A Continuous Probabilistic Scene Model for Aerial Imagery

by

Daniel E. Crispell

B. S., Northeastern University, 2003

Sc. M., Brown University, 2005

Submitted in partial fulfillment of the requirements

for the Degree of Doctor of Philosophy in the

Division of Engineering at Brown University

Providence, Rhode Island

May, 2010

This dissertation by Daniel E. Crispell is accepted in its present form by the Division of Engineering as satisfying the dissertation requirement for the degree of Doctor of Philosophy.

Date _____ _____

Gabriel Taubin, Director

Recommended to the Graduate Council

Date _____ _____

Joseph L. Mundy, Reader

Date _____ _____

Benjamin Kimia, Reader

Approved by the Graduate Council

Date _____ _____

Sheila Bonde
Dean of the Graduate School

Curriculum Vitae

Daniel Elting Crispell was born in Port Jervis, New York on June 28, 1980. He graduated from Wallkill Senior High School in Wallkill, New York in June of 1998 and began studies at Northeastern University in Boston, Massachusetts the following Fall. He graduated from Northeastern in 2003 Summa Cum Laude with a Bachelor of Science degree in Computer Engineering. In the Fall of 2003 Daniel began his graduate studies at Brown University in Providence, Rhode Island as part of the Laboratory for Engineering Man/Machine Systems (LEMS). During his time at Brown he has been both a teaching assistant and a research assistant working in the field of computer vision. He has presented peer-reviewed work on a variety of topics including camera networks, multi-view object reconstruction, and video registration at several conferences and workshops.

This thesis is dedicated to Donald and Jean Crispell, whose love, support, and guidance throughout my academic career has led to this culminating work.

Acknowledgements

I would like to thank my family for providing the best support structure any person could ever ask for, as well as my friends from Wallkill, Northeastern, Brown, and everywhere in between for making my academic experience such an enjoyable one. I would also like to thank Lindsay Slivka for all of her support and encouragement over the past two and a half years.

I would like to thank Professors Gabriel Taubin and Joseph Mundy, both of whom provided daily support and always had my best interests at heart. I have been extremely fortunate to have had the opportunity to learn from and work with both of them.

Above all else, I acknowledge the countless blessings of Almighty God, through whom all things are possible.

Contents

List of Tables

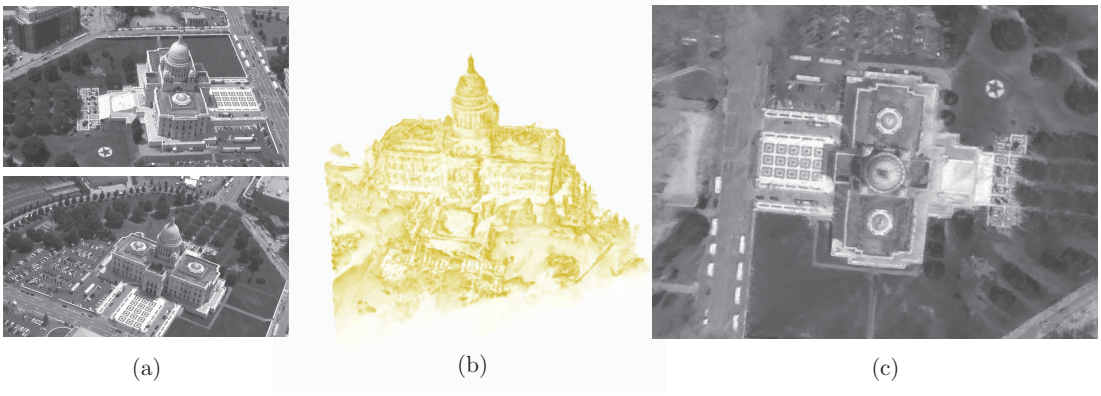
List of Figures

Chapter 1

Introduction

Aerial imagery is a vital source of information for a large number of civil and defense applications, including city planning and land management, surveillance, and mission planning and rehearsal. While the captured imagery itself can provide useful information, applications often require that the data be transformed in some way (e.g. registered to a common coordinate system) or that information be extracted from it for it to be maximally useful. One such application is that of 3-d localization of image points: Given a selected point in a (2-d) image, it is often useful to determine the 3-d location of the corresponding world point which produced the image intensity. Localization in 3-d requires the inference and representation of potentially uncertain scene geometry based on the available image data and is made possible using the reconstruction methods and probabilistic 3-d model presented in this thesis. A second application of aerial imagery is that of generating synthetic images from viewpoints which have not been directly observed based on available imagery from those that have. This process is known as novel view generation and is also made possible using the methods and model presented in this thesis. Although there do exist novel view generation methods which do not use explicit 3-d models (known collectively as image-based rendering methods and discussed in Section 2.2), they are not capable of producing accurate renderings from viewpoints far from any of the available input images. In addition to 3-d point localization and novel view generation, 3-d models are also valuable as tools enabling measurement of objects and distances in three dimensions as well as a way to aggregate information from multiple images in applications such as change detection,

which has been shown to have better performance when using 3-d models [53].



(a) (b) (c)

Figure 1.1: In the proposed system, input images (two of which are shown in (a)) are used to generate a probabilistic model (b) which can be used to locate 3-d points and produce expected images from novel viewpoints (c).

Three-dimensional information can be inferred from imagery, but manual effort is often needed to ensure the accuracy and completeness of the models due to uncertainties and ambiguities present in the image data. It is often the case that exact position is unknowable given the image data alone due to ambiguities and intensity variation caused by unmodeled phenomenon. The precision with which 3-d points can be localized is also inherently limited by the resolution of the images in which they appear. An automatic reconstruction algorithm may handle these uncertainties and ambiguities by either resolving them based on implicit or explicit prior assumptions about the scene geometry or by fully representing them inherently in the reconstructed model. When dealing with real-world scenes, however, prior assumptions about surface geometry are often violated, leading to reconstruction errors. Often times a "smoothness" prior is used which in the case of uncertainty favors the surface with the lowest possible curvature, leading to errors near sharp corners and rough texture. Representations which allow for "fuzzy", or probabilistic, representations of surfaces are able to represent surfaces with uncertain location without making such prior assumptions and are therefore able to avoid potential errors of this type. In order to be useful for novel viewpoint generation, the model must also be capable of providing rich information about the represented scene, including visibility and occlusion information for

points in the scene. Finally, the model must lend itself to practical and efficient representation, enabling large-scale scenes to be represented with fine detail. This is particularly important when modeling and generating views of large and complex outdoor scenes.

Currently, there exists a variety of manual, semi-automatic, and fully automatic 3-d reconstruction algorithms used for the purpose of site modeling from aerial imagery. These methods construct models which are necessarily compact in nature so that they are able to scale to large scenes. (Detailed discussion is presented in Section 2.3.) Manual and semi-automatic method are able to produce accurate reconstructions, but require tedious work on the part of the operator. Automatic site-modeling systems are prone to error and not capable of producing probabilistic representations which can account for the error sources. There do exist, however, several probabilistic reconstruction algorithms in the computer vision literature. These methods are capable of automatically producing volumetric scene models which accurately represent uncertainties and (in some cases) ambiguities present in the image data. These methods are discussed in detail in Section 2.4. Despite their ability to construct probabilistic models these methods do not scale well to large and complex scenes, primarily due to their dependence on large three-dimensional voxel arrays.

The work presented in this thesis generalizes the previous probabilistic models in such a way that multiple orders of magnitude savings in storage are possible, making precise representation and novel view generation of large-scale outdoor scenes possible. Specifically, the inherent dependence on a discrete array of uniformly sized voxels is removed through the derivation of a continuous probabilistic model. The continuous model allows for implementations which non-uniformly sample the volume, providing high resolution detail where needed, and coarser resolutions in areas containing little information. In addition, multiple reconstruction algorithms are presented to accommodate differing modes of operation in which aerial imagery may be captured and used. The proposed model combined with the reconstruction and novel view generation algorithms comprise the first system capable of automatically generating photo-realistic renderings of large and complex scenes from arbitrary viewpoints based on aerial image data alone.

1.1 Probabilistic Scene Models

The utility of a probabilistic model stems primarily from the fact that computing exact 3-d structure based on 2-d images is in general an ill-posed problem. Bhotika et al. [5] characterized the sources of difficulty as belonging to one of two classes: scene *ambiguity* and scene *uncertainty*. Scene ambiguity exists due to the existence of multiple possible photo-consistent scenes and is a problem even in the absence of any sensor noise or violations of appearance assumptions (Figure 1.3). In the absence of prior information regarding the scene structure, there is no reason to prefer one possible reconstruction over another. The term "scene uncertainty", on the other hand, is used to describe sources of error stemming from unmodeled phenomenon including sensor noise, violations of simplified appearance models, (e.g. Lambertian reflectance), and calibration errors (Figure 1.2). The presence of scene uncertainty typically makes reconstruction of a perfectly photo-consistent scene impossible. A probabilistic model allows the scene ambiguity and uncertainty to be explicitly represented, which in turn allows the assignment of confidence values to visibility calculations, expected images, and other data extracted from the model. A probabilistic model can also be used to determine which areas of the scene require further data collection due to low confidence.

1.1.1 Scene Uncertainty

The presence of scene uncertainty means that, even with a perfectly accurate reconstruction, we cannot expect perfect photo-consistency due to various sources of noise. This fact is most often dealt with by modeling the appearance of a surface point in a particular image as a random sample drawn from a distribution (usually Gaussian) associated with the point. In the space carving method of Kutulakos and Seitz [41], a global threshold on the variance of the distribution is specified. A voxel whose projections have a sample variance greater than this threshold is considered non photo-consistent and carved from the model as a result. As Broadhurst et al. [7] point out, this approach can cause holes to be incorrectly carved in the model. They improve on the space carving method by assigning a probability to each voxel based on the likelihood that the image samples originated from a distribution with

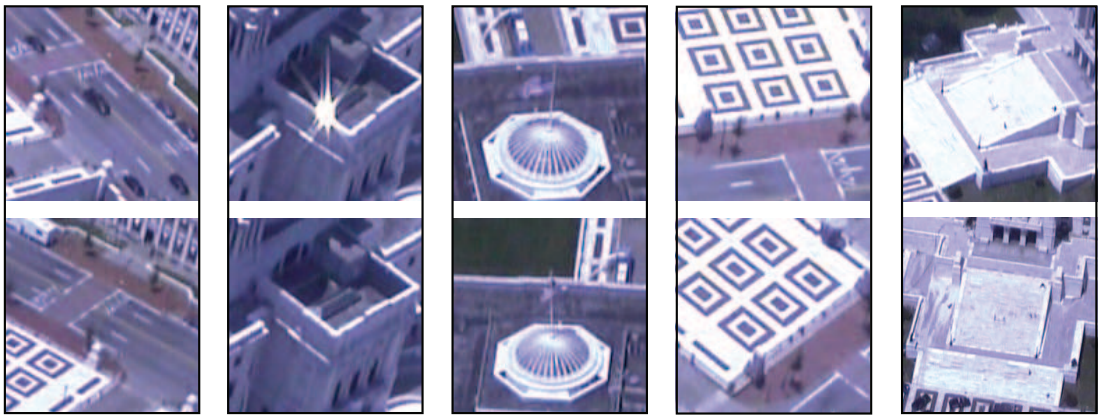


Figure 1.2: Various causes of scene uncertainty in the "Capitol" sequence. (a): Transient foreground objects such as cars (b): Specularities and other non-Lambertian effects (c): Movement of scene objects (caused by wind, for example) (d): Regions with large intensity gradients are particularly sensitive to small errors in camera calibration (e): Sensor noise and non-linear effects can cause appearance variation.

small variance rather than making a binary decision. Similarly, Bhotika et al. [5] carve each voxel with a probability based on the variance of the samples in each of a large number of runs. The final voxel probability is computed as the probability that the voxel exists in a given run. Like these methods, the algorithms presented in this thesis are based on the basic assumption that, all else being equal, voxels whose image projections have a small variance across many views are more likely to be photo-consistent than those that have a larger variance.

1.1.2 Scene Ambiguity

In the absence of any prior knowledge of the scene, the existence of scene ambiguity forces one to concede the impossibility of reconstructing the scene exactly with total confidence. This occurs when multiple reconstructions are possible which are photo-consistent with all available image data. There are varying approaches to dealing with this reality. The simplest approach is to consider any single photo-consistent reconstruction a valid solution to the problem. The obvious drawback of this approach is that nothing can be said about the certainty of the true scene matching the reconstruction or the relationship between the

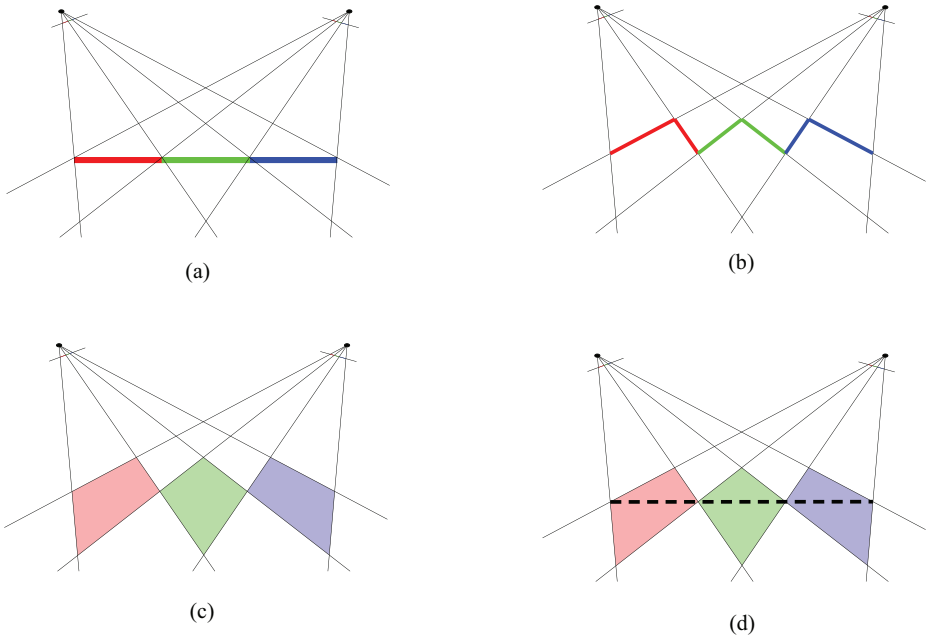


Figure 1.3: Scene Ambiguity. (a): Two cameras view a surface with three uniformly colored regions. (b) The photo hull. (c) A probabilistic model: The true location of the surface may be anywhere within the shaded regions. (d): If a prior on the shape of the surface is available, a more accurate reconstruction may be possible.

reconstruction and the true scene, only that it is *possible* that the reconstruction is accurate. If certain a priori information about the scene is available, the information may be used to choose the photo-consistent reconstruction which best agrees with the prior. This is the approach taken by most stereo algorithms, which use the assumption of smoothly varying depth as a regularizing constraint (e.g. [25, 10, 76, 70]). The reconstruction algorithms presented in this thesis do not utilize any such priors in order to keep them as generally applicable as possible.

Another approach is to define a particular member of the set of photo-consistent reconstructions as "special", and aim to recover that member. This is the approach taken by Kutulakos and Seitz [41] and Bhotika et al. [5]. Kutulakos and Seitz define the *photo hull* as the tightest possible bound on the true scene geometry, i.e. the maximal photo-consistent reconstruction. The photo hull is guaranteed to contain all possible photo-consistent reconstructions, and to be itself a photo-consistent reconstruction. They show that under ideal conditions the photo hull can be recovered exactly, while Bhotika et al. present a

stochastic algorithm for probabilistic recovery of the photo hull in the presence of noise. The photo hull provides a maximal bound on the true scene geometry, but does not contain any information about the distribution of possible scene surfaces within the hull.

A third approach is to explicitly and probabilistically represent the full range of possible scene reconstructions. Broadhurst et al. [7] aim to reconstruct such a representation, as well as Pollard and Mundy [53] for the purpose of change detection. (These approaches are discussed in detail in Section 2.4.) Because a model which fully represents both scene ambiguities and uncertainties is desired, the model and algorithms presented in this thesis are also based on this approach.

1.1.3 Representation

One of the most important aspects of any reconstruction algorithm is the way in which the estimated 3-d geometry is represented. Ideally, a probabilistic reconstruction algorithm would produce a probability distribution over the space of all possible surfaces. In practice, however, the dimensionality of the space is too large to feasibly represent. Previous probabilistic reconstruction algorithms [57, 7, 5, 53] have relied on discrete models which instead represent the probabilities of individual "volume elements", or *voxels*, belonging to a particular set of voxels S whose precise definition varies slightly between algorithms; the set S may represent the volume occupied by the objects in the scene or the boundary of the volume only. In either case, the space of possible surfaces is effectively marginalized for each voxel by ignoring the interdependence of voxel probabilities. Each voxel v in the model stores a probability $P(\mathcal{X}_v)$, where the proposition $\mathcal{X}_v$ represents "$v \in S$".

One possible definition of S is the set of "surface voxels". This definition is based on the assumption that the surface is composed of a set of discrete, opaque voxels. The model of Pollard and Mundy [53] is based on this definition of voxel probability. Although not formulated as a probabilistic model, the opacity values associated with the voxels by Debonet et al. [57] can also be treated as surface probabilities. The second possible definition of S is the set of "occupied" voxels, the union of the set of surface voxels and the set of interior voxels not visible from any viewpoint. This is the definition used by Broadhurst et al. [7]. A third possible definition of S is used by Bhotika et al. [5]: the set of voxels

comprising the photo hull (Section 1.1.2). One potential problem associated with methods which define S to be the set of surface voxels is that of distinguishing between voxels interior to surfaces (which may or may not be occupied) and visible, empty voxels in open space. For example, the probability associated with the interior voxels in Pollard and Mundy's model are not well defined since a voxel is no longer updated once it is occluded from all viewpoints. Although the model presented in this thesis does represent "surface-like" geometry, a method is presented for determining interior and exterior regions in Section 5.3.

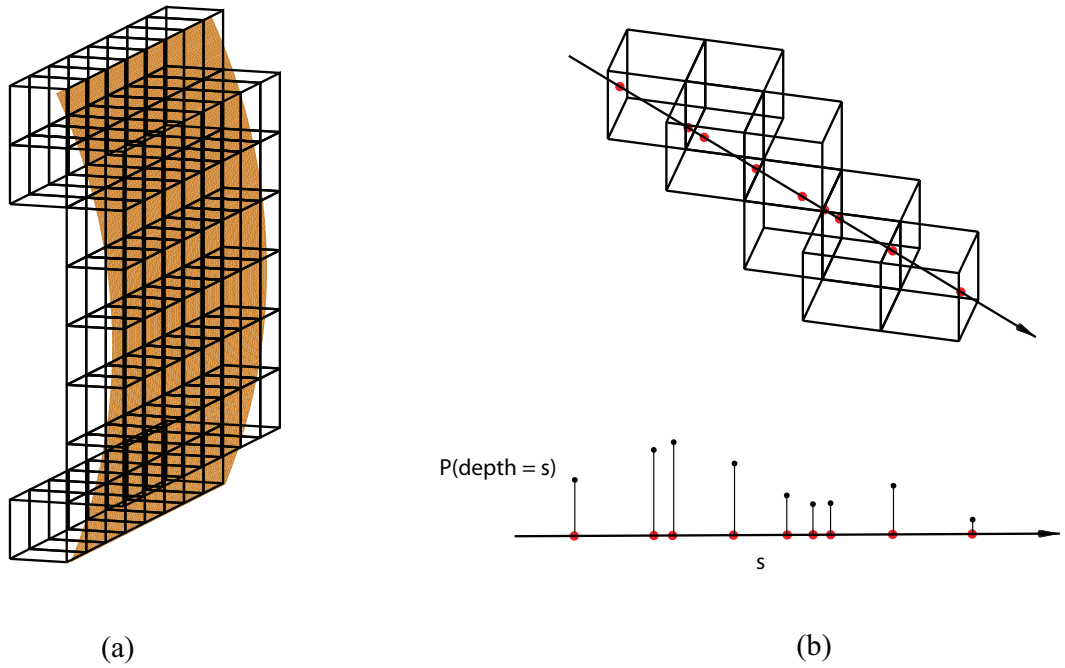


Figure 1.4: Space discretization using voxels. (a) Discretization in scene-space: A best-case reconstruction of the surface. (b) Discretization in "camera space": The depth of the scene along the camera ray is confined to a discrete probability distribution. Possible values correspond to point of intersection with voxel boundaries (marked in red).

Regardless of the definition of the set S, there is no notion of "partial" voxels existing in these discrete models. This means that the boundaries between empty space and objects in the scene (i.e. surfaces) may only exist at voxel boundaries. Assuming a best-case reconstruction (surface location known with infinite precision), the maximum distance between the true surface and a voxel boundary is bounded by $\frac{\ell}{2}$, where ℓ is the side length of a voxel. In general, however, surfaces cannot be localized with infinite precision for the

reasons discussed previously. Instead, there is a some continuous range of possible surface locations with an associated continuous probability distribution. Using traditional voxel models, however, the continuous distribution is approximated by a set of discrete locations which correspond to voxel boundaries.

In addition to representing the 3-d geometry itself, it is often useful to compute depth map images where each pixel value represents the depth of the first world surface intersected by its corresponding camera ray. Because the scene geometry is uncertain, these depth values are represented by a probability distribution in a probabilistic model. Using the discrete voxel models the distribution is also discrete since there are a finite number of possible depth values corresponding to voxel boundaries along the camera ray. This type of model is analogous to stereo methods which assign per-pixel depths based on a discrete set of possible values. The maximum distance between two consecutive possible depth values is $\sqrt{3}\ell$ and occurs when the ray enters and exits a voxel from opposite corners. Although the continuous nature of the surface location and depth distribution are not reflected in the model, it can be made arbitrarily precise by increasing the resolution (decreasing the voxel size) of the model to satisfy a given precision constraint. Unfortunately, the cost of such a representation in terms of both storage and computational requirements is prohibitive for sufficiently complex scenes; unlike deterministic space carving methods [41] where only surface voxels are explicitly represented, a full volumetric representation must be maintained for the probabilistic case. This means that $\left(\frac{L}{\ell}\right)^3$ voxels are required, where $\frac{L}{\ell}$ is the linear resolution desired.

In general, information regarding the location of objects and surfaces in the scene is not known a priori, and so a regular sampling of the volume is a reasonable strategy. As information is learned about the scene, however, a more intelligent sampling of the volume should be possible in order to offset the cubic storage cost. Regions of low surface probability can be sampled at a lower rate to increase efficiency, and regions of high surface probability can be sampled a higher rate to increase effective resolution and accuracy. In voxel-based models, a lower sampling rate translates to larger voxels. Unfortunately, handling variable voxel sizes correctly presents serious problems for models based on discrete probabilities of the form $P(\mathcal{X}_v)$. As voxel size grows, the discrete model becomes a poor representation of

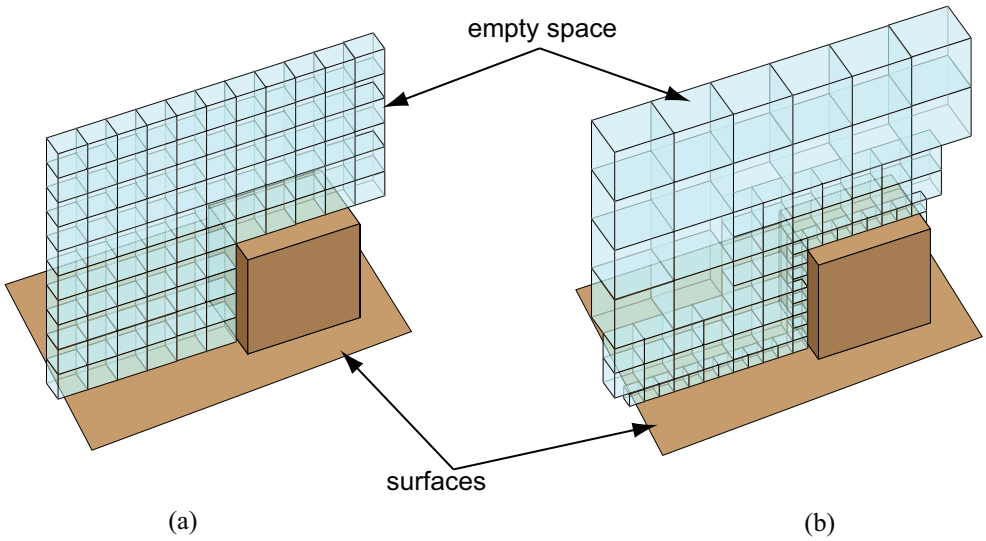


Figure 1.5: Variable voxel size: (a) A regular discretization of space. (b) Variable voxel size allows a finer resolution near surfaces, and a coarser resolution in empty space where there is little information.

the underlying scene knowledge even in the case of low, slowly varying probability regions. As an example, consider a large region of space which is known, based on the data, to have a low (but non-zero) probability of containing a world surface. This type of situation commonly arises when large homogeneous surfaces are imaged. Based on the low, smoothly varying probability, the region would seem to be an ideal candidate for data compression in the model. Modeling the region with a single (for example) large voxel, however, creates a large region with zero probability of containing a world surface (the interior of the voxel), and a single potential location for the surface (the voxel boundary) with a finite probability based on the probability $P(\mathcal{X}_v)$ assigned to the voxel. Data compression is achieved at the cost of an accurate representation of the scene knowledge. An alternate way to view the problem is from the point of view of a camera viewing the scene. Treating the depth of occlusion along a camera ray as a random variable, the depth probability distribution for a ray passing through the unknown region is continuous and close to uniform. Modeling the region with a large voxel size, however, simply moves the possible occlusion depth values further apart (Figure 1.6).

Rather than representing space using discrete elements with probability of belonging

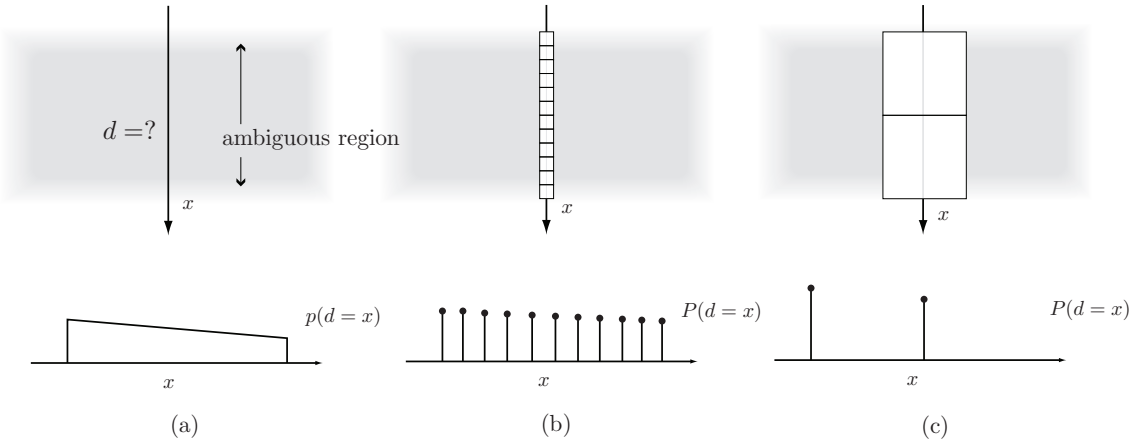


Figure 1.6: The problem with discrete voxel models: (a) a ray passes through a uniformly ambiguous region, resulting in a continuous probability distribution of the depth d of the surface. (b) The distribution is approximated by a finite set of depths with associated discrete probabilities when the region is represented by a series of small voxels. (c) The depth distribution approximation becomes less accurate as the voxel size grows.

to a particular set, the model presented in this thesis represents the probability that a ray passing through a point will be occluded by that point as a density function. Instead of a discrete set of possible locations of the depth of the scene from the camera center measured along a particular camera ray, a fully continuous probability density function describing the depth may be computed. Details of the model are discussed in Chapter 3. The key advantage of the proposed continuous model over the traditional discrete voxel models is the ability to non-uniformly sample the scene space while maintaining accurate representations of the continuously varying occlusion pdfs. This enables implementations to efficiently represent large and complex scenes with fine detail, making it a viable solution for scene modeling from aerial imagery.

1.2 Outline

The remainder of the thesis is laid out as follows. In Chapter 2, the state of the art in 3-d reconstruction from image data in general and aerial imagery specifically is discussed. Existing probabilistic methods are discussed in detail, and the presented approach is compared and contrasted. In Chapter 3, the continuous probabilistic scene model is presented in detail. In chapter 4, an implementation of the scene model is presented which uses a

piecewise-constant assumption and is represented by an octree. An alternate piecewise-linear implementation is also briefly discussed. In Chapter 5, algorithms for reconstructing the probabilistic model are presented. The algorithms are first presented in their general continuous form, and then adapted for use with the piecewise-constant implementation. Both online and offline reconstruction algorithms are presented, as well as a method for recovering a polygonal surface estimate from the probabilistic model. Chapters 6 and 7 discuss the results of the reconstruction algorithms on test and aerial video, respectively. The test data is used in Chapter 6 in order to provide a basis for comparison between the reconstructed 3-d models generated using the presented approach and existing techniques. The 3-d localization and synthetic viewpoint generation methods are evaluated in Chapter 7 using aerial and satellite imagery to showcase the model's ability to handle large outdoor scenes with high levels of detail. In Chapter 8, a method for refining camera calibration based on visual servoing techniques and expected image generation is discussed. Finally, the thesis is concluded in Chapter 9.

1.3 Contributions

The following contributions are provided in this thesis:

- A continuous probabilistic scene model is presented which generalizes and improves upon existing discrete models by enabling higher accuracy and more efficient sampling techniques.
- An implementation of the model based on a piecewise-constant assumption and octree for data storage is described.
- Methods for reconstructing probabilistic scene geometry and appearance based on image data are presented for the general continuous model as well as simplified versions tailored to the piecewise-constant case. Both online and offline algorithms are presented.
- A method for 3-d point localization using the probabilistic model is presented and evaluated.

- A method for the generation of synthetic images from novel viewpoints using using the probabilistic model is presented and evaluated.
- A method for constructing a surface mesh representing the estimated positions of surfaces present in the scene based on the probabilistic model is described for the purposes of evaluating the accuracy of the reconstruction algorithms using standard test data.
- A method for refining camera estimates which uses all available image data is described and evaluated.

1.4 Notation

The following notation is used as consistently as possible throughout this thesis. Vector quantities are denoted by boldface variable names, and scalars by regular font. A capital "P" is used to represent discrete probabilities, and lowercase for continuous probability density values. Propositions to which probabilities are assigned are denoted by capital letters in calligraphic fonts (e.g. $\mathcal{X}$, $\mathcal{D}$, $\mathcal{M}$).

Chapter 2

Related Work

The research presented in this thesis is closely related to work in several other fields. The goal of this chapter is to provide an overview of currently existing methods for constructing and visualizing computerized 3-d models based on image data. Site modeling from aerial imagery is of particular interest, although general 3-d reconstruction and visualization is also discussed. In Section 2.1, a brief discussion of general 3-d reconstruction methods existing in the computer vision literature is presented, followed by the presentation of the related field of image-based rendering in Section 2.2. In Section 2.3, previous work and the current state of the art in site modeling will be discussed, including manual and automatic methods. Previous reconstruction methods involving probabilistic models are discussed in detail in Section 2.4, and the chapter is concluded in Section 2.4.5.

2.1 General 3-d Multi-view Reconstruction

3-d reconstruction from images is one of the fundamental problems of computer vision and the basic principles behind it are discussed in many texts, including Hartley and Zisserman [33] and Ma et al. [48]. Reconstruction methods vary both in the algorithms used and the type of output produced. While traditional dense two-view stereo methods generally produce $2\frac{1}{2}$-d depth maps, multi-view methods generally produce output in one of three forms: 3-d point clouds based on feature correspondences, polygonal meshes, or volumetric models.

2.1.1 2.5-d Methods

Traditional stereo reconstruction methods take as input two (calibrated) images, and produce a disparity or depth map as output. Depth values are calculated by determining the distance between points in the first image and their corresponding match in the second. Due to the epipolar constraint [33], matching can be accomplished with a 1-d search. If a smoothness and ordering constraints are also enforced, depth values can be efficiently computed for each pixel using a dynamic programming scheme [3]. A Comprehensive review of the stereo reconstruction literature as of 2002 is given by Scharstein and Szeliski [59]. While highly accurate results are possible [70, 75], the reconstruction results are limited to functions of the form $f(x, y)$ and cannot completely represent general 3-d scenes.

Figure 2.1: Stereo reconstruction input (two left-most images) and resulting depth map (right). Results and image taken from Wang and Zheng [70]

2.1.2 Point-Based Methods

When reconstructing with a number of input images greater than two, it is reasonable to produce a fully 3-d representation. One alternative is a discrete set of elements with no inherent relationship to each other. The elements may be oriented or non-oriented points, or small surface patches. Perhaps the most straightforward method of extending traditional stereo to larger numbers of views is to perform traditional two-view stereo between image pairs, convert the depth map values to 3-d point clouds, and merge the sets of estimated points in 3-d. This is the approach taken by Bradley et al. [6], who also fit a triangular mesh to the resulting filtered point cloud as an additional step. Other multi-view methods explicitly match features across multiple images in 2-d, and then triangulate the 3-d position using all correspondences simultaneously.

Furukawa and Ponce [25] match small image patches across multiple frames, expanding and filtering the set of matches using visibility constraints as the algorithm progresses. A set of discrete small planar patches is output as a result. Habbecke and Kobbelt [31] have developed another surface-element based reconstruction algorithm; theirs is able to produce a dense set of circular disks with a smoothness assumption on the scene geometry. Many multi-view methods are capable of computing 3-d point locations as well as camera calibration information simultaneously using the constraints imposed by feature matches across multiple images (so called "structure and motion"). One example of a such a method is presented by Pollefeyes et al. [55], who use tracked Harris corner [32] features to establish correspondences across frames of a video sequence. Brown and Lowe [9] and Snavely et al. [65] use SIFT features [47] for the same purpose, with considerable success: Snavley et al. have shown their system to successfully calibrate data sets consisting of hundreds of images taken from the internet. The output of feature-based matching methods (at least in an initial phase) is a discrete and sparse set of 3-d elements. While high quality point-based visualization is possible in general [40], the reconstruction results of these methods are not directly useful for this purpose since some regions (e.g. those with homogeneous appearance) will be void of features and thus also void of reconstructed points. It is possible to estimate a full surface mesh based on the reconstructed features [28, 25], but doing so requires imposing regularizing constraints and hallucinating data to fill in "holes" corresponding to featureless regions. Methods based on dense matching techniques avoid the hole-filling problem, but are dependent on smoothness and ordering assumptions to perform the matching. The methods presented in this thesis are not dependent on any assumptions regarding the 3-d scene geometry, yet produce a dense model suitable for photorealistic rendering.

2.1.3 Surface-Based Methods

Another class of systems stores 3-d information in the form of a polygonal mesh whose properties are optimized based on the image data. Hernández and Schmitt [35] initialize a surface mesh based on the visual hull [42, 23] (which requires foreground segmentation) and optimize the geometry based on silhouette and photo-consistency information. Other methods [2, 17] based on surface optimization have also been proposed. Aside from requiring

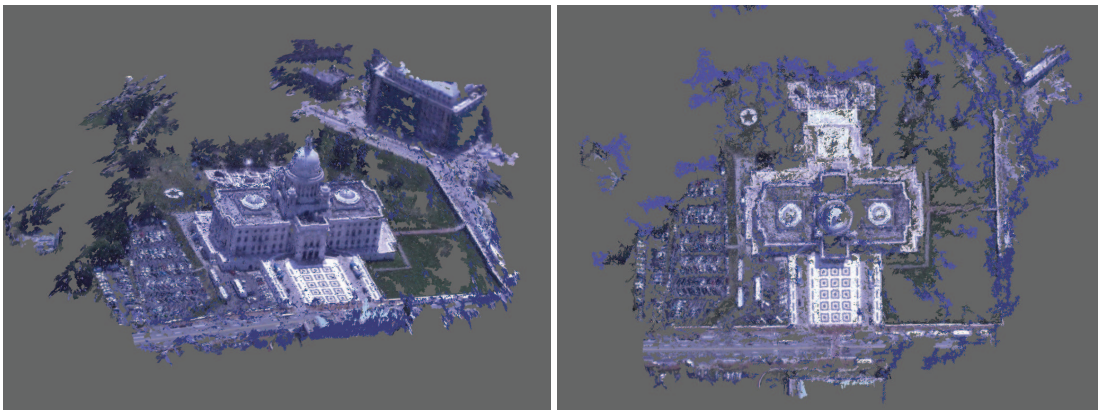


Figure 2.2: Two views of the "capitol" scene reconstructed using the patch-based multiview stereo method of Furukawa and Ponce [25]. Matching is ambiguous in large homogeneous regions such as the grassy area, and so no points are reconstructed there.

that the scene to be reconstructed can be segmented from the background in order to produce an initial estimate, the utilization of a mesh representation requires knowledge of the scene topology, which may be arbitrarily complex and not fully recoverable from the visual hull alone. Zaharescu et al. [79] propose a method for robustly handling topology changes during optimization, but still require an initialized mesh which is based on the visual hull of the segmented foreground object. Because the methods presented in this thesis are based on a volumetric representation, they are able to handle scenes with arbitrarily complex topology. In addition, no segmentation or initial estimate is needed for the presented reconstruction algorithms to operate. Vogiatzis et al. [69] use graph cuts in order to compute an optimal surface which segments the "exterior" and "interior" of the modeled object. They use a volumetric representation of the cost function which allows them to handle topologically complex objects, but do not model occlusions which are instead treated as outliers in the photo-consistency function. The reconstruction and visualization algorithms presented in this thesis fully model occlusions in 3-d, allowing the reconstruction of complex scenes with many objects.

2.1.4 Volumetric Methods

A third class of systems represent the 3-d scene to be reconstructed as a set of voxels, which are classified as either being occupied or empty [41] based on the visual hull and photometric consistency. Volumetric methods are capable of producing a complete representation of the modeled surfaces (as mesh-based reconstructions do), yet (like point-based reconstructions) make no assumptions about scene topology and do not in general require foreground segmentation or an initial estimate. One disadvantage of volumetric reconstruction methods is that they are capable of producing results only as accurate as the resolution of the fixed-size grid on which the voxels are defined, and large grid sizes can potentially become computationally expensive and infeasible to store in memory. A survey of early methods was presented by Slabaugh et al. [63] in 2001. In general, volumetric models are initialized with all voxels labeled as "occupied" and are iteratively carved away using one of two types of constraints. The first type of constraint is the visual hull [42]. If foreground segmentation is available in the input images, voxels which project to background pixels may be carved from the model. The second type of constraint used in volumetric reconstruction methods is that of color consistency: Under the assumption of a Lambertian scene and fixed illumination, a surface point must project to a pixel of the same intensity for each image in which it is visible. Seitz and Dyer [62] showed that an ordinal visibility constraint on the set of voxels is satisfied when no scene point is contained within the convex hull of the camera centers. Using this constraint, voxels may be ordered such that all voxels which potentially occlude a given voxel occur before it in the ordering. The set of voxels is then traversed, and voxels which do not satisfy the color consistency constraint are removed. Those that do are assigned the color of their projections. Kutulakos and Seitz [41] later presented a variation which overcomes the ordinal visibility constraint by traversing the voxel grid in six passes (one for each of the axis-aligned directions), with only a subset of the cameras being considered during each pass. Slabaugh et al. [64] presented their "Generalized Voxel Coloring" algorithm in 2004 which allows the voxels to be scanned in a more general order. They also present a novel projective voxel warping function which allows an infinite volume to be represented with a finite number of voxels.

Because the focus of this thesis is on the automatic modeling of complete scenes, visual

hull methods which require foreground segmentation are not applicable in general. Voxel coloring methods have been shown to produce satisfactory results, but are prone to errors due to violations of the color consistency constraint due to the incorrect assumption of Lambertian surfaces, calibration errors, sensor noise, or other factors. These errors often manifest themselves as incorrectly carved "holes" in the model [57]. To combat these errors, several probabilistic methods have been proposed [57, 7, 5, 53] which do not "carve" voxels, but rather assign each a probability of existing as part of the model. Because the models and algorithms presented in this thesis are closely related to these probabilistic methods, they are discussed in detail separately in Section 2.4.

2.2 Image-Based Rendering

The major application area focused on in this thesis is that of novel viewpoint generation, an operation which can, in general, be performed using one of two strategies. The first is to reconstruct a 3-d model of the scene based on the imagery and then render the model directly. The second possible strategy is to render the novel views by intelligently drawing pixel data directly from the input images; the class of methods which follow this strategy are collectively known as "image-based rendering" (ibr) techniques. In practice, image-based rendering methods range from having no 3-d model [43] to a full 3-d model with input images used for texture mapping [15]. Two pioneering papers in the field were presented by Levoy and Hanrahan [43] and Gortler et al. [29] in 1996. Based on the observation that the plenoptic function [1], which models light visible from all points in all directions for all time, can be reduced to a 4-d function under certain assumptions (static scene, constant lighting, viewpoints in free space). Levoy and Hanrahan represent images as 2-d slices of the 4-d light field, allowing them to generate new views by extracting and resampling the appropriate samples from the data structure. Gortler et al. use a similar model and present a method for utilizing geometric information about the scene if available.

Because most image-based rendering techniques use a crude or non-existent 3-d model, generating images free of artifacts is a major challenge. Fitzgibbon et al. [22] aim to solve the problem by constraining the generated views such that their texture statistics are similar

to those of the input images. Woodford et al. [73] also use patches of the input images as priors on the output image, but formulate the problem using Markov Random Fields with pair-wise potentials, which allows for global optimization. In general, state of the art image-based techniques are capable of producing convincing images under certain assumptions, but have trouble when the synthetic viewpoint is far from that of any input images (See Chapter 6).

2.3 Site Modeling

The term "site modeling" describes the process of constructing a 3-d representation of a geographic area based on sensor data such as aerial imagery. Other forms of data such as LiDAR [78, 67] or digital elevation models (DEM's) may also be used. Geographic site modeling is a large field with research and development efforts taking place both in academia and industry. Hu et al. [37] provide an overview of urban modeling techniques, with a particular focus on large scale modeling. The various approaches can be classified into two categories, manual and automatic. Most state of the art systems in use in industrial settings automate at least part of the modeling process, however for the purposes of this thesis, a site modeling tool requiring any user interaction is classified as manual. Many fully automatic tools do exist, but are in general not yet reliable or robust enough to generate models as complete and accurate as the manual tools.

2.3.1 Manual Methods

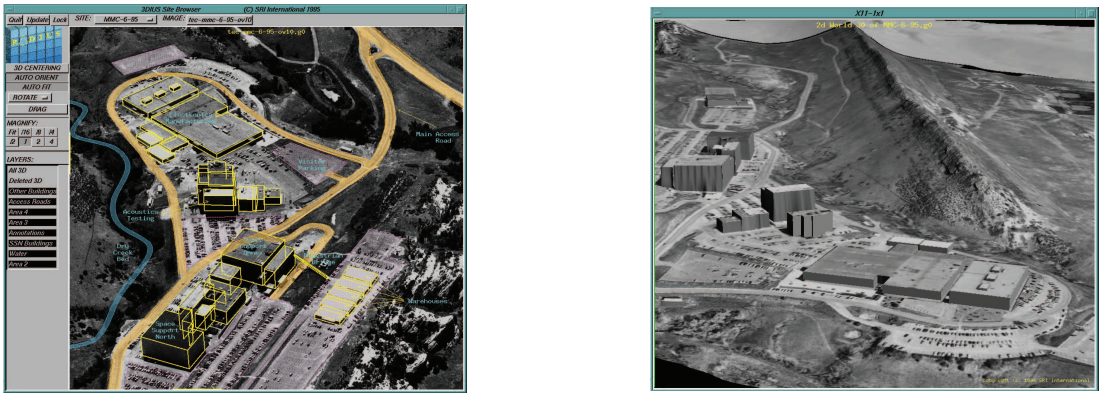
An early site modeling tool was the "Façade" system, presented by Debevec et al. [16] in 1996. The system used manually selected image constraints to optimize parameters of user-defined geometric primitives corresponding to components of structures in the scene. The primitive-based scene representation can then be automatically refined using stereo correspondences between the images. Rendering of the models utilizes view dependent texture mapping, where the texture data is generated by interpolating between input images based on the viewing angle of the virtual camera.

The RADIUS [21] (Research And Development for Image Understanding Systems)

Figure 2.3: Stages of the Façade modeling system (Image from Debevec et al. [16]): (a) Marked edges on an input image. (b) Optimized geometric primitives. (c) The model reprojected back into an input image. (d) A synthetic view generated using view-dependent texture mapping.

project represents another early effort aimed at developing a set of tools enabling a user to rapidly construct site models from imagery. The RADIUS Common Development Environment (RCDE) [34] is an interface which provides the infrastructure for all of the modeling and image operations which are part of the project. Using the RCDE, users can manually create primitive building and terrain models and render texture mapped versions of them. As part of the RADIUS project, Fua [24] developed a method for optimizing manually created building models and terrain features such as roads and rivers based on stereo and shading information present in images, as well as geometric constraints.

A similar tool, the "CyberCity Modeler" [30], was developed at ETH Zurich at about the same time frame. Using the tool, the user manually selects correspondences of feature points in multiple calibrated images. The positions of the features are then triangulated, and polyhedral models are automatically generated using the 3-d point locations. The resulting models can be manually edited to improve accuracy and robustness, and then texture mapped using aerial and ground-based imagery. Other manual methods exist both in the



(a) (b)

Figure 2.4: The RADIUS Common Development Environment (Images taken from Heller and Quam [34]): (a) A screenshot from the RCDE (b) A rendering of a texture mapped site model.

research literature [27], and in the form of production software (Sketchup, 3dvia). In general, simple primitive-based models can be generated with ease, but larger time commitments are necessary for generation of realistic and complex models. The work presented in this thesis differs in that it is a fully automatic system which is able to generate models of scenes with arbitrarily complex structure.

2.3.2 Automatic Methods

Manual and semi-automatic methods for site modeling have proven to be both accurate and robust, but have the obvious disadvantage of requiring human interaction. In order to create larger and more detailed models in a shorter time-frame, automatic methods are needed.

Although the focus of the RADIUS project [21] was primarily on the creation of an interactive site modeling environment, some fully automatic components were developed as well. Collins et al. [11] developed a method to automatically detect line segments in aerial images, and organize the segments into groups corresponding to building rooftops. The rooftop heights are determined using epipolar matching across images, along with a digital terrain map to constrain the heights. The authors show promising results on test data, but the system is only capable of extracting simple roof models consisting of single polygons.

A more sophisticated system that also relies on the assumption of planar scenes was presented by Dick et al. [18] in 2000. An initial reconstruction is first computed based on matching corner features across multiple frames. Using prior models and geometric constraints (vertical and horizontal planes, perpendicular intersection of planes, etc.), a planar model is fit to the reconstructed point cloud. An initial planar model is estimated from the point cloud by first assuming that all planes are perpendicular to a common ground plane. Planes are estimated from the point cloud using RANSAC, and the normal of the ground plane is then estimated. All points are then projected onto the ground plane, and lines are fit to clusters of the projected points. Assuming that the lines correspond to rectangular and vertical walls, a rough model can be estimated. A refined model is constructed by fitting parameterized architectural features (arches, windows, etc.) to manually selected regions of the planes. Structure and motion algorithms can also be formulated to take advantage of the planarity and grid-like structure of urban scenes [60].

Much recent effort has been focused on the reconstruction of urban scenes from ground-based imagery. Pollefeys et al. [54] have developed a complete system for real-time reconstruction of urban models based on street-level video captured from a moving vehicle. Camera pose is estimated using GPS/INS sensor data combined with structure from motion techniques, and depth maps are generated for each image using dense stereo. Finally, the depth maps are fused and meshed to create the model. A similar system has been developed by Cornelis et al. [12] for the purpose of novel view generation in vehicle navigation systems. The authors make the assumption that building facades are ruled surfaces with lines parallel to the gravity vector, and the ground plane is a ruled surface which is known based on the camera height. A novel feedback loop is presented in which occluding objects such as pedestrians and vehicles are detected and removed from the model generation process. These vehicle-based reconstruction methods are capable of generating convincing renderings similar ground-based viewpoints, but are unable to produce full 3-d building models based on the limited input data and, in the case of Cornelis et al., simplifying assumptions which do not hold for general 3-d building shapes.

(a) (b)

Figure 2.5: Reconstruction results from the Urbanscape project. (Image from Pollefeys et al. [54]): (a) reconstruction of building facades viewed from above (b) building models viewed from the side.

2.3.3 Utilizing Range Data

In order to improve accuracy and completeness, many methods employ the use of LiDAR to produce range scans or digital elevation models (DEM's) of the scenes to be modeled. You et al. [78] present a method requiring minimal user interaction in order to optimize a primitive-based site model using airborne LiDAR data. In later work [38], they utilize aerial and ground-based imagery to refine and texture-map the building models. Ground-based LiDAR has also been used to construct detailed building models. Stamos et al. [66] present a method for registering scans with each other, and Liu and Stamos [44, 45] present a method for registering the scans with ground-based imagery for the purposes of photorealistic rendering.

2.3.4 Industrial State of the Art

Several commercial products exist which provide users the ability to construct accurate 3-d site models based on image data and manual input. Examples include PhotoModler from Eos Systems Inc. and ImageModeler, a product of Autodesk. In addition, companies such as CyberCity 3d, LLC (originally a spin-off company of ETH Zurich, developers of the CC-modeler tool [30]) provide customers with ready-made 3-d site models of geographic regions. In general, significant manual effort is still required in order to create site models of industrial quality.

2.4 Probabilistic Reconstruction Methods

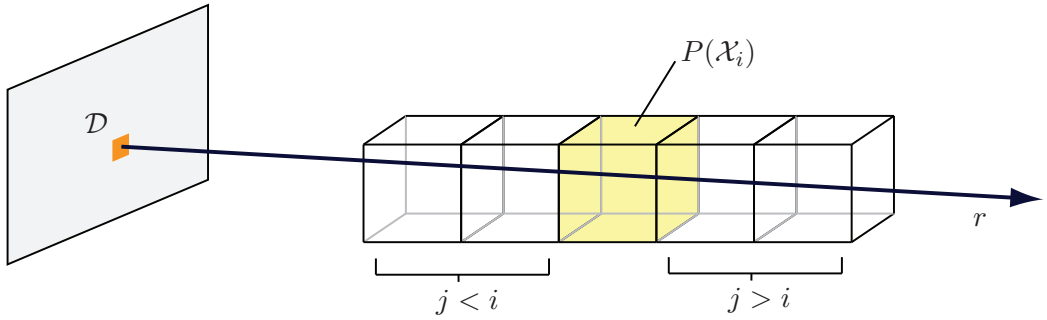


Figure 2.6: Notation used while discussing the discrete probabilistic reconstruction methods. A camera ray r corresponding to data $\mathcal{D}$ passes through a series of voxels, each with occupancy probability $P(\mathcal{X}_i)$.

There exists in the literature several distinct methods for reconstructing a probabilistic volumetric scene model based on image data, all based on discrete voxel models. Although the methods vary in their approach, the goal is the same: to produce a volumetric representation of the 3-d scene, where each point in the volume is assigned a probability based on the likelihood of it being contained in the scene. In the following discussion, a consistent notation (which may differ from that of the original authors) is applied to facilitate comparison. It is assumed that there exists a set S of voxels which are occupied. The probability that a particular voxel v belonging to the set is denoted $P(\mathcal{X}_v)$. The term $\mathcal{D}$ refers to the available image data.

2.4.1 Responsibility Weighted Voxels

Although not formulated as such, De Bonet and Viola [57] presented in 1999 what is essentially the first probabilistic method for volumetric reconstruction. The authors define an imaging model in which each observed pixel is computed as a weighted sum of voxel colors along the viewing ray. These weights are referred to as "responsibilities", and can be computed for each voxel along the ray as the product of the voxel's probability and its visibility probability:

$$\Omega_i = P(\mathcal{X}_i) \prod_{j<i} (1 - P(\mathcal{X}_j)) \tag{2.1}$$

where the index i refers to the ordering of the voxels along the camera ray. Note that the authors did not refer to the $P(\mathcal{X}_i)$ values as probabilities, but as "opacities"; the terms and notations have been changed for consistency with other methods, but the algorithm remains as originally presented. Following the computation of the responsibilities Ω_i^k for each voxel i and image k, the voxel's color is computed as the weighted average of the projections, weighted by the responsibilities. Given the new color estimates, the responsibilities are recomputed using a heuristic based on the agreement of the voxel's color and the projection into the image. From the responsibilities a separate probability $P_k(\mathcal{X}_{xyz})$ for each voxel i and image k can be computed as follows:

$$P_k(\mathcal{X}_{xyz}) = \frac{\Omega_i^k}{1 - \sum_{j<i} \Omega_j^k} \tag{2.2}$$

Finally, the voxel's updated probability $P(\mathcal{X}_i)$ is computed based on the weighted average of the probabilities $P_k(\mathcal{X}_i)$, weighted by the responsibilities Ω_i^k. The process is repeated until the voxel probabilities converge.

2.4.2 Probabilistic Space Carving

The first volumetric reconstruction algorithm to be explicitly formulated using a probabilistic framework was proposed by Broadhurst et al. [7] in 2001 and later elaborated in Broadhurst's thesis [8]. It is a global algorithm which uses Bayes' theorem to compute the probability of each voxel existing in the model.

$$P(\mathcal{X}|\mathcal{D}) = \frac{P(\mathcal{X})P(\mathcal{D}|\mathcal{X})}{P(\mathcal{D})} \tag{2.3}$$

Broadhurst's method is an offline algorithm, and so the data term $\mathcal{D}$ in Equation 2.3 represents all available image data. The prior probability term $P(\mathcal{X})$ is assumed to be a constant (0.5 is given as a nominal value), and the reconstruction is performed (in theory) in a one-time noniterative operation. The denominator is expanded as follows:

$$P(\mathcal{X}|\mathcal{D}) = \frac{P(\mathcal{X})P(\mathcal{D}|\mathcal{X})}{P(\mathcal{D}|\mathcal{X})P(\mathcal{X}) + P(\mathcal{D}|\overline{\mathcal{X}})P(\overline{\mathcal{X}})} \tag{2.4}$$

The term $P(\overline{\mathcal{X}})$ represents the prior probability that the voxel is not occupied, and is equal to $1 - P(\mathcal{X})$. The only remaining issues are how to compute the probabilities $P(\mathcal{D}|\mathcal{X})$ and $P(\mathcal{D}|\overline{\mathcal{X}})$. It is assumed that each voxel has associated with it an appearance model $\{\mu, \sigma\}$ based on the assumption of a spherical gaussian color distribution with mean color μ and covariance matrix $\sigma^2 I$. The appearance models, however, are not estimated explicitly but marginalized using a uniform prior for the mean color and the prior $P(\sigma) = \frac{1}{\sigma}$ for the standard deviation. This prior is chosen primarily for mathematical convenience as it allows a closed-form solution for the probabilities $P(\mathcal{D}|\mathcal{X})$ and $P(\mathcal{D}|\overline{\mathcal{X}})$ to be computed, and also for having the desirable attribute that surface voxels with small variances are more likely than large ones. In order to determine the probability of the data $\mathcal{D}$ given that the voxel does not exist, an independent appearance model for the projection of the voxel into each image is assumed. Given that the voxel does exist, the projection of the voxel into a given image can be explained by the voxel, or by any of the voxels "in front" of the voxel along the ray to the camera center. In order to account for these occlusions, all possible cases are, in theory, marginalized. In practice, this is computationally infeasible, and several simplifying assumptions are made.

2.4.3 Stochastic Space Carving

The concept of the *photo hull distribution* was presented by Bhotika et al. [5] in 2002 and expanded upon in Bhotika's thesis [4]. A clear distinction between scene ambiguity and scene uncertainty is made by the authors. Scene ambiguity is caused by the existence of multiple photo-consistent reconstructions, even in the absence of noise. Scene uncertainty is caused by the presence of image noise. The authors define the photo hull as the union of all photo-consistent reconstructions; the photo hull itself is photo-consistent as well. The objective of the reconstruction algorithm is to recover the (unique) photo hull, thereby removing the problem of scene ambiguity. A probabilistic representation (due to scene uncertainty) is obtained by a stochastic process which draws fair samples from photo hull distribution. The samples are obtained through a carving process which randomly chooses voxels and removes them from the photo hull probabilistically based on their photo-consistency. This process is run repeatedly, and each time a voxel belongs to the drawn sample its counter is

incremented. The probability of the voxel belonging to the photo hull is then approximated as the ratio of the count divided by total number of samples drawn.

2.4.4 Online Reconstruction

A third probabilistic formulation, proposed by Pollard and Mundy [53] in 2007 and later elaborated in Pollard's thesis [52], is an online method. Pollard's model and reconstruction algorithm were proposed primarily as a means for change detection in images, but are also well suited for other purposes including novel view generation [52, 14]. The first and most basic difference between Pollard's reconstruction algorithm and Broadhurst's is a direct reflection of Pollard's method being an online algorithm, meaning that it uses information from the current time step only, and Broadhurst's being a global one (i.e. it uses information from all images simultaneously). In Pollard's equations, the term $\mathcal{D}$ represents information from the current image only, and the prior probability $P(\mathcal{X})$ for the current timestep is represented by the posterior probability from the previous timestep. In order to emphasize this point, Equation 2.3 can be rewritten:

$$P_t(\mathcal{X}|\mathcal{D}_t) = \frac{P_{t-1}(\mathcal{X})P(\mathcal{D}_t|\mathcal{X})}{P(\mathcal{D}_t)} \tag{2.5}$$

In addition to the surface probabilities of each voxel, an appearance model is also explicitly modeled using a mixture of Gaussians distribution. The distribution is updated at each timestep using a method similar to that proposed by Stauffer and Grimson [68], with the addition of an update weight based on the probability that the voxel is occupied and is visible in the current image. The visibility probability is given by:

$$\text{vis}(x) = \prod_{i<x}(1 - P(\mathcal{X}_i)) \tag{2.6}$$

where $i < x$ denotes that the voxel i lies on the camera ray and is closer to the camera center than x. The probability of the image data which corresponds to the projection of x is computed as follows:

$$p(\mathcal{D}) = \sum_{i \in R} P(\mathcal{X}_i)\,\text{vis}(i)p(\mathcal{D}|v = i) \tag{2.7}$$

Essentially, the identity of the unknown voxel v which produced the image data $\mathcal{D}$ is marginalized by summing the appearance probabilities $p(\mathcal{D}|v=i)$ (given by the mixture of Gaussians distribution at i) of all voxels along the camera ray R, weighted by the probability that i is a surface voxel and is visible. The expanded update equation is written:

$$P(\mathcal{X}|\mathcal{D}) = P(\mathcal{X})\frac{\left\{\sum_{i'<i} P(\mathcal{X}'_i)\,\mathrm{vis}(i')p(\mathcal{D}|v=i')\right\} + \mathrm{vis}(i)p(\mathcal{D}|v=i)}{\sum_{i'} P(\mathcal{X}'_i)\,\mathrm{vis}(i')p(\mathcal{D}|i'=v)} \tag{2.8}$$

and is applied to each voxel with the introduction of each new image. Pollard and Mundy show that under certain simplifying assumptions, the model will converge to a state in which all occupied voxels have probability 1 and all empty voxels converge to probability 0 after sufficiently many updates. The update equation is used as the basis for the online updating algorithm presented in Chapter 5.

2.4.5 Conclusions

While the work presented in this thesis is closely related to previously developed probabilistic reconstruction algorithms (particularly that of Pollard and Mundy [53]), a major advantage of this work is the continuous probabilistic representation as discussed in Section 1.1.3. State of the art probabilistic models are not capable of reconstructing large and complex scenes, a critical obstacle to successful site modeling which is overcome by the proposed model and methods.

Chapter 3

The Continuous Probabilistic Scene Model

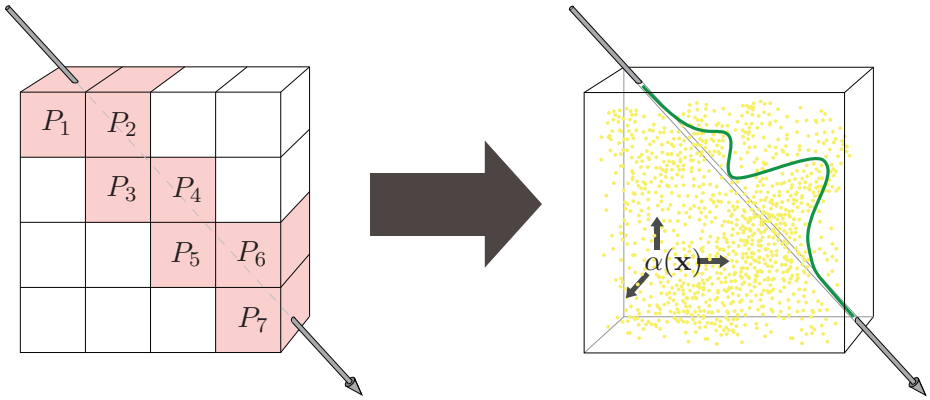


Figure 3.1: Traditionally, a volume is modeling probabilistically using discrete units of space. The proposed model uses a continuously varying scalar field to represent likelihood of occlusion.

Based on the discussion in Section 1.1, the advantages of a probabilistic model are clear. The regular discretization inherent in state of the art probabilistic models [53, 77, 7], however, make the modeling of very large scenes prohibitively expensive in terms of storage costs. In this chapter, a continuously varying probabilistic scene model is proposed which generalizes the discrete model proposed by Pollard and Mundy [53]. The proposed continuous model allows implementations to sample the volume in a non-uniform fashion, leading to the possibility of significantly more space-efficient representations (Examples of

such implementations will be presented in Chapter 4). This chapter is laid out as follows: In Section 3.1, the theoretical basis for the continuous probabilistic geometry model is introduced. In Section 3.2, the inclusion of appearance information in the model is discussed. In Section 3.3, methods for rendering models containing geometry and appearance information are presented, and the chapter is concluded in Section 3.4

3.1 Geometry

A continuous representation of surface geometry is proposed in the form of a scalar function termed the *occlusion density*. The occlusion density at a point in space can be thought of as a measure of the likelihood that the point occludes points behind it along the line of sight of a viewer, given that the point itself is unoccluded. More precisely, the occlusion density value at the point is a measure of occlusion probability per unit length of a viewing ray which is passing through the point. If the occlusion density is defined over a volume, probabilistic visibility reasoning can be performed for any pair of points within the volume. In the case where surface geometry exists and is known completely (e.g. scenes defined by a surface mesh), the occlusion density is defined as infinite at the surface locations, and zero elsewhere.

3.1.1 Occlusion density definition

Given a ray in space defined by its origin point $\mathbf{q}$ and a unit direction vector $\mathbf{r}$, the probability of each point $\mathbf{x}$ along the ray being visible from $\mathbf{q}$ may be computed. It is assumed here that $\mathbf{q}$ is the position of a viewing camera, and $\mathbf{r}$ represents a viewing ray of the camera, but the assumption is not necessary in general. Points along the line of sight may be parameterized by s, the distance from $\mathbf{q}$:

$$\mathbf{x}(s) = \mathbf{q} + s\mathbf{r},\ s \geq 0 \tag{3.1}$$

Given the point $\mathbf{q}$ and viewing ray $\mathbf{r}$, a proposition $\mathcal{V}_s$ may be defined as follows:

$$\mathcal{V}_s \equiv \text{“The point along } \mathbf{r} \text{ at distance } s \text{ is visible from } \mathbf{q} \text{ .”} \tag{3.2}$$

The probability $P(\mathcal{V}_s)$ is a (monotonically decreasing) function of s and can be written as such using the notation vis(s).

$$\text{vis}(s) \equiv P(\mathcal{V}_s) \tag{3.3}$$

Given a segment of r with arbitrary length ℓ beginning at the point with distance s from $\mathbf{q}$, an additional proposition $\mathcal{Q}_s^\ell$ may be defined.

$$\begin{aligned} \mathcal{Q}_s^\ell \quad \equiv \quad & \text{“A ray entering the region of length } \ell \text{ at distance } s \text{ is occluded} \\ & \text{while passing through the region.”} \end{aligned} \tag{3.4}$$

The probability $P(\mathcal{Q}_s^\ell)$ is termed the *occlusion probability* of the segment and can be defined as the probability that the point at distance $s+\ell$ is not visible, given that the point at distance s is visible.

$$\begin{aligned} P(\mathcal{Q}_s^\ell) &= P(\bar{\mathcal{V}}_{s+\ell}|\mathcal{V}_s) \\ &= 1 - P(\mathcal{V}_{s+\ell}|\mathcal{V}_s) \end{aligned} \tag{3.5}$$

Using Bayes' Theorem,

$$P(\mathcal{Q}_s^\ell) = 1 - \frac{P(\mathcal{V}_s|\mathcal{V}_{s+\ell})P(\mathcal{V}_{s+\ell})}{P(\mathcal{V}_s)} \tag{3.6}$$

Substituting vis(s) for the visibility probability at distance s and recognizing that $P(\mathcal{V}_s|\mathcal{V}_{s+ds}) = 1$,

$$\begin{aligned} P(\mathcal{Q}_s^\ell) &= 1 - \frac{\text{vis}(s+\ell)}{\text{vis}(s)} \\ P(\mathcal{Q}_s^\ell) &= \frac{\text{vis}(s) - \text{vis}(s+\ell)}{\text{vis}(s)} \end{aligned} \tag{3.7}$$

If an infinitesimal segment length $\ell = ds$ is used, Equation 3.7 becomes:

$$P(\mathcal{Q}_s^{ds}) = \frac{-d\,\mathrm{vis}(s)}{\mathrm{vis}(s)} \tag{3.8}$$

$$\frac{P(\mathcal{Q}_s^{ds})}{ds} = -\frac{\mathrm{vis}'(s)}{\mathrm{vis}(s)} \tag{3.9}$$

The left-hand side of Equation 3.9 can be thought of as a measure of occlusion probability per unit length, or *occlusion density*. The function $\alpha(s)$ is introduced as notation for the occlusion density as a function of distance s along a viewing ray (Equation 3.10). From Equation 3.11 it is clear that the visibility probability at a point along a viewing ray decreases at a rate proportional to both the occlusion density of the point and the value of the visibility probability itself. The occlusion density at a point has thus far been defined as a property of a particular viewing ray passing through the point. It is assumed, however, that the probability that a point occludes a viewing ray is independent of the direction from which the point is being viewed. It is conceivable that this assumption could be violated for some special cases of material, but it is consistent with standard imaging models of opaque surfaces and allows the occlusion density to be specified as a 3-d scalar function $\alpha(\mathbf{x})$ independent of any particular viewing ray. The ray independence assumption manifests itself in traditional discrete probabilistic models by the fact that voxel probabilities are fixed with respect to the viewing direction.

$$\alpha(s) \equiv -\frac{\mathrm{vis}'(s)}{\mathrm{vis}(s)} \tag{3.10}$$

$$\mathrm{vis}'(s) = -\alpha(s)\,\mathrm{vis}(s) \tag{3.11}$$

In practice, the occlusion density of points in a given scene must be computed using a finite number of observations. In these cases the occlusion density at a point in space is a property of what is known about the scene based on the available data, and not an intrinsic property of the point itself. Given that the underlying true occlusion density is a generalized function with support concentrated at the true surface location, and assuming that the modeled occlusion density approaches this function as the number of unique observations increases, it follows that the expected visibility from any viewpoint also approaches the true

visibility. Empirically, the reconstructed occlusion density function does in fact increase in precision with the number of unique observations. (The modeling of the occlusion density function from a finite set of viewpoints is discussed in detail in Chapter 5.) This allows the prediction of visibility probabilities along viewing rays which have not been observed, which is a critical requirement for novel viewpoint rendering (Section 7.3).

3.1.2 Visibility probability calculation

The visibility probability of the point at distance t can be derived in terms of $\alpha(s)$ by integrating both sides of equation 3.10 with respect to s.

$$\begin{aligned} \int_0^t \alpha(s)ds &= \int_0^t \frac{-d\,\mathrm{vis}(s)}{\mathrm{vis}(s)} \\ -\int_0^t \alpha(s)ds &= \left[\ln\left(\mathrm{vis}(s)\right) + c\right]_0^t \\ -\int_0^t \alpha(s)ds &= \ln\left(\mathrm{vis}(t)\right) - \ln\left(\mathrm{vis}(0)\right) \end{aligned} \tag{3.12}$$

Finally, by recognizing that $\mathrm{vis}(0) = 1$ and placing both sides of Equation 3.12 in an exponential:

$$\mathrm{vis}(t) = e^{-\int_0^t \alpha(s)ds} \tag{3.13}$$

Equation 3.13 gives a simple expression for the visibility probability in terms of the occlusion density values along the viewing ray which will be used heavily throughout the remainder of the thesis.

3.1.3 Relationship to the Beer-Lambert Law

It is perhaps interesting to note that the derivation of the visibility probability from Equation 3.8 onward follows very closely the derivation of the Beer-Lambert law, which relates the absorption of light to the properties of a solution through which it is traveling. Rather than visibility probability, the Beer-Lambert law specifies the ratio of the quantities I_0 and I_1: the intensities of the light before entering the solution and after leaving the solution, respectively.

$$\frac{I_1}{I_0} = e^{-\alpha' l} \tag{3.14}$$

Here, α' is known as the *absorption coefficient* (which is assumed to be constant along the path) and l is the length of the path through the solution.

3.1.4 Relationship with discrete voxel probabilities

The key difference (as discussed in Section 1.1.3) between the discrete voxel probabilities $P(\mathcal{X})$ of existing methods and the preceding formulation of occlusion density is the interpretation of the probability values. Because existing methods effectively model the probability that a voxel boundary is occluding, the path length of the viewing ray through the voxel is irrelevant. By contrast, the occlusion probabilities $P(\mathcal{Q}_s^\ell)$ represent the probability that the viewing ray is occluded at any point on the interval $[s, s+\ell]$. In order to interpret a voxel probability as an occlusion probability, it is therefore necessary to associate a canonical path length $\hat{\ell}$ with the voxel. Assuming the voxels are cubical, a reasonable choice is to simply use the voxel side length. The voxel occlusion probability $P(\mathcal{Q}_s^{\hat{\ell}})$ is the complement of the visibility probability of the point $s+\hat{\ell}$, assuming that the visibility probability $\text{vis}(s)$ of the voxel is certain.

$$P(\mathcal{Q}_s^{\hat{\ell}}) = 1 - e^{-\int_s^{s+\hat{\ell}} \alpha(s')ds'} \tag{3.15}$$

Assuming that the occlusion density within the voxel is constant with value α, the voxel probability is computed as follows.

$$P(\mathcal{Q}_s^{\hat{\ell}}) = 1 - e^{-\alpha\hat{\ell}} \tag{3.16}$$

Likewise, given a canonical segment length $\hat{\ell}$ and occlusion probability value $P(\mathcal{Q}_s^{\hat{\ell}})$, the constant valued occlusion density within the voxel may also be computed.

$$\alpha = -\frac{\ln(1 - P(\mathcal{Q}_s^{\hat{\ell}}))}{\hat{\ell}} \tag{3.17}$$

Although the occlusion probabilities as defined in this thesis and the voxel probabilities used in previous probabilistic methods are defined slightly differently, they can therefore

be compared by assigning a canonical segment length to a voxel. Using Equations 3.16 and 3.17, the voxel probabilities can also be converted to and from occlusion density values.

3.2 Appearance

In addition to the occlusion density, it is often useful to define appearance information for points in space. Here, "appearance" is a general term that describes the predicted pixel value of a particular imaging sensor, given that the point is visible from the sensor. In general, many factors contribute to this value: lighting conditions, viewing angle, particularities of the imaging sensor, and others. There is a large body of work in the computer graphics literature that is focused on modeling these factors' effect on an object's appearance, a comprehensive survey of which is given by Dorsey et al. [19].

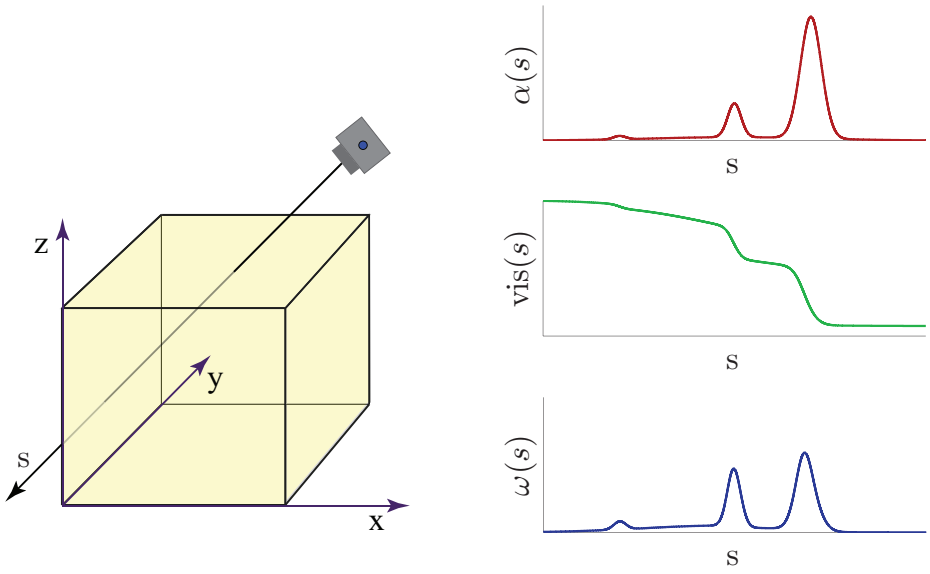
For practical reasons, the appearance models discussed in this thesis are relatively simple and are independent of viewing direction and lighting conditions. It is assumed that for a given point $\mathbf{x}$, a single distribution $p_A(\mathbf{i}, \mathbf{x})$ describes the probability density of the imaged value, $\mathbf{i}$, of $\mathbf{x}$. In the case of greyscale imagery, $\mathbf{i}$ is a scalar value and the distribution is one-dimensional. In the case of multi-band imagery (e.g. RGB color), $\mathbf{i}$ is an n-dimensional vector and the distribution is defined over the n appearance dimensions. Specific distribution models are discussed in Chapter 4.

3.3 Rendering

Given a volume for which both occlusion density and appearance are defined for all points contained in the volume, the probability distribution of a particular pixel value can be computed. It is assumed that each pixel is associated with a corresponding camera ray defined by a origin point $\mathbf{q}$ (usually the camera center), and a viewing direction defined by a unit vector $\mathbf{r}$.

3.3.1 Probability density function of imaged point location

Given a camera ray again parameterized by s, the unknown distance s_o at which the ray is occluded can be modeled as a random variable with cumulative density function $1 - \text{vis}(s)$.



(a) (b)

Figure 3.2: (a) A camera ray parameterized by s passes through a volume over which the scalar field $\alpha(\mathbf{x})$ is defined. (b) Plots of $\alpha(s)$, vis(s), and $\omega(s)$.

(The probability that a point is visible from the camera center is the complement of the probability that the ray is occluded by a point $s_o < s$.) A new function, $\omega(s)$, is defined as the probability density function of the point of occlusion.

$$\begin{aligned} \omega(s) &\equiv \text{PDF}(s_o) \\ \omega(s) &= \frac{d}{ds}\left[1 - \text{vis}(s)\right] \\ \omega(s) &= \alpha(s) e^{-\int_0^s \alpha(s')ds'} \end{aligned} \tag{3.18}$$

By substituting the definition of vis(s) back into Equation 3.18,

$$\omega(s) = \alpha(s)\,\text{vis}(s) \tag{3.19}$$

This definition is not surprising, since in order for s to be the occlusion distance, the point at s must be occluding *and* visible from the camera.

In many cases, it is useful to compute the probability that a camera ray passes through all (modeled) space unoccluded. This is denoted by the probability vis(∞). By definition

of $\omega(s)$:

$$\int_0^\infty \omega(s)ds = 1 - \text{vis}(\infty) \tag{3.20}$$

The complete probability distribution of s_o for a given ray is therefore composed of the continuous density function $\omega(s)$, plus the discrete probability $\text{vis}(\infty)$.

$$\int_0^\infty \omega(s)ds + \text{vis}(\infty) = 1 \tag{3.21}$$

3.3.2 Probability density function of imaged pixel value

Using the density function $\omega(s)$ and the appearance distributions $p_A(\mathbf{i}, s)$ for each point $\mathbf{x}(s)$ along the camera ray, a total appearance distribution describing the imaged pixel value can be computed:

$$p_T(\mathbf{i}) = \int_0^\infty \omega(s)p_A(\mathbf{i}, s)ds + \text{vis}(\infty)p_A(\mathbf{i}, \infty) \tag{3.22}$$

The term $p_A(\mathbf{i}, \infty)$ represents the appearance model in the case where the camera ray passes unoccluded through space.

For the purpose of rendering images of the scene, the expected pixel value $E[p_T(\mathbf{i})]$ is often all that is required to be computed. In this case, the expected values of the appearance model distributions may be accumulated in place of the distributions themselves.

$$E[p_T(\mathbf{i})] = \frac{1}{(R_1 - R_0)} \int_{R_0}^{R_1} \int_0^\infty \omega(s)p_A(\mathbf{i}, s)ds + \text{vis}(\infty)p_A(\mathbf{i}, \infty)d\mathbf{i}$$

The values R_1 and R_0 represent the upper and lower limits of the intensity value i, respectively. Changing the order of integration:

$$\begin{aligned} E[p_T(\mathbf{i})] &= \int_0^\infty \omega(s)\frac{1}{(R_1 - R_0)} \int_{R_0}^{R_1} p_A(\mathbf{i}, s)ds + \text{vis}(\infty)p_A(\mathbf{i}, \infty)d\mathbf{i} \\ &= \int_0^\infty \omega(s)E[p_A(\mathbf{i}, s)]ds + \text{vis}(\infty)E[p_A(\mathbf{i}, \infty)] \end{aligned} \tag{3.23}$$

It is important to note that Equation 3.23 provides a means for computing an expected image pixel value by weighting and accumulating expected values of appearance

models along the corresponding camera ray, allowing implementations to efficiently render expected images from arbitrary viewpoints. Figure 3.3 shows two examples of expected images rendered using this technique.

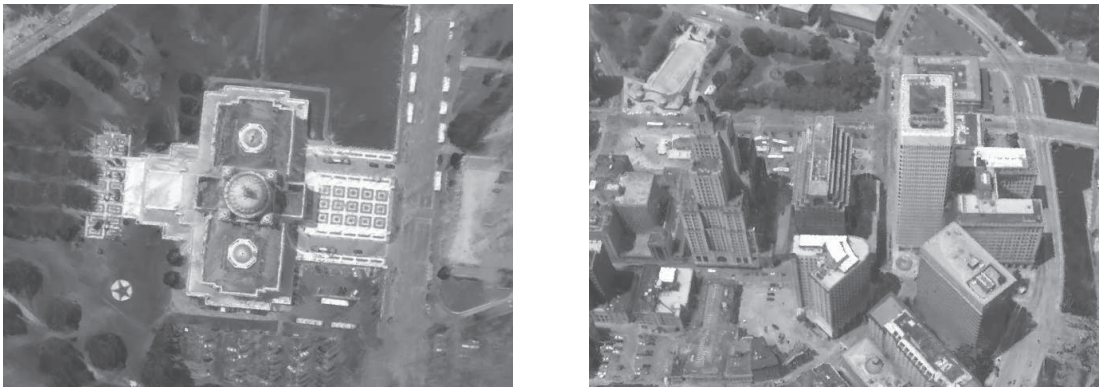


Figure 3.3: Expected images generated from models of the capitol (left) and downtown (right) aerial sequences.

3.4 Conclusion

The model presented in this chapter allows for the probabilistic representation of scene geometry and appearance in a continuous manner. Given a scene model and a camera, visibility information for any point in the scene can be determined, as well as a probability distribution of the value for each pixel in the image. The main motivation behind this model is to provide a theoretical framework for probabilistic scene model implementations which are not inherently tied to the regular sampling of space, i.e. voxel-based models.

Chapter 4

Implementation

In order to make practical use of the continuous probabilistic scene model described in Chapter 3, a finite-sized representation which is able to associate both an occupancy probability and appearance information with each point in the working volume is needed. One such representation involves storing a finite set of sample values from the volume and interpolating values between them. Based on the sampling theorem, it is clear that a perfect reconstruction can be achieved if the volume is sampled with a frequency at least twice that of the highest frequency present in the continuous model. If some simplifying assumptions are made, however, more efficient sampling schemes can be employed. Details are presented in this chapter of an implementation which approximates the underlying occupancy probability and appearance functions as piecewise-constant and assumes that the modeled volume contains large regions in which the functions are slowly varying. Section 4.1 introduces the piecewise-constant model and discusses its representation using an adaptively refined octree. The visibility and rendering equations presented in Chapter 3 are reformulated based on the piecewise-constant assumption in Section 4.2. In addition to intelligent sampling, appearance information at each cell must also be represented efficiently; this is discussed in Section 4.3. An alternative representation based on a piecewise-linear approximation is briefly discussed in Section 4.4, and the chapter is concluded in Section 4.5.

4.1 Piecewise-Constant Model

Most real-world scenes contain large, slowly varying regions of low occlusion probability in areas of "open space" and high, quickly varying occlusion probability near "surface-like" objects. It therefore makes sense to sample $\alpha(\mathbf{x})$ in a nonuniform fashion. The proposed implementation approximates both $\alpha(\mathbf{x})$ and the appearance model as being piecewise constant, with each region of constant value represented by a cell of an adaptively refined octree.

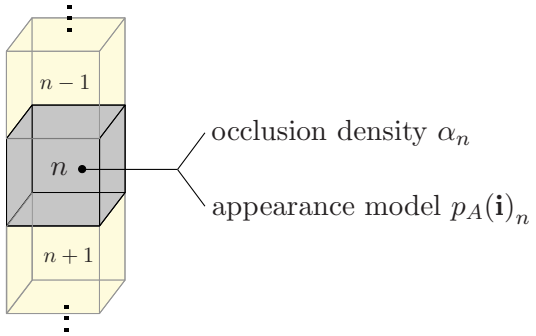


Figure 4.1: Each cell of the octree is associated with a single occlusion density value α_n and a probability density function $p_A(\mathbf{i})_n$ representing the appearance of the cell.

4.1.1 Octree Representation

An octree is a hierarchical spatial data structure which is the three-dimensional extension of its two-dimensional counterpart, the quadtree. (The term "quadtree" is sometimes used to refer to the general class of data structures.) The general principles of the data structure are presented in Section 4.1.2, and a specific implementation of the 2-d quadtree in Section 4.1.3. The implementation is extended to three dimensions in Section 4.1.4. For further details about the quadtree and octree data structures, the reader is referred to Samet's comprehensive treatment [58].

4.1.2 The Quadtree

The quadtree is a hierarchical spatial data structure first proposed by Morton in 1966 [49]. Its main purpose is to provide a subdivision of space which is efficient in terms of storage

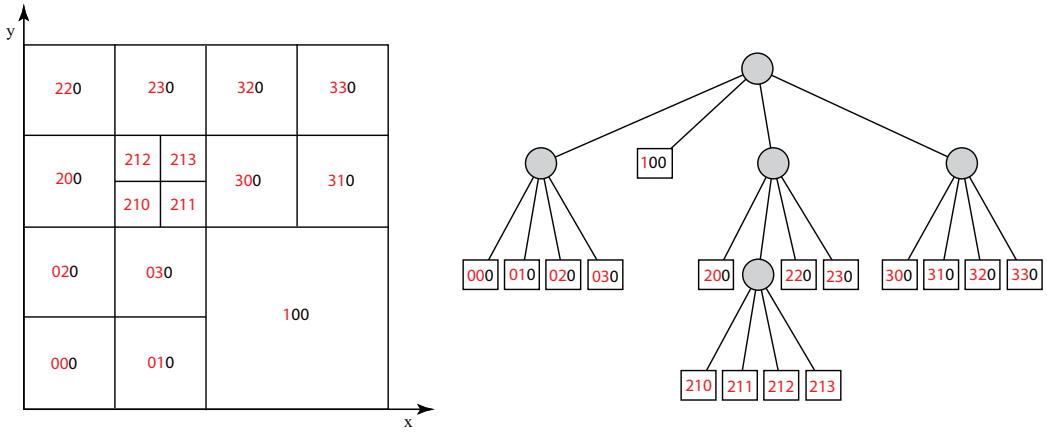


Figure 4.2: A three-level quadtree, with base-4 fd-linear codes assigned to each leaf node and corresponding cell. The significant digits of the fd-linear codes are colored in red. Left: The spatial organization of the quadtree. Right: The tree structure.

costs and computational complexity of basic operations (e.g. query). A tree of degree four is used to represent a hierarchy of square-shaped regions, with the children of each node representing its four quadrants. In a variable resolution quadtree, regions are recursively subdivided until some application-specific criterion in the region is met.

4.1.3 The FD-Linear Quadtree

For some applications, only the leaf nodes of the quadtree need to be explicitly represented. In this case the structure may be fully represented using a set of leaf node locational codes, each composed of a fixed length base-4 index and an integer representing the depth in the tree. The index of a node contains $2d$ significant bits, where d is the depth of the node. Each base-4 (two bit) digit represents an edge in the quadtree in a traversal from the root node. A quadtree represented by a set of leaf nodes with indices constructed in this manner was first proposed by Rosenfeld et al. in 1983 [56] and is known as an FD-Linear quadtree (Figure 4.2). When the trailing insignificant bits are set to 0, traversal of the cells in the order of their indices corresponds to an inorder traversal of the leaf nodes of the corresponding tree.

4.1.4 The FD-Linear Octree and Piecewise-Constant Approximation

The principles of the quadtree can be trivially extended to three dimensions, giving rise to the octree data structure. Rather than quadrants, the eight children of each node in an octree represent octants of its corresponding cubic region. The fd-linear codes used to represent leaves in the tree contain $3d$ significant bits, with each base-8 (three bit) digit representing an edge along the path to the leaf in the degree eight tree. The proposed implementation partitions the space to be modeled into fixed-sized cubes, each cube being represented by an fd-linear octree. Each cell in an octree stores a single occlusion density value α and appearance distribution $p_A(\mathbf{i})$. The appearance distribution represents the probability density function of the pixel intensity resulting from the imaging of the cell. (The representation used for the function $p_A(\mathbf{i})$ is discussed in Section 4.3.) The occlusion density value and appearance distribution are assumed to be constant within the cell. Note that this piecewise-constant assumption can be made arbitrarily accurate since, in theory, any cell in which the approximation is not acceptable can always be subdivided into smaller cells. In practice, however, the amount of useful resolution in the model is limited by the resolution of the input data used to construct it.

4.2 Visibility Reasoning and Rendering using the octree

Taking advantage of the piecewise-constant approximation, the visibility and rendering equations presented in Chapter 3 can be simplified considerably. Given a ray again defined by an origin point $\mathbf{q}$ and a unit direction vector $\mathbf{v}$, and parameterized by the distance s along $\mathbf{v}$ from $\mathbf{q}$, the function $\alpha(s)$ along the ray can be represented by S segments $[s_i, s_{i+1}]$, $i = 0, 1, \ldots S-1$ with corresponding constant occlusion density values α_i and appearance model distributions $p_{A_i}(\mathbf{i})$. The appearance model $p_{A_i}(\mathbf{i})$ represents the probability distribution of the imaged intensity, given that the intensity was produced by a point v along the ray with $s_i < v \leq s_{i+1}$). The distance $(s_{i+1} - s_i)$ is the length of the segment passing through the ith cell along the ray and is abbreviated ℓ_i (Figure 4.3). This simplification allows the integrals in Equations 3.13, 3.18, and 3.22 to be easily computed as discrete summations.

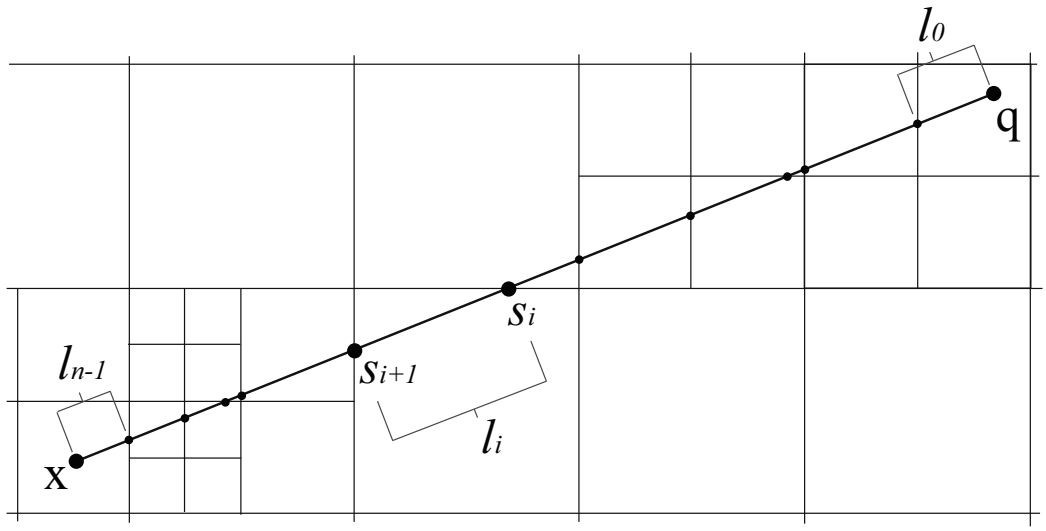


Figure 4.3: A camera ray parameterized by s cuts through cells in the octree. Both the occlusion density and appearance model are approximated as being constant within each cell.

4.2.1 Visibility Reasoning

Using the piecewise constant assumption and a point $\mathbf{x} = \mathbf{q} + s\mathbf{v}$ lying within the nth cell along the ray defined by $\mathbf{q}$ and $\mathbf{v}$, the visibility of $\mathbf{x}$ from $\mathbf{q}$ can now be computed as:

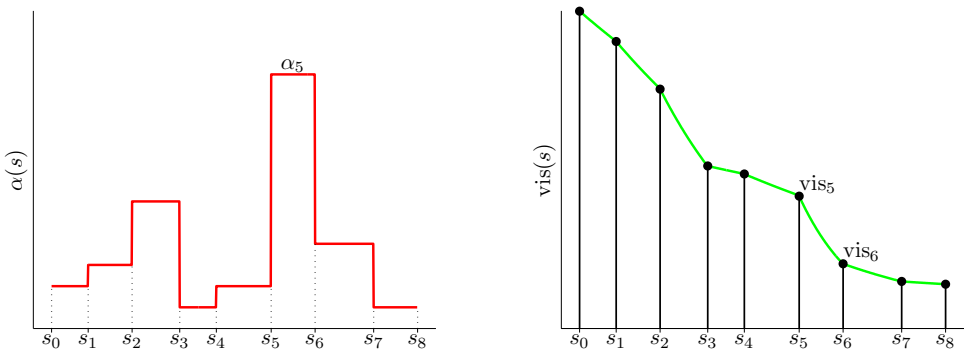
$$\text{vis}(s) = \exp(-\int_0^s \alpha(s)ds) = \exp\left(-\sum_{i=0}^{n-1} \alpha_i \ell_i + \alpha_n(s - s_n)\right) \tag{4.1}$$

The visibility probability of the points $\mathbf{x}(s_i)$ are defined as vis_i:

$$\text{vis}_i = \exp\left(-\sum_{i=0}^{n-1} \alpha_i \ell_i\right) \tag{4.2}$$

4.2.2 Rendering

Given an image pixel and corresponding camera ray parameterized by s, a discrete probability Ω_i can be assigned to each octree cell intersected by the ray representing the probability that a point along the ray within the cell is imaged by the camera. (Here, $\omega(s)$ is as defined in Equation 3.19.)



(a) (b)

Figure 4.4: Plots of a piecewise-constant occlusion density function (a) along a viewing ray, and the corresponding visibility probability function (b). The term α_i is defined as the (constant) value of the occlusion density function between s values s_i and s_{i+1}, while the term vis$_i$ is defined as the visibility probability at $s = s_i$.

$$\begin{aligned}
\Omega_i &= \int_{s_i}^{si+1} \omega(s)ds \\
&= \int_{s_i}^{s_{i+1}} \alpha_i e^{-\int_0^s \alpha(s')ds'} ds \\
&= \alpha_i e^{-\int_0^{s_i} \alpha(s')ds'} \int_{s_i}^{s_{i+1}} e^{-\int_{s_i}^{s_{i+1}} \alpha_i ds''} ds \\
&= \alpha_i e^{-\sum_{n=0}^{i-1} \alpha_n \ell_n} \left[\frac{1 - e^{-\alpha_i \ell_i}}{\alpha_i} \right] \\
&= e^{-\sum_{n=0}^{i-1} \alpha_n \ell_n} (1 - e^{-\alpha_i \ell_i}) \qquad (4.3)
\end{aligned}$$

This expression is equivalent to the difference of the two consecutive visibility probabilities vis$_i$ and vis$_{i+1}$.

$$\Omega_i = \text{vis}_i - \text{vis}_{i+1} \qquad (4.4)$$

Using the Ω_i's as weights, the expected intensity $E[\mathbf{i}]$ of the observed intensity at an image pixel may be computed as a weighted sum of the n distributions $p_{A_i}(\mathbf{i})$ along the corresponding camera ray based on Equation 3.23.

$$E[I] = \sum_{i=0}^{S-1} \Omega_i E[p_{A_i}(\mathbf{i})] + E[p_{A_\infty}(\mathbf{i})] \, \text{vis}_\infty \qquad (4.5)$$

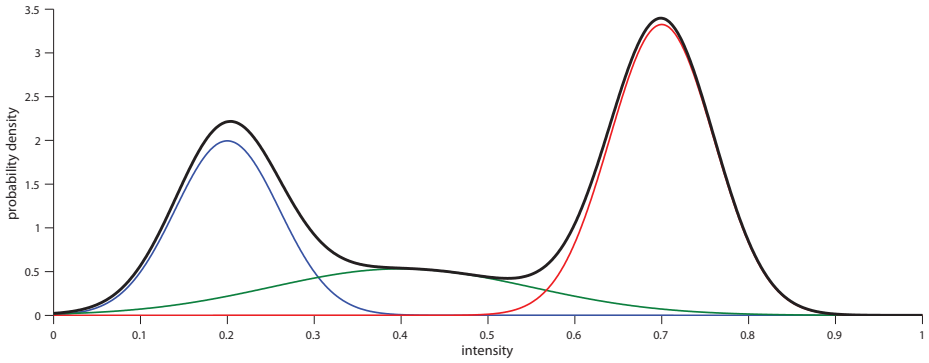


Figure 4.5: A mixture of Gaussians distribution in one dimension with three modes. The distributions of the individual modes are plotted in red, green, and blue. The composite distribution is plotted in black.

The term vis_∞ represents the probability that the camera ray passes unoccluded through the model, and the appearance model $p_{A_\infty}(\mathbf{i})$ represents the intensity distribution of pixels produced by such unoccluded rays. Each pixel and corresponding camera ray are treated as being independent of the others, simplifying computations and allowing the expected intensities of each pixel to be computed in parallel.

4.3 Appearance Models

The appearance model $p_A(\mathbf{i})$ assigned to each cell is represented by a mixture of Gaussians distribution, similar to that proposed by Stauffer and Grimson [68] for 2-d background modeling and used by Pollard and Mundy [53] in their voxel model. The distribution is composed of a fixed number N of Gaussian distributions with means μ_n and standard deviations σ_n. Each Gaussian component is weighted by a scale factor w_n.

$$\begin{aligned} p_A(\mathbf{i}) &= \frac{1}{W}\sum_{n=1}^{N}\frac{w_n}{\sigma_n\sqrt{2\pi}}\exp(-\frac{(\mathbf{i}-\mu_n)^2}{2\sigma_n^2}) \ , \\ W &= \sum_{n=1}^{N} w_n \end{aligned} \tag{4.6}$$

The number of modes needed for a sufficiently flexible representation varies depending on application. For the purposes of this work it is assumed that a single dominant mode in the distribution is sufficient to model the nominal appearance of each scene point, with slight

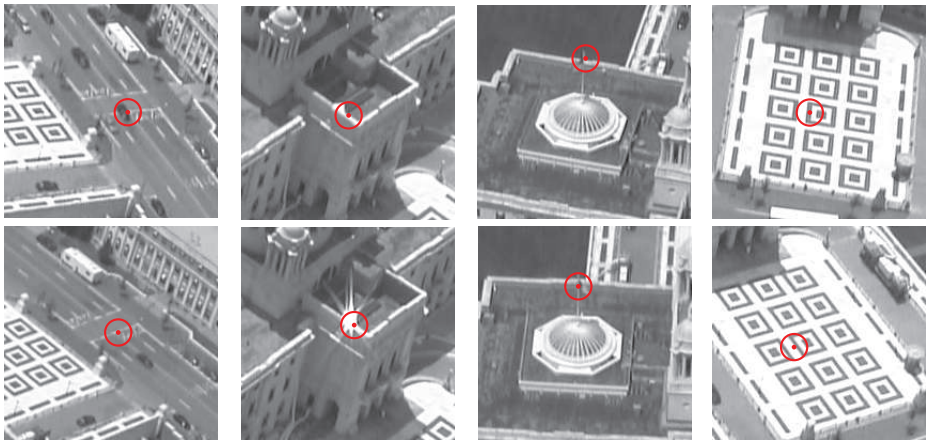


Figure 4.6: Examples of appearance variation from the "capitol" video sequence. From left to right: foreground objects in motion, specular reflection, motion due to wind, camera misregistration.

variations in appearance due to viewing angle, illumination, and color calibration accounted for by the normal variance of the mode. The other modes in the distribution serve to account for any other variation. Appearance changes may occur for many reasons, including: occlusion by moving objects in the scene (e.g. vehicles and pedestrians), small motions of the scene components themselves (e.g. foliage motion due to wind), shadows, (e.g. those cast by moving foreground objects or clouds), specularities, and camera misregistration.

4.3.1 Storage Savings

The storage savings over a fixed size voxel grid made possible by the octree implementation of course vary depending on the scene being modeled, but can be easily examined for the simple case of a completely planar scene. Assuming that the volume to be modeled is a cube, the N^3 voxels are required to represent the scene, where N is the linear resolution (voxels per side length of modeled volume). In the case of the octree, the model is refined iteratively by removing each cell intersected by the plane and replacing it with eight children cells, so the linear resolution at a given iteration i is 2^i. The number of cells intersecting the plane at iteration i is $(2^i)^2$ (or 4^i), meaning the number of additional cells needed at iteration $i+1$ is $7 \cdot (4^i)$. The total number of cells at iteration i can be computed as a geometric series:

$$N = 1 + \sum_{j=1}^{i} 7 \cdot 4^{j-1} = 1 + \frac{7 \cdot (4^i - 1)}{3} \tag{4.7}$$

Written in terms of linear resolution N,

$$ncells(N) = 1 + 7 \cdot \frac{(N^2 - 1)}{3} \tag{4.8}$$

Figure 4.7 shows a plot of the number of cells required for the fixed size voxel grid and octree model for a given linear resolution. Assuming a constant sized storage space allocated per cell/voxel, the fixed-grid voxel model requires $\mathrm{O}\left(N^3\right)$ storage space, and the octree model $\mathrm{O}\left(N^2\right)$.

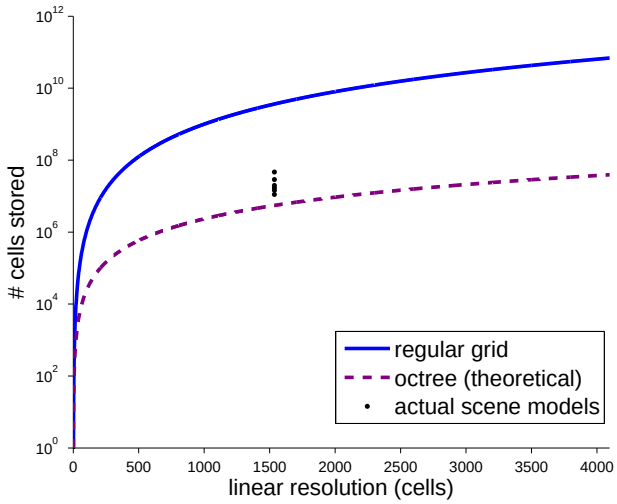


Figure 4.7: The theoretical number of cells stored for a planar scene are plotted based on the linear resolution of the model. Data points for actual (non-planar) scene models are also shown.

4.4 Piecewise-Linear Model

Although the focus of the remainder of this thesis is on the presented piecewise-constant implementation, use of the continuous probabilistic model is not limited to this implementation in general. As an example of a potential alternate implementation, the groundwork for a piecewise-linear model is briefly presented.

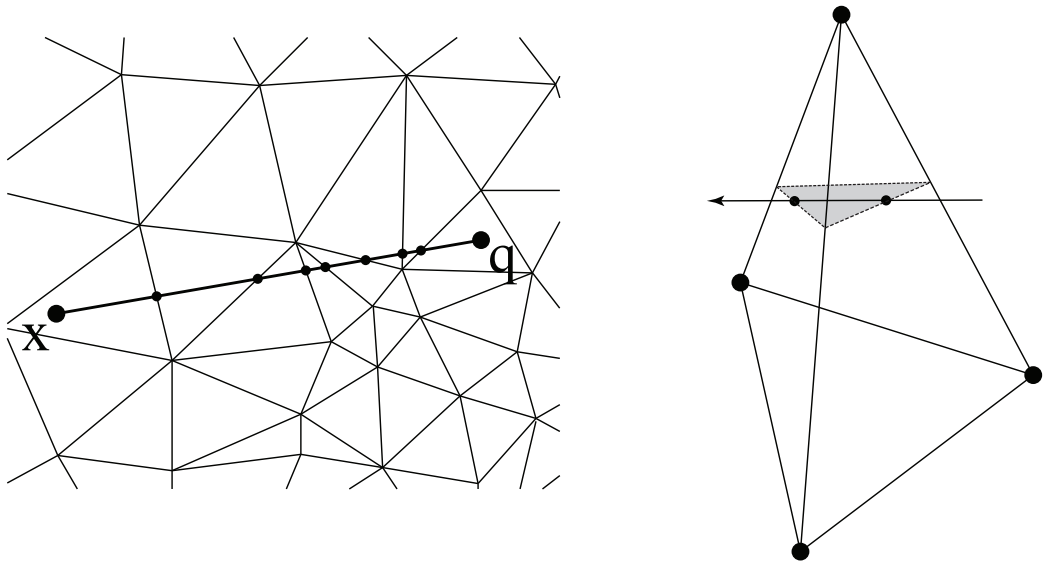


Figure 4.8: A camera ray parameterized by s passes through the tetrahedral mesh. A 2-d cross section is shown on the right. On the left, the ray is shown piercing a single tetrahedron in 3-d.

4.4.1 Tetrahedral Mesh Representation

In a tetrahedral volume mesh, sample values may be stored at each of the vertices and linearly interpolated at any point within the volume represented by the mesh using barycentric coordinates [13]. The barycentric coordinates λ_i of a point $\mathbf{x}$ relative to the four vertices $\mathbf{v}_i$, $i \in [1 \ldots 4]$ of the tetrahedron containing $\mathbf{x}$ are computed as follows:

$$\begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} \mathbf{v}_1 - \mathbf{v}_4 & \mathbf{v}_2 - \mathbf{v}_4 & \mathbf{v}_3 - \mathbf{v}_4 \end{pmatrix}^{-1} (\mathbf{x} - \mathbf{v}_4) \tag{4.9}$$

The fourth coordinate, λ_4, is computed as $1 - \lambda_1 - \lambda_2 - \lambda_3$. The interpolated value of the occlusion density at $\mathbf{x}$ is computed using the barycentric coordinates as weights.

$$\alpha(\mathbf{x}) = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix}^T \begin{pmatrix} \alpha(\mathbf{v}_1) \\ \alpha(\mathbf{v}_2) \\ \alpha(\mathbf{v}_3) \\ \alpha(\mathbf{v}_4) \end{pmatrix} \tag{4.10}$$

Note that it is straightforward but not strictly necessary to use a tetrahedral mesh; Warren presented a method for computing barycentric coordinates for general convex polyhedrons in 1996 [71]. Interpolation of the appearance model is potentially less straightforward, as the weighted sum in Equation 4.10 must be applied to the color probability distributions $p_A(\mathbf{i}, \mathbf{v}_i)$ defined at the vertices. Given the mixture of Gaussians representation described in Section 4.3, one solution is to represent the distribution using $4N$ modes, where N is the number of modes in the distributions stored at each of the vertices. The distribution at point x would then be computed as:

$$p_A(\mathbf{i}, \mathbf{x}) = \sum_{v=1}^{4} \frac{\lambda_v}{W_v} \sum_{n=1}^{N} \frac{1}{\sigma_{n_v}\sqrt{2\pi}} \exp(-\frac{(\mathbf{i} - \mu_{n_v})^2}{2\sigma_{n_v}^2}) \tag{4.11}$$

In addition to the accuracy that a piecewise-linear approximation provides over a piecewise-constant approximation, some added flexibility is provided by the tetrahedral mesh data structure. While the cells of an octree can be made arbitrarily fine, their locations are constrained by the regular subdivision into octants. This is not the case with the tetrahedral cells of a mesh: their vertices can be repositioned if doing so will lead to a more efficient representation. This added flexibility comes at the cost of explicitly storing the positions of each vertex in the mesh; the positions of the octree cells are computed implicitly based on the cell index and location of the octree origin.

4.4.2 Visibility and Rendering

The visibility computation of Equation 3.13 can be simplified for the piecewise-linear approximation in a similar manner as that of the piecewise-constant approximation. The ray is divided into segments based on transition points between tetrahedra of the mesh (Figure 4.9). The values at the transition points s_i are interpolated based on the equations in Section 4.4.1. The visibility probability of a transition point $\mathbf{x}(s_i)$ along the ray can then be calculated as follows:

$$\text{vis}_i = \exp\left(-\frac{1}{2}\sum_{j=0}^{i-1}(\alpha_j + \alpha_{j+1})(s_{j+1} - s_j)\right) \tag{4.12}$$

Using the linearly interpolated function $\alpha(s)$ and $\text{vis}(s)$, the probability density $\omega(s)$ of

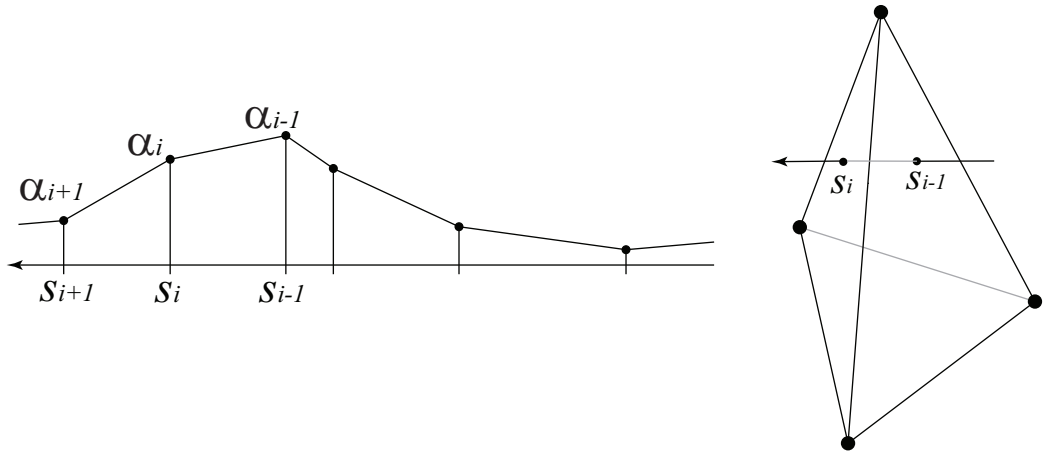


Figure 4.9: The function α as a function of the ray parameter s is shown, linearly interpolated between values of s_i. The values s_i correspond to points at which the ray passes from one tetrahedron to another, and the function values at the points are themselves interpolated based on the values at the tetrahedron's vertices.

the imaged point along a camera ray can also be computed using the equations presented in Section 3.3.1. The weights Ω_i can then also be computed for each appearance model $p_A(\mathbf{i}, s_i)$ in order to compute the total distribution $p_T(\mathbf{i})$.

$$\Omega_i = \frac{1}{2} \int_{s_{i-1}}^{s_{i+1}} \alpha(s)\,\mathrm{vis}(s) ds \tag{4.13}$$

The values of Ω_i can be expanded in terms of the values α_i and vis_i but is omitted here since the remainder of the thesis focuses on the piecewise linear implementation.

4.5 Conclusion

In order to be useful, a theoretical model must lend itself to efficient and practical implementations. Two such implementations have been presented in this chapter based on the continuous model proposed in Chapter 3 and varying levels of simplifying assumptions about the underlying occlusion density and appearance functions. The remainder of this thesis will focus exclusively on the octree-based implementation using the piecewise-constant assumption presented in Section 4.1.

Chapter 5

Reconstruction Algorithms

A continuous probabilistic scene model was introduced in Chapter 3, and an implementation based on a piecewise-constant approximation in Chapter 4. In this chapter, methods for reconstructing and updating an instance of the model based on calibrated images are presented. There are two distinct classes that a reconstruction algorithm may fall into. The first class is that of *online* algorithms, and the second is that of (*offline*) global optimization algorithms. An online algorithm is one that processes input serially, without access to previous or future input data at a given time step. Online reconstruction algorithms maintain the current state of the model and update the state using one image at a time. These algorithms are useful and sometimes necessary when the image size is very large (e.g. satellite imagery) or when new image data is continuously becoming available (e.g. a live video stream). Global optimization algorithms, on the other hand, make use of (potentially) all available data and compute a reconstruction that attempts to satisfy the constraints imposed by all input simultaneously. These algorithms are in general costlier in terms of memory and computation than the online variety, but have the obvious advantage of access to all available data. In cases such as an indefinitely active live video stream where it is not practical to store all available input, a fixed-sized subset of image data can be maintained and used for model reconstruction. For the purposes of this thesis, any algorithm which operates using more than one image at time is classified as a global optimization algorithm.

The remainder of the chapter is laid out as follows: In Sections 5.1 and 5.2, online and global reconstruction algorithms, respectively, for the piecewise-constant implementation of

the continuous probabilistic model are presented. In Section 5.3, a method for estimating an implicit surface based on the probabilistic model is presented for the purpose of post-processing the reconstructed models, and finally the chapter is concluded in Section 5.4.

5.1 Online Updating

As discussed in the chapter introduction, an online reconstruction algorithm may be preferable or even necessary in some circumstances. An online method for updating the probabilistic model presented in Chapter 4 is presented here. The formulation closely parallels that of Pollard and Mundy [53], which is discussed in Section 2.4.4.

Pollard and Mundy's online Bayesian update equation (Equation 2.8) is used to determine the posterior probabilities of a series of voxels along a camera ray, given their prior probabilities and an image observation. Rather than a camera ray intersecting a series of voxels, the equation can be thought of as being applied to a series of N intervals of equal length ℓ along a ray parameterized by s, the distance from the camera center. The probability $P(\mathcal{X}_i)$ of the voxel i being a surface voxel is replaced by $P(\mathcal{Q}_i^\ell)$, the occlusion probability defined in Equation 3.4 of the ith segment. The visibility probability of the ith segment is denoted by the term vis_i.

$$\text{vis}_i \equiv \prod_{j=0}^{i-1}(1 - P(\mathcal{Q}_j^\ell)) \tag{5.1}$$

The posterior probability $P(\mathcal{Q}_i^\ell|\mathcal{D})$ is computed as follows:

$$P(\mathcal{Q}_i^\ell|\mathcal{D}) = P(\mathcal{Q}_i^\ell)\frac{P(\mathcal{D}|\mathcal{Q}_i^\ell)}{P(\mathcal{D})} \tag{5.2}$$

Expanding the numerator and denominator terms,

$$P(\mathcal{Q}_i^\ell|\mathcal{D}) = P(\mathcal{Q}_i^\ell)\frac{\sum_{j=0}^{i-1} P(\mathcal{Q}_j^\ell)\,\text{vis}_j\,P(\mathcal{D}|s_j < v \le s_{j+1}) + \text{vis}_i\,P(\mathcal{D}|s_i < v \le s_{i+1})}{\sum_{j=0}^{N-1} P(\mathcal{Q}_j^\ell)\,\text{vis}_j\,P(\mathcal{D}|s_j < v \le s_{j+1})} \tag{5.3}$$

For convenience, the term pre_i, which represents the probability of the observation taking into account segments 0 through $i-1$ only, is defined:

$$\text{pre}_i \equiv \sum_{j=0}^{i-1} P(\mathcal{Q}_j^\ell)\,\text{vis}_j\,P(\mathcal{D}|s_j < v \leq s_{j+1}) \tag{5.4}$$

Equation 5.3 can then be re-written as:

$$P(\mathcal{Q}_i^\ell|\mathcal{D}) = P(\mathcal{Q}_i^\ell)\frac{\text{pre}_i + \text{vis}_i\,P(\mathcal{D}|s_i < v \leq s_{i+1})}{\text{pre}_\infty} \tag{5.5}$$

where pre_∞ represents the total probability of the observation based on all pixels along the ray. Pollard and Mundy assume that the observed imaged data is produced by a voxel along the camera ray and do not consider the possibility that the observed intensity was produced by a point beyond the region of modeled space. In this thesis an additional term $\text{vis}_\infty\,P(\mathcal{D}|\nexists v)$ is introduced to the denominator of the update equation which accounts for this possibility. The probability of the ray passing unoccluded through the model is represented by vis_∞ and is computed based on Equation 5.6. The term $P(\mathcal{D}|\nexists v)$ represents the probability of the observed intensity given that the ray passes unoccluded through, and can be thought of as a "background" appearance model.

$$\text{vis}_\infty \equiv \prod_{i=0}^{N-1}[1 - P(\mathcal{Q}_i^\ell)] \tag{5.6}$$

The new equation for the posterior occlusion probability is then written:

$$P(\mathcal{Q}_i^\ell|\mathcal{D}) = P(\mathcal{Q}_i^\ell)\frac{\text{pre}_i + \text{vis}_i\,P(\mathcal{D}|s_i < v \leq s_{i+1})}{\text{pre}_\infty + \text{vis}_\infty\,P(\mathcal{D}|\nexists v)}\quad . \tag{5.7}$$

Throughout the remainder of this chapter, the ratio of the posterior probability to the prior probability is denoted by the variable β for convenience.

$$\beta_i = \frac{\text{pre}_i + \text{vis}_i\,P(\mathcal{D}|s_i < v \leq s_{i+1})}{\text{pre}_\infty + \text{vis}_\infty\,P(\mathcal{D}|\nexists v)} \tag{5.8}$$

$$P(\mathcal{Q}_i^\ell|\mathcal{D}) = \beta_i P(\mathcal{Q}_i^\ell) \tag{5.9}$$

Equation 5.9 computes the posterior occlusion probability of a segment along a camera ray given the observation $\mathcal{D}$. What is needed for the continuous model is an analogous

method for updating the continuous occlusion density values along the ray. A straightforward conversion to the continuous case can be arrived at by subdividing the segment into N infinitesimally small segments.

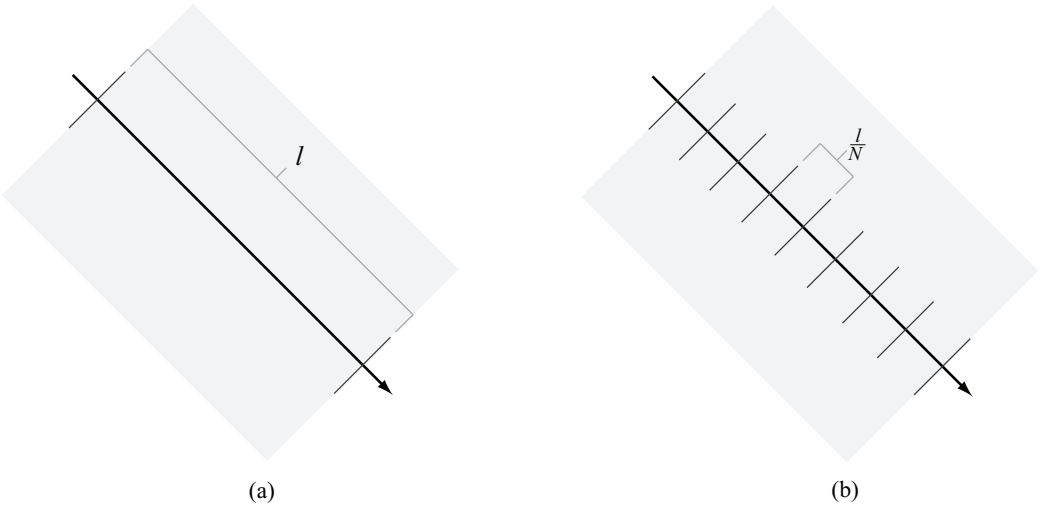


Figure 5.1: (a) A viewing ray passes through a segment of uncertain geometry for length ℓ. (b) Equivalently, the viewing ray passes through N regions of length $\frac{\ell}{N}$.

The probability of an entering ray passing thourgh the segment unoccluded is the complement of $P(\mathcal{Q}^{\ell})$, or $(1-P(\mathcal{Q}^{\ell}))$. Equivalently, the ray can be described as passing through N consecutive intervals of length $\frac{\ell}{N}$.

$$1 - P(\mathcal{Q}^{\ell}) = \left[1 - P(\mathcal{Q}^{\frac{\ell}{N}})\right]^{N} \tag{5.10}$$

Solving for the probability $P(\mathcal{Q}^{\frac{\ell}{N}})$ in terms of $P(\mathcal{Q}^{\ell})$ gives:

$$P(\mathcal{Q}^{\frac{\ell}{N}}) = 1 - \left[1 - P(\mathcal{Q}^{\ell})\right]^{\frac{1}{N}} \tag{5.11}$$

For a given point lying in the ith sub-interval, β_i does not change as the number of intervals is increased by subdivision using equation 5.11. This can be shown in three steps:

1. The total probability contributed by a set of M consecutive identical segments is identical to the probability contributed by the original segment.

2. The value of β for the first cell in the set is identical to the value of β for the original segment.

3. The value of β is identical for all cells in the set.

The first step can be shown by again considering a segment of length ℓ with occlusion probability $P(\mathcal{Q}^{\ell})$, and visibility probability vis_0. The total probability contribution Ω_{ℓ} of the segment is equal to $P(\mathcal{Q}^{\ell})\,\text{vis}_0$. If the segment is divided into N sub-segments, the total probability contribution of the sum can be shown to be equal to Ω_{ℓ}.

$$\Omega_{\ell} = P(\mathcal{Q}^{\ell})\,\text{vis}_0 = \sum_{i=0}^{N-1}[P(\mathcal{Q}^{\frac{\ell}{N}})\,\text{vis}_i] \tag{5.12}$$

By substituting the definition of vis_i, the equation can be simplified as follows.

$$\begin{aligned}
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= P(\mathcal{Q}^{\frac{\ell}{N}})\sum_{i=0}^{N-1}[\text{vis}_0\prod_{j=0}^{i-1}(1-P(\mathcal{Q}^{\frac{\ell}{N}}))] \\
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= P(\mathcal{Q}^{\frac{\ell}{N}})\,\text{vis}_0\sum_{i=0}^{N-1}(1-P(\mathcal{Q}^{\frac{\ell}{N}}))^i \\
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= P(\mathcal{Q}^{\frac{\ell}{N}})\,\text{vis}_0\,\frac{(1-P(\mathcal{Q}^{\frac{\ell}{N}}))^N-1}{(1-P(\mathcal{Q}^{\frac{\ell}{N}}))-1} \\
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= -(\text{vis}_0[(1-P(\mathcal{Q}^{\frac{\ell}{N}}))^N-1])
\end{aligned} \tag{5.13}$$

By substituting the definition of $P(\mathcal{Q}^{\frac{\ell}{N}})$ based on Equation 5.11, Equation 5.13 can be further simplified.

$$\begin{aligned}
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= -\,\text{vis}_0[(1-(1-(1-P(\mathcal{Q}^{\ell}))^{\frac{1}{N}}))^N-1] \\
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= -\,\text{vis}_0[((1-P(\mathcal{Q}^{\ell}))^{\frac{1}{N}})^N-1] \\
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= -\,\text{vis}_0(1-P(\mathcal{Q}^{\ell}))+\text{vis}_0 \\
P(\mathcal{Q}^{\ell})\,\text{vis}_0 &= P(\mathcal{Q}^{\ell})\,\text{vis}_0
\end{aligned} \tag{5.14}$$

Based on this proof, it can be trivially seen that the value of pre_i which encompasses a given interval of segments does not change as the segments are subdivided. Likewise, the

visibility probability vis_i also remains constant by definition of the subdivision equation (Equation 5.11). Assuming that the appearance probability $P(\mathcal{D}|s_i < v \leq s_{i+1})$ of a subdivided segment is equal to the appearance probability of the original segment, it is clear that β_i for the first subdivided segment is equal to β of the original segment since all terms remain constant.

It can then be seen that all subdivided intervals must have the same β value as the original segment by proving that for two consecutive segments with identical appearance probabilities $P(\mathcal{D}|s_j < v \leq s_{j+1})$ and occlusion probabilities $P(\mathcal{Q})$, β is identical. The proof is as follows.

$$\begin{aligned}
\beta_{i+1} &= \frac{\text{pre}_{i+1} + \text{vis}_{i+1}(1 - P(\mathcal{Q}))P(\mathcal{D}|s_i < v \leq s_{i+1})}{pre_\infty} \\
\beta_{i+1} &= \frac{\text{pre}_i + \text{vis}_i\, P(\mathcal{Q})P(\mathcal{D}|s_i < v \leq s_{i+1}) + \text{vis}_i(1 - P(\mathcal{Q}))P(\mathcal{D}|s_i < v \leq s_{i+1})}{pre_\infty} \\
\beta_{i+1} &= \frac{\text{pre}_i + \text{vis}_i\, P(\mathcal{D}|s_i < v \leq s_{i+1})(P(\mathcal{Q}) + 1 - P(\mathcal{Q}))}{pre_\infty} \\
\beta_{i+1} &= \frac{\text{pre}_i + \text{vis}_i\, P(\mathcal{D}|s_i < v \leq s_{i+1})}{pre_\infty} \\
\beta_{i+1} &= \beta_i
\end{aligned} \tag{5.15}$$

A continuous form of Equation 5.9 can then be derived by dividing both sides of Equation 5.9 by ℓ and taking the limit as the number of subdivisions N grows and ℓ approaches an infinitesimal value ds.

$$\begin{aligned}
\lim_{\ell \to 0} \frac{P(\mathcal{Q}_i^\ell|\mathcal{D})}{\ell} &= \lim_{\ell \to 0} \beta_i \frac{P(\mathcal{Q}_i^\ell)}{\ell} \\
\frac{P(\mathcal{Q}_i^{ds}|\mathcal{D})}{ds} &= \beta_i \frac{P(\mathcal{Q}_i^{ds})}{ds}
\end{aligned} \tag{5.16}$$

Substituting in $\alpha(s)$ based on the definition given in Equation 3.10 and expanding provides the full update equation. Using the continuous form of the pre term

$$\text{pre}(s) = \int_0^s \alpha(s')\,\text{vis}(s')P(\mathcal{D}|v = s')ds' \tag{5.17}$$

the update equation is written:

$$\beta(s) = \frac{\text{pre}(s) + \text{vis}(s)P(\mathcal{D}|v = s)}{\text{pre}(\infty) + \text{vis}(\infty)P(\mathcal{D}|\nexists v)} \tag{5.18}$$

$$\alpha(s|\mathcal{D}) = \alpha(s)\beta(s) \tag{5.19}$$

Equations 5.18 and 5.19 describe the full online update equation for the continuous probabilistic model, a trivial extension of the discrete update equation to the continuous case where occlusion probabilities are replaced by occlusion densities. In order to be of practical value, however, a form which is suited to the piecewise-constant model must be derived.

5.1.1 Adaptation for Piecewise-Constant Model

The systems described in this thesis use a piecewise-constant approximation to the continuous model as described in section 4.1. The piecewise-constant model allows the update equations (Equations 5.18 and 5.19) to be simplified by converting the integrals of constant valued segments in the pre and vis terms to summations. It is assumed that each segment i has associated with it a constant occlusion density value and a single appearance model, i.e. appearance is constant in a given segment. The vis_i and pre_i terms of the discrete model are redefined as follows for the piecewise-constant model.

$$\begin{aligned} \text{vis}_i &\equiv \exp(-\int_0^{s_i} \alpha(s')ds') \\ \text{vis}_i &= \exp(-\sum_{j=0}^{i-1} \alpha_j \ell_j) \end{aligned} \tag{5.20}$$

$$\begin{aligned} \text{pre}_i &\equiv \int_0^{s_i} \alpha(s)\,\text{vis}(s)P(\mathcal{D}|v = s)ds \\ \text{pre}_i &= \sum_{n=0}^{i-1}(1 - e^{\alpha_n \ell_n})\,\text{vis}_n\,P(\mathcal{D}|s_n < v \leq s_{n+1}) \end{aligned} \tag{5.21}$$

The probability $P(\mathcal{D}|s_n < v \leq s_{n+1})$ is represented by the distribution $p_{A_n}(\mathbf{i})$ at the value of the intensity $\mathbf{i}$ associated with the camera ray. Likewise, the probability $P(\mathcal{D}|\nexists v)$

is represented by a "background" intensity distribution $p_{A_\infty}(\mathbf{i})$. Substituting into Equation 5.21 gives:

$$pre_i = \sum_{n=0}^{i-1} (1 - e^{\alpha_n \ell_n}) \, \mathrm{vis}_n \, p_{An}(\mathbf{i}) \tag{5.22}$$

Using the new definitions for vis_i and pre_i, the β_i value can be computed using the analogous form of Equation 5.8.

$$\beta(s_i) = \frac{\mathrm{pre}_i + \mathrm{vis}_i \, p_{A_i}(\mathbf{i})}{\mathrm{pre}_\infty + \mathrm{vis}_\infty \, p_{A_\infty}(\mathbf{i})} \tag{5.23}$$

Because vis_i represents the visibility probability of the first point in the ith segment $\mathbf{x}(s_i)$ and pre_i represents the observation probability taking into account all points on the ray prior to $\mathbf{x}(s_i)$, this equation gives the update ratio for $\mathbf{x}(s_i)$. Using the proof of Equation 5.15 and extending to the continuous case by taking the limit as N becomes infinite, however, it can be seen that for a segment of constant occlusion density and appearance, $\beta(s)$ must be constant within the segment as well. Thus the constant-valued occlusion density α_i for segment i can be updated to the posterior value α_i' using Equation 5.23.

$$\beta_i \equiv \beta(s_i) \tag{5.24}$$

$$\alpha_i' = \alpha_i \beta_i \tag{5.25}$$

5.1.2 Updating the Scene Model

The update equations described in Section 5.1.1 give a posterior value of the occlusion density α_i for each constant-valued segment i along a camera ray. In the piecewise-constant model, the transition points $\mathbf{x}(s_i)$ correspond to locations where the camera ray exits the $(i-1)$th cell and enters the ith cell along the ray. The updating of the occlusion density and appearance model cannot, however, be performed independently for each camera ray. This is because multiple camera rays will pass through a single cell and will not, in general, agree on the values of the updated occlusion densities. To resolve this conflict, each camera

ray's contribution to the updated value is weighted based on the corresponding segment length ℓ_i.

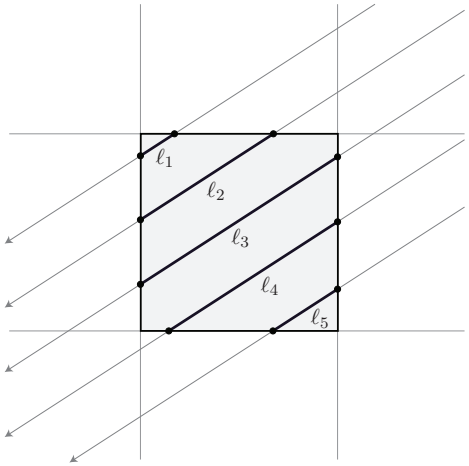


Figure 5.2: The update equations are defined per ray, but a given block of constant occlusion density may contain multiple ray segments. The updated occlusion density value is determined by averaging the contributions from each ray, weighted by their corresponding segment lengths ℓ_i.

The appearance of each cell is represented by a mixture of a fixed number of Gaussian distributions (Section 4.3) and is updated with each new image based on the adaptive update equations proposed by Stauffer and Grimson [68] and used by Pollard and Mundy [53]. Each update to the appearance model is weighted by the visibility probability vis_i of the cell. The appearance models for each visible cell are also updated once per image, using the mean image value of the projected cell. When computing the mean, the image pixel values are also weighted based on their respective camera ray segment lengths.

5.1.3 Adaptive refinement

Upon initialization of the model, each cell corresponds to a leaf at constant depth in the octree, i.e. the volume is regularly sampled. As more information is incorporated into the model, the sampling of regions with high occlusion density may be refined. The proposed implementation refines a cell when its maximum occlusion probability $P(\mathcal{Q}_{\text{max}_i})$ reaches a global threshold. The maximum occlusion probability of cell i is a function of the longest path through the cell and the cell's occlusion density.

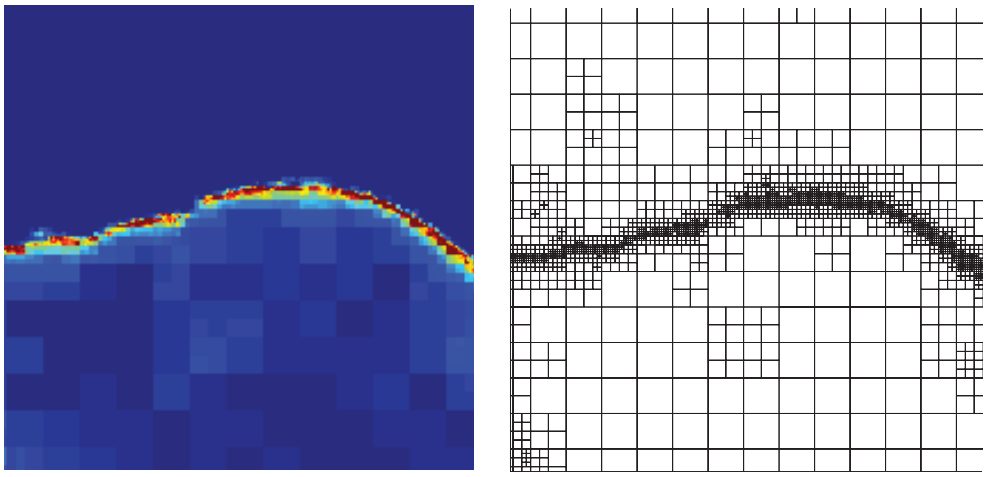


Figure 5.3: A cropped 2-d slice from the "dinoRing" model. Left: The occlusion density values. Right: The structure of the octree.

$$P(\mathcal{Q}_{max_i}) = 1 - \exp(-\ell_{max_i}\alpha_i) \tag{5.26}$$

The occlusion densities of the refined cells are initialized to the value of their parent cell, while the appearance models are reset to a "blank" state to avoid propagating low frequency image information down to the finer levels. This process is executed after each new image update to the model.

5.2 Enforcing Global Consistency

While online reconstruction algorithms can be essential for certain modes of operation, they must deal with the disadvantage of not being able to access all available information simultaneously. For this reason, a second reconstruction algorithm is presented which iteratively optimizes the occlusion density and appearance model parameters for each cell using all available information from a set of input images.

It is assumed that the observed data $\boldsymbol{\mathcal{D}} = \{\mathcal{D}_0, \mathcal{D}_1 \ldots \mathcal{D}_N\}$ is given as a set of N images along with the corresponding set of cameras. The probability of each observed image $P(\mathcal{D}_n)$ is assumed to be independent of all other observation probabilities. The posterior

probability of a given constant-valued cell of the octree depends only on the $M \leq N$ images for which the cell projects within image bounds. In general, there is an arbitrarily complex network of dependence between the cells in the scene which may include circular dependencies, as pointed out by Bhotika et al. [5]. Note that this is not the case for the online updating scheme discussed in Section 5.1. In the case of online updating using a single image, each cell is dependent only on the other cells along the camera rays which pass through it, resulting in a cycle-free dependency graph. In order to produce a computationally tractable global reconstruction algorithm, however, the parameters of each cell are considered independently, with the assumption that all of the other cells' parameters are fixed. The cells are updated incrementally in this way until convergence is achieved over the entire scene. Taking advantage of both the data and cell independence assumptions, the posterior occlusion density for a given cell can be computed as the product of the individual posterior occlusion densities.

$$\beta_i = \prod_{m=0}^{M} \frac{\text{pre}_{im} + \text{vis}_{im} p_{Ai}(\mathbf{i_m})}{\text{pre}_{\infty m} + \text{vis}_{\infty m} p_{A\infty}(\mathbf{i_m})} \tag{5.27}$$

This update equation is based on the implicit assumption that all occlusion probabilities and appearance models in the model are correct. Until the model converges, this assumption is false and so the resulting multiplier β_i must be damped to prevent the model from jumping to or oscillating between erroneous states as the visibility probabilities change at each iteration. The damped multiplier $\hat{\beta}$ is computed as a function of the free parameter κ as follows:

$$\hat{\beta}_i = \frac{(\beta_i + \kappa)}{(\kappa\beta_i + 1)} \quad , \qquad 0 < \kappa < 1 \tag{5.28}$$

The damping function is chosen for its simplicity (controlled by a single parameter κ) and the fact that it imposes straight-forward minimum and maximum $\hat{\beta}$ values of κ and $\frac{1}{\kappa}$, respectively. Figure 5.4 illustrates how the damped update ratio $\hat{\beta}$ varies with β for various values of κ. Figure 5.5 shows plots of the occlusion density value of a surface point as a function of the batch update iteration number for various κ values. The undamped ($\kappa = 0.0$) case shows an initial oscillation and eventual convergence to a relatively low value, while the damped cases show a continual increase.

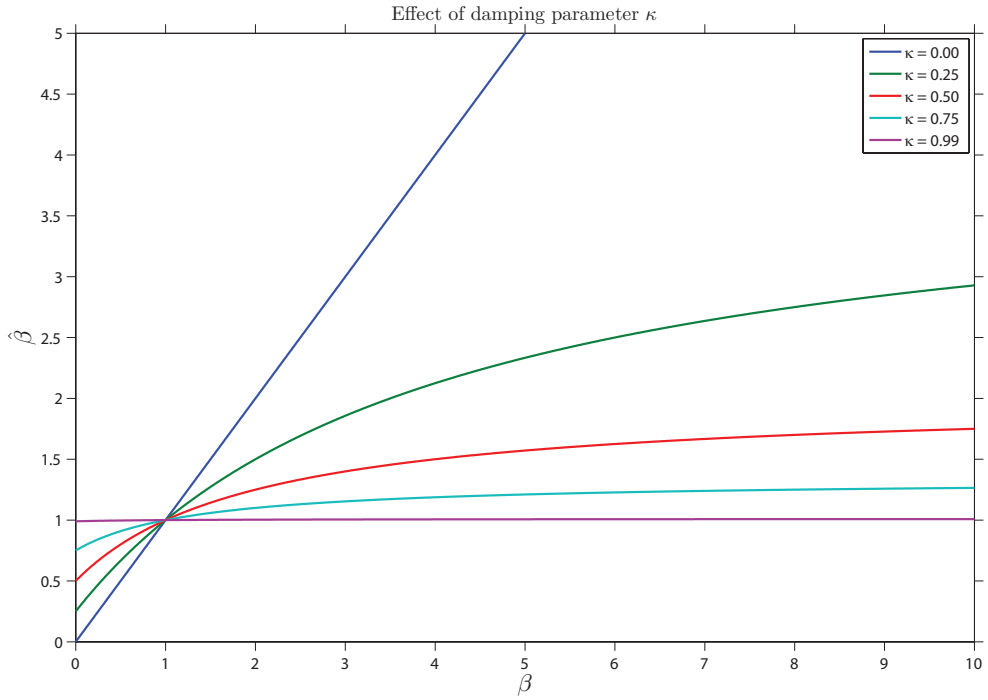


Figure 5.4: The damped update ratio $\hat{\beta}$ is shown as a function of the undamped value β for various values of the damping parameter κ.

5.2.1 Appearance calculation

Because the updating algorithm uses information from all images simultaneously, appearance modeling is not limited to online updating algorithms such as those used in Section 5.1. Instead, an expectation maximization (EM) algorithm is used to compute the maximum likelihood parameters of the appearance independently for each cell at each timestep. A single Gaussian distribution is used to model the probability density $p_{A_i}(\mathbf{i}_m)$ with two free parameters: the mean μ and variance σ^2. It is assumed that each observed intensity $\mathbf{i}_m$ is produced by one of three potential sources:

- $\mathcal{J}_m \equiv$ Observation m is produced by a point within the cell.
- $\mathcal{K}_m \equiv$ The cell is not visible; observation m is generated by a point "in front of" the cell.
- $\mathcal{L}_m \equiv$ The cell is visible, but observation m is produced by a point "behind" the cell.

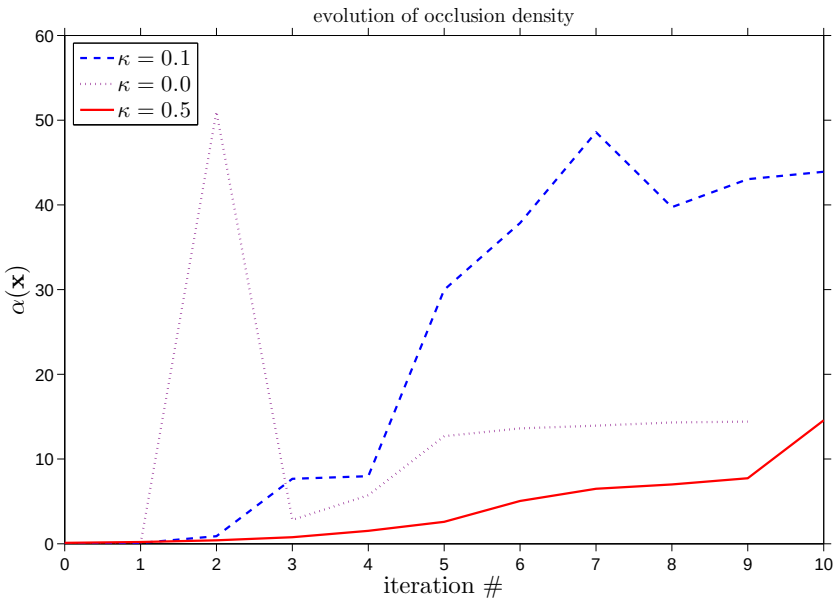


Figure 5.5: The occlusion density value for a surface point as a function of iteration number with various amounts of damping. For the undamped case, the model became too large after the ninth iteration due to excessive subdividing of octree cells and was terminated.

For the purposes of appearance estimation the cell is assumed to be occluding, and so $P(\mathcal{L}_m) = 0$. This assumption is valid because if the cell is not occluding, the appearance model is not relevant. The "expectation" step consists of computing the probabilities $P(\mathcal{J}_m)$ and $P(\mathcal{K}_m)$, which are calculated as follows:

$$P(\mathcal{J}_m) = \text{vis}_i \, p_{Ai}(\mathbf{i}_m) \tag{5.29}$$

$$P(\mathcal{K}_m) = \text{pre}_i \tag{5.30}$$

The "maximization" step then consists of computing new values for μ and σ^2 using the observed intensities $\mathbf{i}_m$ and the corresponding probabilities w_m as weights.

$$w_m = \frac{P(\mathcal{J}_m)}{P(\mathcal{J}_m) + P(\mathcal{K}_m))} \tag{5.31}$$

The parameters are computed using the standard definitions of weighted sample mean and weighted sample variance s^2. Note that the sum W of observation weights may be interpreted as the expected number of observations.

$$\mu = \sum_{m=0}^{M} \frac{w_m}{W} \mathbf{i}_m \quad , \qquad W = \sum_{m=0}^{M} w_m \tag{5.32}$$

$$s^2 = \frac{1}{1 - W_2} \sum_{m=0}^{M} \frac{w_m}{W} (\mathbf{i}_m - \mu)^2 \quad , \qquad W_2 = \sum_{m=0}^{M} \left(\frac{w_m}{W} \right)^2 \tag{5.33}$$

The sample variance s^2 is clipped to a fixed minimum value $s^2_{\min}$ if it falls below in order to help prevent the EM algorithm from converging to a degenerate solution dominated by a single observation. If the computed values of μ and s^2 do not differ from the previous values by more than a predetermined tolerance value, the computation is assumed to have converged. Otherwise, the algorithm iterates using the updated values as input to the "expectation" step.

Once convergence is reached, an estimate of the true variance σ^2 must be estimated based on the sample variance s^2. The computation of the sample variance in Equation 5.33 can itself be interpreted as a sample from a chi-square distribution with $M - 1$ degrees of freedom based on Cochran's theorem [72].

$$\frac{(M-1)s^2}{\sigma^2} \sim \chi^2_{M-1} \tag{5.34}$$

The practical result of Equation 5.34 is that the sample variances computed in Equation 5.33 tend to underestimate the true appearance variances σ^2 for small sample sizes (Figure 5.6). This underestimation can lead to the occlusion density of cells with few observations growing larger than expected due to the overestimate of $p_{A_i}(\mathbf{i}_m)$ caused by the underestimated variance. In order to prevent this, an estimate $\hat{\sigma}^2$ of σ^2 is computed such that

$$P(\hat{\sigma}^2 < \sigma^2) = \varepsilon \quad , \tag{5.35}$$

where ε is a parameter which specifies the probability of underestimating the true variance σ^2. The estimated variance $\hat{\sigma}^2$ is computed as the sample variance s^2 multiplied by a scale factor a which is determined by the parameter ε.

$$P(as^2 < \sigma^2) = \varepsilon \tag{5.36}$$

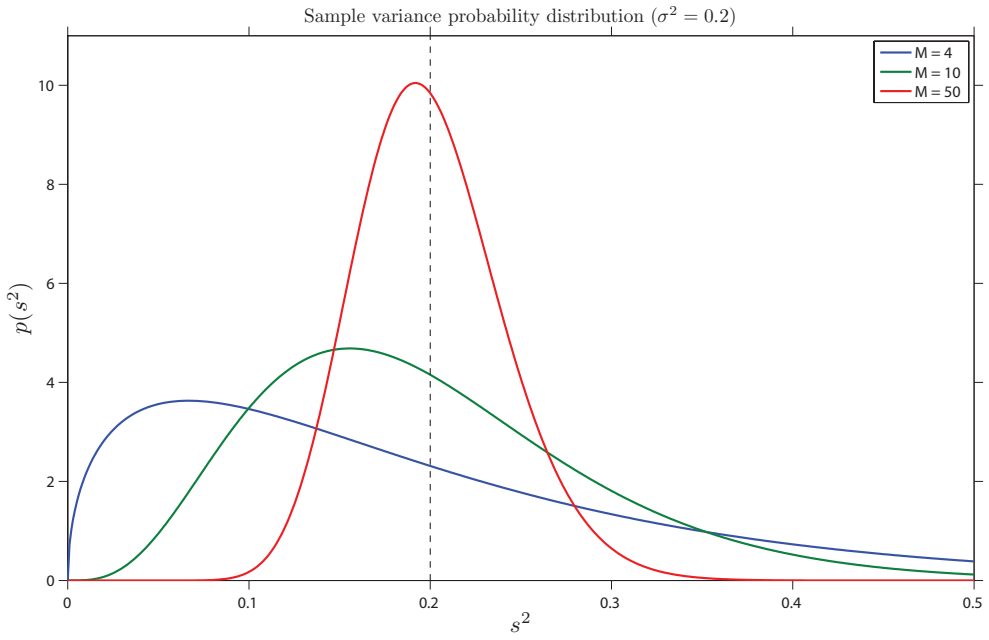


Figure 5.6: The probability distribution of the sample variance is plotted for various sample sizes M. For small sample sizes the variance tends to be underestimated. As the sample size grows, the most probable sample variance approaches the true variance ($\sigma^2 = 0.2$ in this case).

$$\hat{\sigma}^2 = as^2 \tag{5.37}$$

Equation 5.36 can be rewritten using the cumulative distribution function of the chi-squared distribution.

$$\frac{\gamma((M-1)/2, (as^2)/2)}{\Gamma((M-1)/2)} = \varepsilon \tag{5.38}$$

The functions $\Gamma(k, x)$ and $\gamma(k, x)$ are the regularized Gamma function and lower incomplete Gamma function, respectively. By solving Equation 5.38 for a as a function of the expected number of observations W, the scale factors with which to multiply the sample variances to achieve the equality specified in Equation 5.36 are computed. Figure 5.7 shows the scale factors as a function of W for $\varepsilon = 0.1$, 0.25, and 0.75. The estimated variance can then be computed using the sample variance and scale factor a using Equation 5.37.

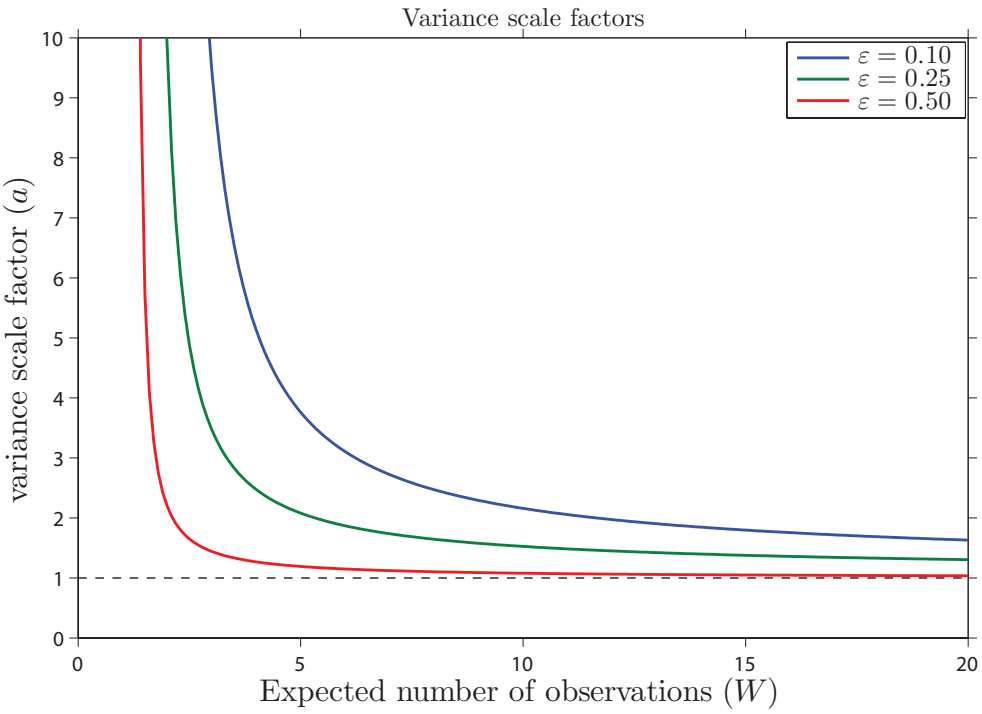


Figure 5.7: The scale factors a with which the appearance sample variances s^2 are multiplied in order to satisfy Equation 5.36 are shown as a function of the expected number of observations W for some representative values of the parameter ε. The scale factors converge to unity (dashed line) for very large values of W.

5.2.2 Iteration and convergence

Once the updated occlusion density $\hat{\beta}\alpha$ and appearance parameters μ,$\hat{\sigma}^2$ have been computed, the cells are independently updated with the new values. The cells are then refined as needed using the method described in Section 5.1.3. The occlusion density and appearance update steps are then repeated until an acceptable level of convergence is achieved.

5.3 Post Processing

Once converged, it is often desirable to represent the model with an octree which has as few leaves as possible. This reduces the storage space needed for the model and also allows for higher computational efficiency in operations such as expected image generation.

5.3.1 Compression of Exterior cells

Octree cells located in "open space" have corresponding occlusion density values which approach zero, and therefore contribute close to nothing to the expected image pixel intensities based on Equation 4.5. Merging neighboring cells with negligible occlusion densities will therefore improve the efficiency of the model while having little effect on the quality of the representation.

Like the cell refinement process, the merging of cells requires a hard decision to be made, and therefore requires a threshold on some attribute of a given cell to determine if it is eligible for merging. The merging process should not introduce artifacts into the expected images generated using the model, so the maximum possible contribution to an expected pixel value is used as the value to be thresholded. A cell's contribution to an expected pixel value is dependent upon its occlusion density, visibility probability, and the length of the corresponding pixel ray segment which passes through the cell (Equation 4.3). The visibility probability may vary with viewpoint, but is assumed to be close to one in the worst case. The worst-case segment length $\ell_{\max}$ is a diagonal path between opposite corners of the cell and is equal to $\sqrt{3}\ell_s$, where ℓ_s is the side length of the cell. The maximum contribution $\Omega_{\max}$ can then be computed:

$$\Omega_{\max} = 1 - e^{\alpha \ell_{\max}} \tag{5.39}$$

If all siblings of a given node at level n in the octree have $\Omega_{\max}$ less than some threshold value $\hat{\Omega_{\max}}$, they are merged into a single parent cell at level $n-1$ with an occlusion density and appearance model equal to the mean of the children's. This process is continued recursively until no further nodes can be merged.

5.3.2 Compression of Interior cells

Based on the definition of the occlusion density function in Equation 3.10, it can be seen that for points which are never visible (e.g. located in the interior of solid objects) $\alpha(s)$ is undefined since $\text{vis}(s) = 0$ for all viewpoints. In practice, the values for these points computed using the iterative reconstruction algorithms presented in this chapter are updated

initially, and then remain unchanged once the model converges sufficiently such that their visibility probability approaches zero for all viewpoints. In order to detect these points, a new scalar function, $\Lambda_\infty(\mathbf{x})$, is introduced.

$$\Lambda_\infty(\mathbf{x}) = \|[\text{vis}(\mathbf{q}_i, \mathbf{x})]\|_\infty \,,\ \mathbf{q}_i \in Q \tag{5.40}$$

where Q is a set of viewing positions, and the infinity-norm is used to select the maximum visibility probability at a point over the set. It is assumed that surfaces are "one-sided", i.e. orientable with an outside and an inside that is never viewed. Points inside of solid objects therefore have $\Lambda_\infty(\mathbf{x})$ close to zero, points in open space have $\Lambda_\infty(\mathbf{x})$ close to one, and an implicit surface can be generated by selecting the level set of any value in between.

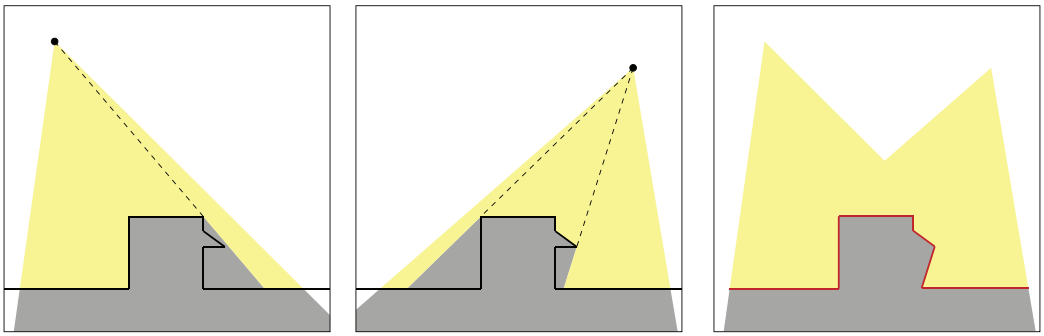


Figure 5.8: The construction of the Λ_∞ implicit function from two views of a simple scene. Figures (a) and (b) show the visibility values for the two cameras (yellow indicates vis = 1, and grey vis = 0). The black lines indicate regions of high occlusion density values. Figure (c) shows the resulting Λ_∞ values, with the red line indicating the implicit surface $\Lambda_\infty = 0.5$.

Interior points are detected by selecting the interior points of a level set corresponding to a very small maximum visibility value $\Lambda_\infty = \epsilon$. The occlusion density of these interior points is then set to zero, allowing the corresponding cells to be compressed as the exterior cells are.

5.4 Conclusion

Two methods for constructing a probabilistic scene model using the variable resolution piecewise-constant model of Chapter 4 have been presented. The first method uses an

online updating algorithm which uses information from a single image at each update step, while the second uses information from all images simultaneously to construct the model using a global optimization. The online algorithm has the advantage of using a constant amount of storage independent of the number of images used since it only needs to store the current state of the model and update information for the current image at any given time. The global optimization technique must store information which is proportional to the number of images being used, but is able to produce better results than the online method. Results achieved using the two methods to reconstruct outdoor scenes from aerial video are compared in Chapter 7.

Chapter 6

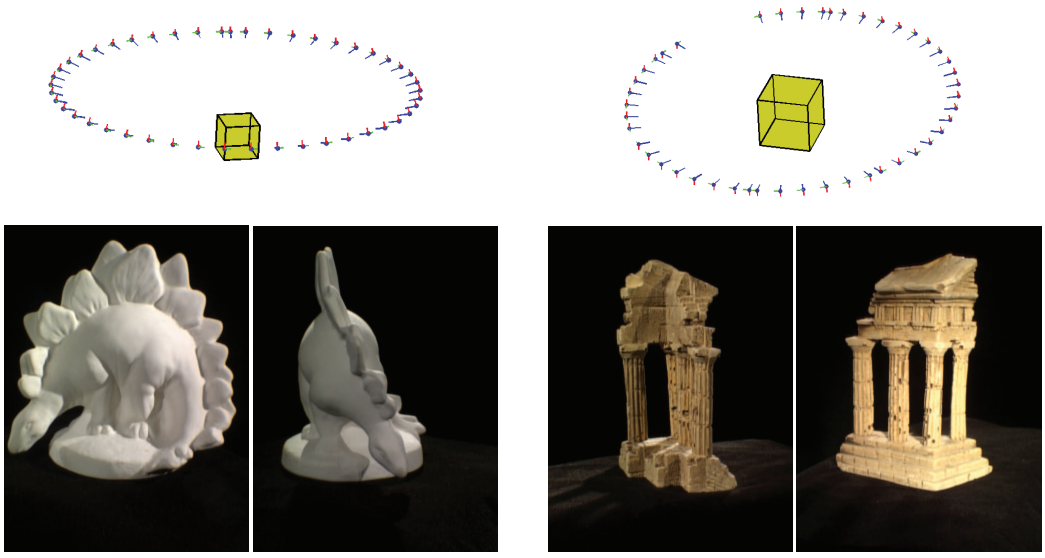
Experiments: Test Data

In order to accurately locate the positions of 3-d points or produce renderings of novel views using the technique described in Section 4.2.2, the reconstruction algorithms presented in Chapter 5 must produce accurate representations of the true scene geometry. In order to evaluate the level of accuracy, the reconstruction algorithms are applied to standard multi-view stereo test sets for which the ground truth geometry is known. Unfortunately, there does not currently exist any standard method for evaluating the accuracy of probabilistic reconstructions. There do exist, however, standard data sets and evaluation techniques for traditional multi-view stereo algorithms [61]. Because the standard method of evaluation requires a triangular mesh as the output of the reconstruction algorithm, a method is presented which produces one based on the probabilistic model. Finally, the accuracy of the resulting mesh is compared with state of the art multi-view stereo methods.

6.1 Evaluation Datasets

In recent years, the Middlebury multi-view stereo evaluation project [61] has become the most widely accepted method for comparing multi-view reconstruction results in the computer vision community. Four sets of images (with corresponding cameras) are available, each captured using a spherical gantry from viewpoints ranging most of a full hemisphere around the test objects. A black background is placed around the object, making straightforward foreground segmentation possible for algorithms which utilize it. The test objects

are then captured using a laser scanner in order to provide a ground truth model with which triangular meshes produced by the reconstruction algorithms are compared. Two of the datasets whose camera configuration resembles those of the aerial sequences are used for evaluation purposes in this chapter.



(a) (b)

Figure 6.1: (a) The cameras and modeled volume for the "dinoRing" dataset, and two representative images. (b) The cameras and modeled volume for the "templeRing" dataset, and two representative images.

6.2 Mesh Generation

The online and batch update algorithms presented in Chapter 5 were run on both of the test data sets with the "background" distribution $p_{A_\infty}(\mathbf{i})$ set to a Gaussian distribution with a mean 0 and $\sigma^2 = 0.031$ to account for the black background present in the images. The σ value was determined empirically by sampling manually selected background pixels from the images. In order participate in the Middlebury multi-view evaluation, reconstructions in the form of a triangle mesh must be produced. In order to satisfy this requirement, a level set $\Lambda_\infty(\mathbf{x}) = 0.5$ of the scalar field $\Lambda_\infty(\mathbf{x})$ presented in Section 5.3.2 is extracted which estimates the location of visible surface points in the volume. The level set partitions the

volume into regions which are more likely to lie outside the object ($\Lambda_\infty(\mathbf{x}) > 0.5$), and regions which are more likely to be inside and therefore not visible from any viewpoint ($\Lambda_\infty(\mathbf{x}) < 0.5$). A mesh-based representation of the level set can then be extracted using the marching cubes method [46].

6.3 Results

dataset	accuracy (mm)	completeness (%)
"dinoRing" (Furukawa and Ponce [26])	0.28	99.8
"dinoRing" (online)	2.61	91.4
"dinoRing" (batch)	5.08	91.5
"templeRing" (Vu et al. [36])	0.45	99.8
"templeRing" (online)	1.89	92.1
"templeRing" (batch)	2.56	93.1

Table 6.1: Accuracy results obtained using the Middlebury multi-view evaluation site for the presented algorithms and the accuracy leaders as of September 2009. Results were obtained using an accuracy threshold of 90% and a completeness threshold of 1.25mm.

Figures 6.2 and 6.3 show renderings of the meshes created from the "dinoRing" and "templeRing" datasets, using the online and batch update algorithms, respectively. The meshes were submitted to the Middlebury multi-view evaluation website (`http://vision.middlebury.edu/mview`) for evaluation, and the results are reported in Table 6.1. Given a threshold percentage P, the accuracy metric indicates the maximum error of the most accurate P% of points. Given a distance threshold c, the completeness metric indicates the percent of vertices of the ground truth model which lie within c of the reconstructed model. The results in Table 6.1 were computed using the site's default thresholds $P = 90\%$ and $c = 1.25$mm.

Based on the accuracy and completeness metrics, the quality of the models constructed with the online method are competitive with state of the art multi-view reconstruction algorithms, although not among the leaders. The primary reason for this is the lack of a regularization term which enforces smoothness of the surfaces in regions of homogeneous color. These homogeneous regions have ambiguous geometry, resulting in a thick region of low occlusion density rather than a thin sheet of high occlusion density as is produced in

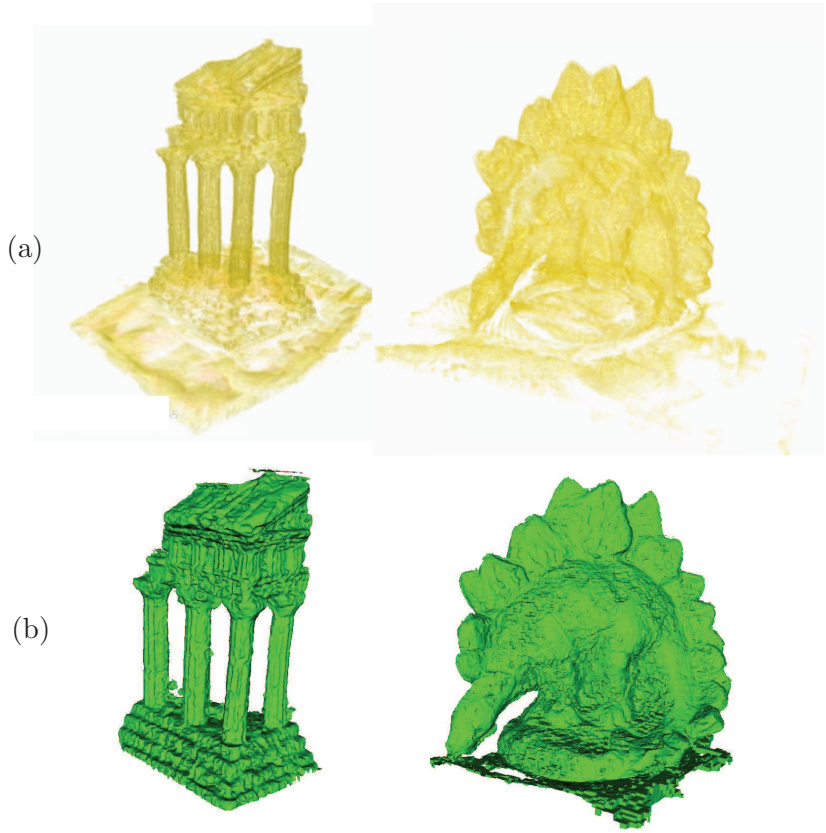


Figure 6.2: (a) Volume renderings of the occlusion densities for the "templeRing" and "dinoRing" datasets using the online construction method. (b): The generated meshes.

more well-defined and textured regions. Because the models are in fact smooth in these regions (particularly the dinosaur model), the reconstruction algorithms which enforce the smoothness assumption with regularization terms are able to produce reasonable surface estimates, as opposed to the presented surface extraction method which becomes very locally sensitive to the choice of Λ_∞. This ambiguity does not, however, affect the model's ability to produce accurate renderings (Figure 6.4) and allows for a more complete representation of the knowable surface geometry free of any smoothness assumptions. Such a representation is important where not only the location of the surface is important, but the precision with which it is known as well. Using the batch method, the completion percentage improves slightly, but the accuracy drops. In practice the accuracy is sensitive to the damping parameter κ, with higher values (and slower convergence) leading to more accuracy. Future

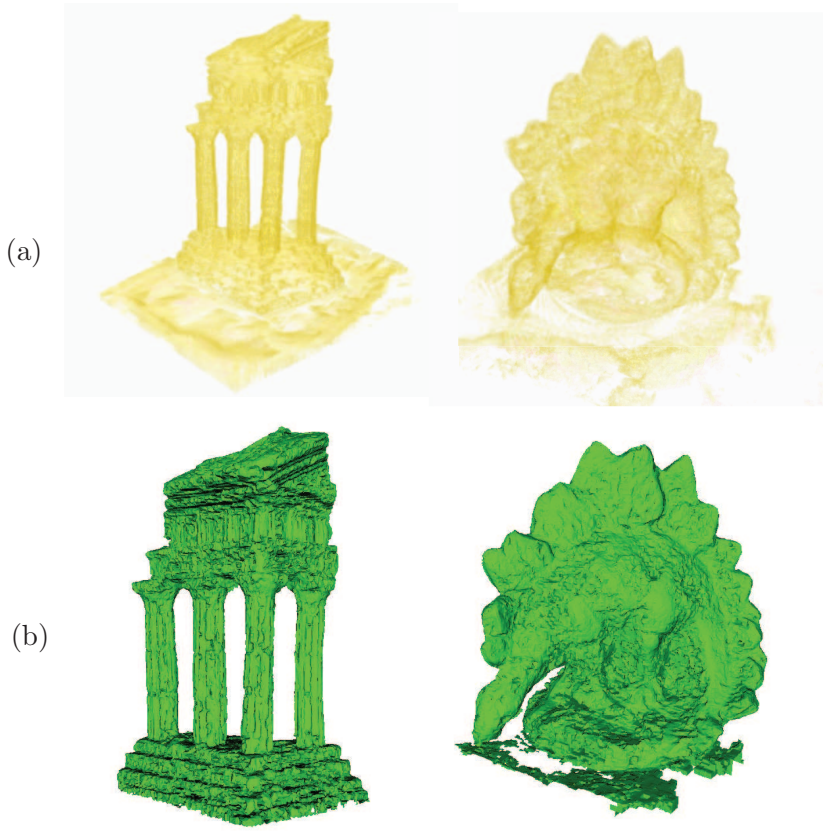
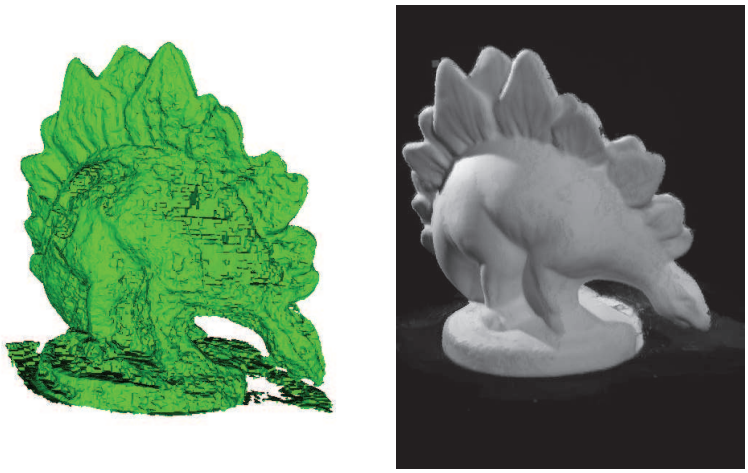


Figure 6.3: (a) Volume renderings of the occlusion densities for the "templeRing" and "dinoRing" datasets using the batch construction method. (b): The generated meshes.

work will focus on the development of optimization algorithms which do not depend on such a damping parameter. Despite the apparent shortcomings of the batch algorithm, it does allow points which are not visible in many views to be modeled more effectively than with the online algorithm. This is demonstrated in Chapter 7, particularly in the datasets where complicated visibility relationships exist between points in the scene (e.g. the "downtown" sequence).

6.4 Conclusions

While not particularly well suited to probabilistic representations, the Middlebury multi-view stereo evaluation system allows the accuracy of the reconstructed models to be independently evaluated with known ground truth. The presented mesh generation algorithm



(a) (b)

Figure 6.4: (a) A view of some regions containing large errors in the surface estimation. (b) An expected image generated from a similar viewpoint.

works well in regions with unambiguous surface geometry, but is not well defined in ambiguous regions due to the lack of any regularization terms in the mesh fitting process. While leading to poorer results on the smooth Middlebury models (particularly the dinosaur), the lack of surface geometry assumptions is in fact a strength of the probabilistic model, since it enables a more complete representation of the 3-d information embedded in the input images. Despite this lack of regularizing assumptions, the experiments demonstrate that an accurate reconstruction of surface geometry is possible which is competitive with state of the art multi-view stereo methods.

Chapter 7

Experiments: Aerial Imagery

The probabilistic model and reconstruction algorithms presented in this thesis have many applications in GIS and visualization, two of which are focused on in this chapter: 3-d point localization and novel view generation. Results for the presented single-image 3-d localization method are presented and compared with multi-view triangulation. Results for the expected image rendering algorithm are presented and compared with state of the art image-based rendering methods.

7.1 Aerial Video Datasets

The aerial video used for experimentation in this thesis was collected using a digital camcorder with resolution 1280×720 from a helicopter flown above Providence, Rhode Island during July of 2006. Three representative sequences are used throughout this chapter for experimentation. The "capitol" sequence is a roughly 270° pass around the Rhode Island capitol building composed of 255 images. The sequence contains finely detailed as well homogeneous regions and a moderate amount of occlusion, mainly due to the capitol building itself. The "downtown" sequence is a roughly complete circular pass around a group of tall buildings in Providence consisting of 180 images and containing many occlusions. The "steeple" sequence is composed of 100 images taken from a linear path above North Main St.. The sequence contains a moderate amount of occlusion but has a much lower variation of viewpoints than the other sequences.

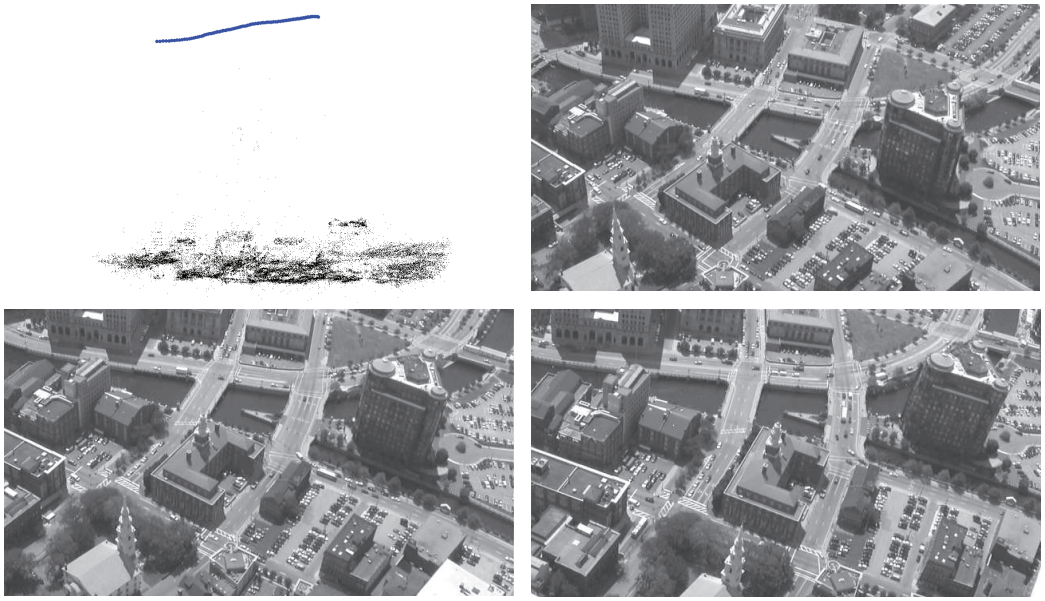


Figure 7.1: The "steeple" dataset. A plot of the feature points and camera centers generated by the structure from motion calibration algorithm [65] as well as three representative images from the set are shown.

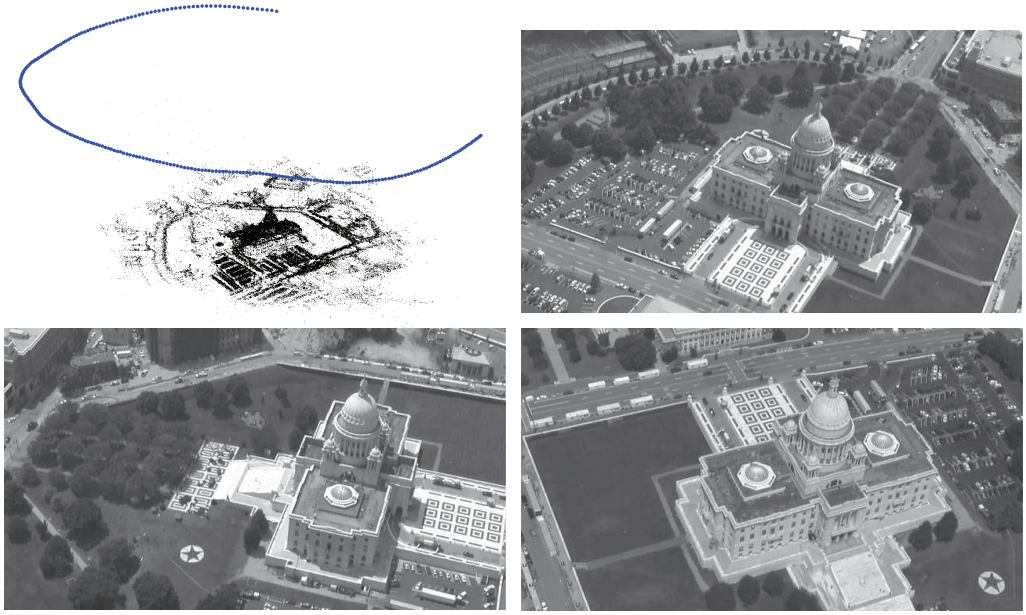


Figure 7.2: The "capitol" dataset. A plot of the feature points and camera centers generated by the structure from motion calibration algorithm [65] as well as three representative images from the set are shown.

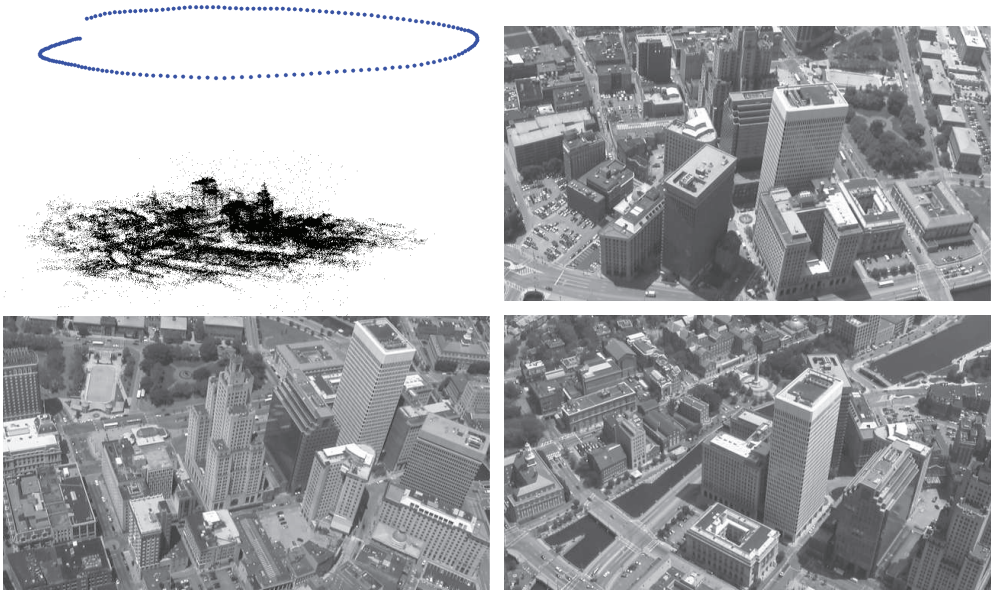


Figure 7.3: The "downtown" dataset. A plot of the feature points and camera centers generated by the structure from motion calibration algorithm [65] as well as three representative images from the set are shown.

The cameras corresponding to each video frame were automatically calibrated using structure from motion software [65] made available by the authors, producing full intrinsic and extrinsic camera calibration for each image as well as a sparse point cloud. The point clouds were not used as part of any probabilistic scene modeling algorithms, but were needed as input to the image-based rendering algorithms of Woodford et al. as shown in Section 7.3.1.

In order to place the calibrated cameras in a real-world coordinate frame with meaningful units, manual point correspondences were selected in a small ($\sim$ 10) subset of the images as well as in a pair of pre-calibrated satellite images of the region purchased commercially. After triangulating the 3-d positions of the points in both the (arbitrary) structure from motion coordinate frame and the (real-world) satellite coordinate frame, a similarity transform (rotation, translation, and a single scale parameter) which maps between the two frames was computed for each sequence.

7.2 3-d Localization

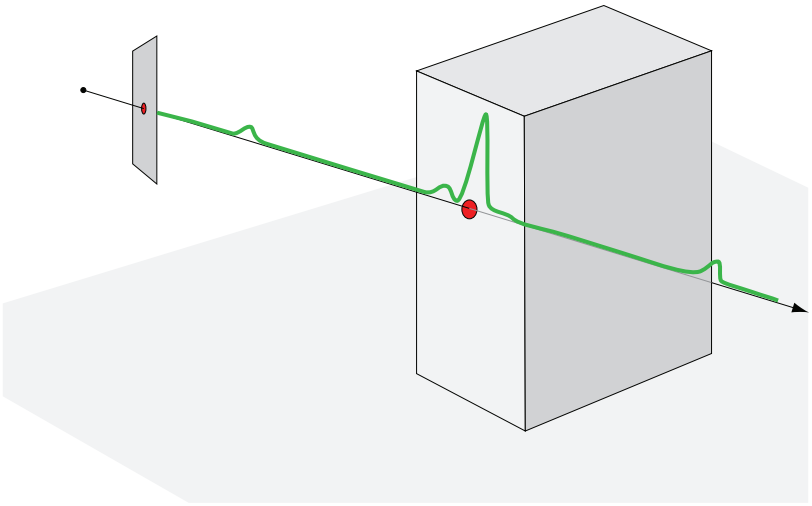


Figure 7.4: Given accurate calibration, a camera ray can be computed for each pixel location (small red dot) in a given image. The distance along the ray of the corresponding world point (large red dot) can then be computed as a probability density function (shown in green) based on the probabilistic scene model.

Given a probabilistic scene model and an image with a corresponding camera, the full probability distribution of the depth of each imaged point can be computed as the function $\omega(s)$ (Section 3.3.1), where s is the distance from the camera center. Because the camera center and ray corresponding to any given point are known based on the calibration parameters, the 1-d distribution can be embedded into 3-d space. Adding the assumption that the camera ray does in fact intersect a point in the scene, Bayes' Law can be used to normalize the density function, producing a new function, $\hat{\omega}(s)$. As shown in section 3.3.1, $\omega(s)$ integrates to the complement of vis_∞. The normalized density function $\omega(s)$ integrates to one.

$$\hat{\omega}(s) = \frac{\omega(s)}{\int_0^\infty \omega(s')ds'} \tag{7.1}$$

The density functions $\omega(s)$ and $\hat{\omega}(s)$ are derived from occlusion density values along the ray only, and do not take into account any appearance information. A more accurate estimate of the occluding point depth for an image pixel can be computed by including the probability of each point along the ray producing the imaged pixel value. This new density

function is termed $\psi(s)$ and is computed as follows:

$$\psi(s) = \frac{\omega(s)p_A(\mathbf{i}, s)}{\int_0^\infty \omega(s')p_A(\mathbf{i}, s')ds'} \tag{7.2}$$

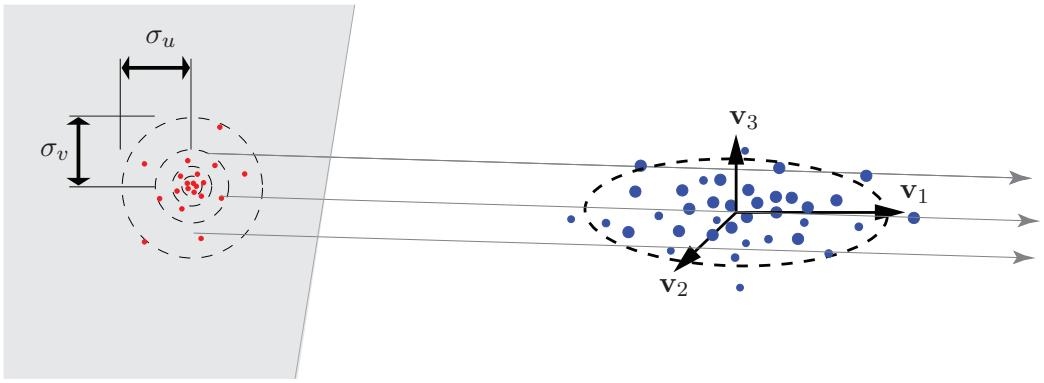


Figure 7.5: Localization precision: Given an image location accuracy represented by the standard deviation σ in pixels, a set of random image samples may be drawn (red dots). For each image sample, a random sample along the corresponding camera ray is drawn based on the depth probability density function. The set of 3-d position samples (blue dots) is represented by its principal component vectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$. The lengths of the vectors indicate the variance of the point set in the respective directions.

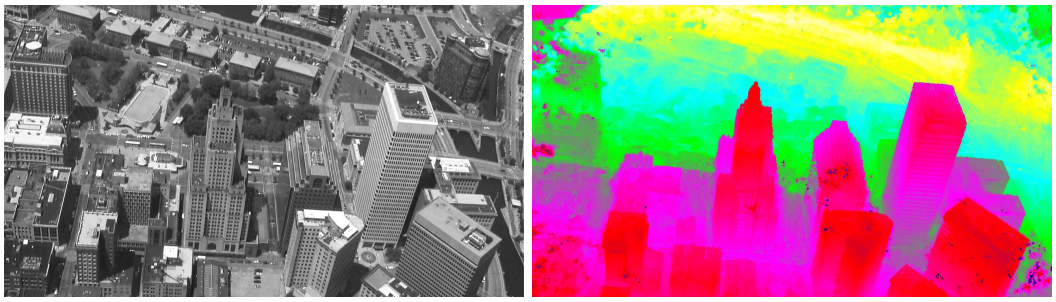
The 1-d probability density functions $\hat{\omega}(s)$ and $\psi(s)$ represent the likelihood of a given camera ray being occluded as a function of depth along the ray. Often times, it is useful to instead have a 3-d representation indicating the expected 3-d point location and a measure of the certainty of the point's location. For this type of measurement it is necessary to know the accuracy of the camera ray itself, which depends on two factors: the accuracy of the camera calibration, and the accuracy of the 2-d image point location.

Assuming that these two error sources are normally distributed and independent in the image x and y directions as well as independent relative to each other, the total ray accuracy can be represented using two variances σ_u^2 and σ_v^2 for the image x and y directions, respectively. The sum of these error sources is also normally distributed, and σ_u^2 and σ_v^2 are computed by simply adding the variances of the calibration and localization distributions. Typical backprojection errors of the structure from motion algorithm are on the order of 0.5 pixels and the conservative assumption of 1 pixel accuracy in image point localization is assumed. These figures are used as the standard deviations of the calibration and image

localization distributions, giving a total variance in each image direction of $\sigma_u^2 = \sigma_v^2 = 1^2 + 0.5^2 = 1.25$ pixels^2.

Based on the distribution of image samples with mean position (u, v) and covariance $\left[\sigma_u^2 \sigma_v^2\right] \mathbf{I}_2$, a set of random samples is chosen, each corresponding to a unique camera ray. For each camera ray, a random depth sample s_d is then chosen according to the depth probability density function $\psi(s)$ (or $\hat{\omega}(s)$ if no appearance information is available) corresponding to the ray. In this way the image location distribution and depth distributions are used to generate a set of random 3-d point locations corresponding to the location of the imaged point. The distribution of this set of 3-d points is analyzed using principal component analysis, which produces three orthogonal basis vectors indicating the directions of maximal variance. The length of the vectors indicate the variance in their respective directions.

7.2.1 Results



(a) (b)

Figure 7.6: A frame from the "downtown" sequence (a) and a visualization of the expected per-pixel depth (b).

In order to evaluate the accuracy of the depth probability functions, ground truth point locations were established by manually selecting their projections in multiple images and triangulating based on standard methods [33]. The location of each point was then manually selected in a test image and a set of 3-d location samples were generated. Figure 7.7 shows the points selected for 3-d localization in the three datasets. Figures 7.8 – 7.16 show plots of the depth cumulative density functions corresponding to the camera rays of the selected image points, as well as the set of 3-d point samples generated according to the process described in Section 7.2. Finally, Tables 7.1 – 7.9 show the standard deviations of the 3-d

point distributions as well as the error of the expected location relative to the triangulated ground truth point location.

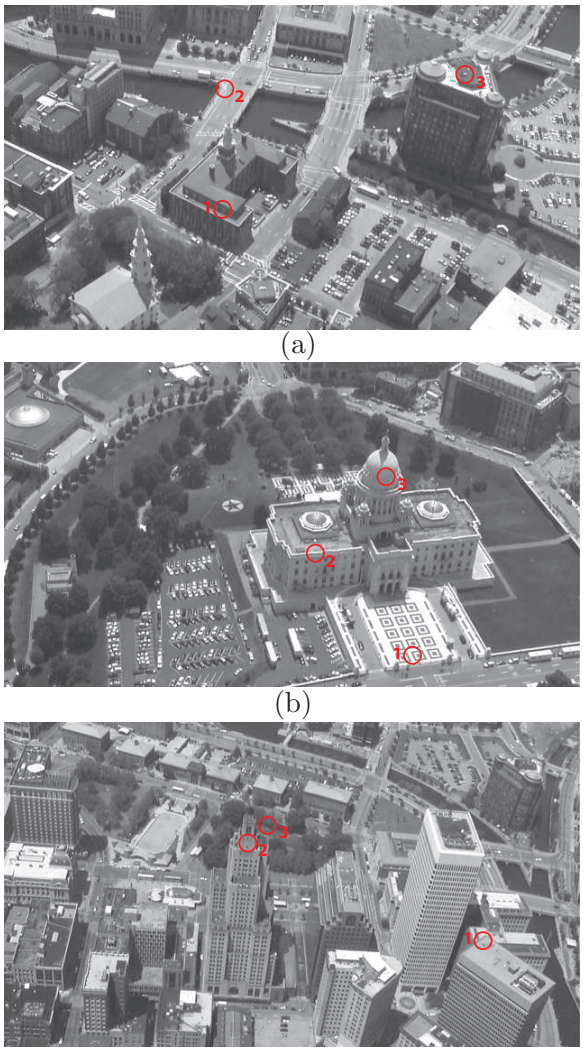


Figure 7.7: Test points from the "steeple" (a), "capitol" (b), and "downtown" (c) sequences used for the localization tests.

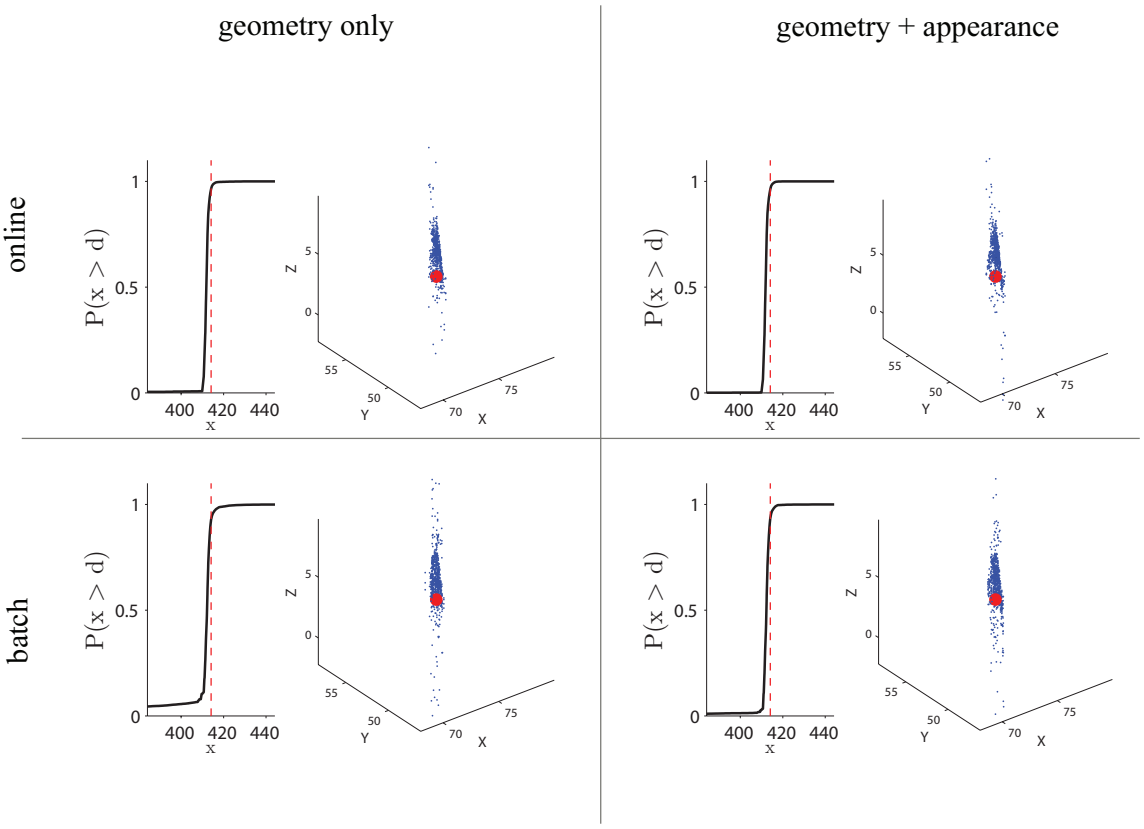


Figure 7.8: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "steeple" test point 1.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	4.88	1.86
batch	2.11	1.49

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	16.44	7.49
batch	6.88	3.80

Table 7.1: Test point 1 for the "steeple" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

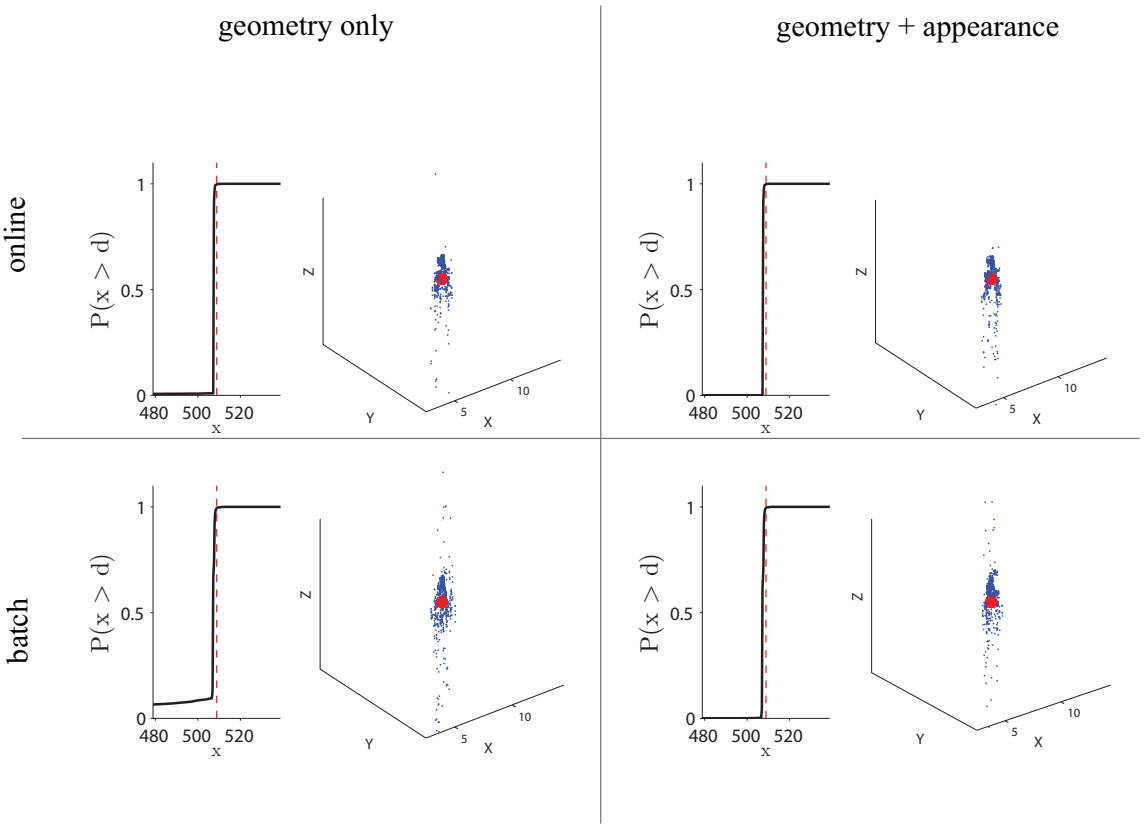


Figure 7.9: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "steeple" test point 2.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	6.09	0.66
batch	0.73	0.14

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	25.4	5.23
batch	8.13	3.48

Table 7.2: Test point 2 for the "steeple" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

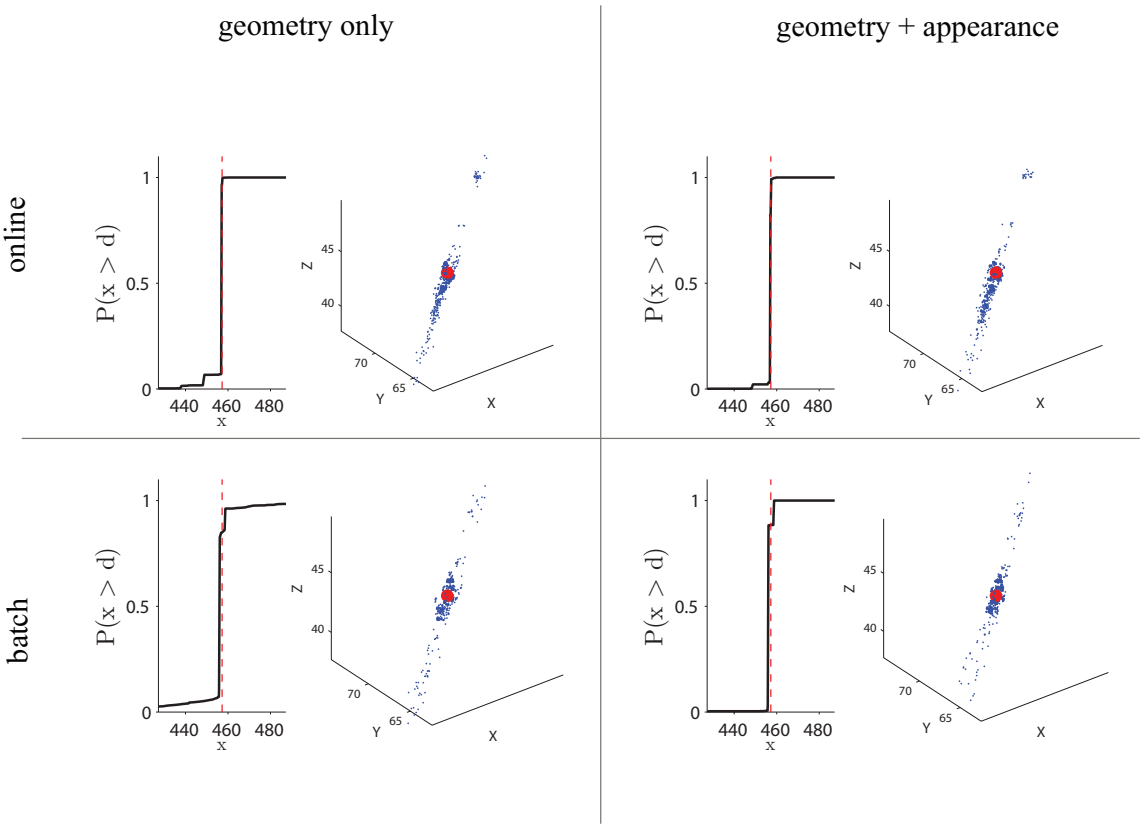


Figure 7.10: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "steeple" test point 3.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	3.51	0.14
batch	0.55	0.78

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	20.5	16.6
batch	7.14	5.5

Table 7.3: Test point 3 for the "steeple" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

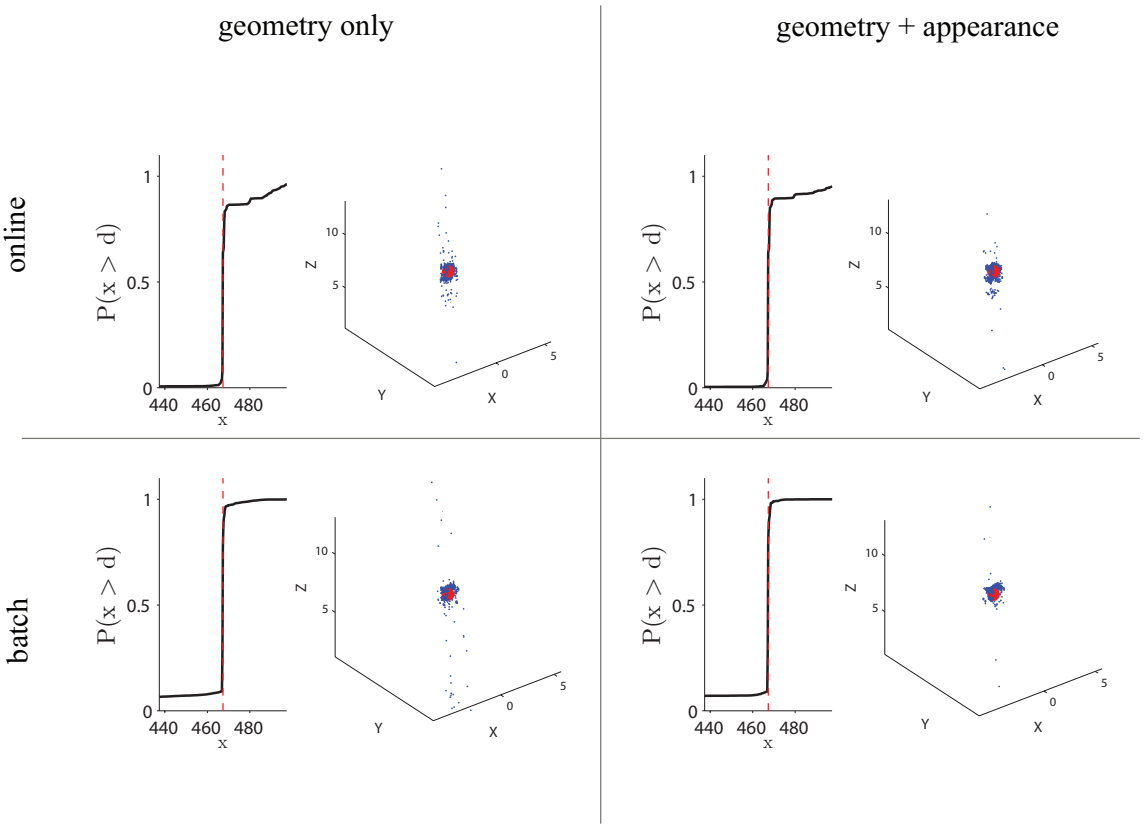


Figure 7.11: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "capitol" test point 1.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	6.53	4.68
batch	2.70	2.07

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	27.86	23.08
batch	12.75	9.83

Table 7.4: Test point 1 for the "capitol" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

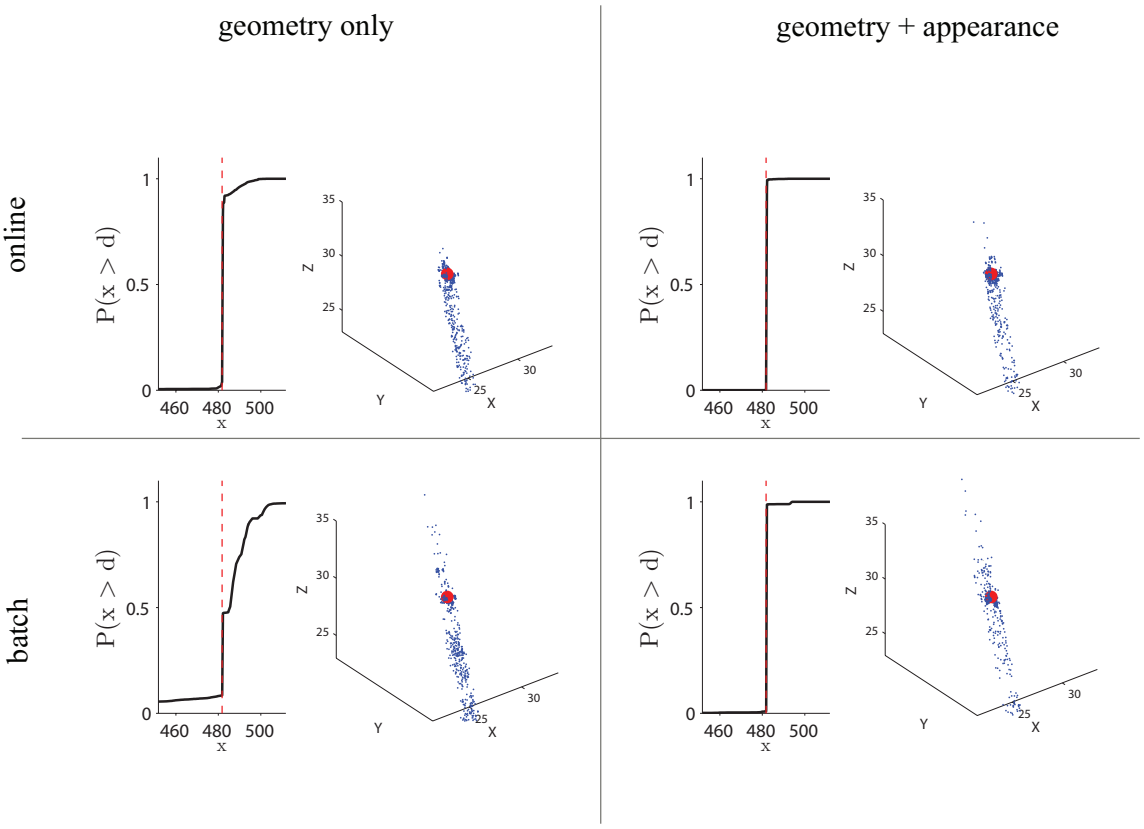


Figure 7.12: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "capitol" test point 2.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	0.05	3.21
batch	2.81	1.85

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	24.47	21.20
batch	9.56	4.53

Table 7.5: Test point 2 for the "capitol" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

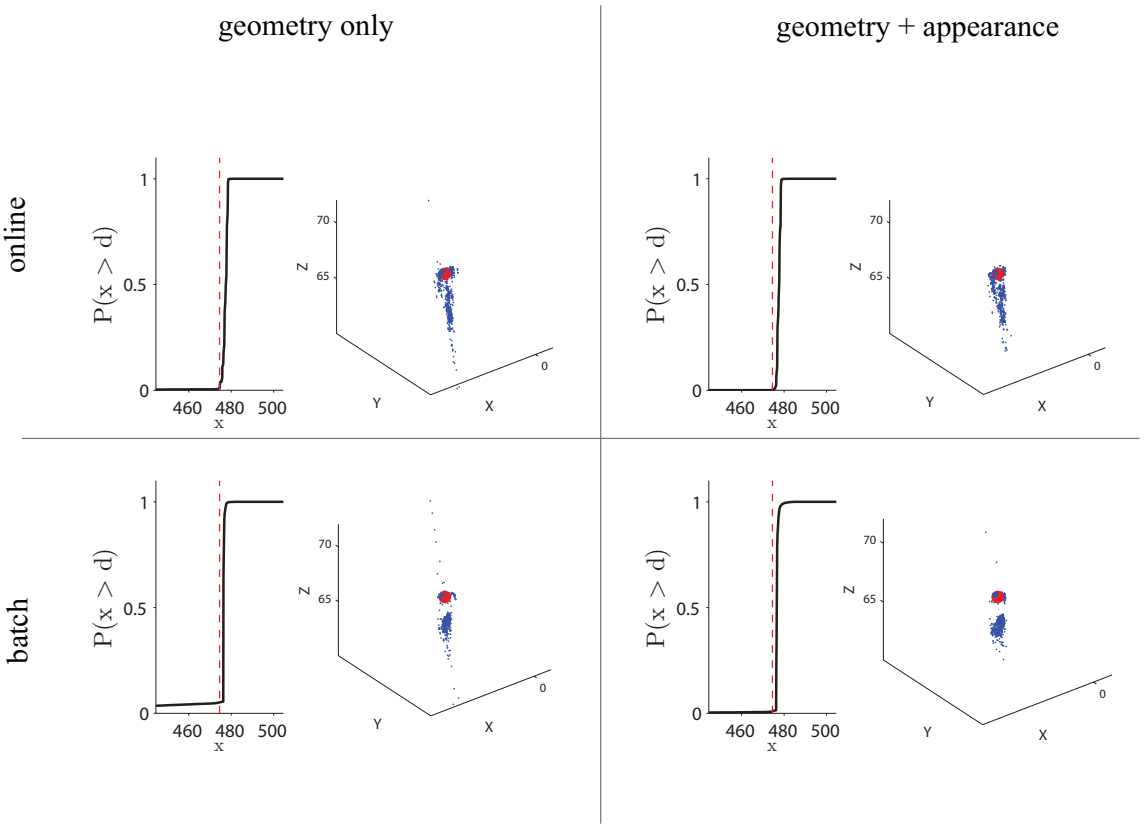


Figure 7.13: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "capitol" test point 3.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	1.39	1.05
batch	1.28	1.35

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	14.05	5.45
batch	3.99	2.13

Table 7.6: Test point 3 for the "capitol" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

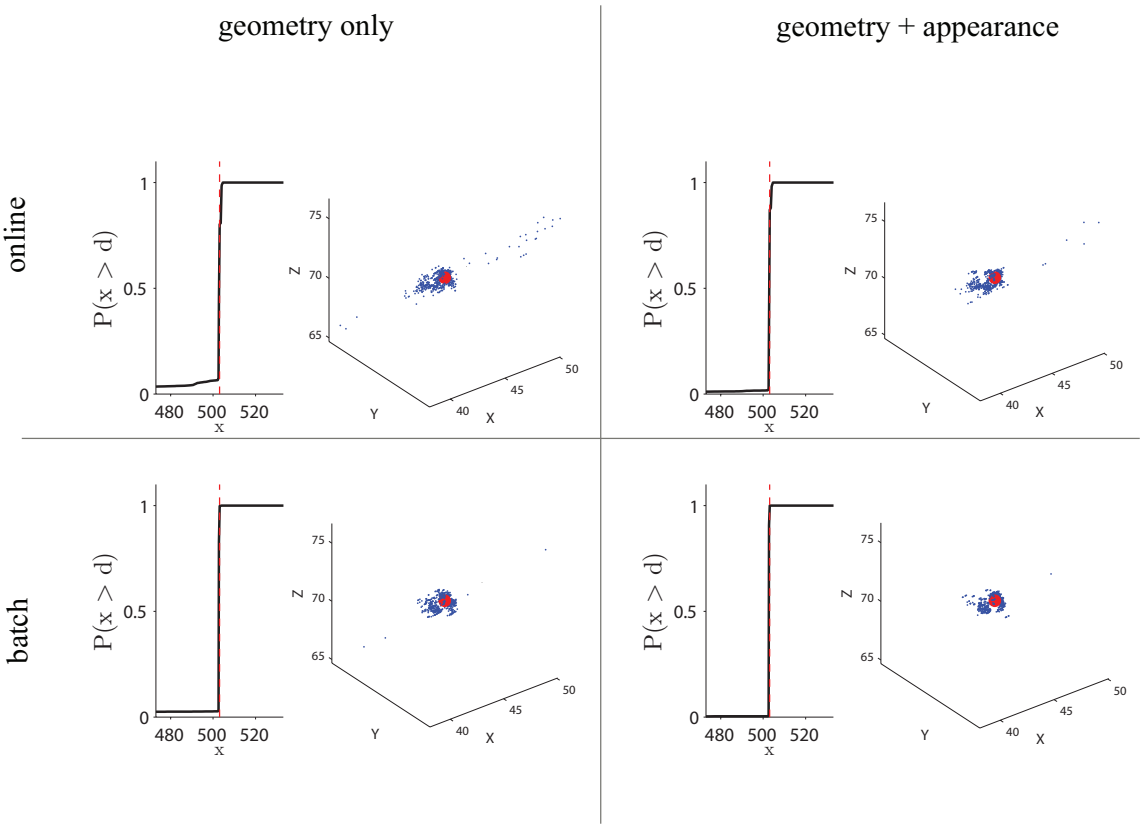


Figure 7.14: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "downtown" test point 1.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	2.54	0.76
batch	4.42	0.57

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	15.02	9.31
batch	20.43	8.85

Table 7.7: Test point 1 for the "downtown" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

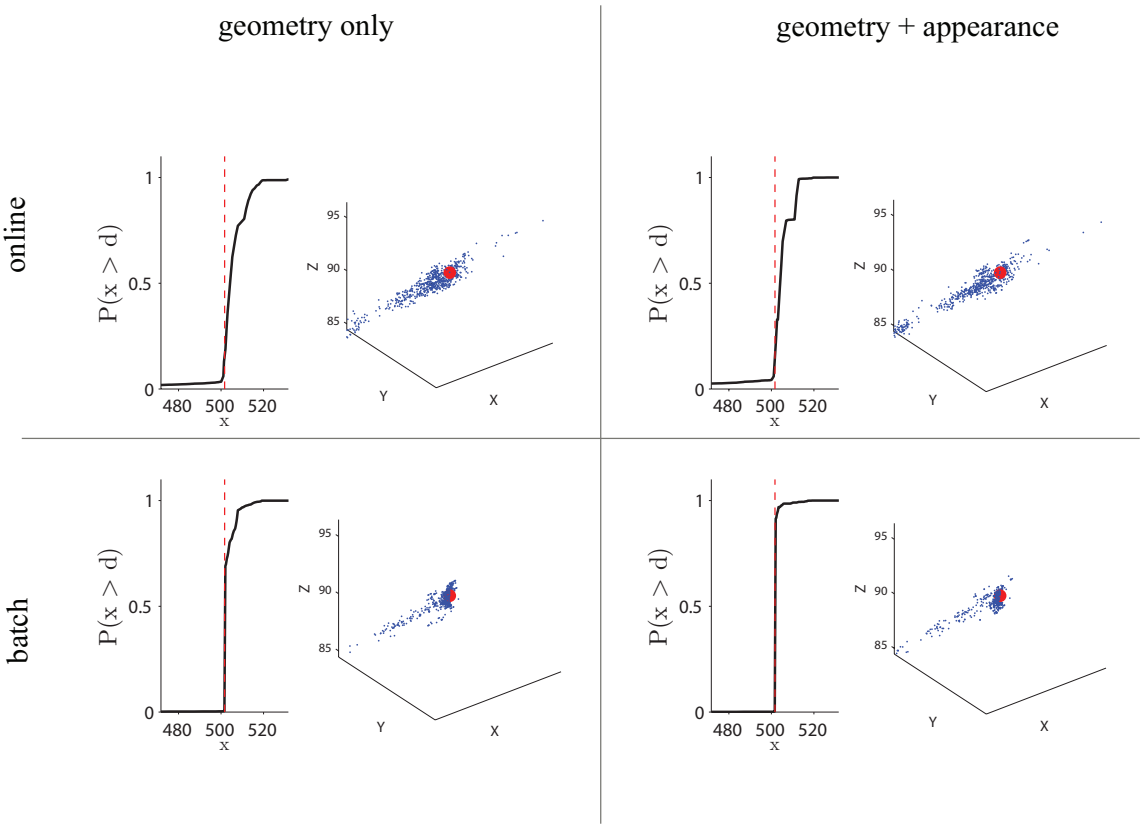


Figure 7.15: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "downtown" test point 2.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	1.90	1.79
batch	1.04	1.24

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	9.56	11.68
batch	3.85	4.37

Table 7.8: Test point 2 for the "downtown" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

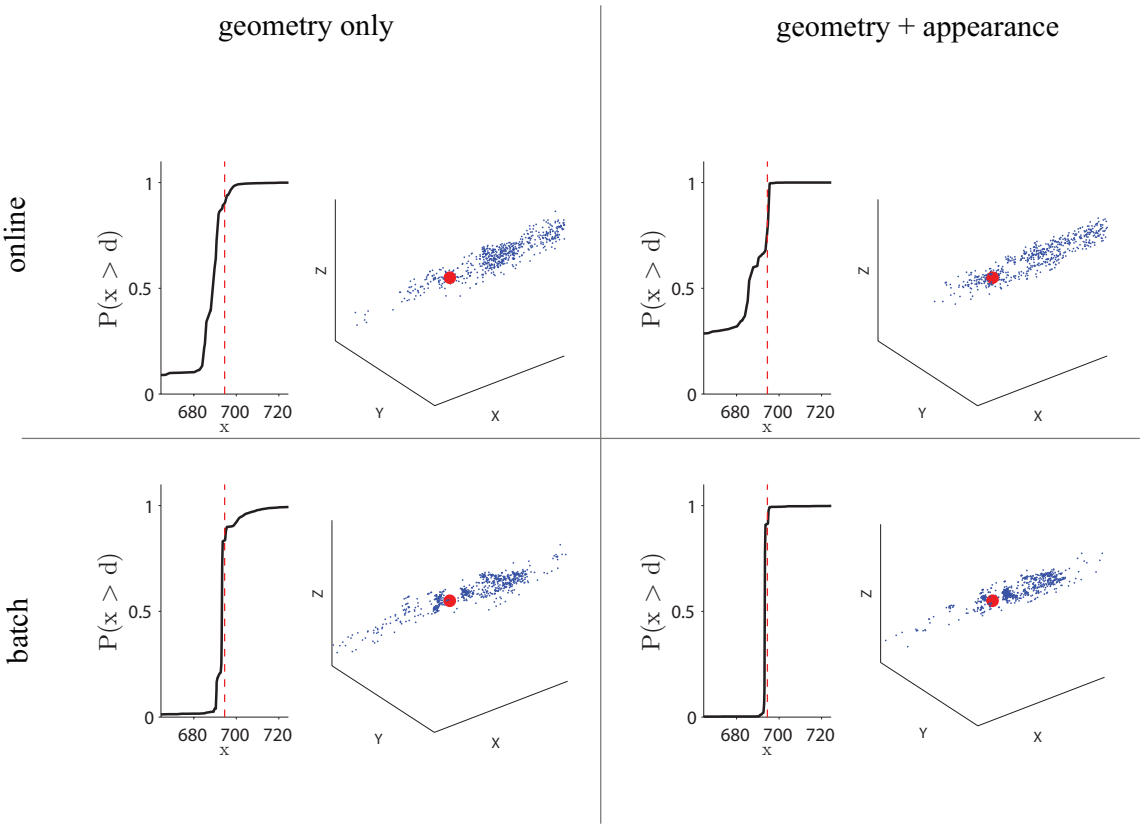


Figure 7.16: Plots of the depth cdf and generated random point samples (triangulated point shown in red.) for the "downtown" test point 3.

(a) Expected Point Error (m)

	appearance info?	
	no	yes
online	16.90	22.99
batch	2.84	5.81

(b) $\sigma_{\max}$ (m)

	appearance info?	
	no	yes
online	44.36	56.01
batch	15.36	24.27

Table 7.9: Test point 3 for the "downtown" dataset. (a): distance from the triangulated point to the expected 3-d sample. (b): Standard deviation of the point sample distribution in the direction of maximum variance.

By accumulating information from previous images, the probabilistic model is able to produce accurate point location estimates based on an image correspondence from a single image. In addition to the expected position, a full probability distribution of the location can be generated which is useful for GIS applications where the precision of 3-d point estimates is of high importance. Based on the expected point errors and distribution variances, it is clear that the appearance information adds significant accuracy and precision to the estimated point locations. The models constructed with the batch algorithm are able to localize better than those built using the online algorithm, perhaps not surprising since the batch algorithms have access to a greater amount of input image data at each iteration of the reconstruction. The depth cumulative density functions reveal the sources of error and uncertainty which lead to unreasonably large values of the expected point errors and $\sigma_{\max}$. In some cases, such as test point 3 of the "downtown" sequence, there are regions of non-zero occlusion density in the open space between the camera center and the true point of occlusion. When this occurs, the cdf will be greater than zero at the true depth. Alternatively, in some cases the occlusion densities around the surface location have not sufficiently converged, leaving a significant probability of the ray passing through the surface region unoccluded. This can be seen in the cases of test point 2 of the "capitol" sequence and test point 2 of the "downtown" sequence. In general, however, the depth cumulative density functions closely approximate the ideal case of a step function, with the sharp transition from 0 to 1 occurring near the true depth value.

7.3 Novel View Generation

By generating an expected intensity value for each pixel in an image using the algorithm described in Section 3.3.2 (implementation described in Section 4.2.2), an image may be generated from any arbitrary viewpoint using a probabilistic scene model. In order to evaluate the accuracy of these images with respect to those produced by state of the art image-based rendering algorithms, two distinct test case paradigms are used. The first, sometimes referred to as a "leave one out" test, involves generating an image from the viewpoint of one of the images from the original sequence but was not used as input to the algorithm. Using the original image as ground truth, the accuracy of the rendered images may be quantitatively evaluated by computing the root mean square error over all pixels in the image. In the case of video sequences, the viewpoint from which the view is rendered, although not identical, is typically fairly similar to input images which are temporally close to the "left out" image. In image-based rendering terminology, this is often referred to as the problem of view interpolation. It should be noted that, while widely used, the evaluation metric based on pixel differences is not a perfect indicator of "photorealism" since it does not penalize various type of artifacts such as discontinuities and "salt and pepper" noise which can be disturbing to the human visual system. The second, more challenging, type of test involves generating a view of the scene from a viewpoint which is far from any of the input viewpoints. Unfortunately, no suitable ground truth images were captured which image the scenes from a view disparate from those of the input sequences. Because of this, the rendered results must be evaluated qualitatively on the basis of photorealism alone.

7.3.1 Results

	Woodford07a	Woodford07b	Pollard09	psm (online)	psm (batch)
"steeple"	0.068	0.071	0.051	0.053	0.046
"capitol"	0.109	0.060	0.098	0.085	0.075
"downtown"	0.122	0.103	0.109	0.106	0.079

Table 7.10: RMS errors for the image-based rendering algorithms evaluated over the intersection of masks. Results are based on normalized pixel values with range $[0, 1]$

Figures 7.17, 7.18, and 7.19 show rendered images using two state of the art image-based

rendering algorithms ([73] and [74]), Pollard and Mundy's fixed-grid voxel model [53, 52], and the octree-based models based on the continuous model presented here (online and batch reconstruction methods). Table 7.3.1 shows the root mean square error of the images, using the ground truth image contained within the intersection of all valid reconstruction regions as the basis for evaluation. All five methods perform well on the "steeple" sequence, which is not surprising given that it is the least complex of the three sequences in terms of occlusions and viewpoint variation. The "capitol" sequence is more challenging due to the large viewpoint variation and resulting occlusions, mostly due to the capitol building itself. All methods perform reasonably well with the exception of Woodford07a [73], which produces noticeable artifacts in the patterned concrete in front of the building. The fixed-grid voxel method [52] produces a reasonable image but suffers around regions of high frequency due to the inherent practical resolution limitations of fixed-grid methods. As is the case with all of the test images, the model constructed with the batch algorithm outperforms that constructed with the online method. The "downtown" sequence presents the greatest challenge for the rendering algorithms due to the large viewpoint variation combined with the large amount of occlusions present. The image-based rendering algorithms suffer because of this, as evidenced by the large amount of rendering artifacts present. The fixed-grid probabilistic method again suffers due to the lack of sufficient resolution to cover the entire scene at the resolution of the input images. Along with the online octree method, it also appears to suffer in regions that are occluded in the majority of views, such as the lower portions of the tall building sides. This is presumably due to the fact that the online methods did not processes a sufficient number of views in which the regions were not occluded. The batch algorithm does not suffer from this problem since it is able to process information from the entire range of viewpoints at each iteration, resulting in a significantly lower RMS error rendering.

While capable of rendering images for which nearby input images are available, the quality of the images produced by image-based rendering algorithms rapidly degrades as the viewpoint moves farther away. Because they explicitly represent 3-d scene structure, the probabilistic methods are capable of predicting views which vary greatly from any available input. Figures 7.20, 7.21, and 7.22 show two rendered viewpoints for each of the aerial

datasets rendered using the fixed-grid voxel model and the octree-based continuous model. Although the relatively small variation in viewpoints in the "steeple" sequence makes view interpolation an easier problem, it makes rendering from disparate viewpoints more difficult since the 3-d information is not as accurate. This is evidenced in the relatively poor quality of the rendered images in Figure 7.20, especially around homogeneous regions which are impossible to precisely localize given the image data. Figure 7.21 shows renderings from two distant viewpoints for the "capitol" sequence. The first view is a low-altitude view looking at the back of the capitol building, which the models produced by both online methods (discrete voxel and online octree-based) are not able to satisfactorily render. It should be mentioned that the discrete voxel-based method has an implementation issue which prevents it from correctly handling camera rays at shallow angles, which may have contributed to the poor image quality. The model produced by the batch update method produces a fairly crisp and photo-realistic rendering. The second view, a nadir view of the building and surrounding land, is rendered fairly successfully by all three probabilistic methods, although the octree-based batch method again reproduces the finest high frequency details and contains the fewest number of artifacts. Similar types of viewpoints are rendered for the "downtown" sequence, shown in Figure 7.22. The octree-based batch method again outperforms the online methods in terms of image fidelity, especially in regions that are not seen in the majority of images such as the lower regions of the building sides. The octree-based methods are able to predict imagery for a larger geographic region in greater detail thanks to the storage savings of the octree model.

7.4 Conclusions

In this chapter, the utility of the octree-based implementation of the continuous probabilistic scene model in two distinct application areas was explored: 3-d point localization and novel viewpoint rendering. The 3-d point localization technique is novel in that it enables a user to view a probability distribution of the location of any imaged 3-d point based on a scene model and it's projection in a single image. Two localization methods were presented: one that uses the occlusion density information only and one that also incorporates appearance

information, with the later shown to be significantly more effective. Using the rendering equations presented in Chapters 3 and 4, novel views of aerial scenes with varying amounts of occlusion and viewpoint variation were generated and compared with results produced by state of the art image-based rendering algorithms. Particularly in the scenes with a large number of occlusions, the probabilistic models fared better in general. Unlike the image-based rendering algorithms, the probabilistic models were also able to generate photo-realistic renderings from viewpoints far from any of the available input views. In general, the octree-based models built using the batch update algorithm fared better than the online version and the octree-based models were able to represent larger areas with finer detail than the fixed-grid models using less storage space.

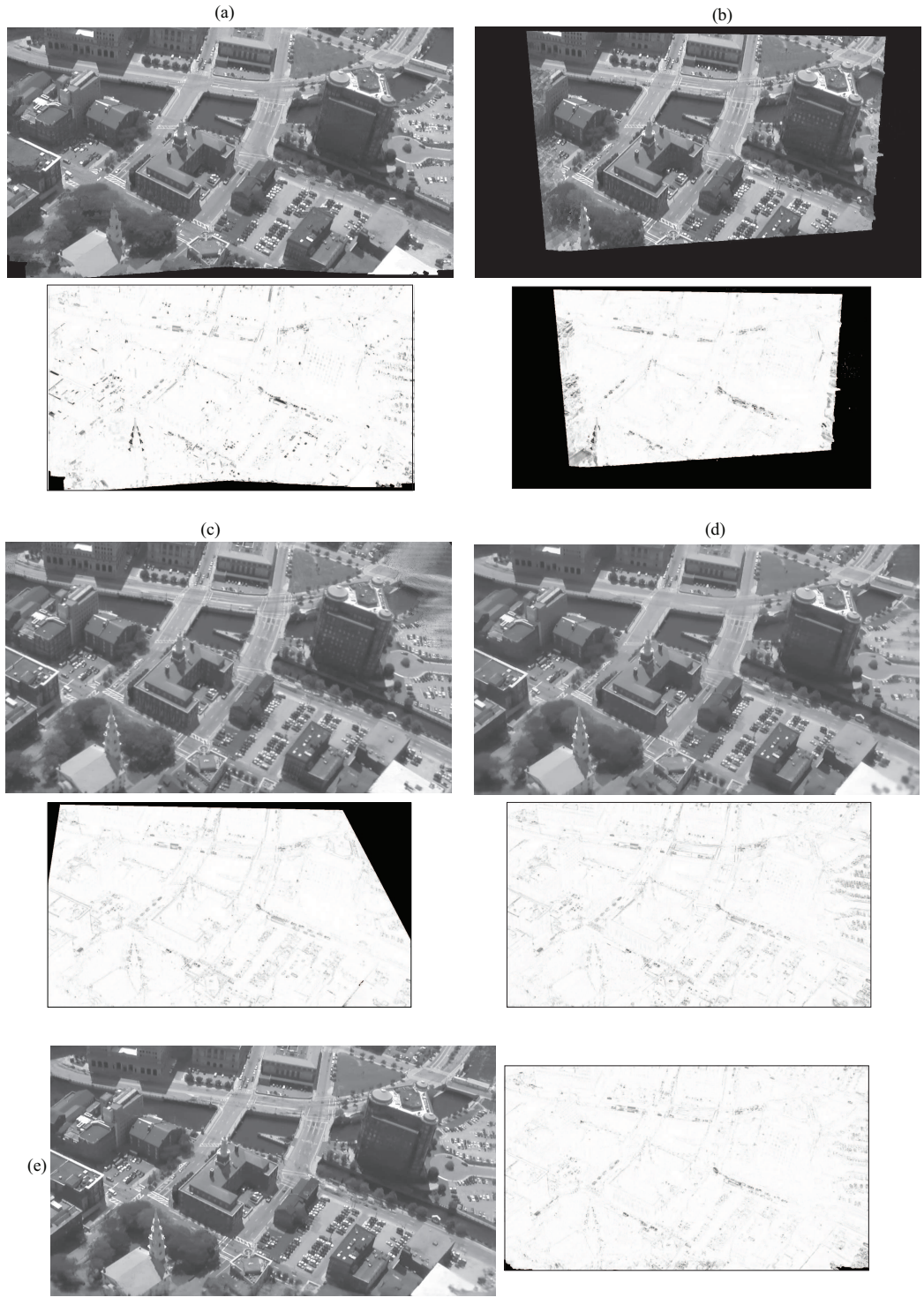


Figure 7.17: Image-based rendering results for the "leave-one-out" test on the "steeple" sequence. The pixel error images are shown below the rendered results. (a): Woodford et al. [73] (b): Woodford at al. [74] (c): Pollard [52], (d) Continuous model (online construction): (e): Continuous model (batch method)

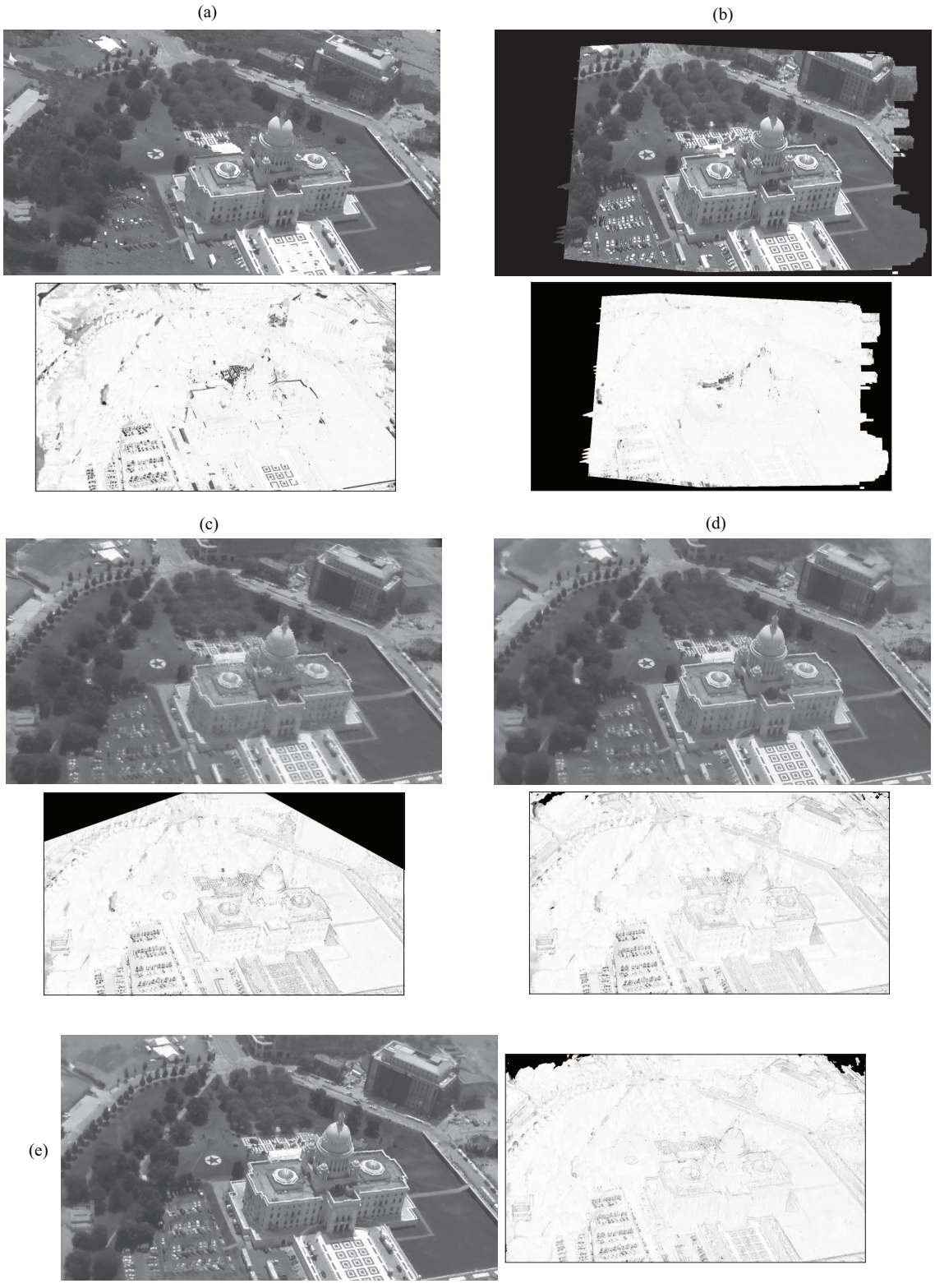


Figure 7.18: Image-based rendering results for the "leave-one-out" test on the "capitol" sequence. The pixel error images are shown below the rendered results. (a): Woodford et al. [73] (b): Woodford at al. [74] (c): Pollard [52], (d) Continuous model (online construction): (e): Continuous model (batch method)

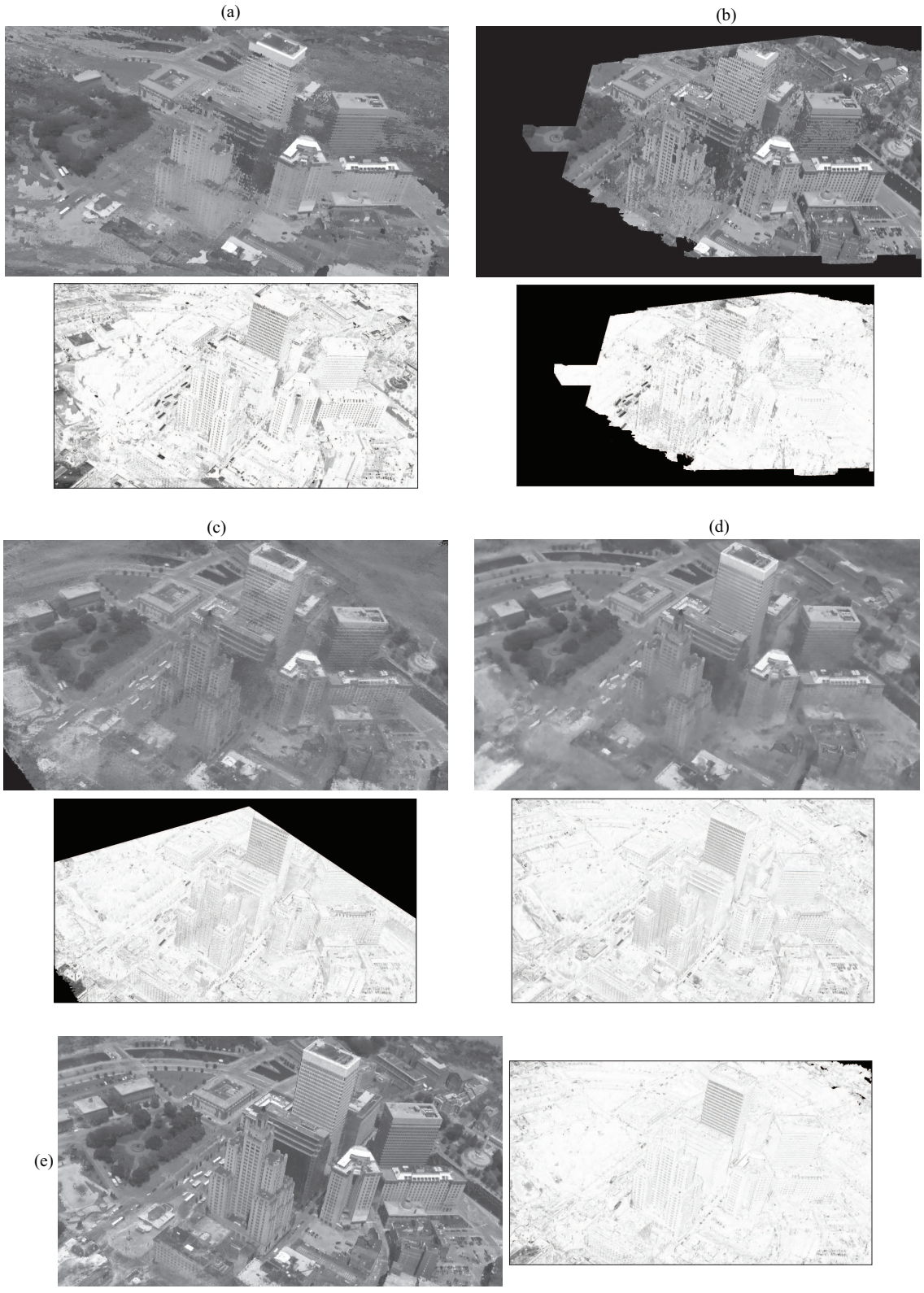


Figure 7.19: Image-based rendering results for the "leave-one-out" test on the "downtown" sequence. The pixel error images are shown below the rendered results. (a): Woodford et al. [73] (b): Woodford at al. [74] (c): Pollard [52], (d) Continuous model (online construction): (e): Continuous model (batch method)

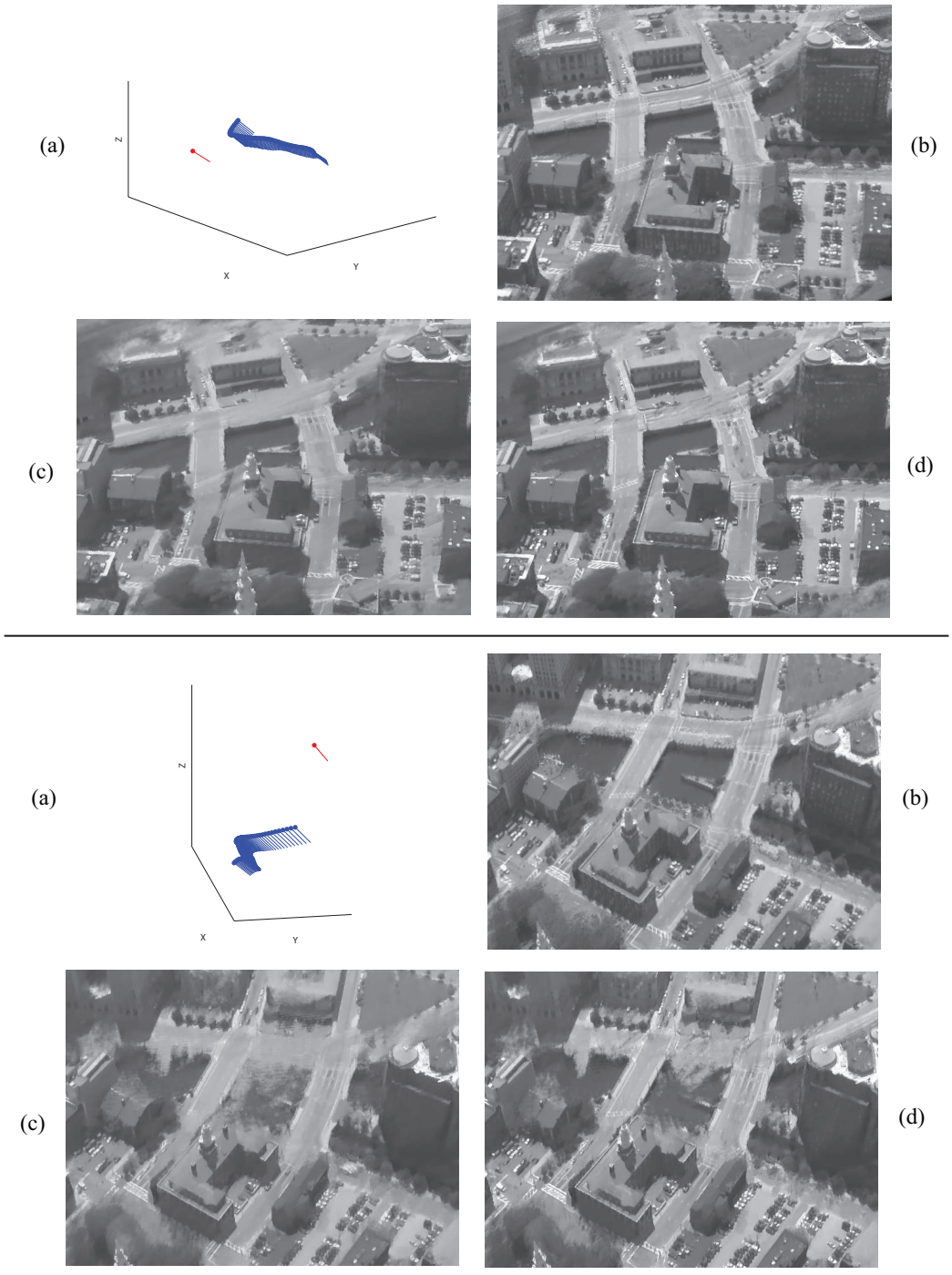


Figure 7.20: Two novel viewpoints far from any input images ("steeple" sequence). (a): The virtual camera (red) plotted with the input cameras (blue). (b): Using Pollard's voxel model. (c): Continuous model, online constuction. (d): Continuous model, batch method.

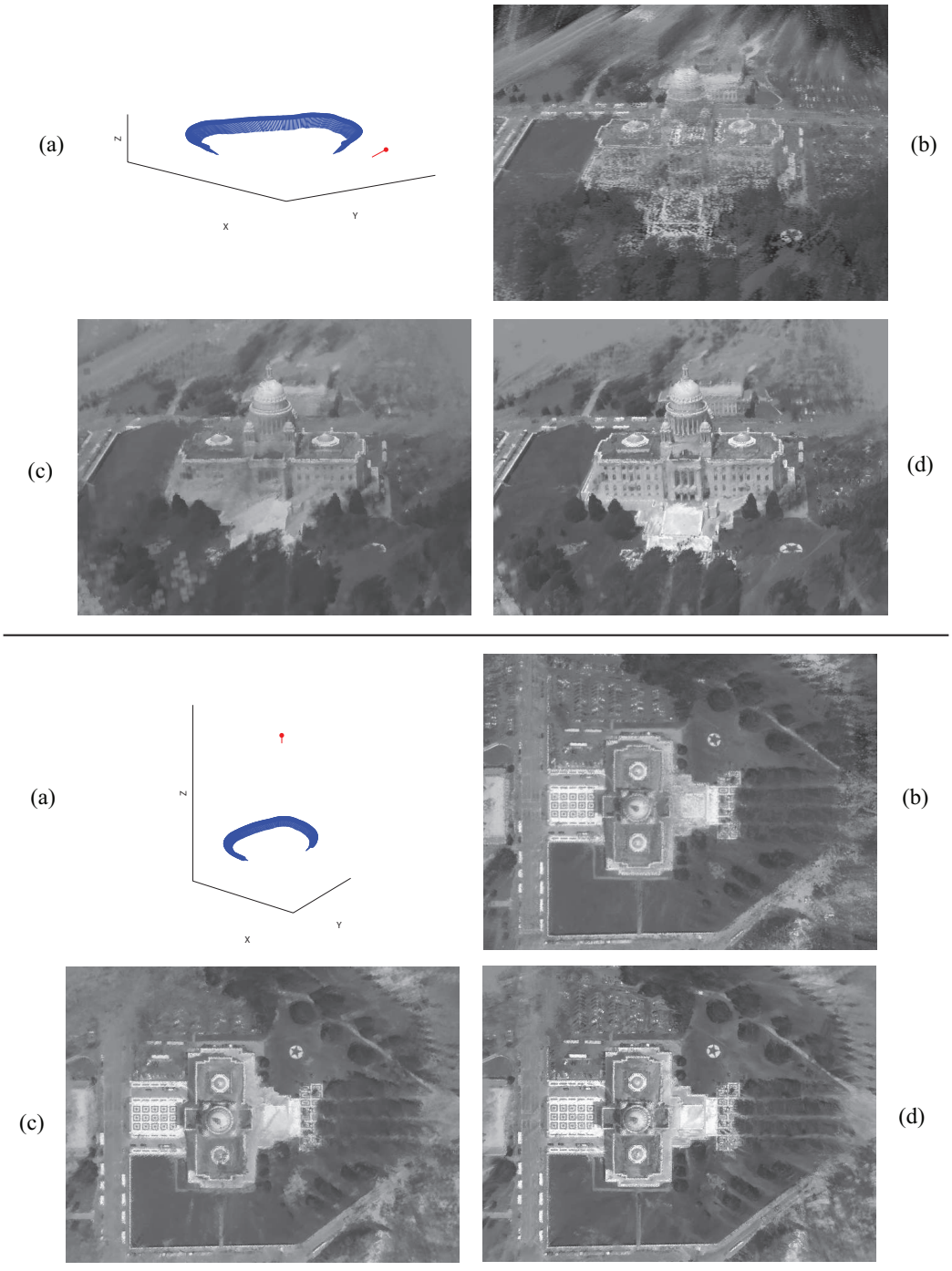


Figure 7.21: Two novel viewpoints far from any input images ("capitol" sequence). (a): The virtual camera (red) plotted with the input cameras (blue). (b): Using Pollard's voxel model. (c): Continuous model, online constuction. (d): Continuous model, batch method.

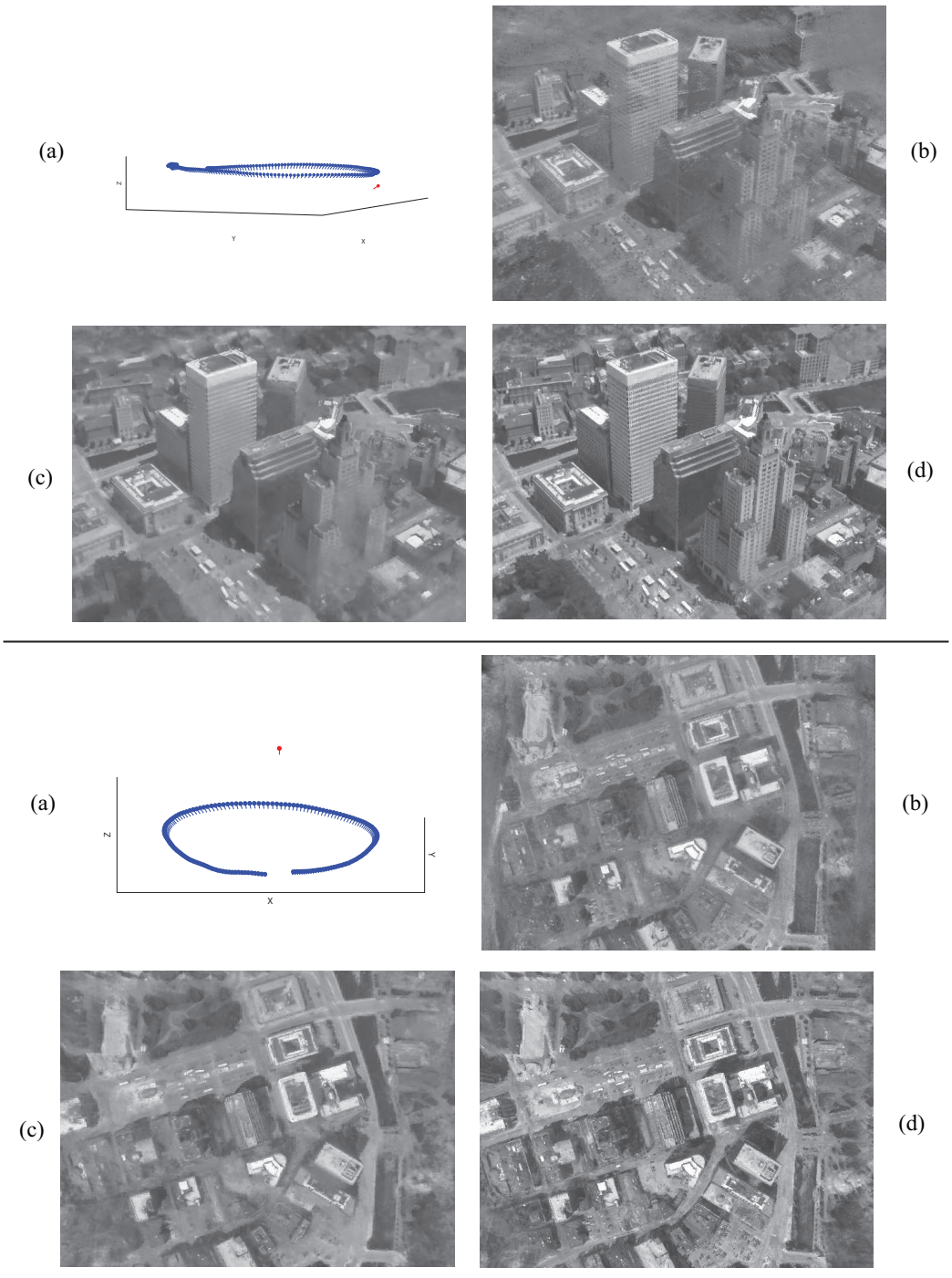


Figure 7.22: Two novel viewpoints far from any input images ("downtown" sequence). (a): The virtual camera (red) plotted with the input cameras (blue). (b): Using Pollard's voxel model. (c): Continuous model, online constuction. (d): Continuous model, batch method.

Chapter 8

Camera Refinement

There exist a large number of automatic and manual camera calibration algorithms in the computer vision and photogrammetry literature. The vast majority of these algorithms produce or take as input a sparse set of feature correspondences across images and optimize the camera parameters (and potentially the 3-d feature locations) to minimize the reprojection error in the images. For example, the camera models for the aerial video frames used in this thesis were generated using the Structure from Motion algorithm of Snavely et al. [65] with features automatically detected and matched using Lowe's SIFT [47] keypoints. While these algorithms have proven to be sufficiently accurate for many applications, their reliance on sparse feature locations means that much of the available image information goes unused. In this chapter, a method for refining these camera estimates which uses all available image information is presented. The method is based on the visual servoing technique from the robotics field and requires that the provided initial camera estimates are accurate enough to construct a scene model from which expected images can be rendered. This chapter is based on material originally published by the author in 2008 [14].

8.1 Background and Overview

Visual servoing is a term used in the robotics community referring to the use of video and video processing as a form of feedback in a position control loop. A tutorial on the field was put together by Hutchinson et al. [39]. A typical goal is to move the end-effector of

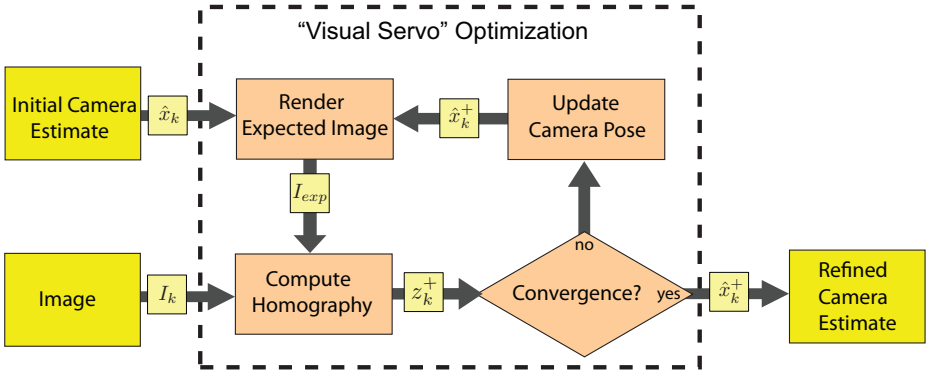


Figure 8.1: Flowchart of the camera pose optimization algorithm.

a robotic arm to a pre-determined position and orientation relative to a stationary object, with a single camera located on the end-effector (the "eye-in-hand" configuration). The arm is first manually moved to the desired position, and an image of the target is taken and stored. If the target is planar, a homography can be computed which maps the current image to the stored image, and the end-effector is moved with the goal of bringing the homography to the identity matrix. Drummond and Cipolla [20] showed that by using Lie Algebra representations, 3-d information about the world is implicitly embedded into the 2-d image transformations.

Rather than providing feedback to a physical servoing system, here it is the camera estimate that is being adjusted. Since it is not possible to capture real images from the estimated viewpoints, expected images generated using the scene model are used to instead. The goal of the refinement algorithm is to produce a camera model whose corresponding expected image is mapped to the true image by the identity transformation.

8.2 Representation of 2-d and 3-d Transformations

The 2-d general affine matrix group $GA(2)$ is used to represent image homographies, and the special Euclidian matrix group $SE(3)$ to represent camera motions. Drummond and Cipolla [20] showed that by using Lie Algebra representations, 3-d information about the world is implicitly embedded into the 2-d image transformations. Although the goal is to accurately handle non-planar scenes, an assumption is made that the translation between

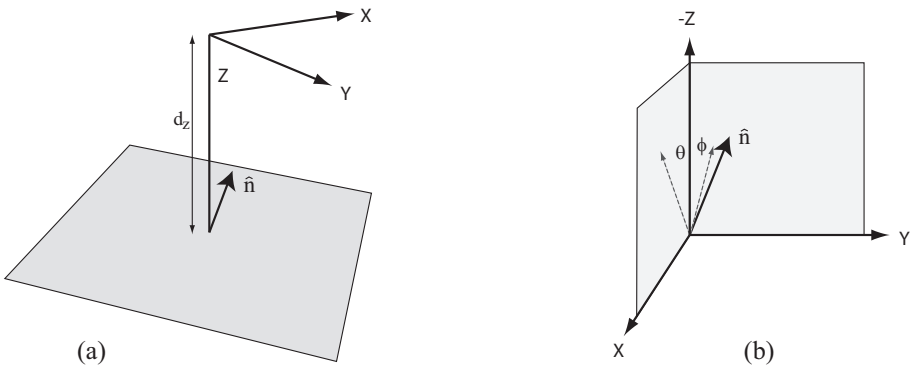


Figure 8.2: (a) The dominant world plane is shown in the camera coordinate system. (b) The plane normal $\hat{n}$ is projected onto the X-Z and Y-Z planes, giving the plane parameters θ and ϕ.

the estimated and true camera positions is sufficiently small that the corresponding image transformation can be approximated by a 2-d homography induced by a dominant world plane $\Pi = \begin{bmatrix} \hat{n}_x & \hat{n}_y & \hat{n}_z & d \end{bmatrix}^T$, where $\|\hat{n}\| = 1$ and the plane parameters are specified in the camera coordinate system (i.e. the z axis is the principal axis of the camera). Disregarding degenerate cases, Π can be represented using three parameters:

$$\theta = \tan^{-1} \frac{\hat{n}_x}{-\hat{n}_z}, \quad \phi = \tan^{-1} \frac{\hat{n}_y}{-\hat{n}_z}, \quad d_z = \frac{-d}{\hat{n}_z} \tag{8.1}$$

(See Figure 8.2). The Lie group $SE(3)$ has an associated Lie algebra $se(3)$, which is spanned by the so-called $SE(3)$ *generator matrices* E_i, $i \in \{1, 2 \ldots 6\}$.

$$E_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad E_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad E_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$E_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad E_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad E_6 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{8.2}$$

The six $se(3)$ bases correspond to translation in x, y, and z, and rotation about the x,y,

and z axes, respectively. Likewise, the Lie group $GA(2)$ has an associated Lie algebra $ga(3)$ which is spanned by the $GA(2)$ generator matrices G_i, $i \in \{1, 2 \ldots 6\}$.

$$G_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad G_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad G_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$
$$G_4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad G_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad G_6 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{8.3}$$

The six $ga(2)$ bases correspond to shift in x, shift in y, rotation, scaling, shear at 90°, and shear at 45°, respectively. Using these bases, the vectors $\hat{\mathbf{x}} \in \Re^6$ and $\hat{\mathbf{z}} \in \Re^6$ are defined, representing infinitesimal 3-d Euclidean and 2-d affine transformations, respectively. Using the dominant world plane parameterized by θ, ϕ, and d_z, The Jacobian matrix which maps infinitesimal changes in the camera pose to changes in the induced homography can then be defined, i.e. $J_{i,j} = \frac{\delta \hat{z}_i}{\delta \hat{x}_j}$.

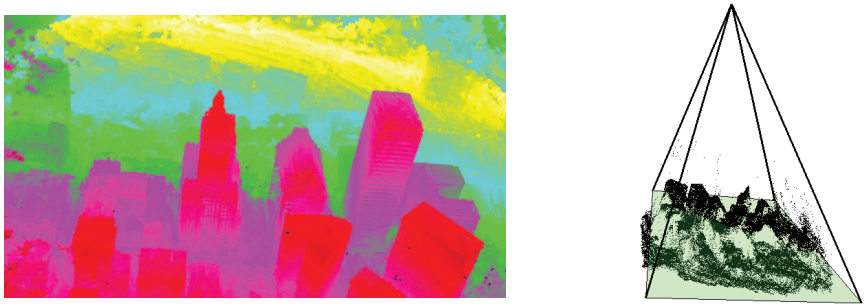
$$J = \begin{bmatrix} 1/d_z & 0 & 0 & 0 & 1 & 0 \\ 0 & 1/d_z & 0 & -1 & 0 & 0 \\ \frac{\tan(\phi)}{2d_z} & \frac{-\tan(\theta)}{2d_z} & 0 & 0 & 0 & 1 \\ \frac{-\tan(\theta)}{2d_z} & \frac{-\tan(\phi)}{2d_z} & -1/d_z & 0 & 0 & 0 \\ \frac{-\tan(\theta)}{2d_z} & \frac{\tan(\phi)}{2d_z} & 0 & 0 & 0 & 0 \\ \frac{-\tan(\phi)}{2d_z} & \frac{-\tan(\theta)}{2d_z} & 0 & 0 & 0 & 0 \end{bmatrix} \tag{8.4}$$

The derivation of this matrix is very similar to the one presented by Drummond and Cipolla [20], and the reader is referred there for the details. One difference between the Jacobian J presented here and Drummond and Cipolla's is that the world plane normal is not constrained to lie in the YZ plane here, and thus requires three parameters (as opposed to the two used by Drummond and Cipolla). Note that columns 3 through 5 are approximate only, because in general a full projective image transformation is needed to model changes caused by translation along the camera axis and rotation around an axis other than the camera's principal axis.

8.3 Refinement Algorithm

The camera refinement algorithm has three essential steps. First, a planar fit to the scene geometry being viewed must be determined in order for the Jacobian J (Equation 8.4) to be computed. Second, an expected image is rendered from the viewpoint of the current camera estimate. Finally, an image transformation mapping the expected image to the original image is computed, leading to an incremental adjustment to the camera parameters based on J. The second and third steps are then repeated until convergence is reached. Section 8.3.1 describes the first step in detail, and Section 8.3.2 the refinement iteration.

8.3.1 Plane Estimation



(a) (b)

Figure 8.3: (a) A depth map computed for a frame from the "downtown" sequence. (b) The depth map is converted to a point cloud, and a plane is fit to the points.

In order to compute the Jacobian matrix J, a dominant world plane estimate must first be computed. Using the estimated camera position, a depth value for each pixel is computed using the expected value $E[\omega(s)]$ of the depth distribution discussed in Section 3.3.1. The depth values are converted to 3-d points by backprojecting along the camera rays, and a plane can be fit to the set of backprojected points. The fit is in general poor due to the non-planarity of the scene, but is sufficient since a coarse representation of scene depth is all that is needed [20].

8.3.2 Camera Parameter Update

The vector $\hat{\mathbf{x}}$ is used to represent the Euclidean transformation which maps the original camera coordinate system to that of the refined camera estimate, and is updated iteratively until convergence is reached. The system is initialized with the identity transformation, i.e. $\hat{\mathbf{x}} = 0$. At each step, an expected image is rendered from the viewpoint defined by $\hat{\mathbf{x}}$ using the method described in Section 3.3. A homography represented by the vector $\hat{\mathbf{z}}$ is then computed which minimizes the sum of squared pixel intensity differences between the original image and the transformed expected image. (The reader is referred to Zitovia's 2003 survey [80] for a review of 2-d image registration techniques.) The inverse Jacobian J^{-1} is then used to move the estimated camera position towards the correct state, and a new expected image is generated using the adjusted state. The processes is repeated until the adjustments to $\hat{\mathbf{x}}$ fall below a fixed threshold or a maximum number of iterations is reached, in which case the refinement process is considered a failure. This refinement process is essentially a novel application of Drummond and Cipolla's [20] visual servoing algorithm to camera calibration made possible by the ability of the probabilistic scene model to render realistic and complete expected images from arbitrary viewpoints.

8.4 Validation

In order to validate the camera refinement process, the algorithm was run on the "capitol" and "downtown" sequences, using the camera estimates produced by Snavely et al.'s structure from motion algorithm [65] (manually aligned to a real-world coordinate frame as described in Section 7.1). The scene models were generated using the batch update method presented in Section 5.2 with 32 out of the 255 total images and 31 out of the 180 total images, respectively. The cameras were first refined using the structure from motion cameras as initial estimates. The mean displacements over all cameras were 0.18 and 0.16 meters for the "capitol" and "downtown" sequences, respectively (max 0.53, 0.50). The mean respective orientation offsets were 0.02 and 0.01 degrees (max 0.08, 0.04). This small movement of the refined cameras indicates that the refinement algorithm "agrees" with the cameras generated by the structure from motion algorithm.

In order to test the robustness of the refinement algorithm, the estimated camera centers and "look" directions (i.e. the z axis orientation of the camera coordinate frame) were perturbed in random directions by various amounts. Tables 8.1 and 8.2 show the displacement and orientation errors of the refined "capitol" cameras for various amounts of perturbation of the initial estimates for a randomly chosen camera which was not used in the construction of the model. Tables 8.3 and 8.4 show the results for the "downtown" sequence.

		translation offset (m)										
		0.0	**1.0**	**2.0**	**3.0**	**4.0**	**5.0**	**6.0**	**7.0**	**8.0**	**9.0**	**10.0**
orientation offset (deg.)	**0.0**	0.43	0.52	1.00	0.53	0.29	0.48	0.46	0.49	0.41	0.43	0.36
	0.5	0.66	0.42	0.18	0.51	0.48	0.37	0.34	0.30	0.32	0.57	1.39
	1.0	0.56	0.57	0.21	0.63	0.40	0.27	0.37	0.45	0.48	0.79	0.26
	1.5	0.48	0.57	0.21	3.53	0.92	31.70	5.33	6.80	0.31	8.25	16.37
	2.0	0.25	0.88	0.62	0.14	5.67	2.49	18.32	3.43	9.19	52.57	3.56
	2.5	2.34	39.12	11.61	2.88	8.19	14.43	172.31	5.49	18.54	18.06	1.01
	3.0	22.50	9.90	46.45	10.86	22.06	98.96	16.76	15.64	53.48	11.62	41.80
	3.5	3.19	31.08	10.32	28.40	23.67	20.91	25.41	21.84	78.39	15.34	10.32
	4.0	39.30	9.10	10.19	10.17	8.66	8.47	56.31	6.37	96.28	104.26	132.06

Table 8.1: Displacement errors of the refined camera estimates for the "capitol" sequence, using various amounts of random perturbation of structure from motion cameras as initial estimates.

		translation offset (m)										
		0.0	**1.0**	**2.0**	**3.0**	**4.0**	**5.0**	**6.0**	**7.0**	**8.0**	**9.0**	**10.0**
orientation offset (deg.)	**0.0**	0.07	0.09	0.16	0.06	0.03	0.05	0.05	0.05	0.04	0.04	0.04
	0.5	0.10	0.07	0.03	0.07	0.08	0.04	0.03	0.04	0.04	0.07	0.16
	1.0	0.09	0.07	0.03	0.10	0.07	0.04	0.06	0.07	0.04	0.11	0.06
	1.5	0.08	0.08	0.05	0.56	0.15	3.93	1.48	1.51	0.05	1.01	1.77
	2.0	0.07	0.12	0.08	0.04	0.99	0.31	2.27	0.57	1.13	5.91	0.57
	2.5	2.54	6.78	1.40	0.36	2.18	0.44	20.87	1.80	2.34	1.73	0.15
	3.0	2.63	3.34	5.45	2.77	3.84	11.78	3.19	3.09	7.72	2.33	3.27
	3.5	3.13	5.83	2.99	6.11	3.18	1.20	5.33	5.78	10.80	4.76	0.87
	4.0	3.87	4.86	4.40	4.49	2.66	4.75	5.90	4.47	10.09	10.05	13.35

Table 8.2: Orientation errors of the refined camera estimates for the "capitol" sequence, using various amounts of random perturbation of structure from motion cameras as initial estimates.

It is clear from the tables that the refinement algorithm robustly converges to a close approximation of the true camera parameters when the orientation of the initial estimate is accurate within roughly 1°. The accuracy of the camera center of the initial estimate

		translation offset (m)										
		0.0	**1.0**	**2.0**	**3.0**	**4.0**	**5.0**	**6.0**	**7.0**	**8.0**	**9.0**	**10.0**
orientation offset (deg.)	**0.0**	0.16	0.37	0.41	0.37	0.52	0.51	0.79	0.98	0.45	0.75	0.99
	0.5	1.06	0.84	0.79	0.77	0.42	0.89	0.65	0.49	0.79	0.53	0.95
	1.0	0.39	0.42	0.57	0.71	0.84	1.06	0.35	0.78	1.10	5.96	0.39
	1.5	39.52	0.71	0.52	0.59	16.95	0.29	0.39	12.76	1.65	8.21	1.02
	2.0	21.79	59.33	0.91	55.90	11.13	15.59	0.65	7.53	9.15	12.01	11.22
	2.5	0.38	0.90	60.31	0.93	6.14	12.66	5.41	6.89	8.66	0.97	55.97
	3.0	15.54	97.79	48.76	59.02	5.90	40.83	5.86	6.62	65.24	32.98	1.95
	3.5	13.31	14.06	66.92	31.63	33.70	27.98	16.71	17.30	37.19	14.98	20.89
	4.0	79.20	36.38	2.86	2.24	33.46	11.21	5.15	34.78	18.80	11.55	8.97

Table 8.3: Displacement errors of the refined camera estimates for the "downtown" sequence, using various amounts of random perturbation of structure from motion cameras as initial estimates. All units are meters.

		translation offset (m)										
		0.0	**1.0**	**2.0**	**3.0**	**4.0**	**5.0**	**6.0**	**7.0**	**8.0**	**9.0**	**10.0**
orientation offset (deg.)	**0.0**	0.01	0.02	0.03	0.03	0.04	0.04	0.06	0.08	0.03	0.07	0.10
	0.5	0.10	0.07	0.09	0.07	0.04	0.08	0.05	0.06	0.06	0.05	0.07
	1.0	0.04	0.06	0.05	0.06	0.08	0.10	0.03	0.07	0.09	1.50	0.03
	1.5	3.98	1.50	0.04	0.04	2.69	0.02	0.03	2.09	0.21	1.91	0.08
	2.0	3.17	6.34	0.13	5.75	2.41	2.55	0.05	1.83	1.77	0.85	1.95
	2.5	0.04	2.49	5.91	0.12	2.24	2.55	1.88	2.51	2.41	0.08	6.03
	3.0	2.15	5.92	4.98	6.36	0.62	4.02	3.00	0.88	5.86	1.58	3.15
	3.5	2.70	2.54	4.23	5.43	1.45	1.48	3.65	4.35	5.40	3.86	1.63
	4.0	3.48	3.35	3.56	3.47	2.54	3.53	3.73	4.41	3.92	3.17	3.99

Table 8.4: Orientation errors of the refined camera estimates for the "downtown" sequence, using various amounts of random perturbation of structure from motion cameras as initial estimates. All units are degrees.

did not appear to have much of an effect on the accuracy of the refined camera within the range of 0 to 10 meters of initial error. This approximate region of convergence is within the limits of what is currently available from global positioning devices and inertial navigation systems.

Although it is useful to determine what range of initial error the refinement algorithm is capable of recovering from, these numbers do not tell the whole story. When the cameras are reasonably distant from the scene (as typically are in aerial imagery), small errors in translation can be offset by errors in orientation, resulting in a minimal effect on the projection errors of points in the scene. It is therefore useful to measure the projection error

of points in the scene before and after the optimization process. Figure 8.4 shows a scatter plot of the mean projection error of 3-d scene points using the refined camera estimates vs. the initial (noisy) camera estimates based on the experiments presented above. The 3-d point locations and ground truth projections are again provided by the structure from motion algorithm. The refinement algorithm is consistently able to recover from projection errors of up to 50 pixels, but does not reliably converge for larger errors.

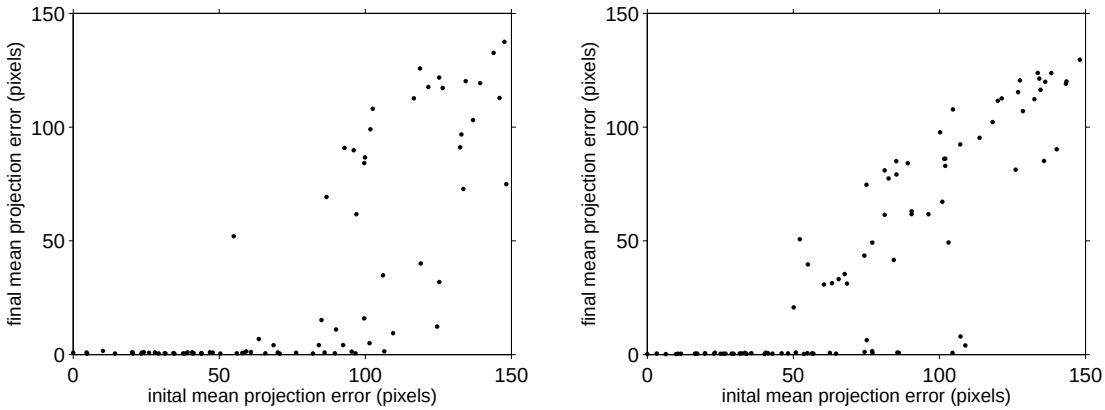


Figure 8.4: Mean projection errors after refinement as a function of the mean initial projection error for the (a) "capitol" and (b) "downtown" sequences.

8.5 Conclusions

The camera refinement algorithm presented in this chapter is a robust and useful tool, provided that a sufficiently accurate model exists for the purpose of generating the expected images used for visual feedback. The existing model can produced using by previously collected and calibrated data or, if they are sufficiently accurate, the camera estimates themselves. The refinement algorithm itself is essentially a novel application of visual servoing, and demonstrates the usefulness of the probabilistic model in the application area of camera calibration.

Chapter 9

Conclusions and Future Work

This thesis has presented a novel continuous probabilistic model for representing scene geometry and appearance based on aerial imagery. By generalizing previous (discrete) probabilistic models in this way, implementations such as the presented octree-based model can achieve multiple orders of magnitude in storage savings by non-uniformly sampling the volume of interest. A generalization of a previous online probabilistic reconstruction algorithm [53] was presented, as well as a novel reconstruction algorithm which is able to process all available data at once. The utility of the model for two critical application areas, 3-d point localization and novel viewpoint rendering, was presented and evaluated, leading to the conclusion that the ability of the octree-based implementation to represent large areas with fine detail gives it a distinct advantage over previous methods based on discrete fixed-grid models. Because the presented algorithms rely on a form of photo-consistency metric, pixel-accurate camera calibration is critical. For this reason, a camera refinement algorithm based on the model was presented in Chapter 8 was presented which shown to be able to recover from up to 50 pixels of calibration error.

The utility of the presented model is not however limited to the applications focused on in this thesis. Much future work will focus on exploring new applications which can benefit from the increased resolution and coverage made possible by the continuous model. One example of such an application is object recognition. Previous work by Mundy and Ozcanali [51] used a discrete, fixed grid voxel model. When replaced with the octree-based model, Ozcanali has shown recognition results to improve significantly, as shown in

Figure 9.1.

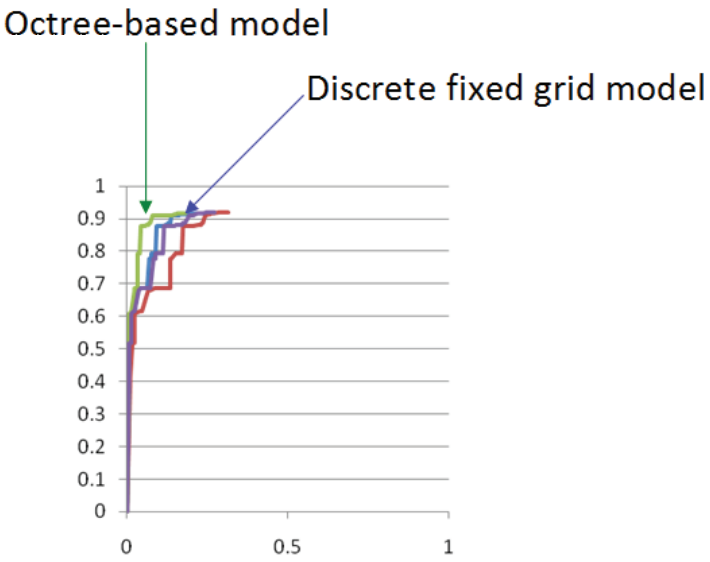


Figure 9.1: A series of object detection [51] ROC curves. Using the presented octree-based continuous model produces an increase in accuracy over the discrete fixed model. Graph courtesy of Ozge C. Ozcanali.

In addition to exploring new application areas, several means of potential improvement to the model and implementation will be explored in the near future. One example of such work is that of speed improvements to the reconstruction and rendering algorithms. Off the shelf computer graphics cards are becoming more and more powerful, with current models able to process data multiple orders of magnitude faster than CPU's for parallelizable algorithms. Work is underway to leverage this enormous processing power to speed up the algorithms presented in this thesis, and a real-time interactive novel viewpoint rendering implementation has already been realized.

A second area of future work involves exploring the representation of the occlusion densities themselves. Currently, unobserved regions are initialized with low occlusion density values which are permitted to increase as evidence from the image data mounts. Ideally the lack of knowledge about a given point could be represented by a degree of uncertainty, and not by the occlusion density itself. In other words, the model should be able to distinguish between a point which is probably not occluding and a point for which there is little or no

evidence either way. Mundy has explored this concept using subjective logic [50] for the discrete voxel case. Future work will involve developing and refining the theory as it relates to the continuous model. It is expected that such a model will ease or eliminate the need for the damping control to the batch update algorithm, which is partially a result of the insufficiently accurate initial prior state of the model.

Finally, although this thesis has focused primarily on aerial imagery, the model is of course not limited to that class of data, as evidenced by its ability to produce accurate reconstructions using the Middlebury multi-view stereo evaluation test data 6. Experiments which focus on ground-based data or combinations of ground-based and aerial data will be explored, giving rise to the potential challenges of dealing with modes of data containing vastly varying resolution.

The significant gains over traditional probabilistic scene models demonstrated here, along with the many avenues for potential improvement and expansion of capabilities, indicate a promising future for the model whose basis has been presented in this thesis.

Bibliography

[1] E. H. Adelson and J.R. Bergen. *Computational Models of Visual Processing*, chapter Chapter 1, The Plenoptic Function and the Elements of Early Vision. The MIT Press, Cambridge, MA, 1991.

[2] Adrien Auclair, Nicole Vincent, and Laurent D. Cohen. Using point correspondences without projective deformation for multi-view stereo reconstruction. In *International Conference on Image Processing*, 2008.

[3] H.H. Baker and T.O. Binford. Depth from edge and intensity based stereo. In *Proc. 7th International Joint Conference on Artificial Intelligence*, pages 631–636, 1982.

[4] Rahul Bhotika. *Scene-Space Methods for Inference of 3-d Shape and Motion*. PhD thesis, University of Rochester, 2003.

[5] Rahul Bhotika, David J. Fleet, and Kiriakos N. Kutulakos. A probabilistic theory of occupancy and emptiness. In *European Conference on Computer Vision*, pages 112–132, 2002.

[6] Derek Bradley, Tamy Boubekeur, and Wolfgang Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *Computer Vision and Pattern Recognition*, 2008.

[7] A. Broadhurst, T.W. Drummond, and R. Cipolla. A probabalistic framework for space carving. In *International Conference on Computer Vision*, pages pp. 288–393, 2001.

[8] Adrian Broadhurst. *A Probabilistic Framework for Space Carving*. PhD thesis, Cambridge University, 2001.

[9] M. Brown and D.G. Lowe. Unsupervised 3d object recognition and reconstruction in unordered datasets. In *5th International Conference on 3D Imaging and Modeling (3DIM)*, 2005.

[10] Neill D.F. Campbell, George Vogiatzis, Carlos Hernández, and Roberto Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *10th European Conference on Computer Vision*, volume 5302 of *LNCS*, pages 766–779, 2008.

[11] Robert Collins, A. Hanson, E. Riseman, C. Jaynes, F. Stolle, X. Wang, and Y. Cheng. Umass progress in 3d building model acquisition. In *Arpa Image Understanding Workshop*, pages 305–315, February 1996.

[12] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool. 3d urban scene modeling integrating recognition and reconstruction. *International Journal of Computer Vision*, 78(2-3):121–141, July 2008.

[13] H.S.M. Coxeter. *Introduction to Geometry, 2nd ed.*, chapter 13.7: "Barycentric Coordinates", pages 216–221. Wiley, 1969.

[14] Daniel Crispell, Joseph Mundy, and Gabriel Taubin. Parallax-free aerial video regisration. In *British Machine Vision Conference*, 2008.

[15] Paul Debevec, Yizhou Yu, and George Borshukov. Efficient view-dependant image-based rendering with projective texture-mapping. In *9th Eurographics Rendering Workshop*, 1998.

[16] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs. In *SIGGRAPH '96*, August 1996.

[17] A. Delaunoy, E. Prados, P. Gargallo, J.-P. Pons, and P. Sturm. Minimizing the multi-view stereo reprojection error for triangular surface meshes. In *British Machine Vision Conference*, 2008.

[18] Anthony Dick, Phil Torr, and Roberto Cipolla. Automatic 3d modelling of architecture. In *BMVC*, 2000.

[19] Julie Dorsey, Holly Rushmeier, and François Sillion. *Digital Modeling of Material Appearance*. Morgan Kauffmann / Elsevier, 2007.

[20] Tom Drummond and Robert Cipolla. Application of lie algebras to visual servoing. *International Journal of Computer Vision*, 37(1):pp 21–41, 2000.

[21] O. Firschein and T.M. Strat, editors. *Radius: Image Understanding for Imagery Intelligence*. Morgan Kaufmann, 1997.

[22] Andrew Fitzgibbon, Yonatan Wexler, and Andrew Zisserman. Image-based rendering using image-based priors. *International Journal of Computer Vision*, 63(2):141–151, July 2005.

[23] Jean-Sébastien Franco and Edmond Boyer. Exact polyhedral visual hulls. In *Britich Machine Vision Conference (BMVC)*, 2003.

[24] Pascal Fua. Model-based optimization: An approach to fast, accurate, and consistent site modeling from imagery. In O. Firschein and T.M. Strat, editors, *RADIUS: Image Understanding for Intelligence Imagery*, pages 903–908. Morgan Kaufmann, 1997.

[25] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multi-view stereopsis. In *Computer Vision and Pattern Recognition*, 2007.

[26] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (To Appear)*, 2009.

[27] S. Gibson, R.J. Hubbold, J. Cook, and T.L.J. Howard. Interactive reconstruction of virtual environments from video sequences. *Computers & Graphics*, 27:293–301, 2003.

[28] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2007.

[29] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *SIGGRAPH '96*, 1996.

[30] Armin Gruen and Xinhua Wang. Cc-modeler: A topology generator for 3-d city models. *International Archives of Photogrammetry and Remote Sensing*, 32(4):286–295, 1998.

[31] M Habbecke and Leif Kobbelt. A surface-growing approach to multi-view stereo. In *Computer Vision and Pattern Recognition (CVPR)*, 2007.

[32] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, 1988.

[33] Richard Hartley and Andrew Zisserman. *Multiple View Geometery in Computer Vision*. Cambridge University Press, 2nd edition, 2003.

[34] Aaron J. Heller and Lynn H. Quam. The radius common development environment. In O. Firschein and T.M. Strat, editors, *RADIUS: Image Understanding for Intelligence Imagery*. Morgan Kaufmann, 1997.

[35] Carlos Hernández, George Vogiatzis, and Roberto Cipolla. Multi-view photometric streo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):548–554, March 2008.

[36] Vu Hoang Hiep, Renaud Keriven, Patrick Labatut, and Jean-Philippe Pons. Towards high-resolution large-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[37] Jinhui Hu, Suya You, and Ulrich Neumann. Approaches to large-scale urban modeling. *IEEE Computer Graphics and Applications*, 23(6):62–69, 2003.

[38] Jinhui Hu, Suya You, and Ulrich Neumann. Integrating lidar, aerial image and ground images for comlete urban building modeling. In *Proc. 3rd International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, 2006.

[39] Seth Hutchinson, Greg Hagar, and Peter Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):pp 651–670, 1996.

[40] Leif Kobbelt and Mario Botsch. A survey of point-based techniques in computer graphics. *Computer & Graphics*, 28(6):801–814, 2004.

[41] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000.

[42] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):150–162, 1994.

[43] Marc Levoy and Pat Hanrahan. Light field rendering. In *SIGGRAPH '96*, 1996.

[44] Lingyun Liu and Ioannis Stamos. Automatic 3d to 2d registration for the photorealistic rendering of urban scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 137–143, 2005.

[45] Lingyun Liu, Ioannis Stamos, Gene Yu, George Wolberg, and Siavash Zokai. Multiview geometry for texture mapping 2d images onto 3d range data. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[46] W.E. Lorensen and H.E. Cline. Marching cubes: a high resolution 3d surface construction algorithm. In *SIGGRAPH '87*, pages 163–169, 1987.

[47] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[48] Yi Ma, Stefano Soatto, Jana Košecká, and S. Shankar Sastry. *An Invitation to 3-D Vision*. Springer-Verlag, 2004.

[49] G. M. Morton. A computer oriented geodetic data base and a new technique in file sequencing. Technical report, IBM Ltd., Ottawa, Canada, 1966.

[50] Joseph L. Mundy. What is obseved, what is unknown. Brown University internal report, April 2009.

[51] Joseph L. Mundy and Ozge C. Ozcanli. Uncertain geometry: A new approach to modeling for recognition. In *Proceedings of SPIE Defense, Security, and Sensing Conference*, April 2009.

[52] Thomas Pollard. *Comprehensive 3-d Change Detection Using Volumetric Appearance Modeling*. PhD thesis, Brown University, 2009.

[53] Thomas Pollard and Joseph Mundy. Change detection in a 3-d world. In *Computer Vision and Pattern Recognition*, July 2007.

[54] M. Pollefeys, D. Nister, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3-d reconstruction from video. *International Journal of Computer Vision*, 78(2-3):143–167, July 2008.

[55] Marc Pollefeys, Luc Van Gool, Maarten Vergauwen, Frank Verbiest, Kurt Cornelis, Jan Tops, and Reinhard Koch. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232, September 2004.

[56] A. Rosenfeld, H. Samet, C. Shaffer, and R.E. Webber. Application of hierarchical data structures to geographic information systems: phase II. Technical Report TR-1327, University of Maryland, College Park, MD, 1983.

[57] Jeremy s. De Bonet and Paul Viola. Roxels: Responsibility weighted 3d volume reconstruction. In *International Conference on Computer Vision*, pages 418–425, 1999.

[58] Hanan Samet. *Applications of spatial data structures: computer graphics, image processing, and GIS*. Addison-Wesley Publishing Company, Inc., 1990.

[59] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1–3):7–42, April 2002.

[60] Grant Schindler, Panchapagesan Krishnamurthy, and Frank Dellaert. Line-based structure from motion for urban environments. In *3DPVT*, 2006.

[61] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Computer Vision and Pattern Recognition*, volume 1, pages 519–526, 2006.

[62] Steven M. Seitz and Charles R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *In Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 1067–1073, 1997.

[63] Greg Slabaugh, Bruce Culbertson, Tom Malzbender, and Ron Schafer. A survey of methods for volumetric scene reconstruction from photographs. In *Proceedings of the International Workshop on Volume Graphics*, 2001.

[64] Gregory G. Slabaugh, W. Bruce Culbertson, Thomas Malzbender, Mark R. Stevens, and Ronald W. Schafer. Methods for volumetric reconstruction of visual scenes. *International Journal of Computer Vision*, 57(3):179–199, 2004.

[65] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *International Journal of Computer Vision*, 80(2):189–210, November 2008.

[66] Ioannis Stamos and Marius Leordeanu. Automated feature-based range registration of urban scenes of large scale. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 555–561, 2003.

[67] Ionnis Stamos. Automated 3d modeling of urban environments. In *ISPRS International Workshop on 3D Virtual Reconstruction and Visualization of Complex Architectures (3D-ARCH)*, 2009.

[68] Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition*, pages pp. 246–252, 1999.

[69] George Vogiatzis, Carlos Hernández, Philip H.S. Torr, and Roberto Cipolla. Multi-view stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12), 2007.

[70] Zeng-Fu Wang and Zhi-Gang Zheng. A region based stereo matching algorithm using cooperative optimization. In *Computer Vision and Pattern Recognition*, 2008.

[71] Joe Warren. Barycentric coordinates for convex polytopes. *Advances in Computational Mathematics*, 6:97–108, 1996.

[72] Eric W. Weisstein. Chi-squared distribution. From MathWorld-A Wolfram Web Resource.

[73] O.J. Woodford, I.D. Reid, and A. W. Fitzgibbon. Efficient new-view synthesis using pairwise dictionary priors. In *Computer Vision and Pattern Recognition (CVPR)*, 2007.

[74] O.J. Woodford, I.D. Reid, P.H.S. Torr, and A. W. Fitzgibbon. On new view synthesis using multiview stereo. In *Proceedings of British Machine Vision Conference (BMVC)*, 2007.

[75] Qingxiong Yang, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3), March 2009.

[76] Qingziong Yang, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. In *Computer Vision and Pattern Recognition*, pages 2347–2354, 2006.

[77] Annie Yao and Andrew Calway. Dense 3-d structure from image sequences using probabilistic depth carving. In *British Machine Vision Conference*, 2003.

[78] Suya You, Jinhui Hu, Ulrich Neumann, and Pamela Fox. Urban site modeling from lidar. In *Proc. 2nd International Workshop on Computer Graphics and Geometric Modeling (CGGM)*, 2003.

[79] Andrei Zaharescu, Edmond Boyer, and Radu P. Horaud. Transformesh: a topology-adaptive mesh-based approach to surface evolution. In *Proceedings of the Eighth Asian Conference on Computer Vision*, 2007.

[80] Barbara Zitová and Jan Flusser. Image registration methods: A survey. *Image and Vision Computing*, 21:977–1000, 2003.