Ray Markov Random Fields for Image-Based 3D Modeling: Model and Efficient Inference

Shubao Liu David B. Cooper
Laboratory for Engineering Man/Machine Systems (LEMS)
Division of Engineering, Brown University
Providence, RI 02912
{sbliu, cooper}@lems.brown.edu

Abstract

In this paper, we present an approach to multi-view image-based 3D reconstruction by statistically inversing the ray-tracing based image generation process. The proposed algorithm is fast, accurate and does not need any initialization. The geometric representation is a discrete volume divided into voxels, with each voxel associated with two properties: opacity (shape) and color (appearance). The problem is then formulated as inferring each voxel's most probable opacity and color through MAP estimation of the developed Ray Markov Random Fields (RayMRF). RayMRF is constructed with three kinds of cliques: the usual unary and pairwise cliques favoring connected voxel regions, and most importantly ray-cliques modelling the ray-tracing based image generation process. Each ray-clique connects the voxels that the viewing ray passes through. It provides a principled way of modeling the occlusion without approximation. The inference problem involved in the MAP estimation is handled by an optimized belief propagation algorithm. One unusual structure of the proposed MRF is that each ray-clique usually involves hundreds/thousands of random variables, which seems to make the inference computationally formidable. Thanks to the special property of the ray-clique functional form, we investigate the deep factorization property of ray-clique energy and get a highly efficient algorithm based on the general loopy belief propagation, which has reduced the computational complexity from exponential to linear. Both of the efficient inference algorithm and the overall system concept are new. Combining these results in an algorithm that can reverse the image generation process very fast. 3D surface reconstruction in a 100x100x100, i.e., 10^6 *voxel space with 10 images requires roughly 3 minutes on a 3.0 GHz single-core CPU. The running time grows linearly with respect to the number of voxels and the number of images. And the speed could be further improved with a hierarchical sparse representation of the volume, like octree. Experiments on several standard datasets show the quality and speed of the proposed models and algorithms.*

1. Introduction

Image-based 3D modeling is the problem of recovering scenes' 3D geometry and appearance from images. Nowadays, with the prevalence of digital cameras, image-based 3D modeling has the clear advantage over other 3D modeling techniques in terms of equipment availability, affordability and amount of user input, besides advantages on operation conditions, scalability, etc. Image-based 3D modeling is usually the preferred solution for 3D modeling if satisfactory algorithms can be developed to accomplish its goal under normal working conditions.

There have been large amount of related work on Image-based 3D modeling, thanks to its long history and popularity. Here the historical context of this work is briefly reviewed, then a few related papers are picked up for more detailed review. Image-based 3D modeling started from stereo vision through image-matching and triangulation. Later multi-view extension of triangulation is developed, resulting in the various kinds of multi-view geometries [7]. Recent focus has shifted to statistical models, trying to optimize the re-projection error. This work is no exception. A good summary and evaluation of the recent developments can be found in [17]. Past works can be roughly divided into two categories based on the visibility model they used. The first approach estimates the visibility (i.e., occlusion) based on initial surface [19] or currently estimated surface [6, 12]; The other approach estimates the visibility based on the probabilistic opacity of the volume (usually for voxel based approach) [4, 15]. Our method falls into the second category.

Next several representative papers, which share common elements with our proposed approach, are reviewed.

The iterative space carving idea was first introduced in [12], where a discrete volume is carved based on a photo-consistency function. The shape shrinks from a super shape (usually a bounding box) to "photo-hull". It shares the same kind of weakness as other approaches based on local photo-consistency measurement: If the photo-consistency is defined stringently, then the shape will continue shrink, which may penetrate into a hole, or shrink to nothing. If it is defined loosely, then the shape is a superset of the ground truth shape. Our approach overcomes the problem by having viewing rays that pass through the same voxel compete with each other to reach a final agreement over the voxel's opacity. In this way, it avoids the local consistency checking, which leads to inferior local minimum. In [19], the idea of MRF based modeling and graph-cut based optimization was introduced to multi-view modeling. Their MRF model is a simple extension of 2D MRF commonly used in image processing, where each hidden variable having a direct observation. In [19] the observation of each voxel is defined in a heuristic way, relying on an initial surface to help compute an approximation of the visibility. Our approach shares the idea of using MRF for multi-view stereo, but defines a more realistic model by modeling the image generation process directly and does not require an initial shape. In [4], space carving is extended to the probabilistic setting, where the visibility is modeled probabilistically. The volumetric rendering function is directly used to model the image generation process. But to get a computationally affordable solution, they resort to defined the photo-consistency function based on 2-3 views. And in [15], the Bayesian online updating of a voxel world is extended to model a dynamic 3D scene for change detection. The inference of the model is done through simple iterative online updating based on Bayesian theorem. But the online upating nature make it not guaranted to satisfy all the image constraints. Our approach shares the idea of direct modeling the image rendering process. We build a MRF and carry out inference with more efficient techniques, including EM for voxel color estimation and optimized belief propagation for voxel opacity estimation. In this work, we need to handle high order MRFs to model the occlusion relationship between voxels. Related to this, in [16], Potetz proposed a BP algorithm for a class of higher-order potential functions – linear constraint functions. However, the functional form of our ray clique does not fall into that category. In [21], Wei and Quan expressed the visibility constraints as a large number of $O(N^2)$ of pairwise nodes, and then solved it with graph-cuts. Here we deal with occlusion differently by exploring deep factorization property of the ray-tracing rendering equation.

This paper is organized in the following way. In Sec. 2, a statistical image formation model is introduced and further a Markov Random Field model, based on the image formation model, is formulated. The inference of this model is studied in Sec. 3.2 and 3.1. Sec. 3.2 solves the problem of estimating each voxel's intensity through estimating a marginalized distribution over the surface shape. Sec. 3.1 develops the optimized belief propagation algorithm for the large clique inference problem by taking advantage of the special property of ray-clique functional form. After that, experiments and discussions follow, which demonstrates the quality of proposed solution, and discuss possible extensions of the current work.

2. Ray Markov Random Fields

2.1. Volumetric Representation

The scene's geometry and appearance is modeled with an evenly divided discrete volume, each unit being a voxel. Each voxel has two properties: opacity and color. Opacity is a binary variable, 0 for empty and 1 for solid. The color of the solid voxel represents the appearance of the object. We also assign a color for the empty voxels, just for the convenience of computation and representation, because we don't know which voxel is empty a priori. On the other hand, the color of the empty voxel does not violate the image generation process, because it cannot be observed. Figure 1 and 2 illustrates the basic concepts. Table 1 summaries some of the notations used throughout the paper.

2.2. Statistical Rendering Equation and Ray-Clique

Image is record of the amount of lights received at the imaging sensor. The pixel value I_R of a ray R is proportional to the amount of light it received, equivalently, the amount of light that the surface point x_R^* reflected towards direction R (the geometry relationship between I_R, x_R^*, R is illustrated in Fig. 1):

$$I_R = \gamma L_o(\mathrm{x}_R^*, R) \tag{1}$$

where γ is a scale constant. With suitable units, γ can be set as 1. $L_o(\mathrm{x}_R^*, R)$ is a function of its environmental light and its material. This relationship is modeled by the rendering equation, introduced by Kajiya in 1986 [8]:

$$L_o(\mathrm{x}_R^*, R) = \int_\Omega \rho(\mathrm{x}_R^*, R, R') L_i(\mathrm{x}_R^*, R')(-R' \cdot \mathbf{n}) dR' \tag{2}$$

where $L_o(\mathrm{x}_R^*, R)$ is the radiance of light that point x_R^* sends towards direction R, $L_i(\mathrm{x}_R^*, R')$ is the radiance of light that point x receives from direction R', $\rho(\mathrm{x}_R^*, R, R')$ is the Bidirectional Reflectance Distribution Function (BRDF) of the surface material at x_R^*, which measures the proportion of light reflected from R' to R at position x_R^*. As the first step towards reversing the rendering equation, in this work we do not intend to recover the surface BRDF, instead we

R	viewing ray, which start from the camera center and going through a pixel
I_R	intensity value of the pixel, where the viewing ray R back-projects from
X	the set of voxels in the volume
X_R	the set of voxels on R which passed through volume X, i.e., $X_R = R \cap X$. The points (voxels) are ordered in the viewing ray traversal order.
x	a voxel in discrete volume
x_R	a voxel in X_R, i.e., $\mathrm{x}_R \subset X_R$
$\mathcal{S}$	the set of surface points
x^*_R	the intersection point of R and surface $\mathcal{S}$, i.e., $\mathrm{x}^*_R = X_R \cap \mathcal{S}$
$L_o(\mathrm{x}^*_R, R)$	the radiance of surface point x^*_R towards direction R
R'	an incoming light ray
$L_i(\mathrm{x}^*_R, R')$	the incoming light radiance from direction R' to surface point x^*_R
x^o	opacity of a point (voxel) x
x^c	color intensity of a point (voxel) x
X^o_R	opacity of the points (voxels) on Ray R
X^c_R	color intensity of the points (voxels) on Ray R

Table 1. Notations

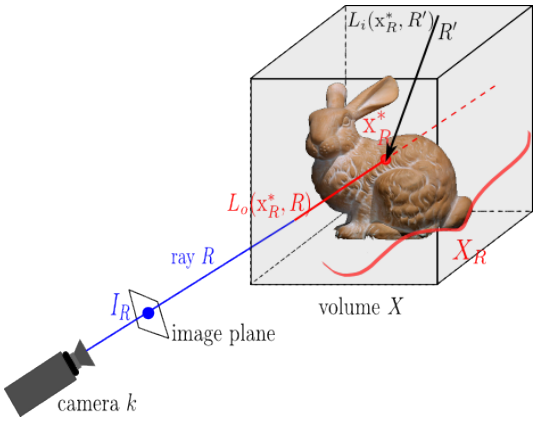
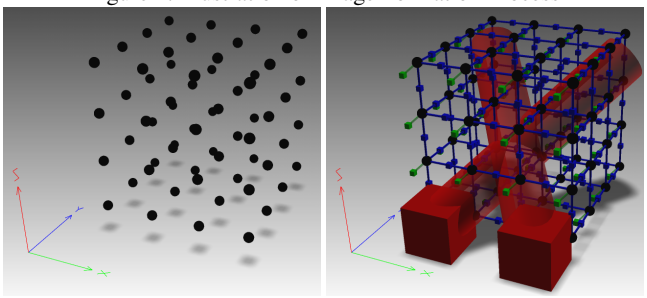


Figure 1. Illustration of Image Formation Process



(a) voxel based representation of a discrete volume

(b) RayMRF

Figure 2. A toy RayMRF model (visualized with the factor graph representation [11]): black balls represent voxels, square cubes represent clique factors and tubes represent clique connections: green for unary-clique, blue for pairwise-clique, and red for ray-clique.

resort to simply the rendering equation by building a statistical rendering model marginalized over BRDF. Under the common assumption that the lighting is ambient (reasonable approximate for most outdoor and indoor scenarios), $\rho(\mathrm{x}^*_R, R, R') * L_i(\mathrm{x}^*_R, R')$ is only a function of (x^*_R, R). Further we use the Phong shading model to approximate the BRDF with two components: diffuse reflection and specular reflection. And based on the property that specular reflection have constant chromatic components for all direction of R, we can build a simple statistical model as

$$I_R \sim \mathcal{N}(\mathrm{x}^{c*}_R, \Sigma_{x^*_R}). \tag{3}$$

where I_R is measured in the CIELab color space. Here we assume the three channels of the color is independent, so the co-variance matrix $\Sigma_{x^*_R}$ is diagonal. Different component of $\Sigma_{x^*_R}$ measures different variance of illumination and chromatic values. (Illumination component normally have larger variance than chromatic values due to the existence of specular reflectance.) Expressed with the probability density function, we have

$$f_R(I_R) = \frac{1}{(2\pi)^{3/2}|\Sigma_{x^*_R}|^{1/2}} \exp(-\frac{1}{2}(I_R - \mathrm{x}^{c*}_R)^T \Sigma^{-1}_{x^*_R}(I_R - \mathrm{x}^{c*}_R)) \tag{4}$$

This statistical model is a minor generalization of the Lambertian model. This pixel value distribution defines a clique, (referred as ray-clique after wards), which is a observation of the points (voxels) along the viewing ray. The ray-clique is unusual due to the fact that it involves (commonly) hundreds to thousands of voxels, in contrast to 2 (pairwise clique) or dozens of (higher order clique) random variables for traditional MRF. It seems formidable to infer this model with ray-clique involved, from previous MRF experience in the first glance, but we will show that there exists very fast efficient inference for ray-cliques in Sec. 3.1.

2.3. Ray MRF

Besides the ray-clique, which models the observation process, there are two other cliques: unary-cliques and pairwise-cliques, modeling the prior distribution of the discrete volume. Unary-cliques model the preference of a voxel being solid or empty; And pairwise-cliques model the smoothness of the volume. Formally, the whole MRF can be expressed in the following energy form:

$$E(X) = \sum_R E_R(X_R) + \alpha_p \sum_{<i,j>\in\mathcal{N}_p} E_p(\mathrm{x}_i, \mathrm{x}_j) + \alpha_u \sum_k E_u(\mathrm{x}_k) \tag{5}$$

where E_R is the clique energy for the ray R, X_R is the set of voxels that ray R passes through; E_u is the unary clique energy; E_p is the pairwise clique energy; α_u and α_p are respectively the weight for unary clique and pairwise clique. The functional form of each energy function is defined as following:

$$E_u(\mathrm{x}_k) = \begin{cases} 0, & \mathrm{x}_k^o = 1 \\ 1, & \mathrm{x}_k^o = 0 \end{cases} \tag{6}$$

$$E_p(\mathrm{x}_i, \mathrm{x}_j) = \begin{cases} 0, & \mathrm{x}_i^o = \mathrm{x}_j^o \\ 1, & \mathrm{x}_i^o \neq \mathrm{x}_j^o \end{cases} \tag{7}$$

$$E_R(X_R) = (I_R - \mathrm{x}^{c*}_R)^T \Sigma^{-1}_{\mathrm{x}^*_R} (I_R - \mathrm{x}^{c*}_R) \tag{8}$$

where

$$\mathrm{x}^{c*}_R = \begin{cases} \mathrm{x}_0^c, & X_R^o = 1, \times, \times, \times, \times, \cdots \\ \mathrm{x}_1^c, & X_R^o = 0, 1, \times, \times, \times, \cdots \\ \cdots & \\ \mathrm{x}_i^c, & X_R^o = \underbrace{0, \cdots, 0}_{i}, 1, \times, \cdots \\ \cdots & \end{cases} \tag{9}$$

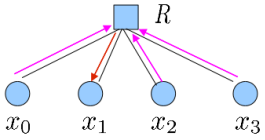
Note that the value of X_R^o is a vector of opacity values for the voxels sorted in the traversal order of the viewing ray. The graphical structure of the MRF is illustrated in Fig. 2, where cliques are represented with a cube representing the clique factor, a tube connecting the random variables (i.e., voxels), following the conventions of factor graph [11].

3. Efficient Inference

3.1. Voxel Opacity Estimation Through Optimized Belief Propagation

The inference of the proposed RayMRF is to estimate voxels' most probable opacity (shape) and color (appearance). In this paper, this is done through a two-stage process. First each voxel's color is estimated through an approximate multi-view image generation process. Then the opacity of voxels can be estimated by highly optimized belief propagation. In this section, the focus is on estimating voxels' opacity, which is one of the main contributions of this paper. The color estimation will be discussed in the next subsection. For MRF, two approximate inference methods, graph cuts [10] and loopy belief propagation (BP) [9] [14] [18], are usually deployed for their good inference quality and speed. Usually cliques involve only a few random variables for most MRF., To our knowledge, there is no literature on how graph-cuts can be applied in the case that each ray-clique involves hundreds/thousands of voxels (random variables). Here we will investigate how loopy belief propagation can be applied here. The direct implementation of BP for ray-clique, which has hundreds of voxels involved, is formidable computationally. The computation for each message sent from a clique to a node increases exponentially with the number of nodes in each clique. In some sense, cliques with more than hundreds of voxels involved violates the essence of MRF, which try to factor the whole joint distribution (energy) into small clique distribution (energy). In [16], Potetz proposed a BP algorithm for a class of higher-order potential functions – linear constraint functions. However, the functional form of our ray clique does not fall into that category. Here we will show that although ray-clique cannot be further factorized, it has some special structure that allow efficient computing of BP messages, resulting a highly optimized belief propagation.

(a) ray-voxel relationship



(b) message passing from R to x_1

Figure 3. Toy Ray-Clique Example

Message sending for self-clique and pairwise-cliques are no special, compared to other MRF applications. To save space, we omit the technical details for them. Here we devote the space on deriving the optimized ray-clique message passing, which plays a key role in the whole inference. For a clique with N binary random variables, the direct implementation of belief propagation takes $O(2^N)$ computation. Here for optimized message passing of ray-clique, the computation is reduced to $O(N)$. Intuitively, this is possible because the ray-clique energy function can only take N possible values, although there are 2^N configurations for the random variables. Detailed formulations are discussed in the the single-column page 6.

3.2. Voxel Color Estimation

We first show that voxels' color can be estimated fairly well independently of voxels' opacity. A voxel can be observed by a camera if it is solid and there is no occlusion between the camera center and the voxel. Each voxel has potentially N (the number of cameras) observations if it can be observed in all cameras. Since we don't know which

To make things clear, let's start from a simple example – a ray-clique which has 4 voxels: $E_R(\mathrm{x}_0, \mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3)$, as illustrated in Figs. 3. Let's derive the message that the ray clique send to voxel x_1, for example. In the following derivation, we follow the convention: The messages are represented in log scale to be immune to over-floating, so the BP is in min-sum form; The message is a 2D vector (one component for $\mathrm{x} = 0$ (empty), another one for $\mathrm{x} = 1$ (solid)). Here we normalize the message to make the messages send from node x to ray clique R have $m_{\mathrm{x}\to R}(0) = 0$. This simplifies the notation below and also saves memory in implementation. The "$\times$" symbol denotes either 0 or 1.

$$\begin{aligned} m_{R\to \mathrm{x}_1}(0) &= \min \begin{pmatrix} E_R(0,0,0,0) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(0), \\ E_R(0,0,0,1) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(1), \\ E_R(0,0,1,0) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(0), \\ E_R(0,0,1,1) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(1), \\ E_R(1,0,0,0) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(0), \\ E_R(1,0,0,1) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(1), \\ E_R(1,0,1,0) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(0), \\ E_R(1,0,1,1) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(1) \end{pmatrix} \\ &= \min \begin{pmatrix} E_R(0,0,0,0), \\ E_R(0,0,0,1) + m_{\mathrm{x}_3\to R}(1), \\ E_R(0,0,1,\times) + m_{\mathrm{x}_2\to R}(1) + \min(0, m_{\mathrm{x}_3\to R}(1)), \\ E_R(1,0,\times,\times) + m_{\mathrm{x}_0\to R}(1) + \min(0, m_{\mathrm{x}_2\to R}(1)) + \min(0, m_{\mathrm{x}_3\to R}(1)) \end{pmatrix} \end{aligned} \tag{10}$$

$$\begin{aligned} m_{R\to \mathrm{x}_1}(1) &= \min \begin{pmatrix} E_R(0,1,0,0) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(0), \\ E_R(0,1,0,1) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(1), \\ E_R(0,1,1,0) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(0), \\ E_R(0,1,1,1) + m_{\mathrm{x}_0\to R}(0) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(1), \\ E_R(1,1,0,0) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(0), \\ E_R(1,1,0,1) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(0) + m_{\mathrm{x}_3\to R}(1), \\ E_R(1,1,1,0) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(0), \\ E_R(1,1,1,1) + m_{\mathrm{x}_0\to R}(1) + m_{\mathrm{x}_2\to R}(1) + m_{\mathrm{x}_3\to R}(1) \end{pmatrix} \\ &= \min \begin{pmatrix} E_R(0,1,\times,\times) + \min(0, m_{\mathrm{x}_2\to R}(1)) + \min(0, m_{\mathrm{x}_3\to R}(1)), \\ E_R(1,1,\times,\times) + m_{\mathrm{x}_0\to R}(1) + \min(0, m_{\mathrm{x}_2\to R}(1)) + \min(0, m_{\mathrm{x}_3\to R}(1)) \end{pmatrix} \end{aligned} \tag{11}$$

For a general ray-clique involving N voxels: $E_R(\mathrm{x}_0, \cdots, \mathrm{x}_{i-1}, \mathrm{x}_i, \mathrm{x}_{i+1}, \cdots, \mathrm{x}_{N-1})$, let's derive the message that the ray sends to the ith voxel, $m_{R\to \mathrm{x}_i}$. To simplify the notation, define $E_R^j = E_R(\underbrace{0, \cdots, 0}_{j}, 1, \times, \cdots, \times)$, and $E_R^N = E_R(0, \cdots, 0, \cdots, 0)$ which represents the ray-clique energy when the ray hits the background.

$$m_{R\to \mathrm{x}_i}(0) = \min \begin{pmatrix} E_R^N, \\ E_R^{N-1} + m_{\mathrm{x}_{N-1}\to R}(1), \\ \cdots \\ E_R^{i+1} + m_{\mathrm{x}_{i+1}\to R}(1) + \sum_{k=i+1+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) \\ E_R^{i-1} + m_{\mathrm{x}_{i-1}\to R}(1) + \sum_{k=i-1+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) - \min(0, m_{\mathrm{x}_i\to R}(1)), \\ \cdots \\ E_R^0 + m_{\mathrm{x}_0\to R}(1) + \sum_{k=0+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) - \min(0, m_{\mathrm{x}_i\to R}(1)) \end{pmatrix} \tag{12}$$

$$m_{R\to \mathrm{x}_i}(1) = \min \begin{pmatrix} E_R^i + \sum_{k=i+1+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)), \\ E_R^{i-1} + m_{\mathrm{x}_{i-1}\to R}(1) + \sum_{k=i-1+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) - \min(0, m_{\mathrm{x}_i\to R}(1)), \\ \cdots \\ E_R^0 + m_{\mathrm{x}_0\to R}(1) + \sum_{k=0+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) - \min(0, m_{\mathrm{x}_i\to R}(1)) \end{pmatrix} \tag{13}$$

It can be written more concisely as:

$$m_{R\to \mathrm{x}_i}(0) = \min \begin{pmatrix} \min_{j=i+1}^{N-1} \left(\left\{ E_R^j + m_{\mathrm{x}_j\to R}(1) + \sum_{k=j+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) \right\}, E_R^N \right), \\ \min_{j=0}^{i-1} \left\{ E_R^j + m_{\mathrm{x}_j\to R}(1) + \sum_{k=j+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) - \min(0, m_{\mathrm{x}_i\to R}(1)) \right\} \end{pmatrix} \tag{14}$$

$$m_{R\to \mathrm{x}_i}(1) = \min \begin{pmatrix} E_R^i + \sum_{k=i+1}^{N-1} \min(0, m_{k\to R}(1)), \\ \min_{j=0}^{i-1} \left\{ E_R^j + m_{j\to R}(1) + \sum_{k=j+1}^{N-1} \min(0, m_{\mathrm{x}_k\to R}(1)) - \min(0, m_{\mathrm{x}_i\to R}(1)) \right\} \end{pmatrix} \tag{15}$$

Denote

$$\begin{aligned} &\lfloor m \rfloor_{\mathrm{x}_k\to R} = \min(0, m_{k\to R}(1)), \quad k \in [0, N-1] \\ &\lfloor m \rfloor^+_{\mathrm{x}_j\to R} = \textstyle\sum_{k=j}^{N-1} \lfloor m \rfloor_{\mathrm{x}_k\to R}, \quad j \in [0, N-1]; \quad \lfloor m \rfloor^+_{\mathrm{x}_N\to R} = 0. \\ &\lfloor m \rfloor^+_{R\to \mathrm{x}_i} = \min_{j=i+1}^{N-1} (\left\{ E_R^j + m_{\mathrm{x}_j\to R}(1) + \lfloor m \rfloor^+_{\mathrm{x}_{j+1}\to R} \right\}, E_R^N), \quad i \in [0, N-1] \\ &\lfloor m \rfloor^-_{R\to \mathrm{x}_i} = \min_{j=0}^{i-1} \left\{ E_R^j + m_{j\to R}(1) + \lfloor m \rfloor^+_{\mathrm{x}_{j+1}\to R} - \lfloor m \rfloor_{\mathrm{x}_i\to R} \right\}, \quad i \in [0, N-1] \end{aligned}$$

then

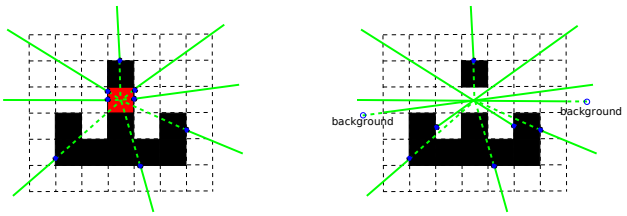
$$\begin{aligned} m_{R\to i}(0) &= \min(\lfloor m \rfloor^+_{R\to \mathrm{x}_i}, \lfloor m \rfloor^-_{R\to \mathrm{x}_i}), & i \in [0, N-1] \\ m_{R\to i}(1) &= \min(E_R^i + \lfloor m \rfloor^+_{\mathrm{x}_{i+1}\to R}, \lfloor m \rfloor^-_{R\to \mathrm{x}_i}), & i \in [0, N-1] \end{aligned} \tag{16}$$

where $\lfloor m \rfloor^+_{R\to \mathrm{x}_i}$ and $\lfloor m \rfloor^-_{R\to \mathrm{x}_i}$ can be computed in a sweep for i taking values from 0 to N.

$$\lfloor m \rfloor^+_{R\to \mathrm{x}_i} = \min(E_R^{i+1} + m_{\mathrm{x}_{i+1}\to R}(1) + \lfloor m \rfloor^+_{\mathrm{x}_{i+2}\to R}, \lfloor m \rfloor^+_{R\to \mathrm{x}_{i+1}}), \quad \lfloor m \rfloor^+_{R\to \mathrm{x}_{N-1}} = E_R^N \tag{17}$$

$$\lfloor m \rfloor^-_{R\to \mathrm{x}_i} = \min(E_R^{i-1} + m_{\mathrm{x}_{i-1}\to R}(1) + \lfloor m \rfloor^+_{\mathrm{x}_i\to R} - \lfloor m \rfloor_{\mathrm{x}_{i-1}\to R}, \lfloor m \rfloor^-_{R\to \mathrm{x}_{i-1}}), \quad \lfloor m \rfloor^-_{R\to \mathrm{x}_0} = +\infty \tag{18}$$

The computational cost for each message for a ray-clique with N random variables has reduced from $O(2^N)$ (the usual implementation) to $O(N)$.



(a) solid voxel case: the 8 camera rays' observations come from either this voxel or occluding voxels (called foreground voxels)

(b) empty voxel case: the 8 camera rays' observations come from either occluding voxels or background voxels

Figure 4. 2D illustration of the voxel's multi-view generation model

cameras can see this voxel without first knowing voxels' occupancy, we collect all the potential observations and make inference based on these N values. As shown in Fig. 4, the pixel value I_R can come from any voxel along the ray, or from background (the region that are out of the volume of interest). We summarize all the cases into two categories: either from the point x_R^* or from foreground / background (foreground means that the object before this voxel, background means that the object after this voxel (if the voxel is transparent). Formally, we have this two mode mixture model: one mode modeling the first case where the statistical rendering equation is used, the second mode modeling the case that it is occluded or transparent:

$$f(I_R) = \lambda f_R(I_R) + (1-\lambda) * f_H(I_R) \tag{19}$$

where $f_R(I_R)$ is the statistical rendering equation (4), $f_H(I_R)$ is the probability density function for the color value of the occluding foreground and background. Here we assume all the voxels share the same H distribution, which is represented with a histogram. Since it is a mixture of Gaussian and H distribution, we refer to it as MOGH.

To estimate voxel color distributions, some general prior knowledge of the distribution of the variance of the color intensity is assumed. Specifically we assume that change of the color variance falls into the Rayleigh distribution. Rayleigh distribution is related to normal distribution through the fact that: The magnitude of a vector, whose components are normal distributed, is Rayleigh distributed. So for the square-root quantities, such as the standard deviation, it is natural to pick up the Rayleigh distribution.

$$f(\sigma;\omega) = \frac{\sigma}{\omega^2}\exp(\frac{-\sigma^2}{2\omega^2}) \tag{20}$$

The Maximum A Posterior (MAP) estimation of mean and variance of the color is formulated as:

$$\begin{aligned}\{\mu^*,\sigma^*\} &= \arg\max_{\mu,\sigma} f(\sigma;\omega)f(X|\mu,\sigma) \\ &= \arg\max_{\mu,\sigma} \tfrac{\sigma}{\omega^2}\exp(\tfrac{-\sigma^2}{2\omega^2}) \\ &\quad \textstyle\prod_{j=1}^{M}(\lambda\frac{1}{\sqrt{2\pi}\sigma}\exp\{\frac{-(x_j-\mu)^2}{2\sigma^2}\} \\ &\quad +(1-\lambda)f_H(x))\end{aligned} \tag{21}$$

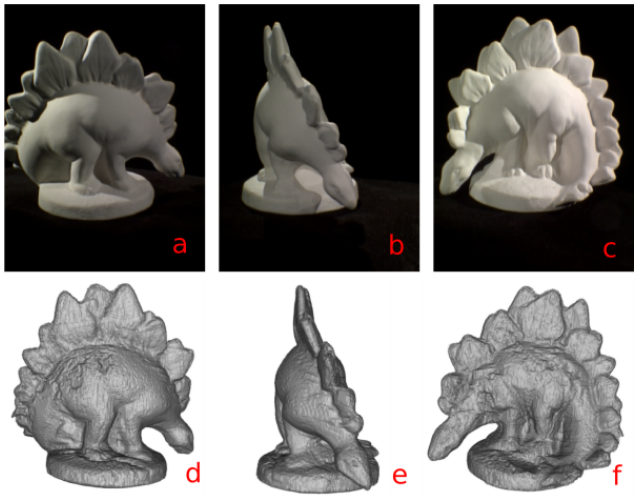


Figure 5. Experiment on dinoSparseRing data-set: (a, b, c) 3 out of 16 input images; (d, e, f) reconstructed surface mesh model

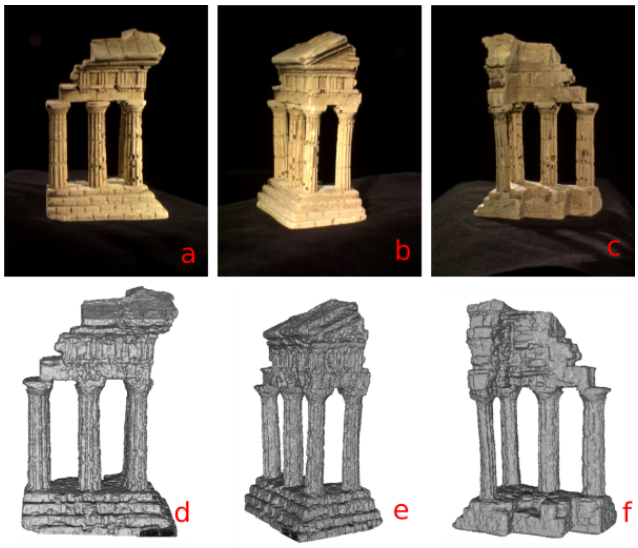


Figure 6. Experiment on templeSparseRing data-set: (a, b, c) 3 out of 16 input images; (d, e, f) reconstructed surface mesh model

where X is the set of observations in different images for each voxel. The MAP solution can be computed through the Expectation-Maximization (EM) algorithm.

4. Experiments

Here we first show some experiments on the middlebury standard data-set [17], which is publicly available at [2]. In all the experiments, the parameters are set as $\alpha_u = 6$, $\alpha_p = 8, \omega = \{4,4,4\}$. These parameters are selected manually (ideally should be learned over a large data-set) and are kept the same for all experiments. Better parameters should be selected through extensive search of the parameter space or through learning. For the dinoSparseRing data-

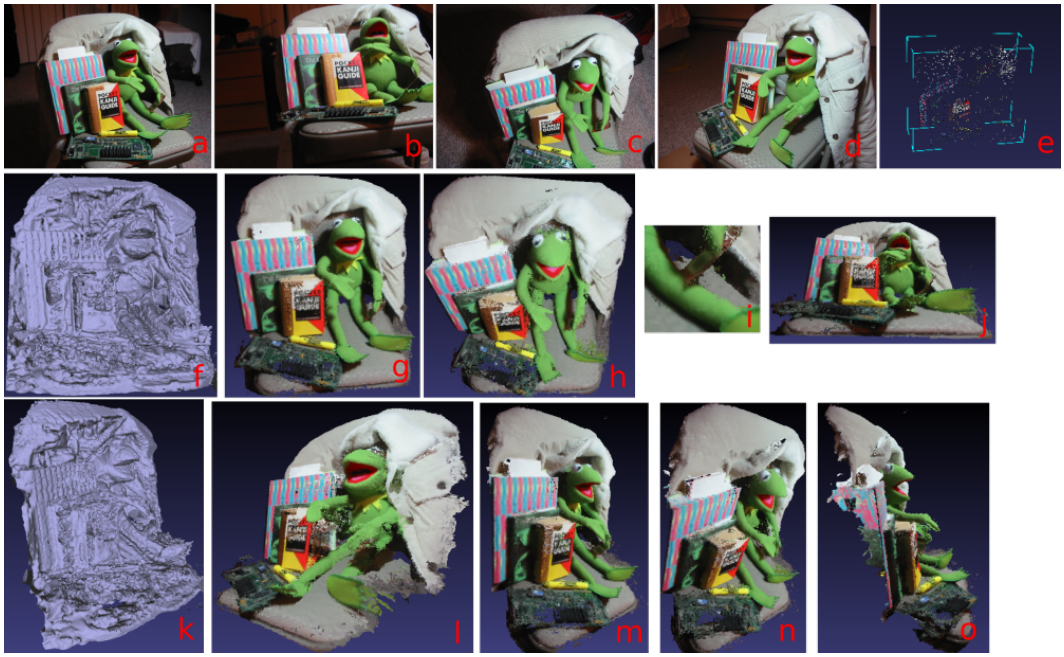


Figure 7. Experiment on kermit data-set: (a,b,c,d) 4 out of 11 input images; (e) point cloud generated by bundler software; (f,k) rendered mesh model in two different views; (g–j, l–o) rendered images in different novel views;

set, the volume size is 283x238x238. And it takes 47 minutes on a 3.0 GHz CPU with 100 iterations. Fig. 5 shows 3 out of 16 input images in the first row, and the reconstructed surface shape in the second row. From which we can see that the surface is reconstructed fairly good. For the templeSparseRing data-set, the volume size is 376x176x241, and it takes 52 minutes. Fig. 6 shows the reconstruction results for the templeSparseRing data-set. Quantitatively it is better than other volumetric methods, and competitive to the state-of-the-art feature based methods, e.g. [5]. Quantitative comparison with other methods is posted on the evaluation website `http://vision.middlebury.edu/mview/eval/`.

Next we test the algorithm on another data-set, kermit, which is provided by the bundler structure from motion software [1]. Fig. 7(a-d) shows 4 out of 11 input images. The cameras are calibrated automatically from these 11 images with bundler software. Fig. 7(e) shows the point cloud that bundler generates. We use the bounding-box of these point cloud as the volume for our reconstruction. In the experiments, the resolution for this model is 360x320x140. The running time is 34 minutes on a 3.0 GHz CPU. Figs. 7(f-o) show our reconstruction results in different views, rendered mesh, and rendered images. There are several interesting parts worth paying attention to. First the shape of the texture-less coat is reconstructed fairly well, and the flat region of the book is also good (see image(o)). Also pay attention to image (i), which shows the reconstruction of the thin, flat price tag.

Figure 8. Experiment on elephant data-set: (a) one of the 22 input images; (b, c) rendering of the reconstructed model

Next we show another two experimental results to demonstrate the system's capability to reconstruct complex objects in cluttered background. These datasets were captured casually with a consumer camera, and all the images are calibrated with the bundler software. Fig. 8 shows the reconstruction of a wood-carved elephant, with the holes reconstructed nicely (see Fig. 8(c).) Fig. 9 shows the reconstruction of a out-door shiny statue. The result is faily good, considering the relfective nature aof the statue surface and the cluttered background of the scene. All the above reconstructed are performed completely automatically.

5. Discussion

This paper has proposed a novel approach for multiview stereo based on a MRF model with ray-cliques modeling the image formation process. With a multi-view image generation probabilistic mixture model, and optimized belief propagation, the MRF model can be inferred efficiently. Primary experiments demonstrate the quality, speed and gen-

Figure 9. Experiment on statue data-set: (a) one of the 29 input images; (b, c, d) rendering of the reconstructed model

eral applicability of the proposed approach. By directly modeling the image generation process, the framework is general to tackle multi-view reconstruction problems under general conditions. To get full understanding and estimation of the power and limitation of this approach, more study and experiments are under way. In this paper, we have assumed simplified statistical rendering model. Extending it to more detailed statistical rendering models, e.g., putting BRDF into the model, would give us better accuracy and reveal more information from images. In the paper, a grid of voxels are used to represent a volume space. Considering the fact that most of the space is either totally empty or solid, hierarchical sparse data structures like octree can be deployed to save both memory and computation. This paper solves the hard occlusion problem in multi-view stereo with a principled way, which could have implications beyond the problem domains of multiview stereo, where occlusion inference is required. We expect to see more developments of this approach in other applications.

Acknowledgement. This work was supported by the NSF IIS Program Grant NO 0808718.

References

[1] http://phototour.cs.washington.edu/bundler.

[2] http://vision.middlebury.edu/mview.

[3] M. Agrawal and L. S. Davis. A probabilistic framework for surface reconstruction from multiple images. In *CVPR*, volume 2, pages 470–476, 2001.

[4] A. Broadhurst, T. Drummond, and R. Cipolla. A probabilistic framework for space carving. In *ICCV*, 2001.

[5] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. In *CVPR*, pages 1–8, 2007.

[6] P. Gargallo, E. Prados, and P. Sturm. Minimizing the reprojection error in surface reconstruction from images. In *ICCV*, pages 1–8, 2007.

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

[8] J. T. Kajiya. The rendering equation. In *SIGGRAPH*, pages 143 – 149, 1986.

[9] J. H. Kim and J. Pearl. A computational model for combined causal and diagnostic reasoning in inference systems. In *IJCAI*, pages 190–193, 1983.

[10] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2), February 2004.

[11] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

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

[13] S. Liu, K. Kang, J.-P. Tarel, and D. Cooper. Free-form object reconstruction from silhouettes, occluding edges and texture edges: A unified and robust operator based on duality. *PAMI*, 30(1):131–146, January 2008.

[14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, second edition, 1988.

[15] T. Pollard and J. Mundy. Change detection in a 3-d world. In *CVPR*, pages 1–6, 2007.

[16] B. Potetz. Efficient belief propagation for vision using linear constraint nodes. In *CVPR*, pages 1–8, 2005.

[17] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, volume 1, pages 519–526, 2006.

[18] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *ICCV*, volume 2, pages 900–906, Nice, France, October 2003.

[19] G. Vogiatzis, C. H. Esteban, P. H. Torr, and R. Cipolla. Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *PAMI*, 29(12):2241–2246, 2007.

[20] H. Vu, R. Keriven, P. Labatut, and J.-P. Pons. Towards high-resolution large-scale multi-view stereo. In *CVPR*, 2009.

[21] Y. Wei and L. Quan. Asymmetrical occlusion handling using graph cut for multi-view stereo. In *CVPR*, volume 2, pages 902–909, 2005.