion trajectory.

4. Volumetric Decomposition

In this section, we discuss how the $3D$ scene is adaptively decomposed into a set of voxels. Then a robust photo-motion variance measure is defined to evaluate how well the voxels correspond to the original images.

4.1. Bounding volume estimation

With the $2D$ motion regions and $3D$ camera poses obtained in previous processes, we can determine the $3D$ bounding volumes of the static background and moving objects. Most previous approaches build a visual hull by intersecting the optical rays originated from image silhouette boundaries [6]. However, this is not possible in the scene with moving objects, as the optical rays back-projected from $2D$ motion regions do not intersect at the same point.

In our situation, the ground plane is used to bound the object volume. Originated from each camera center, four optical rays pass the image corners until they hit the ground plane. The resulted quadrangle is connected to the camera center and forms a $3D$ slanted pyramid. This pyramid serves as the visual hull in our situation, as is easy to prove that every object voxel lies in the pyramid. Similarly, a $3D$ pyramid is determined for each moving object by projecting the boundaries of motion regions in each image to the ground plane. Fig. 2 illustrates the $3D$ pyramid generated by a single camera. The final bounding volume is the union of the pyramids generated by all cameras. One advantage of using the ground plane is that the bounding volume can be determined automatically, without knowing the scene dimension in advance.

4.2. Adaptive volume tessellation

The $3D$ object volume is divided into a set of voxels adapting to different levels of details in the scene. Each voxel can be subdivided into eight sub-voxels with smaller

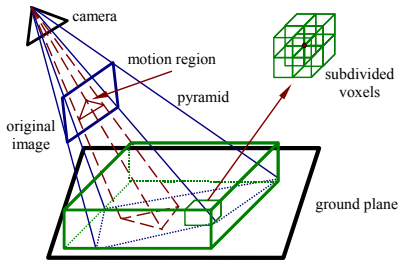


Figure 2. $3D$ volumetric decomposition of the scene. The bounding pyramid (in blue) is determined by projecting $3D$ optical rays passing the corners of image rectangles (in blue) and motion regions (in red) onto the ground plane. The voxels in the pyramid may be sub-divided into smaller voxels, which are organized in an octree structure.

sizes, as shown in Fig. 2. In order to determine the subdivision level of a certain voxel, we project the bounding box of this voxel into all the original images and measure the sizes of the projected polygons. The polygon size affects the accuracy of photo consistency tests and ultimately the reconstruction quality. Therefore, a voxel is sub-divided continuously until its projected polygon is no larger than the maximal region size. In our experiments, it is set to be 20×20 for the background area and 5×5 for motion regions.

An hierarchical spatial structure, namely octree [16], is utilized to organize these voxels in a coarse-to-fine order. Only the leaf nodes in the octree are used in the shape inference process. Hereafter, any voxel we refer to is indeed a leaf node in the octree.

4.3. Photo-motion variance

In the voxel labeling process, if a voxel belongs to the static background or any moving object, the voxel itself and its corresponding voxel at any other time must have consistent photo appearances and similar $3D$ motion. In order to measure this consistency, we define a robust photo-motion variance function between two voxels at two different time instants. If the two voxels belong to the static background, they are indeed identical. Otherwise, the two voxels reside at different locations being related by $3D$ object motion.

Let a voxel in the $3D$ space by $\mathbf{v}$ with its bounding box as $\mathbf{B}(\mathbf{v})$ and its center as $\mathbf{C}(\mathbf{v})$. Let $t \in \mathbf{T} = \{1, \ldots, T\}$ denote a time instant where T is the number of key frames. The photo-motion variance is defined between voxel $\mathbf{v}$ at time t and voxel $\mathbf{v}'$ at time t' as follows,

$$\phi(\mathbf{v}, t, \mathbf{v}', t') = \lambda_p \phi_p(\mathbf{v}, t, \mathbf{v}', t') + \lambda_m \phi_m(\mathbf{v}, t, \mathbf{v}', t') \quad (1)$$

where ϕ_p and ϕ_m are respectively the photo and motion variance normalized into $[0, 1]$. $\lambda_p, \lambda_m \in (0, 1)$ balance the contribution of two variances as $\lambda_p + \lambda_m = 1$. The larger the variance is, the less likely the two voxels belong to the same part of object surface.

Similar to [17], the photo variance is computed based on the normalized cross correlation (NCC) between image patches. The NCC is affected by the orientation of image patches and will reach its maximum if the $2D$ patch orientations are consistent with the corresponding $3D$ surface orientation. Since the $3D$ surface orientation is unknown, we need to extract multiple oriented image patches for the same voxel to increase the probability of solving the true $3D$ surface orientation.

A number of orientation angles (usually 3-4) are estimated at the projected voxel center by finding the maxima in $1D$ gradient histograms [7]. From each orientation angle θ_i, an image patch, $\mathbf{w}_i(\mathbf{v}, t)$, is extracted at the projection of voxel center $\mathbf{C}(\mathbf{v})$ in image $\mathbf{I}_t$. Similarly a number of image patches, $\mathbf{w}'_j(\mathbf{v}', t')$, are extracted from image $\mathbf{I}_{t'}$ for voxel $\mathbf{v}'$.

The final photo variance is computed by selecting the maximum NCC between pairs of image patches :

$$\phi_p(\mathbf{v}, t, \mathbf{v}', t') = \alpha(t, t')(1 - \max_{i,j} \text{NCC}[\mathbf{w}_i(\mathbf{v}, t), \mathbf{w}'_j(\mathbf{v}', t')]) \quad (2)$$

where $\alpha(t, t') \in [0, 1]$ is a weighting factor inversely proportional to the distance between camera centers at time t and t'. α is scaled such that it is 1 for the same camera and 0.5 for the largest distance between camera centers. Note that, (2) does not always prefer the closest view. Instead, it selects the window orientations which maximize NCC between any pair of views.

Fig. 3 gives an example of computing photo variance between multiple oriented image patches. Among all the patches (labeled in green) in each image, the one that leads to maximum NCC or equivalently minimum photo variance is selected as the best patch (in red). The $2D$ orientation of the best patch is consistent with the $3D$ orientation of the ground plane.

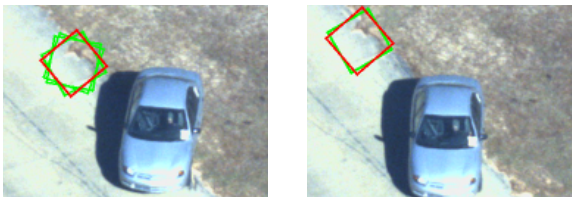


Figure 3. Examples of oriented image patches projected from the same voxel into two images. Among all the extracted image patches (in green), the one that leads to minimum photo variance is selected as the best patch (in red).

The motion variance between two voxels depends on the $3D$ displacement between two voxels. Let the $3D$ rigid motion of object k from time t to t' be denoted by $\mathbf{M}^k_{t,t'} = (\mathbf{R}^k_{t,t'}, \mathbf{T}^k_{t,t'})$ where $\mathbf{R}$ and $\mathbf{T}$ are respectively $3D$ rotation and translation of the object. For the static background, $\mathbf{R}^k_{t,t'} = \mathbf{I}$ and $\mathbf{T}^k_{t,t'} = \mathbf{0}$. Then the motion variance

1	2	3	4	5
EMP	EMP	EMP	EMP	EMP
BAK	BAK	BAK	BAK	BAK
EMP	OBJ_1	OBJ_1	OBJ_1	EMP
OBJ_1	EMP	OBJ_2	OBJ_2	EMP

Table 1. Examples of voxel label sequences at different time instants: EMP (empty), BAK (background), OBJ_k (the k^{th} moving object)

is defined by evaluating how closely the center of voxel $\mathbf{v}$ is driven by motion $\mathbf{M}_{t,t'}^k$ to the center of voxel $\mathbf{v}'$:

$$\phi_m(\mathbf{v}, t, \mathbf{v}', t') = 1 - \frac{\|\mathbf{R}_t^k \mathbf{C}(\mathbf{v}) + \mathbf{T}_t^k - \mathbf{C}(\mathbf{v}')\|}{r(\mathbf{v}')} \quad (3)$$

where $r(\mathbf{v}')$ is the half diagonal length of voxel box $\mathbf{B}(\mathbf{v}')$. Intuitively, when the voxel center $\mathbf{C}(\mathbf{v})$ moves to overlap with the other voxel center $\mathbf{C}(\mathbf{v}')$, the motion variance between two voxels is 0. When the voxel center $\mathbf{C}(\mathbf{v})$ lies on the surface of voxel box $\mathbf{B}(\mathbf{v}')$, the motion variance is 1.

5. Voxel Labeling Process

In this section, we will first introduce preliminary definitions and then propose a three-step voxel labeling method.

A label $l = L(\mathbf{v}, t)$ is assigned to voxel $\mathbf{v}$ at a time instant $t \in \{1, \ldots, T\}$. The label l can be EMP (empty), BAK (static background), or OBJ_k (the k^{th} moving object), $k = 1, \ldots, K$ where K is the number of moving objects. The static background can be considered as the 0^{th} object, namely BAK=OBJ_0.

The label for a voxel may change over time, resulting in a label sequence $\mathbf{L}(\mathbf{v}) = \{L(\mathbf{v}, t) | t = 1, \ldots, T\}$.A voxel can be either invisible to a certain camera (INV), or empty (EMP), or belonging to the background structure (BAK). Once a voxel is labeled as the background, its label remains at BAK all the time. Otherwise, a voxel may be transit from being empty to being occupied by a moving object, or even occupied by different moving objects. Typical examples of label sequences are shown in Table 1.

5.1. Initialization with surface points

A number of $3D$ points have already been computed in the SfM process. These points, which lie on the surface of background structure and moving objects, are used to initialize a subset of voxel labels. Let $\mathbf{P}_t^k$ denote a $3D$ point belonging to object k at time t. A $3D$ line segment, $\overrightarrow{\mathbf{C}_t \mathbf{P}_t^k}$, is established by connecting $\mathbf{P}_t^k$ to camera center $\mathbf{C}_t$, which is a part of the optical ray from $\mathbf{C}_t$. All the voxels intersected with the line segment are labeled as empty at time t, while the voxel containing $\mathbf{P}_i^k$ itself is labeled as OBJ_k at time t.

More voxel labels can be initialized with further information about the scene. For example, if a $3D$ surface patch is known to be part of the k^{th} object, the voxels intersected with the patch are assigned OBJ_k and all the voxels penetrated by the corresponding line segments are labeled as empty. This is especially useful when the ground plane is known to occupy a large portion of the original scene.

5.2. Deterministic labeling method

We propose a deterministic voxel labeling method to assign each voxel a label sequence based on the photo-motion variance. It is assumed that the scene does not lie in the convex hull of all the camera centers. Therefore, we can iterate through the voxels in the ascending order of the distance of voxel centers to the surface of camera bounding box, which guarantees that the labeling decision for a voxel only affects the decision for subsequently visited voxels, but not the previous ones [13].

For each voxel $\mathbf{v}$ that is visible at time t, we compute its photo-motion variance as if it is assigned OBJ_k. If $\mathbf{v}$ is assigned the background (BAK) label, $\mathbf{v} = \mathbf{v}'$. The photo variance is computed by averaging its variance scores between t and other time instants:

$$\phi(\mathbf{v}, t, \text{BAK}) = \frac{1}{|\mathbf{T}_b(\mathbf{v})|} \sum_{t' \in \mathbf{T}_b(\mathbf{v})} \phi(\mathbf{v}, t, \mathbf{v}, t') \quad (4)$$

where $\mathbf{T}_b(\mathbf{v}) \subseteq \mathbf{T}$ is the set of time instants at which $\mathbf{v}$ is visible and belongs to the static background. Since $\phi_m(\mathbf{v}, t, \mathbf{v}, t') = 1$, Eq. (4) is similar to the photo variance functions defined in [13, 17].

if $\mathbf{v}$ is assigned OBJ_k ($k \geq 1$) at time t, the new position of voxel center $\mathbf{C}(\mathbf{v})$ at any other time t' is predicted by its motion from t to t'. if the moved voxel center does not lie in the bounding volume of object k, the variance is not evaluated. Otherwise, the voxel $\mathbf{v}'$ containing the moved voxel center is searched in the octree and is used to compute the variance $\phi(\mathbf{v}, t, \mathbf{v}', t')$. The final variance for $\mathbf{v}$ being assigned label OBJ_k is computed as,

$$\phi(\mathbf{v}, t, \text{OBJ}_k) = \frac{1}{|\mathbf{T}_m(\mathbf{v})|} \sum_{t' \in \mathbf{T}_m(v)} \phi(\mathbf{v}, t, \mathbf{v}', t') \quad (5)$$

where $\mathbf{T}_m(\mathbf{v}) \subseteq \mathbf{T}$ is the set of time instants satisfying: 1) $\mathbf{v}$ is visible to the camera time t; 2) $\mathbf{v}$ moves to overlap with $\mathbf{v}'$; 3) $\mathbf{v}'$ is visible to camera at t' and remains in the bounding volume of object k. Eq. (5) is a generalized version of the photo variance defined in [18].

The overall deterministic labeling method is summarized in Algorithm 1. The voxels are iterated in the ascending order of the distance between its center to the bounding box of camera centers. For each voxel $\mathbf{v}$, the photo-motion variance, $\phi(\mathbf{v}, t, l)$, is computed at time t with every possible object label l (BAK or OBJ_k). Let l^* denote the label which minimizes $\phi(\mathbf{v}, t, l)$. If the minimum variance, $\phi(\mathbf{v}, t, l^*)$,

is below a threshold δ, the voxel is labeled as l^* at time t, $L(\mathbf{v}, t) = l^*$. Otherwise, the voxel is labeled as empty at time t, or say the voxel is "carved". Notice that we need to make T labeling decisions for each voxel $\mathbf{v}$, instead of only one in the case of static scenes.

```
Sort all the voxels in the ascending order based on
their distances to the camera convex hull;
for every voxel v in the sorted list do
    for every time instant t ∈ {1, ..., T} do
        if v is occluded by other voxels then
            go to next voxel;
        for every possible object label l do
            compute photo-motion variance φ(v, t, l);
        end
        find the label l* which minimizes φ(v, t, l) ;
        if φ(v, t, l*) < δ then
            L(v, t) = l*;
        else
            L(v, t) =EMP;
        end
    end
end
```

Algorithm 1: Deterministic voxel labeling algorithm

This deterministic method, however, has a few limitations. First, it relies on a pre-determined threshold which is hard to find with different sequences, even on different image regions. Second, it may make decision for a voxel that conflicts with another one for a voxel visited earlier, as it only considers "local" information available to the current voxel. Third, the labeling method does not impose any global smoothness constraints onto the voxel labels.

5.3. Graph cuts based optimization

The limitations of the deterministic labeling method can be avoided by considering all the voxels at the same time. Let the total set of labels per each voxel per each time instant be denoted by $\mathbf{L} = \{L(\mathbf{v}, t) | v = 1, \ldots, V, t = 1, \ldots, T\}$. We define an energy function over the label set $\mathbf{L}$ [1] as follows,

$$E(\mathbf{L}) = \sum_{v,v'} \sum_{t,t'} D(\mathbf{v}, t, \mathbf{v}', t', l) + \sum_{v,v'} \sum_{t,t'} V(l, l') \quad (6)$$

where $(\mathbf{v}, \mathbf{v}')$ denotes the pair of voxels adjacent in space and (t, t') the pair of adjacent time instants.

This energy function is expected to reach its minimum when all the voxels are given correct labels. Therefore, the voxel labeling program is converted into a global energy minimization problem. The graph cuts algorithm is applied to efficiently solve this problem, with both the α-expansions and $\alpha\beta$-swap iterations repeated until the global energy converges to its minimum [1].

The data cost function $D(\cdot)$ measures how well the voxel labels match the original images. In our implementation, it is defined as the photo-motion variance $\phi(\mathbf{v}, t, \mathbf{v}', t', l)$ with additional surface constraints. Specifically, when the voxel $\mathbf{v}$ is known to be on the surface of the k^{th} object, $D(\mathbf{v}, t, l) = +\infty$ if $l \neq \text{OBJ}_k$. This guarantees that the resulted surface passes the existing surface points.

The second cost $V(\cdot)$ imposes smoothness constraints over voxels adjacent in both spatial and temporal domain. For voxel $\mathbf{v}$, $(\mathbf{v}, t)$ is the neighbor of the voxel itself at any other time instant $(\mathbf{v}, t')$. For a pair of voxels $\mathbf{v}$ and $\mathbf{v}'$, $(\mathbf{v}, t)$ and $(\mathbf{v}', t')$ are neighbors only if $\mathbf{v}$ and $\mathbf{v}'$ are adjacent in the $3D$ space and $|t - t'| < \delta_t$ where δ_t is a constant time interval. The smoothness term $V(l, l')$ is set to be the classical Potts energy model [1].

5.4. Summary of the labeling process

The final method combines all the three steps, as is summarized in Algorithm 2. The combination of deterministic method and Graph Cuts based method is repeated a few times with different variance threshold drawn within the valid range of photo-motion variance. Indeed, this changes the starting point of the optimization process and reduces the probability of getting stuck at local minimum. The labeling which leads to the minimum energy is kept as the final result.

```
Initialize the voxels with surface points;
repeat
    Randomly select a deterministic threshold within
    the range of photo-motion variances;
    Label the voxels by the deterministic method;
    Refine the labels by the graph cuts method;
    Keep the labeling results with minimum energy;
until enough trials have been finished ;
```

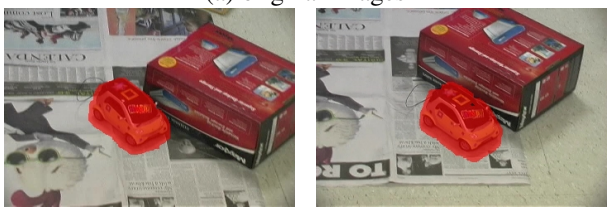
Algorithm 2: Voxel labeling process

The time complexity of our algorithm is analyzed as follows. Let L, V, T denote respectively the number of possible object labels, the total number of voxels, and the number of time instants. The deterministic method makes $O(LT)$ operations at each voxel by finding the best label sequence, resulting in totally $O(LTV)$ operations. The graph cuts method evaluates the data cost of each voxel at each time and the smoothness cost between each voxel-time pairs, resulting in $O(LTV + LT^2V^2)$ operations. Therefore, the total complexity of the voxel labeling method is $O(LT^2V^2)$.

6. Experimental Results

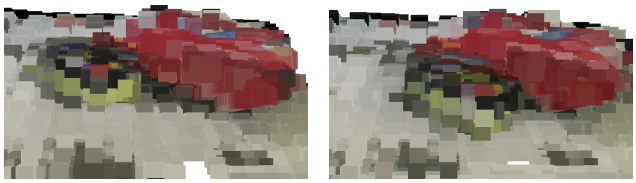
We demonstrate the effectiveness and robustness of our method by testing it against two video sequences. The first video, called "toy-car", is a 150-frame sequence shot indoors by a hand-held camera. A toycar moves on the

(a) original images



(b) motion regions

(c) sparse $3D$ shape



(d) dense $3D$ space (after deterministic coloring)

(e) dense $3D$ shape (after graph cuts)

(f) dense $3D$ shape (after graph cuts; another view)

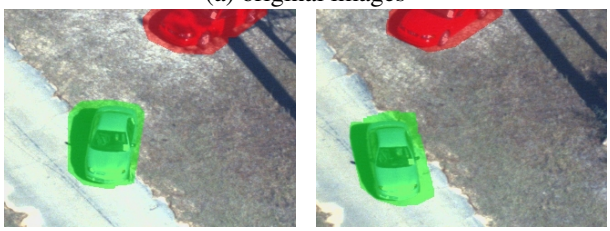
Figure 4. Results of frame 90 (left) and 110 (right) from the "toy-car" sequence.

ground, while a box is placed as the background structure. The results of two frames, out of totally 8 key frames, are shown in Fig. 4.

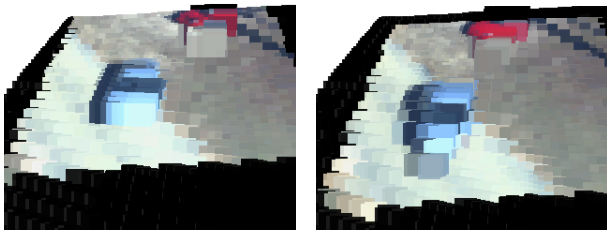
The original images in Fig. 4(a) are segmented into motion regions belonging to the toycar (labeled in red) and the static background shown in Fig. 4(b). Fig. 4(c) shows the $3D$ background structure (the ground plane and the box), as well as the $3D$ object trajectory (in gray), in which the sparse object shape at the current frame is indicated by red color.

The inferred dense $3D$ shape of both the background and moving objects is shown in Fig. 4(d)-(f), where each

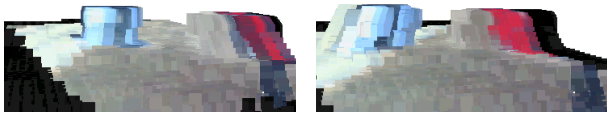
(a) original images



(b) motion regions



(c) dense $3D$ shape (after graph cuts)



(d) dense $3D$ shape (after graph cuts; another view)

Figure 5. Results of frame 1200 (left) and 1232 (right) from the "forest" sequence.

opaque voxel is rendered with the average color of its corresponding $2D$ image patches. Fig. 4(d) shows the labeling results by the deterministic coloring step, which does not impose global smoothness constraint. After refining the labeling results by the graph-cuts step, we can easily identify the $3D$ shape of the static box and moving toy car from Fig. 4(e). We can also clearly see that the toy car occupies different voxels at two frames from a different view point as in Fig. 4(f).

The second sequence, called "forest", is a 100-frame one shot by an airborne camera which follows two cars making turns on the ground. The original images in Fig. 5(a) are segmented into the motion regions (different colors indicate different object labels) and the ground plane as shown in Fig. 5(b). The $3D$ dense shape of both cars and the ground plane is correctly inferred from 9 key frames, as shown in Fig. 5(c) and (d). The $3D$ object motion can be clearly identified by comparing the two images in Fig. 5(e).

Below are a few numbers about space and time complexity of our method. The scene in "toycar" sequence is decomposed into a total of 135,217 voxels , of which 32%

belongs to the toy car. The three processes (motion segmentation and tracking, SfM, and dense shape inference) take approximately 20 minutes, 5 minutes and 1.5 hours respectively, in which the three labeling steps take respectively 5 minute, 20 minutes and 1 hour. The intensive parallel computation in these processes can be accelerated to 10 or more times faster by GPU implementation [4].

7. Conclusion and Future Work

We have proposed a novel approach to inferring $3D$ dense shape of the scene where objects move rigidly on a ground plane being observed by a moving camera. The scene is decomposed into a set of voxels organized in an adaptive octree structure. A robust photo-motion variance measure was introduced to evaluate the inconsistency between voxels and original images. We proposed a three-step method for finding the best dynamic label for each voxel to minimize the total variance. The experimental results demonstrated the effectiveness and robustness of our method.

There are a few directions for future research. It is necessary to investigate how to reliably extract surface meshes, which change due to object motion, from the labeled voxels. Furthermore, it would be interesting to extend the rigid object motion to non-rigid deformations, with the potential applications in $3D$ human modeling and motion capture.

Acknowledgments

This research was funded, in part, by the U.S. Government VACE program.

References

[1] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. *IEEE Trans. PAMI*, 20(12):1222–1239, 2001.

[2] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of 4th Alvey Vision Conference*, 1988.

[3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[4] A. Hornung and L. Kobbelt. Robust and efficient photo-consistency estimation for volumetric 3d reconstruction. In *ECCV*, vol. II, pp. 179–190, 2006.

[5] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *IJCV*, 38(3):199–218, Jul 2000.

[6] A. Laurentini. The visual hull concept for silhouette-basedimage understanding. *IEEE Trans. PAMI*, 16(2):150–162, 1994.

[7] D. Lowe. Distinctive Image Features from Scale-invariant Keypoints. *IJCV*, 60(2):91–110, 2004.

[8] A. Maki, M. Watanabe, and C. Wiles. Geotensity: Combining motion and lighting for 3d surface reconstruction. *IJCV*, 48(2):75–90, 2002.

[9] E. Malis and R. Cipolla. Camera self-calibration from unknown planar structures enforcing the multiview constraints between collineations. *IEEE Trans. PAMI*, 24(9):1268–1272, 2002.

[10] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, and J. Tops. Visual modeling with a hand-held camera. *IJCV*, 59(3):207–232, 2004.

[11] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47:7–42, 2002.

[12] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. of IEEE CVPR*, pp. 519–526, 2006.

[13] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. *IJCV*, 35(2):151–173, 1999.

[14] G. G. Slabaugh, W. B. Culbertson, T. Malzbender, M. R. Stevens, and R. W. Schafer. Methods for volumetric reconstruction of visual scenes. *IJCV*, 57(3):179–199, 2004.

[15] D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *Proc. of IEEE CVPR*, pp. 345–352, 2000.

[16] R. Szeliski. Rapid octree construction from image sequences. *CVGIP: Image Understanding*, 57:23–32, 1993.

[17] S. Tran and L. Davis. 3d surface reconstruction using graph cuts with surface constraints. In *ECCV*, vol. II, pp. 219–231, 2006.

[18] S. Vedula, S. Baker, S. Seitz, and T. Kanade. Shape and motion carving in 6d. In *Proc. of IEEE CVPR*, pp. 592 – 598, 2000.

[19] G. Vogiatzis, P. H. S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *Proc. of IEEE CVPR*, pp. 391–398, 2005.

[20] C. Yuan and G. Medioni. 3d reconstruction of background and objects moving on ground plane viewed from a moving camera. In *Proc of IEEE CVPR*, pp. 2261–2268, 2006.

[21] C. Yuan, G. Medioni, J. Kang, and I. Cohen. Detecting motion regions in presence of strong parallax from a moving camera by multi-view geometric constraints. *IEEE Trans. PAMI*. to appear.