DRESDEN UNIVERSITY OF TECHNOLOGY

Faculty of Information Technology
Institute of Artificial Intelligence
Neurocomputation Group

- Diploma Thesis -

Concurrent Stereo Reconstruction

Martin R. Oswald

supervised by

Doz. Dr. rer. nat. habil. Boris Flach

Dresden, June 4, 2007

Abstract

This thesis presents a probabilistic approach to multi view stereo reconstruction from calibrated images. Together with a voxel based world model a Markov Random Field is used to describe the problem of stereo reconstruction as a structural recognition task. A specific scene representation has been developed for stereo reconstruction with the ability to keep ambiguities within the reconstruction result for further evaluation. Based on several implementation specific methods being used in combination with Gibbs sampling an efficiently computable and flexible framework for stereo reconstruction is presented to solve the Bayesian estimation task. Different a priori and observation models are investigated to adapt the general model to specific reconstruction tasks and unsupervised learning methods have been used to reduce the number of model parameters. Evaluation on sufficient test data proves the viability of the proposed method.

Kurzfassung

Diese Arbeit diskutiert einen wahrscheinlichkeitsbasierten Ansatz zur Stereorekonstruktion anhand mehrerer kalibrierter Kamerabilder. Auf einem Voxel-basierten Weltmodell wird ein Markovsches Zufallsfeld definiert, welches die Aufgabe der Stereorekonstruktion als strukturelles Erkennungsproblem beschreibt. Darauf aufbauend wurde eine Szenenbeschreibung entwickelt, die es ermöglicht bestehende Mehrdeutigkeiten zur weiteren Verarbeitung im Rekonstruktionergebnis zu halten. Mit Hilfe von Gibbs Sampling und mehreren implementierungsspezifischen Methoden wird ein flexibles Modell vorgestellt, welches die Stereorekonstruktion als Bayessche Aufgabe effizient löst.
Zusätzlich werden in der Arbeit unterschiedliche A-priori- und Beobachtungsmodelle untersucht, welche die vorgestellte, allgemeine Erkennungsmethode an spezifische Rekonstruktionsaufgaben anpasst. Unter der Verwendung von Methoden des unüberwachten Lernens konnte die Anzahl der benötigten Modellparameter reduziert werden. Experimente mit mehreren natürlichen und künstlichen Beispielszenen zeigen die Anwendbarkeit des Modells für die Stereorekonstruktion.

Konkurrente Stereorekonstruktion

Student	Oswald, Martin
Betreuer	Doz.Dr. Boris Flach
Beginn	01.10.2006

Zielstellung:

Die 3D-Rekonstruktion allgemeiner Szenen ist auch im Fall mehrerer Aufnahmen eine komplexe Aufgabe. Ursache dafür sind zum Einen Verdeckungen und zum Anderen die nicht-lambertsche Reflexion von Objektoberflächen. Deswegen sind gute Rekonstruktionsresultate mit allgemeinen Verfahren (wie z.B. dem Space-carving) nur unter sehr rigiden Voraussetzungen erreichbar. Lässt man aber letztere fallen, so ist die Auflösung der auftretenden Mehrdeutigkeiten höchstwahrscheinlich nur durch Rückkopplung von abstrakteren, semantischen Interpretationen einer Szene möglich. Ziel der Arbeit ist deswegen die Entwicklung eines Ansatzes zur konkurrenten 3D-Rekonstruktion von Szenen anhand von kalibrierten Kameraaufnahmen. Dabei sollen statt wie bisher eine 3D-Szenenbeschreibung zu suchen, hier mehrere konkurrierende Szenenbeschreibungen als Antwort zugelassen werden. Bei der Modellierung der (schwachen) a-priori Annahmen über die Szene und des Zusammenhangs zwischen Szene und Messung sind folgende Gesichtspunkte zu berücksichtigen:

- Die Farbdifferenzen korrespondierender Pixel lassen sich durch eine Verteilung beschreiben, die anzulernen ist.
- Die zu entwickelnde formale Beschreibung der (Mengen von) Szenenkonfigurationen soll durch Verdeckungen und homogene Farbtexturierung hervorgerufene Mehrdeutigkeiten berücksichtigen. Ergebnis der Erkennung ist eine (natürlich möglichst kleine) Schar von Szenenkonfigurationen.
- Das Modell soll Rückkopplungen semantisch orientierter Verarbeitungsschichten zulassen.

Aus dem entwickelten Modell ist ein effizienter Algorithmus abzuleiten, der die entstehende Erkennungsaufgabe zumindest näherungsweise löst. Dieser ist zu implementieren und anhand künstlicher und realer Szenen experimentell zu untersuchen.

Literatur:

1. Michail I. Schlesinger and Vaclav Hlavac. Ten Lectures on Statistical and Structural Pattern Recognition, volmne 24 of Computational Imaging and Vision. Kluwer Academic Press, 2002.
2. Stan Z. Li. Markov Raudom Field Modeliug in Image Analysis. Computer Science Workbench. Springer Verlag, 2001.
3. A. Broadhurst, T. Drummond, and R. Cipolla. A probabilistic framework for the Space Carving algorithm. In Proc. 8th International Conference of Computer Vision, pages 388-393, Vancouver, Canada, 2001. IEEE Computer Society Press.

Neben der Papierversion sind dem betreuenden Hochschullehrer eine Zusammenfassung (etwa 10 Zeilen) und der Volltext in elektronischer Form als Pdf- oder Ps-file zu übergeben.

Contents

Abstract iii

Topic v

1 Introduction 1

2 Stereo Reconstruction 3
- 2.1 Issues and Problems . . . 3
  - 2.1.1 Visual Hull . . . 3
  - 2.1.2 Scene Illumination and Object Materials . . . 4
  - 2.1.3 Pixel Span Volume . . . 4
  - 2.1.4 Surface Texture . . . 5
  - 2.1.5 Visibility and Occlusion . . . 6
  - 2.1.6 Photo Hull . . . 6
- 2.2 World Model . . . 7
  - 2.2.1 World Properties . . . 7
  - 2.2.2 Model Description . . . 9
- 2.3 Summary . . . 10

3 Probabilistic Model 11
- 3.1 Notation . . . 11
  - 3.1.1 Scene . . . 11
  - 3.1.2 Voxel Relations and Labels . . . 12
  - 3.1.3 Helpful Definitions . . . 13
- 3.2 Model Definition . . . 15
  - 3.2.1 A Priori Model . . . 15
  - 3.2.2 Observation Model . . . 15
  - 3.2.3 Joint Model . . . 18
- 3.3 Summary and Discussion . . . 18

4 Recognition 21
- 4.1 Bayesian Estimation . . . 21
- 4.2 Color Distribution Learning . . . 22
  - 4.2.1 Voxel Color Distribution . . . 23
  - 4.2.2 Background Color Distribution . . . 24
- 4.3 Gibbs Sampling . . . 26
- 4.4 Visibility Function and Label Semantics . . . 27
- 4.5 Feedbacks from an other module . . . 29
- 4.6 Summary . . . 29

5 Implementation 31

5.1 Logarithmic Model . . . 31
5.2 Sampling Process . . . 32
5.3 Pixel-Voxel Size Proportion . . . 34
5.4 Geometry Calculations . . . 35
5.4.1 Coordinate Systems . . . 35
5.4.2 Coordinate Transformations . . . 35

6 Evaluation **37**
6.1 Data Sets . . . 37
6.1.1 Handmade data sets . . . 37
6.1.2 Middlebury data sets . . . 37
6.1.3 Image RGB-Clustering . . . 38
6.2 Functionality of the Model . . . 38
6.3 Sampling . . . 40
6.4 Reconstruction Results . . . 42
6.5 Color A Priori Model . . . 48
6.6 Parameters . . . 49
6.7 Summary and Discussion . . . 51

7 Summary and Outlook **53**
7.1 Conclusion . . . 53
7.2 Future Work . . . 53

A Mathematical Issues **55**
A.1 Transformation Matrices . . . 55
A.2 Derivations and Lemmas . . . 55
A.2.1 From Bayesian to MAP-Estimation . . . 55
A.2.2 Normalization Factor for the Foreground Observation Model . . . 57
A.2.3 Shannon Theorem . . . 59

B Implementation Details **61**
B.1 Ray Traversal . . . 61
B.2 Speed Ups and Memory Savings . . . 61
B.2.1 Visibility Maps . . . 61
B.2.2 Next Occupied Voxel Maps . . . 61
B.2.3 Projection Cache . . . 62
B.2.4 Bit Data Field . . . 62
B.2.5 Color Map Index Cache . . . 63
B.3 Processing Time . . . 63
B.4 Software Structure . . . 64

Notation **65**

References **67**

Declaration **69**

1 Introduction

Stereo reconstruction has been a research interest for many years and is still a very challenging task. The field of applications for stereo reconstruction is huge and apart from the well-known idea to reconstruct 3D objects of the real world from photographs it is an essential part not only in medicine diagnostics and research, but also in applied science, e.g. for the nondestructive examination of materials with supersonic, x-ray or magnetic resonance imaging. Therefore, stereo reconstruction is applicable to a wide field of tasks, but most of the presented methods assume very specific reconstruction tasks to handle the many ambiguities in stereo reconstruction. This thesis investigates a flexible probabilistic approach to the problem of stereo reconstruction in which specific knowledge about a particular reconstruction is rather additional information than a fixed part of the model.

It is common in stereo reconstruction research that the result of the reconstruction is a single scene being as close as possible to the true scene. In contrast, the task of stereo reconstruction is effectively ambiguous by definition. These ambiguities are usually dissolved by strong assumptions while loosing generality at the same time. In contrast, the proposed method will explicitly allow ambiguities while making only weak assumptions about the scene. The main aspect is to reduce the number of possible reconstruction results while keeping the ambiguities. Therefore, this work can be considered as a preprocessing step in stereo reconstruction. The results can then be used for further postprocessing with more knowledge about the scene or stronger assumptions. Furthermore, the proposed model can also be used within an iterative processing scheme as depicted in figure 1.1, but only the first module is discussed in this thesis. The input of this framework is a set of camera images with known camera positions and orientations which is then processed to obtain a (small) set of possible scene configurations.

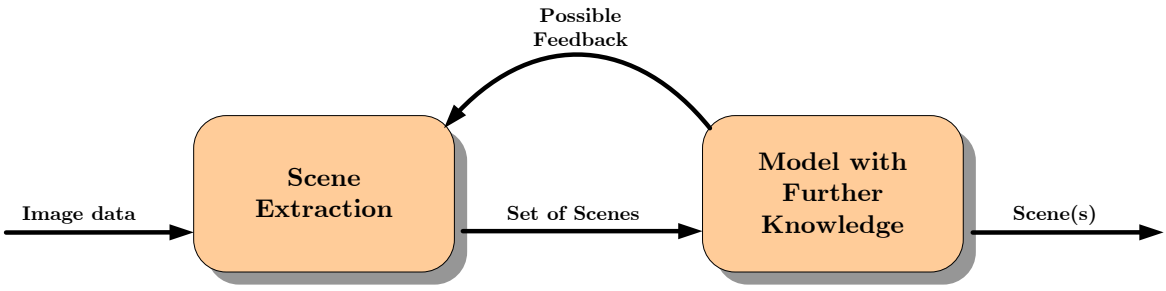


Figure 1.1: Processing Scheme

The thesis is organized as follows. Chapter 2 gives an introduction to stereo reconstruction and focuses on its difficulties. A probabilistic model for stereo reconstruction is defined in chapter 3. Chapter 4 then discusses methods to make the calculations of the probabilistic model feasible and how the number of parameters can be reduced. This is continued in chapter 5 which discusses important issues with respect to the implementation of the model. Chapter 6 presents test and evaluations of the proposed model before the thesis finishes with a summary and an outlook in chapter 7.

2 Stereo Reconstruction

The ambition in the field of stereo reconstruction is often the imitation of the human visual system for depth recovery, but stereo reconstruction can be found in many applications, e.g. in medicine for evaluation of computed tomography data, and is therefore not necessarily bounded to two rectified cameras. In general, the goal of stereo reconstruction is to use information observed from a subspace to obtain information about the unobservable space from which the data is originally coming from. In this thesis, information about a scene in three dimensional (3D) space is obtained by combining the information of 2D camera images. Given the unobservable data in space, the observed information can be described by a function which discards data (by projection) and is thus not bijective. In most cases, the inverse function does not have a single solution. The difficulty of this task is therefore to extract as much information as possible from the observed data to reduce the large number of possible solutions.

2.1 Issues and Problems

The following subsections describe the basics of stereo reconstruction, give some elementary definitions and identify the main difficulties of this task.

2.1.1 Visual Hull

Consider n cameras observing an arbitrary scene and let S^i be the viewing volume of camera i (camera indices will always be superscripted in this thesis) the union of all pairwise camera view volume intersections is the reconstructable scene volume RSV.

$$RSV = \bigcup_{i=1}^{n} \left[\forall j \neq i, j = 1, \ldots, n : S^i \cap S^j \right] \quad (2.1)$$

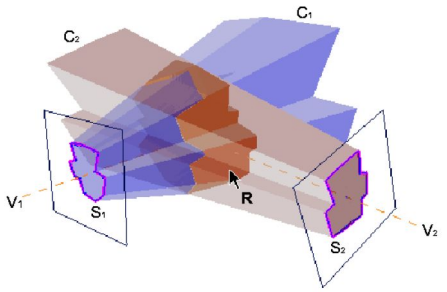


Figure 2.1: Silhouette volume intersection space (picture from [Lau03])

Only scene parts in the RSV can potentially be reconstructed since it needs at least the observation of two cameras two obtain depth information from images.

Without any assumptions about the scene, surface structure, lightning models, etc. the RSV is the best reconstruction result which can be obtained from the images because the "RSV-chunk" would be a consistent reconstruction if it is painted according to the camera images.

However, the RSV can be further reduced if silhouette information of the scene parts to be reconstructed are available for the images. Using this information the RSV can be reduced to the $VisualHull$ which has been defined by Laurentini [Lau94] and can be described similary to equation (2.1). An example of the $VisualHull$ for two cameras can be seen in figure 2.1.

2.1.2 Scene Illumination and Object Materials

The combination of light and surface material has large influence on how difficult it is to solve the reconstruction task in which knowledge about the scene illumination is usually not given. Especially reflections on object surfaces cause their appearances in camera images to change with the camera position and make the reconstruction task more complex. Thus, the kind of illumination of the scene (directed or diffuse light) has strong impact on the one hand and the material and surface properties on the other. The latter ones can be approximated with a class of functions called *bidirectional reflection distribution functions* (BRDF). They can be described as a (material dependent) weighted sum of diffuse, specular and glossy reflections (see fig. 2.2). However, to model specular and glossy reflections one needs to know about the light direction (mainly light source positions). Since the estimation of light sources only from camera images is a very difficult task, a Lambertian-like radiance model describing only diffuse reflections will be assumed in this thesis. It is worth to mention that there exist approaches to handle the problems of reflections with better local consistency functions [Wan] which might be applicable to the proposed reconstruction model too. Moreover, the presence of semi-transparent objects also leads to a very complex reconstruction task and for this reason they are assumed to be absent.

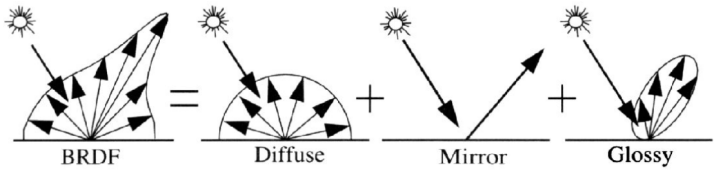


Figure 2.2: *Bidirectional reflection distribution function* as model for surface reflection.(from [Deu01])

2.1.3 Pixel Span Volume

Using perspective cameras, the span volume intersection of pixels from different cameras is a diamond shaped area (see figure 2.3). Considering the case for several cameras with arbitrary positions observing an object, the shape of the areas containing the same color information from all cameras will be a lot more complex. Furthermore, there will be many areas with contradicting color information from different camera pairs due to discretization and calibration errors. In this thesis the problem will be tackled by using a world

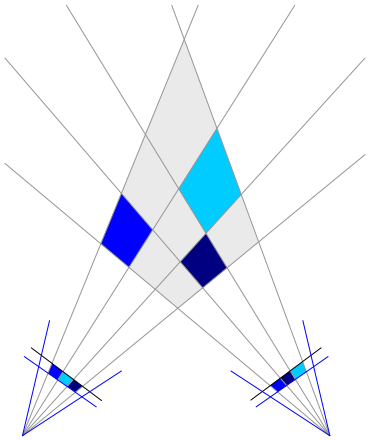


Figure 2.3: Pixel span volume

model which is described by small volume elements (voxels). This approach simplifies the fusion of color information from different cameras and will be introduced in section 2.2.

2.1.4 Surface Texture

Homogeneously Colored Surfaces

Imagine the marginal cases of completely homogeneously colored surfaces and perfectly textured surfaces[1]. Figure 2.4(a) depicts a top view of a wall (thick line) which consists of a homogeneously colored part in the middle (magenta) and perfectly textured surface parts (black) on both sides. The union of all pixel span volume intersections of pixels having the same color leads to a large area (red) of uncertainty. Thus, the surface characteristics in this area are lost due to projection and can not be correctly estimated without additional (e.g. a priori) information.

Artifacts

The same situation as above gets more complicated if two cameras observe several homogeneously colored surface parts having the same color. Then, multiple competing scene parts of uncertainty regarding the reconstruction arise. This means, it cannot be distinguished between the cases of reconstructing a pillar in front of the wall, reconstructing something in between the walls or assume all these areas are empty and reconstruct something behind the wall. Based on the observation all competing reconstruction cases are evenly likely. Thus, one solution excludes the others.

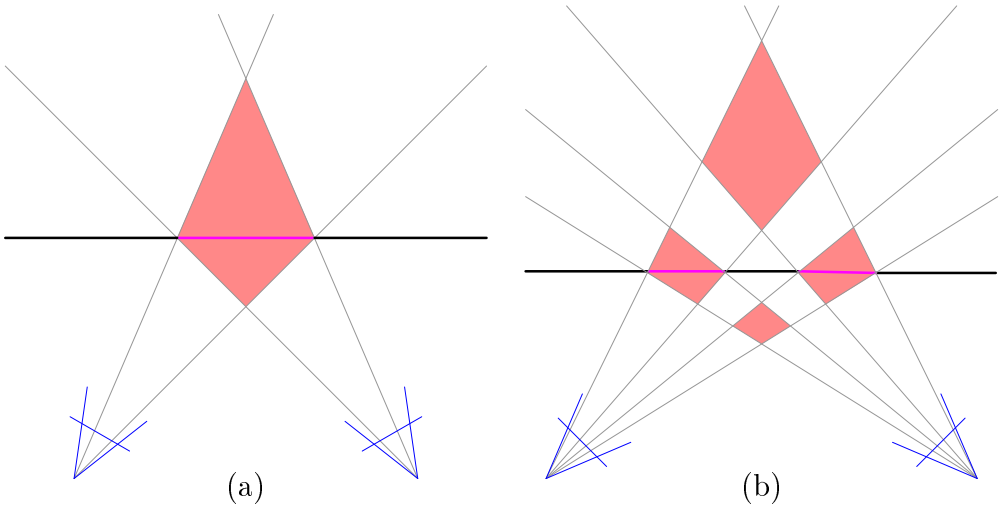


Figure 2.4: Lack of information due to homogeneously colored surface parts: (a) Area of uncertainty due to a homogeneous texturing. (b) Competing areas of uncertainty due to several homogeneously colored scene parts having the same color.

[1] meaning the opposite of completely homogeneously colored surfaces with respect to stereo reconstruction

2.1.5 Visibility and Occlusion

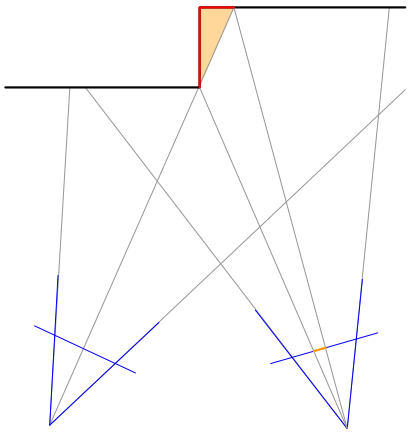


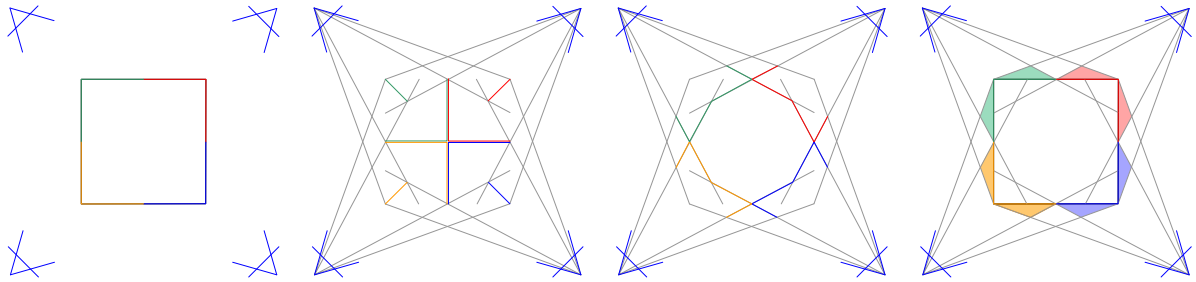
Figure 2.5: Effect of occlusion.

A big problem in stereo reconstruction is the estimation of visibility which - together with the reconstruction task - constitutes a chicken-egg-problem: 1. In order to reconstruct a scene one needs to know the visibility configuration and 2. to calculate the visibility of surfaces one needs to know the scene. The proposed method will simplify this problem because for the presented Markov-Random-Field approach it is enough to describe this global problem (semi-) locally.

Another problem is the occurrence of occlusions due to scene geometry and camera positions resulting in scene parts being not binocularly observable and thus not reconstructable since it needs at least two cameras to infer about depth. In case that only one camera observes part of a surface similar areas of uncertainty regarding the reconstruction occur because one cannot infer a depth value from a single camera. Figure 2.5 illustrates an example showing a top view of a wall with a corner observed by two cameras where the occluded area is shown in orange.

2.1.6 Photo Hull

Consider figure 2.6 which shows a top view of a homogeneously textured cube observed by four cameras (a). The figure shows some examples of the wide range of possible reconstructions (b)-(d). Kutulakos et al. [Kut99] defined the property of photo consistency. In short, photo consistency is defined incrementally beginning from a point in the reconstruction space which is said to be photo consistent if the radiance of the point in camera direction is equal to the color in its projection. This is then used to define the photo consistency of shapes or objects with a set of cameras.



(a) (b) (c) (d)

Figure 2.6: Example Scene showing a top view of a homogeneously colored cube observed by four cameras. (a) scene configuration. (b) minimal photo consistent scene. (c) another minmal photo consistent scene. (d) photo hull of the scene.

Based on the definition of photo consistency the *Photo Hull* is defined as the union of all possible reconstructed shapes which are photo consistent with the camera images. Trivially, the *Photo Hull* is also photo consistent with the all camera images and is the

largest possible reconstructed shape having this property. In the example of figure 2.6 the picture in (d) shows the *Photo Hull* of scene object. Furthermore, the *Photo Hull* is the outcome of the space carving algorithm which is presented in the same paper [Kut99]. Due to the fact that the algorithm uses a single threshold to separate the set of empty voxels from the set of object (surface) voxels it does not check if there are scenes inside the *Photo Hull* being more likely, because the algorithm works like a sculptor while carving away all voxels which do not fit to the camera images.

2.2 World Model

A crucial part of every stereo reconstruction model is its world model since it will limit the generality and accuracy of the reconstruction method on the one hand and will be the mathematical base for the algorithm on the other. Before a world model is defined for the given task some general properties in world modeling with respect to stereo reconstruction are investigated.

2.2.1 World Properties

Using the assumptions described in the previous section (2.1), e.g. the Lambertian-like radiance model, the properties of materials and their surface become unimportant and the view on the world can be strongly simplified by distinguishing between gaseous parts and solid parts only - or simply (*empty*, *material*). Besides that the view on the world in stereo reconstruction is additionally limited by the cameras and often many scene parts remain unknown. How should these areas been labeled?

In short, the visibility configuration of a scene does not require any further labels for reconstruction, but it is important to interpret the results of a reconstruction in the correct way. To illustrate this issue the set of labels is extended regarding the possible visibility configurations to (*empty*, *monocular*, *surface*, *unknown*) which have the following properties:

- *empty* and *surface* areas are visible in more than one camera
- *monocular* areas are visible in exactly one camera
- *unknown* is visible in no camera

Consider figure 2.7 (a) which depicts a scene with nine quadratic pillars observed by three cameras. Figure 2.7 (b) shows the reconstructable parts of the scene which evolve from all surface parts being visible in more than one camera. Furthermore, 2.7 (c) visualizes the visibility properties of the reconstructed scene parts in comparison to 2.7 (d) which depicts the visibility properties of the true scene. In addition, the figure illustrates the properties of visibility and reconstructability dividing the scene into parts which have the following properties:

- *empty* areas are trivially empty
- *surface* areas are surfaces
- *monocular* areas *can* potentially contain a surface which would change the visibility properties behind (when viewing from a camera on a ray into the scene)

- *unknown* areas *can* potentially contain everything and thus represent a set of scenes by definition

Therefore, even the results of reconstruction algorithms looking for a single (most likely) scene represent a set of scenes (in most cases), because reconstructed surfaces let usually occur unknown areas.

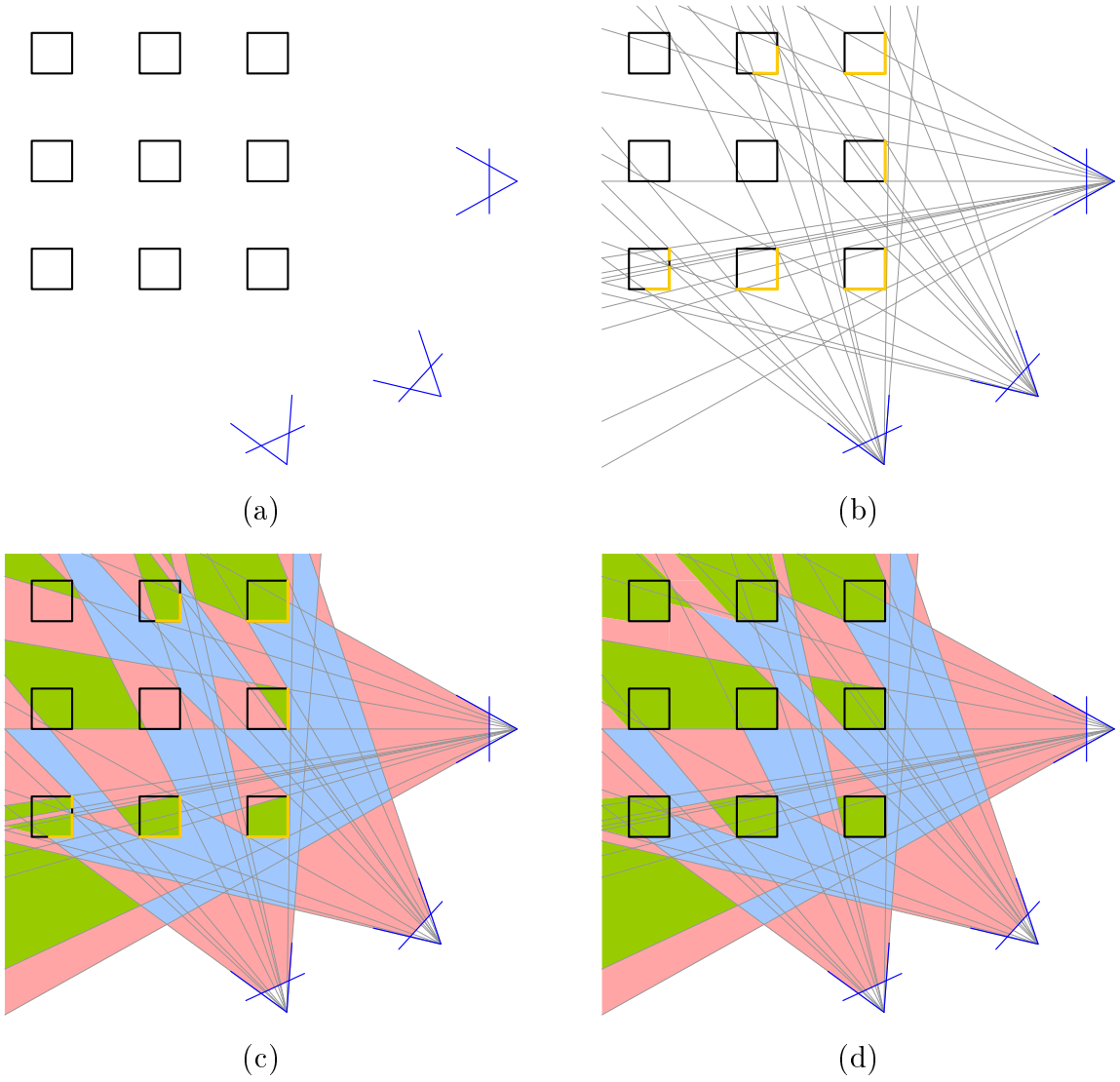


Figure 2.7: Example scene illustrating visibility and reconstructability. (a) Scene overview. (b) Some camera rays (gray) and reconstructable scene parts (orange). (c) Visibility properties of the reconstructed scene. (d) Visibility properties of the true scene. Visibility properties are divided into invisible/unknown (green), monocular visible (red) and visible in more than one camera (blue).

Visibility properties are deterministic for a given scene with given camera positions and can be calculated from a binary scene labeling. In fact, the problem of visibility in stereo reconstruction is equivalent to the shadowing problem in computer graphics if the cameras are accounted to be light sources and the objects in the scene absorb any light. Hence, visibility on a ray from a camera center into the scene is a binary function over the depth value. The visibility function will be defined in section 4.4.

2.2.2 Model Description

Many authors in the field of stereo reconstruction simplify their models and use two rectified cameras and describe the reconstruction result as a depth map. For the general multi-view stereo problem this approach gets more difficult since the depth maps from all cameras need to be merged somehow. To tackle this problem most authors divide it in a multi-pass method, e.g. estimate the depth from camera pairs or a small subsets of cameras and then merge the depth maps in a second pass (as done in [Pon06]). This approach has several disadvantages:

1. The effort of this method increases enormously with the number of cameras, because the number of camera pairs in a set of n cameras is $\frac{n!}{2(n-2)!}$.
2. Due to its structure this method pre-selects data and does not use all available information for the reconstruction (on the other hand this prevents calibration errors from sum up).
3. The view point related world model hinders to express a priori knowledge in the 3D space.

Other possibilities for world models are, e.g. volumetric representations or the creation of non-connected surface patches. Furthermore, the world models for snake and level set methods [Fau] need to assume smooth surfaces and often strong connectivity between (all) surfaces. The use of a volumetric world representation has several advantages:

- The fusion of observation data from several images (remember the pixel span volume in section 2.1.3) is a lot easier.
- Since the intention is to describe the reconstruction task in a probabilistic way, voxel grids and Markov random fields are well suited for each other.
- It is easy to express camera view independent a priori knowledge.
- All other world representations can be calculated from a volumetric representation and it is thus possible to use the advantages (e.g. depth maps) of such world models as well.

The disadvantage is the vast amount of data and this is also essential for the number of necessary operations. Despite this drawback the volumetric world representation has been chosen because of the advantages mentioned. The reconstruction scene therefore consists of a limited area being a cuboid which is tesselated into voxels. Moreover, there are cameras with arbitrary positions and orientations which observe (parts of) the reconstruction volume. The mathematic definition of the world model is given in the next chapter, but for illustration figure 2.8 depicts a sample scene with one of the Middlebury data sets [Col]. It shows 16 cameras observing a cuboid in the scene center and additionally, each camera projection plane shows its corresponding camera image.

Using a volumetric representation of the world one still has to consider the meaning of labels that will be attached to each voxel. To keep ambiguities about the reconstructed surface with respect to homogeneously colored surfaces and occluded areas within the reconstruction result one needs to define a special world model, because the reconstruction result will no longer be a set of depth *values* per camera but rather a set of depth *intervals* per camera. Adapting this thought to the voxel world model makes it necessary to introduce a special label which itself describes a set of scenes. This label will be called

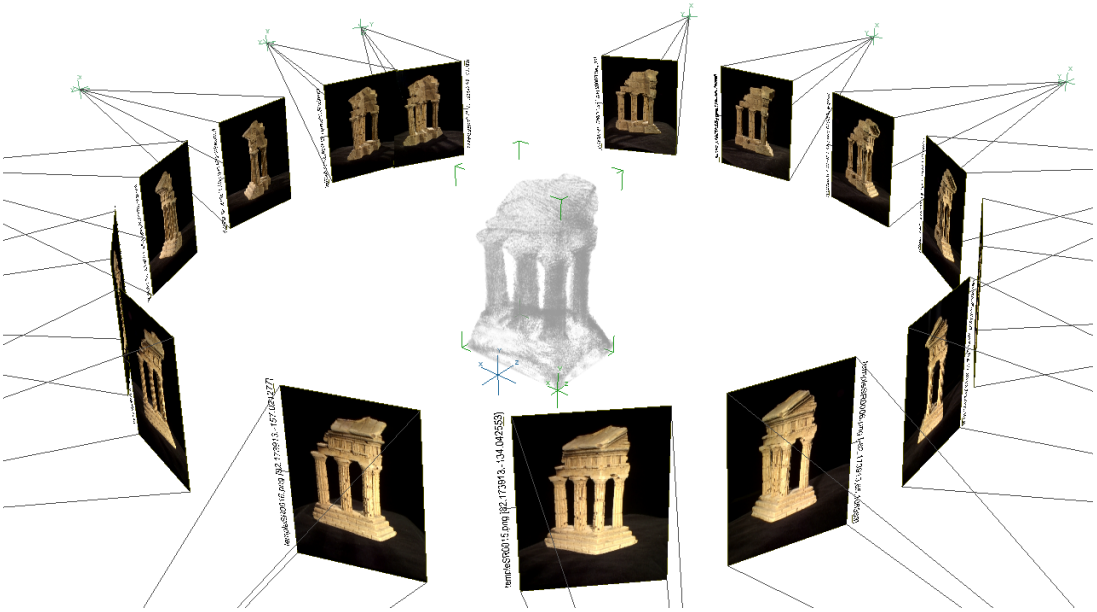


Figure 2.8: Screenshot of the test framework showing an example scene configuration for the temple Middlebury dataset [Col]

fuzzy label according to its meaning. That is, an area of voxels carrying this label means that a surface *is* somewhere in this area (in contrast to the *monocular* areas mentioned above which *can* contain a surface). Therefore, one cannot look through an area of *fuzzy* labeled voxels. In case that a *fuzzy* labeled area has only a "thickness" of one voxel, it becomes equivalent to a *surface* label because there is no further ambiguity about the surface characteristics.

More precisely, the binary labeling (*fuzzy*, *empty*) within areas being thicker than one voxel will represent ambiguity and certainty otherwise. Thus, the (*fuzzy*, *empty*) labeling is able to keep ambiguities about the scene and is more general than the pure (*surface*, *empty*) labeling. Therefore, the semantic of a *fuzzy* label is different to the semantic of a *surface* label, but it will be shown that this aspect has no influence on the probabilistic model. The only difference will be the definition of the visibility function which is discussed in more detail in section 4.4.

Chapter 3 and 4 will show that the probabilistic model will be able reconstruct with a (*fuzzy*, *empty*) labeling as well as with a (*surface*, *empty*) labeling. For this reason, an abstract label *occupied* is introduced which semantics will be defined by the used visibility function (either *fuzzy* or *surface*).

2.3 Summary

The goal of this work is to build a model which handles the problems mentioned above with the help of the clarified assumptions. Ambiguities due to homogeneously colored surfaces and occlusions shall be kept in the reconstruction result to pass them over to a module with further knowledge which is then able to give feedbacks and constitutes - together with the proposed model - an iterative framework for stereo reconstruction. Furthermore, the proposed model represents a stand-alone method for stereo reconstruction when the visibility function is exchanged.

3 Probabilistic Model

It has been shown that the task of stereo reconstruction consists of many ambiguities and it is reasonable to describe them with probabilities mathematically. Before the probabilistic model can be defined the used notation will be introduced.

3.1 Notation

3.1.1 Scene

Consider a scene containing n cameras with arbitrary locations $C^i \in C = \{C^1, C^2, \ldots, C^n\}$, $C \subset \mathbb{R}^3$ observing a reconstruction volume. Given for each camera is an image I^i with dimensions $I^i_{width} \times I^i_{height}$. Without loss of generality all images are assumed to have the same dimensions. An image is defined on a grid of pixels

$$U = \left\{(\mathbf{x},\mathbf{y})^T | \mathbf{x} = 1, \ldots, I^i_{width}, \mathbf{y} = 1, \ldots, I^i_{height}\right\} \subset \mathbb{Z}^2 \tag{3.1}$$

as a function

$$I^i_{\mathbf{xy}} : U \mapsto \mathcal{C} \tag{3.2}$$

where $\mathcal{C} = \mathbb{Z}^d_{255}$ is the color space with $d = 1$ for gray values or $d = 3$ for the RGB-color space.

For the sake of readability a set x of all pixels is defined and will be called observation in the following sections

$$x^i = \left\{x^i_u \in \mathcal{C} \mid \forall u = (\mathbf{x},\mathbf{y})^T \in U, x^i_u = I^i_{\mathbf{xy}}\right\} \tag{3.3}$$

$$x = \left\{x^1, x^2, \ldots, x^n\right\} \tag{3.4}$$

The cameras observe a reconstruction volume, a cuboid of arbitrary size and location, which is tesselated into voxels and is based on the following grid of nodes

$$\mathcal{R} = \left\{(k,l,m)^T \mid k = 1, \ldots, V_{width}, l = 1, \ldots, V_{height}, m = 1, \ldots, V_{depth}\right\} \subset \mathbb{Z}^3 \tag{3.5}$$

The parameters $V_{width}, V_{height}, V_{depth}$ define the dimension of the voxel grid.

Connected to the grid is a set of voxel positions $V \subset \mathbb{R}^3$ containing the voxel center $\mathbf{v}_r \in V$ for each node r in the world coordinate system.

Furthermore, given for each camera i is a function $proj^i : V \mapsto \mathbb{Z}^2$ which summarizes all algebraic transformations that are necessary to obtain the 2D pixel index $u \in U$ from a given 3D voxel center position $\mathtt{v} \in V$. Note that the result of $proj^i$ is not necessarily an element of U. The definition of function $proj^i$ is given in section 5.4.2.

3.1.2 Voxel Relations and Labels

Based on the node grid $\mathcal{R}$ a Graph $\mathcal{G} = (\mathcal{R}, \mathcal{E})$ is defined containing two distinct sets of edges $\mathcal{E} = \mathcal{E}^s \cup \mathcal{E}^v$ described in the following. These two subsets will be needed to describe the neighborhood with respect to the *state* and the *visibility* of voxels and will be distinguished with a superscripted s (state) and v (visibility).

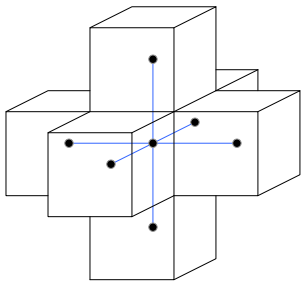


Figure 3.1: 6-neighborhood

$$\mathcal{E}^s = \big\{\{r, r'\} \;\big|\; \forall r, r' \in \mathcal{R},\;\; \|r - r'\| = 1\big\} \tag{3.6}$$

The definition of $\mathcal{E}^s$ is equivalent to the definition of a local 6-neighborhood

$$\mathcal{N}_r^s = \big\{r' \in \mathcal{R} \;\big|\; \|r - r'\| = 1\big\} \tag{3.7}$$

which is depicted in figure 3.1. In addition, a neighborhood regarding the visibility of voxels which depends on the camera positions needs to be defined.

The following list (or ordered set) of voxel indices is described by all voxels hit by the ray from voxel center $\mathtt{v}_r$ to camera position C^i.

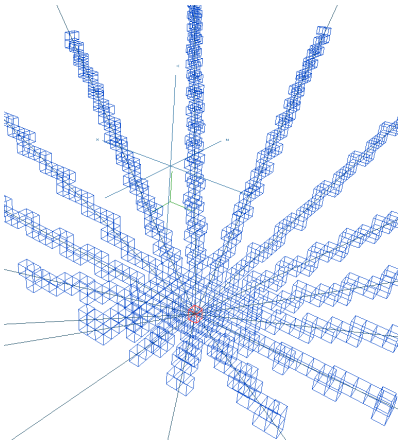


Figure 3.2: Neighborhood wrt. visibility

$$ray(\mathtt{v}_r, C^i) = (r_1^i, r_2^i, \ldots, r_{l_{r^i}}^i) \tag{3.8}$$

where $r_j^i \in \mathcal{R}$ is the j^{th} voxel index starting from voxel $\mathtt{v}_r$ ($r_1^i = r$) and l_{r^i} is the number of voxels between $\mathtt{v}_r$ and C^i intersected by the ray. Detailed information about the $ray(\mathtt{v}_r, C^i)$ function can be found in [Woo] and some information is also given in appendix B.1. Note, that the ray function does not necessarily interect all voxels which project to the same pixel. Its functionality is therefore different to the principle of the $proj^i$ function.

Now, the visibility relation can be defined with the set of all edges between voxel r and every voxel on the ray towards the camera center

$$\mathcal{E}^v = \big\{\{r, r'\} \;\big|\; r, r' \in \mathcal{R}, r \neq r', \exists i = 1, \ldots, n :\; r' \in ray(\mathtt{v}_r, C^i)\big\} \tag{3.9}$$

and for the (semi) local neighborhood analogously:

$$\mathcal{N}_r^v = \big\{r' \;\big|\; \exists i = 1, \ldots, n :\; r' \neq r, r' \in ray(\mathtt{v}_r, C^i)\big\} \tag{3.10}$$

For brevity, let $\mathcal{N}_r = \mathcal{N}_r^s \cup \mathcal{N}_r^v$.

Also attached to each node r is a label $k_r \in \mathcal{L}$ being a complex label consisting of state and color information for each voxel:

$$\text{label } k_r = \begin{pmatrix} \text{voxel state} \\ \text{voxel color} \end{pmatrix} = \begin{pmatrix} k_r^s \in \{\text{empty} = 0, \text{occupied} = 1\} = \mathcal{L}^s \\ k_r^c \in \mathcal{L}^c \subseteq \mathcal{C} \end{pmatrix} \tag{3.11}$$

where $\mathcal{L} = \mathcal{L}^s \times \mathcal{L}^c$ is the set of possible labels. The labels of the nodes are summarized with the set $k = \{k_1, k_2, \ldots, k_{|\mathcal{R}|}\}$ which will be called the labeling.

Intuitively, $k^s_{\mathcal{N}_r^s}$ defines the set of all states in the neighborhood $\mathcal{N}_r^s$ of node r. Thus, $k^s_{\mathcal{N}_r^s} = \{k_{r'} \mid r' \in \mathcal{N}_r^s\}$.

Additionally, a function $vis^i(\mathtt{v}_r) : V \mapsto \{0 = invisible, 1 = visible\}$ which determines the visibility of a voxel by returning a binary value. Hence, the visibility of a voxel can also be considered to be an additional part of the label which is obtained deterministically from the overall labeling and the camera positions. Moreover, the visibility function evaluates the set of edges $\mathcal{E}^v$. The function is defined in section 4.4.

3.1.3 Helpful Definitions

For the derivation of the model a bunch of subsets is needed and will be defined in the following. The first definition is the subset of all nodes $\mathcal{R}_u^i \subset \mathcal{R}$ which attached voxel centers project to pixel u in image i:

$$\mathcal{R}_u^i = \{r \in \mathcal{R} \mid proj^i(\mathtt{v}_r) = u\} \tag{3.12}$$

Based on this definition one can easily define the set of foreground pixels $FG^i \subseteq U$ and the set of background pixels $BG^i \subseteq U$ for camera i:

$$FG^i = \{u \in U \mid \exists r \in \mathcal{R}_u^i : k_r^s = 1\} \tag{3.13}$$
$$BG^i = \{u \in U \mid \neg\exists r \in \mathcal{R}_u^i : k_r^s = 1\} \tag{3.14}$$

which obviously have the following properties: $FG^i \cap BG^i = \emptyset \quad \wedge \quad U = FG^i \cup BG^i \cup \mathcal{IP}^i$, where $\mathcal{IP}^i$ is the set of ignored pixels, i.e. pixels which do not observe the reconstruction volume. According to these definitions the subsets of nodes that project to foreground and background pixels, respectively, can be defined for each camera i:

$$\mathcal{R}_{FG^i} = \{r \in \mathcal{R} \mid proj^i(\mathtt{v}_r) = u, u \in FG^i\} \tag{3.15}$$
$$\mathcal{R}_{BG^i} = \{r \in \mathcal{R} \mid proj^i(\mathtt{v}_r) = u, u \in BG^i\} \tag{3.16}$$

Analogous to the pixel subsets defined above one gets $\mathcal{R} = \bigcup_{i=1}^n (\mathcal{R}_{FG^i} \cup \mathcal{R}_{BG^i} \cup \mathcal{IN}^i)$, where $\mathcal{IN}^i$ is the set of nodes which are ignored in the observation process with respect to camera i (because they are not observable in i) but they may *not* be ignored by other cameras or by the a priori model. Figure 3.3 explains the most important subsets visually.

Together with the subset of pixels FG^i, the visibility function $vis^i(\mathtt{v}_r)$ and a given voxel

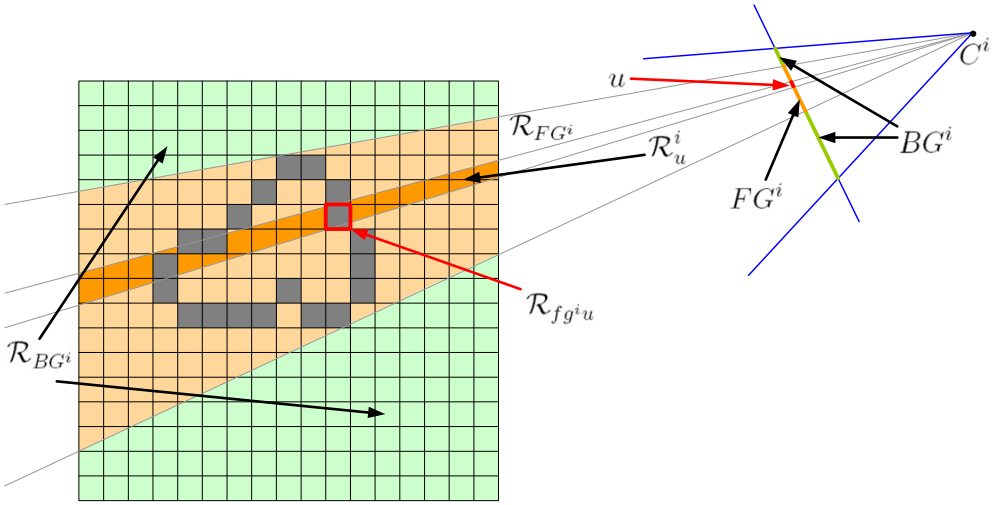


Figure 3.3: Illustration of the subsets.

center $\mathtt{v}_r$ one can define a subset of U which is either empty or it contains the single pixel index u to which $\mathtt{v}_r$ projects if and only if the voxel r is occupied, visible and projects to a foreground pixel. This can analogously be defined for the background, but with the requirement that the regarded voxel is empty (which could actually be omitted because all voxels $r \in \mathcal{R}^i_u$ for a background pixel u are empty per definition (3.14)).

$$fg^i(\mathtt{v}_r) = \left\{u \in U \;\middle|\; proj^i(\mathtt{v}_r) = u \wedge u \in FG^i \wedge vis^i(\mathtt{v}_r) = 1 \wedge k^s_r = 1\right\} \tag{3.17}$$

Note that $|fg^i(\mathtt{v}_r)| \leq 1$ will always hold. Moreover, the set $fg^i(\mathtt{v}_r)$ implicitly contains all dependencies of the visibility neighborhood $\mathcal{N}^v_r$.

Using the visibility function again a further subset of $\mathcal{R}^i_u$ can be defined containing only those nodes being occupied, visible and project to foreground:

$$\mathcal{R}_{fg^iu} = \left\{r \in \mathcal{R} \;\middle|\; proj^i(\mathtt{v}_r) = u \wedge u \in FG^i \wedge vis^i(\mathtt{v}_r) = 1 \wedge k^s_r = 1\right\} \tag{3.18}$$

Note that the set $\mathcal{R}_{fg^iu}$ depends on the choice of the visibility function. If, for example, the visibility function allows only the first occupied voxel being closest to the camera to be visible then $\mathcal{R}_{fg^iu}$ will contain exactly one voxel for foreground pixels and none for background pixels.

3.2 Model Definition

Let $K = \{K_1, \ldots, K_{|\mathcal{R}|}\}$ be a set of random variables where each random variable K_r is connected to a node in $\mathcal{R}$ and takes a value k_r in $\mathcal{L}$. This elementary event will be denoted as $K_r = k_r$ and $P(K_r = k_r)$, abbreviated $P(k_r)$, will denote the probability that random variable K_r takes the value k_r. Furthermore, let $X = \{X^1, \ldots, X^n\}$ with $X^i = \{X^i_1, \ldots, X^i_{|U|}\}$ be the sets of random variables which are connected to the observation. Thus, each random variable X^i_u takes a value $x^i_u \in \mathcal{C}$.

Together with the graph $\mathcal{G} = (\mathcal{R}, \mathcal{E})$ and the sets K and X of random variables a (hidden) Markov random field (MRF) is defined in the following two subsections. The joint probability distribution $P(K = k, X = x)$ can be divided into two parts being an a priori model and an observation model.

3.2.1 A Priori Model

In general the a priori model represents all knowledge which does not depend on the observation. Due to the voxel representation of the world the choice of the a priori model led to the efficient Potts model which favors locally connected scene parts. The Potts model is used to weight the state label k^s_r of a single voxel independently according to the states $k^s_{\mathcal{N}^s_r}$ of its direct neighbors (remember fig. 3.1). Being a generalization of the Ising model the Potts model makes artifacts less likely, is able to close small gaps in scene parts which are not observable due to occlusion or parts which have ambiguous observations due to camera calibration errors.

It is defined as a function $g^s(k^s_r, k^s_{r'}) : \mathcal{L}^s \times \mathcal{L}^s \mapsto \mathbb{R}$

$$g^s(k^s_r, k^s_{r'}) = \begin{cases} \alpha & \text{if } k^s_r = k^s_{r'} \\ \beta & \text{otherwise} \end{cases} \qquad \text{with } \alpha \geq \beta, r' \in \mathcal{N}^s_r \tag{3.19}$$

With the help of this function the a priori probability distribution can be defined a as homogeneous Gibbs distribution [Hla02, Li00] being a product over all edges $\mathcal{E}^s$ (assuming independence of all edges). In view of the joint model, the edge evaluation is defined as product over all nodes and their neighbors (every edge is evaluated twice):

$$P(k) = Z^{-1}_{g^s} \prod_{r \in \mathcal{R}} \prod_{r' \in \mathcal{N}^s_r} g^s(k^s_r, k^s_{r'}) \tag{3.20}$$

where $Z_{g^s} = \sum_k \prod_{r \in \mathcal{R}} \prod_{r' \in \mathcal{N}^s_r} g^s(k^s_r, k^s_{r'})$ is a normalization constant to ensure the requirements of a probability distribution.

3.2.2 Observation Model

The observation model describes the relation between the (unknown) voxel labelings and the measured observation. Based on the fact of having a generative model two distinct models are required, namely a **foreground model** making the labeled voxels coloring the pixels in their projections, and a **background model** coloring all pixels which cannot be explained by the reconstructed shape. The presented approach for the background

model is similar to the one presented by [Str06] in which background pixels are regarded as outliers.

Foreground model

Assuming a Lambertian-like radiance model (section 2.1.2) the voxel color will not differ significantly from the color in its projections. For this reason, it is assumed that the pixel color is normally distributed around the voxel color if the voxel has generated the pixel. The function $\hat{q}(x_u^i, k_r^c)$ describes the probability that voxel r has generated the color in pixel u.

$$\hat{q}(x_u^i, k_r^c) = \frac{1}{(\sqrt{2\pi}\sigma)^d} \exp\left[-\frac{\|x_u^i - k_r^c\|^2}{2\sigma^2} \right] \tag{3.21}$$

Background model

Since nothing is known about the background, it will be modeled as a random generator with unknown distribution independently for each camera. The probability of a pixel u observing the background and having color x_u^i will be

$$f^i(x_u^i) \quad \text{with properties: } \forall c, i = 1, \ldots, n : 0 \leq f^i(c) \leq 1 \ \wedge \ \forall i = 1, \ldots, n : \sum_{c \in \mathcal{C}} f^i(c) = 1 \tag{3.22}$$

This implies the assumption that the color distribution of background pixels differs from the color distribution of the foreground pixels significantly. This is discussed in the summary of this chapter.

Joint observation model

Taking the visibility configuration of the regarded voxel r into account yields the probability of pixel u to have color x_u^i given the current voxel labeling $k_{\mathcal{R}_u^i}$:

$$P(x_u^i | k_{\mathcal{R}_u^i}) = Z_{qfu}^{-1} \underbrace{\prod_{r \in \mathcal{R}_{fg^i u}} \hat{q}(x_u^i, k_r^c)}_{foreground} \cdot \underbrace{f^i(x_u^i)^{bg^i(u)}}_{background} \tag{3.23}$$

where $k_{\mathcal{R}_u^i} = \{k_r | r \in \mathcal{R}_u^i\}$ and $bg^i : U \mapsto \{0, 1\}$ is a function which is 1 if u is a background pixel and 0 otherwise:

$$bg^i(u) = \begin{cases} 1 & \text{if } u \in BG^i \\ 0 & \text{otherwise} \end{cases} \tag{3.24}$$

Note that either the foreground part or the background part of equation (3.23) will be 1. If $bg^i = 1$ then $\mathcal{R}_{fg^i u} = \emptyset$ or otherwise if $bg^i = 0$ then $\left|\mathcal{R}_{fg^i u}\right| \geq 1$. The product over the elements of an empty set will be defined to be 1.

Assuming independent pixel coloring the probability for the observation can be defined as a product

$$P(x|k) = Z_{qf}^{-1} \prod_{i=1}^{n} \prod_{u \in U} P(x_u^i | k_{\mathcal{R}_u^i}) \tag{3.25}$$

With the subsets defined in section 3.1.3, the product over pixels can be transformed to a product over voxels for the foreground model:

$$P(x|k) = Z_{qf}^{-1} \prod_{i=1}^{n} \prod_{u \in FG^i} \prod_{r \in \mathcal{R}_{fg^i u}} \hat{q}(x_u^i, k_r^c) \cdot \prod_{u \in BG^i} f^i(x_u^i) \tag{3.26}$$

$$= Z_{qf}^{-1} \prod_{i=1}^{n} \prod_{r \in \mathcal{R}_{FG^i}} \prod_{u \in fg^i(\mathrm{v}_r)} \hat{q}(x_u^i, k_r^c) \cdot \prod_{u \in BG^i} f^i(x_u^i) \tag{3.27}$$

$$= Z_{qf}^{-1} \prod_{i=1} \underbrace{\prod_{r \in \mathcal{R}} \prod_{u \in fg^i(\mathrm{v}_r)} \hat{q}(x_u^i, k_r^c)}_{foreground} \cdot \underbrace{\prod_{u \in BG^i} f^i(x_u^i)}_{background} \tag{3.28}$$

where Z_{qf}^{-1} is the normalization factor ensuring that $P(x|k)$ sums up to 1 over all possible observations. Unfortunately, this factor depends on the labeling and will trouble the derivation of $P(x|k)$ when the unknown parameters are estimated with unsupervised learning. The background observation part is already properly normalized since the functions f^i are probability distributions over pixel colors. In case that the visibility function allows only one voxel (e.g. the first) on a ray through a pixel to be visible, each pixel is either assigned a single Gaussian or a value of the background distribution. In this case the sum over all observations would be 1 (thus, $Z_{qf} = 1$). If the visibility function assigns several occupied voxels on a ray to be visible a product of Gaussians is assigned to the related pixel. As a result, the sum over all observations in this pixel would be less than 1 and $Z_{qf} \neq 1$.

With the introduction of a new function q the normalization factor Z_{qf} can be eliminated and the observation model can be described to have the following, locally normalized form:

$$P(x|k) = \prod_{i=1}^{n} \prod_{r \in \mathcal{R}} \prod_{u \in fg^i(\mathrm{v}_r)} q(x_u^i, k_r^c) \cdot \prod_{u \in BG^i} f^i(x_u^i) \tag{3.29}$$

There are, however, several possibilities to deal with this problem. One could assume, that a group of voxels "sharing" one pixel observation could be represented by a mixture of Gaussians (i.e., a weighted sum of Gaussians, instead of a product). While this would change the probability distribution, it would not change the intended functionality of the observation model. Another way is to require all voxels in such a group to have the same color. This can be done since the uncertainty areas due to homogeneously colored surfaces have the same property. In this case, the function q is found to be:

$$q(x_u^i, k_r^c) = \left(\frac{\sqrt{|\mathcal{R}_{fg^i u}|}}{\sqrt{2\pi}\sigma} \right)^{\frac{d}{|\mathcal{R}_{fg^i u}|}} \exp\left[- \frac{\|x_u^i - k_r^c\|^2}{2\sigma^2} \right] \tag{3.30}$$

The new normalization factor of the Gaussian ensures that the product over all members of a group of visible occupied voxels on ray is a single properly normalized Gaussian. The detailed derivation of this function can be found in appendix A.2.2. Consequently, even if there are several occupied voxels visible in a pixel each pixel is still assigned either a single Gaussian or a single value of the background distribution.

3.2.3 Joint Model

The joint probability distribution is the product of the a priori distribution and the conditional observation distribution. Thus, combining equation (3.20) and (3.29) results in:

$$\begin{aligned} P(x,k) &= P(k) \cdot P(x|k) \\ &= Z^{-1} \prod_{r \in \mathcal{R}} \prod_{r' \in \mathcal{N}_r^s} g^s(k_r^s, k_{r'}^s) \cdot \prod_{i=1}^{n} \Big[\prod_{r \in \mathcal{R}} \prod_{u \in fg^i(\mathrm{v}_r)} q(x_u^i, k_r^c) \cdot \prod_{u \in BG^i} f^i(x_u^i) \Big] \end{aligned} \tag{3.31}$$

where $Z = Z_{g^s}$ is the normalization constant from the a priori model.

The joint probability $P(x,k)$ cannot be calculated efficiently but it can be approximated with the Gibbs sampler making it possible to use the probability distribution for recognition. This will be described in section 4.3.

3.3 Summary and Discussion

A probabilistic model for general multi-view stereo reconstruction has been defined under weak assumptions. The model defines a probability distribution over possible scene configurations in which especially competing scene configurations (in the meaning of section 2.1) will have similar probabilities.

In sum, the probability of the label for a single voxel is weighted by two components. The voxel label is weighted with a Gaussian of each color difference between its color state and its projected observation given the current visibility configuration. Additionally, the state label of the voxel is weighted with respect to the state labels of its direct neighbors. This is done independently for all voxels/nodes in the MRF and defines the joint probability distribution $P(x,k)$.

The chosen Potts a priori model is efficient and favors locally connected scene parts and is therefore able to close small gaps within reconstructed surfaces which may occur due to occlusions or erroneous pixel color matches. On the other hand the Potts model does not only favor smooth surfaces it also favors compact shapes which impedes the correct reconstruction of thin objects.

Alternatives to the Potts Model

The advantage of the Potts Model regarding efficiency is its disadvantage regarding the a priori information one can encode. With the use of higher order a priori models, i.e., with the evaluation of larger neighborhood structures, one could express a priori information more accurately which could lead to better reconstruction results. On the other hand, this requires a lot more computation time.

Labeling

The labeling has been chosen to be a complex label consisting of state and color information. The state labeling is reasonable and straightforward whereas the question about the necessity of a color label may arise. There are, however, several possible ways to deal with the color value of voxels within a probabilistic stereo reconstruction model. The following possibilities have been regarded:

1. the voxel colors can be regarded as unknown parameters
2. the voxel colors can be modeled as random variables and can be excluded from the joint model via summation
3. the voxel colors can be modeled as random variables and constitute a part of the label

The first possibility has been excluded because of its inflexibility, e.g. it is not possible to express a priori knowledge about colors if available. The second choice is only possible with the use of the additional assumption that the voxel mean observation colors are equally distributed (as done in [A.S]). The third option needs more effort but is the most general of the regarded possibilities and has therefore been selected. Note, that the defined probability model assumes independent voxel coloring.

The model requires voxels on a ray through a $fuzzy$ area to have the same color which is a justifiable attitude since the color values are equal by definition. In comparison with the assumption of a constant normalization factor this should be the better approach. Furthermore, the approach has been chosen due to the implementation where the voxel color labels will be limited to a color map and small color variations would end up with the same color map entry anyway.

4 Recognition

Recognition is the last step of the concurrent stereo reconstruction method. After the definition of a probability distribution over the vast number of possible scene labelings it is necessary to make a decision based on the distribution. To this end, Bayesian estimation will be used. Moreover, this chapter discusses further steps necessary to (approximately) calculate the probabilities defined in the previous chapter which are: learning of unknown parameters, sampling from the distribution, definition of the visibility function and how to deal with feedbacks from another model.

4.1 Bayesian Estimation

Bayesian estimation is defined as the optimization task which minimizes the expected value of the Bayesian risk.

$$k^* = \arg\min_{k'} \sum_{k \in \mathcal{K}} P(x, k) c(k, k') \tag{4.1}$$

where a cost function $c(k, k')$ needs to be defined stating the costs for a correct and a wrong decision, respectively. An intuitive cost function is to summarize the costs for decisions in each node:

$$c(k, k') = \sum_{r \in \mathcal{R}} c(k_r, {k_r}') \tag{4.2}$$

which makes it necessary to define a cost function for a decision of a single label $l \in \mathcal{L}$. The most simple cost function is to pay 1 in case of a wrong decision and 0 otherwise:

$$c(l, l') = 1 - \delta_{ll'} = \begin{cases} 0 & \text{if } l = l' \\ 1 & \text{otherwise} \end{cases} \tag{4.3}$$

It is commonly known that Bayesian estimation with this particular cost function is equivalent to a maximum a posteriori (MAP) decision being defined as

$$\forall r \in \mathcal{R}: \qquad {k_r}^* = \arg\max_{{k_r}'} P(K_r = {k_r}' \mid x) \tag{4.4}$$

A derivation of this proposition can be found in appendix A.2.1. The decision strategy is therefore to decide for the most likely label in each node.

4.2 Color Distribution Learning

The observation model defined above uses two different kinds of probability distributions to describe the color distribution of the input images with both depending on unknown parameters. The Gaussian distributions for the image foreground depend on the unknown standard deviation σ. Moreover, the distribution of the background is completely unknown and each probability distribution f^i can be considered as an unknown parameter. Thus, all unknown parameters can be summarized with the set $\theta = \{\sigma, f^1, f^2, \ldots, f^n\}$.

Given an observation x these parameters can be calculated by maximizing the probability of the observation $P(x;\theta)$ with respect to the unknown parameters θ. This is commonly known as maximum likelihood estimation (MLE) and is defined by the task

$$\theta^* = \arg\max_\theta P(x;\theta) = \arg\max_\theta \sum_k P(x,k;\theta) \tag{4.5}$$

The task can be solved using the EM-Algorithm [Hla02] where the approach of maximizing the log-likelihoods of (4.5) leads to the following task:

1. **Expectation:** Calculate the expected value of the joint probability under the current (fixed) parameter set.

$$E_{k|x;\theta^{t-1}}\Big\{\ln P(x,k;\theta)\Big\} = \sum_k P(k|x;\theta^{t-1}) \cdot \ln P(x,k;\theta) \tag{4.6}$$

2. **Maximization:** Maximize the expected value with respect to the parameter set

$$\theta^t = \arg\max_{\theta'} \sum_k P(k|x;\theta^{t-1}) \cdot \ln P(x,k;\theta') \tag{4.7}$$

Since the EM-Algorithm works iteratively the unknowns at time step t are improved using the results of the previous time step $t-1$. This implies the necessity of an initial parameter set θ^0. The choice of these initial values is discussed later in section 6.6.

Applying the definition of the joint model (3.31) to equation (4.7) results in:

$$\theta^t = \arg\max_{\theta'} \sum_k P(k|x;\theta^{t-1}) \ln\Big[P(k) \cdot P(x|k;\theta')\Big] \tag{4.8}$$

$$= \arg\max_{\theta'} \sum_k P(k|x;\theta^{t-1}) \Big[\ln P(k) + \ln P(x|k;\theta')\Big] \tag{4.9}$$

$$\tag{4.10}$$

The a priori model and its normalization constant does not depend on θ' and can be excluded from the maximization. Inserting the definition of the observation model (3.29) leads to:

$$= \arg\max_{\theta'} \sum_k P(k|x;\theta^{t-1}) \ln \prod_{i=1}^{n} \Big[\prod_{r\in\mathcal{R}} \prod_{u\in fg^i(\mathtt{v}_r)} q(x_u^i, k_r^c) \cdot \prod_{u\in BG^i} f^i(x_u^i) \Big] \tag{4.11}$$

$$= \arg\max_{\theta'} \sum_k P(k|x;\theta^{t-1}) \sum_{i=1}^{n} \Big[\ln \prod_{r\in\mathcal{R}} \prod_{u\in fg^i(\mathtt{v}_r)} q(x_u^i, k_r^c) + \ln \prod_{u\in BG^i} f^i(x_u^i) \Big] \tag{4.12}$$

Now, it can be seen that the maximization task can be decomposed into two independent maximizations since $q(x_u^i, k_r^c)$ only depends on σ and $f^i(x_u^i)$ is the unknown parameter itself. These two maximization tasks are discussed in the following two subsections.

4.2.1 Voxel Color Distribution

$$\sigma^t = \arg\max_{\sigma} \sum_k P(k|x;\theta^{t-1}) \sum_{i=1}^{n} \ln \prod_{r\in\mathcal{R}} \prod_{u\in fg^i(\mathtt{v}_r)} q(x_u^i, k_r^c) \tag{4.13}$$

$$= \arg\max_{\sigma} \sum_k P(k|x;\theta^{t-1}) \sum_{i=1}^{n} \ln \left(\prod_{r\in\mathcal{R}} \prod_{u\in fg^i(\mathtt{v}_r)} \left(\frac{\sqrt{|\mathcal{R}_{fg^iu}|}}{\sqrt{2\pi}\sigma} \right)^{\frac{d}{|\mathcal{R}_{fg^iu}|}} \exp\left[-\frac{\|x_u^i - k_r^c\|^2}{2\sigma^2} \right] \right) \tag{4.14}$$

$$= \arg\max_{\sigma} \sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left(\frac{d}{|\mathcal{R}_{fg^iu}|} \ln \frac{\sqrt{|\mathcal{R}_{fg^iu}|}}{\sqrt{2\pi}} - \frac{d \ln\sigma}{|\mathcal{R}_{fg^iu}|} - \frac{\|x_u^i - k_r^c\|^2}{2\sigma^2} \right) \tag{4.15}$$

The first summand in the parentheses does not depend on σ and can be removed from the maximization:

$$= \arg\max_{\sigma} \underbrace{\sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left(-\frac{d}{|\mathcal{R}_{fg^iu}|} \ln\sigma - \frac{\|x_u^i - k_r^c\|^2}{2\sigma^2} \right)}_{\varphi(\sigma)} \tag{4.16}$$

Calculating the first derivative and equating it with 0 yields :

$$\frac{\partial\varphi(\sigma)}{\partial\sigma} = \sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \Big(-\frac{d}{|\mathcal{R}_{fg^iu}|} + \frac{\|x_u^i - k_r^c\|^2}{\sigma^2} \Big) = 0 \tag{4.17}$$

Later in section 5.1 it will be shown that the square root operation is unnecessary and it is enough to know σ^2 which is found to be:

$$\sigma^2 = \frac{\sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left\|x_u^i - k_r^c\right\|^2}{\sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} d \left|\mathcal{R}_{fg^i u}\right|^{-1}} \tag{4.18}$$

$$= \frac{\sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left\|x_u^i - k_r^c\right\|^2}{d \cdot \sum_k P(k|x;\theta^{t-1}) \sum_{i=1}^{n} \sum_{r\in\mathcal{R}} \sum_{u\in fg^i(\mathtt{v}_r)} \left|\mathcal{R}_{fg^i u}\right|^{-1}} \tag{4.19}$$

$$= \frac{\sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left\|x_u^i - k_r^c\right\|^2}{d \cdot \sum_k P(k|x;\theta^{t-1}) \sum_{i=1}^{n} \left|\mathcal{R}_{fg^i u}\right| \cdot \left|\mathcal{R}_{fg^i u}\right|^{-1}} \tag{4.20}$$

$$= \frac{\sum_k P(k|x;\theta^{t-1}) \sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left\|x_u^i - k_r^c\right\|^2}{d \cdot n} \tag{4.21}$$

$$= E_{k|x;\theta^{t-1}} \left\{ \frac{\sum_{r\in\mathcal{R}} \sum_{i=1}^{n} \sum_{u\in fg^i(\mathtt{v}_r)} \left\|x_u^i - k_r^c\right\|^2}{d \cdot n} \right\} \tag{4.22}$$

where E denotes the expected value of the term in braces. The learning rule declares to use the mean squared color difference between foreground pixels and visible voxels over all foreground pixels.

4.2.2 Background Color Distribution

For the sake of readability let $f = \{f^1, f^2, \ldots, f^n\}$. The maximization task for the background model is then given by:

$$f^t = \arg\max_f \sum_k P(k|x;\theta^{t-1}) \ln \prod_{i=1}^{n} \prod_{u\in BG^i} f^i(x_u^i) \tag{4.23}$$

$$= \arg\max_f \sum_k P(k|x;\theta^{t-1}) \sum_{i=1}^{n} \sum_{u\in BG^i} \ln f^i(x_u^i) \tag{4.24}$$

This task is again separable since the distributions for each camera are independent of each other which results in n maximization tasks of the following form:

$$f^{i,t} = \arg\max_{f^i} \sum_k P(k|x;\theta^{t-1}) \sum_{u\in BG^i} \ln f^i(x_u^i) \tag{4.25}$$

Let's consider the meaning of function $f^i(c)$ again before the derivation is continued. It returns a probability value for each color $c \in \mathcal{C}$ and each of these values are unknown. Since the color space will have more than 16 million color entries in the implementation it is infeasible to regard each value as an unknown parameter. To tackle this problem,

the function $f^i(c)$ is assumed to be piecewise constant. Thus, the color space $\mathcal{C}$ is divided into b distinct partitions of equal size $\mathcal{C} = \mathcal{C}_1 \cup \ldots \cup \mathcal{C}_b$.

This means, that $f^i(c)$ will return the same value for all colors $c \in \mathcal{C}_j$ of a particular partition. Consequently, the function $f^i(c)$ can be defined with independent functions f^i_j where each of them is defined on its dedicated domain $\mathcal{C}_j$ of the color space.

$$f^i(c) = \begin{cases} f^i_1 & \text{if} \quad c \in \mathcal{C}_1 \\ f^i_2 & \text{if} \quad c \in \mathcal{C}_2 \\ \vdots & \\ f^i_b & \text{if} \quad c \in \mathcal{C}_b \end{cases} \tag{4.26}$$

With the help of a binary function $s : \mathcal{C} \mapsto \{0, 1\}$ which selects the appropriate color partition

$$s_j(c) = \begin{cases} 1 & \text{if} \quad c \in \mathcal{C}_j \\ 0 & \text{otherwise} \end{cases} \tag{4.27}$$

the background distribution function can also be described as a product over the parameters f^i_j

$$f^i(c) = \prod_{j=1}^{b} \left(f^i_j \right)^{s_j(c)} \tag{4.28}$$

Applying this definition to the last derivation step in equation (4.24) yields:

$$f^{i,t} = \arg\max_{f^i} \sum_k P(k|x;\theta^{t-1}) \sum_{u \in BG^i} \ln \prod_{j=1}^{b} \left(f^i_j \right)^{s_j(x^i_u)} \tag{4.29}$$

$$= \arg\max_{f^i} \sum_k P(k|x;\theta^{t-1}) \sum_{u \in BG^i} \sum_{j=1}^{b} s_j(x^i_u) \cdot \ln f^i_j \tag{4.30}$$

$$= \arg\max_{f^i} \sum_{j=1}^{b} \underbrace{\sum_k P(k|x;\theta^{t-1}) \sum_{u \in BG^i} s_j(x^i_u)}_{\alpha_j} \cdot \ln f^i_j \tag{4.31}$$

Shannon's theorem [Hla02, page 242] which can be found in appendix A.2.3 leads to the following solution:

$$f_j^i = \frac{\alpha_j}{\sum_{j=1}^{b} \alpha_j} \tag{4.32}$$

$$= \frac{\sum_k P(k|x;\theta^{t-1}) \sum_{u \in BG^i} s_j(x_u^i)}{\sum_{j=1}^{b} \sum_k P(k|x;\theta^{t-1}) \sum_{u \in BG^i} s_j(x_u^i)} \tag{4.33}$$

$$= \frac{E_{k|x;\theta^{t-1}}\left\{ \sum_{u \in BG^i} s_j(x_u^i) \right\}}{E_{k|x;\theta^{t-1}}\left\{ \sum_{j=1}^{b} \sum_{u \in BG^i} s_j(x_u^i) \right\}} \tag{4.34}$$

verbally, this is

$$= \frac{E_{k|x;\theta^{t-1}}\Big\{\text{number of background pixels in } \mathcal{C}_j\Big\}}{E_{k|x;\theta^{t-1}}\Big\{\text{number of background pixels}\Big\}} \tag{4.35}$$

The learning rule in equation (4.35) implies to use the color histogram. The distributions f_j^i are therefore called histogram distributions. Loosely speaking, the current normalized histogram of pixels observing the background defines the probability (over the color space) of being a background pixel and this is done separately for each camera.

4.3 Gibbs Sampling

Since it is computational expensive to calculate the joint probability $P(k, x)$ for Gibbs random fields (GRF) (being NP-complete for the general case) the Gibbs sampler (introduced by [GG84]) is an algorithm allowing to approximate this probability. The algorithm iteratively generates a new valid sample ${k_r}^{t+1}$ at time step t out of the previous sample ${k_r}^t$. This is done by generating a new state for each node r in the GRF according to its conditional probability while exploiting the Markovian property:

$$P(K_r = k_r | k_{\mathcal{R}\setminus\{r\}}, x) = P(K_r = k_r | k_{\mathcal{N}_r}, x) \tag{4.36}$$

Where a time step (= one sampling cycle) is defined by one loop over all nodes $r \in \mathcal{R}$. Let $c[r, k_r]^t$ be a variable counting the number of times a label k_r has been generated in node r until time step t. For large values of t the approximation

$$P(K_r = k_r|x) \approx \frac{c[r, k_r]^t}{\sum_{k_r'} c[r, {k_r}']^t} \tag{4.37}$$

holds and is proven ([GG84]) to become an equality for $t \to \infty$. Thus, an arising task is then to find a "good" initial labeling k^0 which is discussed in subsection 6.3.

Conditional Label Distribution

To locally sample from the probability distribution defined in (3.31) one needs to calculate the conditional distribution for a single node r while the labels in all other nodes $\mathcal{R}\backslash\{r\}$ are fixed

$$P(k_r|k_{\mathcal{R}\backslash\{r\}}, x) = \frac{P(k_r, k_{\mathcal{R}\backslash\{r\}}, x)}{\sum_{k_r'} P(k_r', k_{\mathcal{R}\backslash\{r\}}, x)} \tag{4.38}$$

Applying equation (3.31) leads to

$$\begin{aligned} &P(k_r|k_{\mathcal{R}\backslash\{r\}}, x) \\ &= Z_r^{-1} \prod_{r'\in\mathcal{N}_r^s} g^s(k_r^s, k_{r'}^s)^2 \cdot \prod_{i=1}^{n} \Big[\prod_{u\in fg^i(\mathtt{v}_r)} q(x_u^i, k_r^c) \cdot \prod_{w\in\mathcal{N}_r^v} \prod_{u\in fg^i(\mathtt{v}_w)} q(x_u^i, k_w^c) \cdot \prod_{u\in BG^i} f^i(x_u^i) \Big] \end{aligned} \tag{4.39}$$

with the normalization factor

$$Z_r = \sum_{k_r} \prod_{r'\in\mathcal{N}_r^s} g^s(k_r^s, k_{r'}^s)^2 \cdot \prod_{i=1}^{n} \Big[\prod_{u\in fg^i(\mathtt{v}_r)} q(x_u^i, k_r^c) \cdot \prod_{w\in\mathcal{N}_r^v} \prod_{u\in fg^i(\mathtt{v}_w)} q(x_u^i, k_w^c) \cdot \prod_{u\in BG^i} f^i(x_u^i) \Big] \tag{4.40}$$

Equation (4.39) is the conditional probability being estimated by the Gibbs sampler.

Incremental Learning

In each sampling step one also needs to update the unknown parameters according to the learning rules described in the last section. The learning rules are due to the EM-algorithm an iterative process, but they have been derived from the original distribution function $P(x, k)$. However, the Gibbs sampler only approximates this function in the sampling process and does not return the expected value which makes it necessary to average the learning results over several sampling steps (batch mode). This introduces a new, but intuitively selectable parameter. Equivalently, one could also define a *learn rate* which weights the new learning results against the old during the update (online mode).

4.4 Visibility Function and Label Semantics

The visibility function $vis^i(\mathtt{v}_r) : V \mapsto \{0 = invisible, 1 = visible\}$ determines the visibility of voxels in the voxel grid according to the camera positions and all voxels between camera i and the regarded voxel r. As already pointed out, two different visibility functions are considered in this thesis which change the semantic of the *occupied* label into either a *surface* label or a *fuzzy* label. For brevity, the functions will be called surface visibility and fuzzy visibility.

The surface visibility function is a very intuitive function. One can look through empty

voxels but not through occupied ones. Hence, from the camera point of view the visibility ends behind the first occupied (and visibile) voxel. The fuzzy visibility function allows to model areas of uncertainty (discussed in section 2.2) where it becomes necessary that the camera can look through occupied voxels in such areas. These areas have been defined to contain a surface and therefore a camera cannot look through such an area.

However, both visibility functions have the property that everything between a particular point in space and the camera is visible and everything behind this point (on the ray through the camera center) is *not* visible. Therefore, the visibility of a voxel r regarding camera i can be defined locally and depends only on its direct neighbor on the ray $ray(\mathtt{v}_r, C^i)$ towards the camera center C^i. Remember the definition (3.8) of the ray function: $ray(\mathtt{v}_r, C^i) = (r_1^i, r_2^i, \ldots, r_{l_{r^i}}^i)$. Thus, the visibility of node r_1^i depends only on the state of node r_2^i and its visibility, because r_2^i encodes the visibility of all previous voxels.

Imagine to view from the camera center C^i on a single ray into the scene. The surface visibility will end after the first occupied voxel whereas the fuzzy visibility will end after the last occupied voxel within the first intervall of occupied voxels. These statements are expressed in table 4.1 and illustrated in figure 4.1.

config. no.	voxel 1 state $k^s_{r_1^i}$	voxel 2 state $vis^i(\mathtt{v}_{r_2^i})$	voxel 2 vis. $k^s_{r_2^i}$	voxel 1 surf. vis. $vis^i(\mathtt{v}_{r_1^i})$	voxel 1 fuzz. vis. $vis^i(\mathtt{v}_{r_1^i})$
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	0	0
3	0	1	1	0	0
4	1	0	0	0	0
5	1	0	1	1	1
6	1	1	0	0	0
7	1	1	1	0	1

Table 4.1: Surface and fuzzy visibility function.

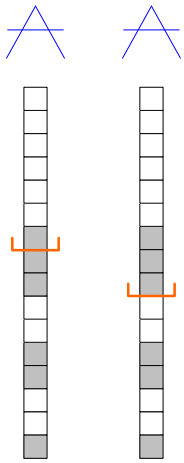


Figure 4.1: Visibility functions.

Thus, one gets the following visibility functions:

$$\text{surface: } vis^i(\mathtt{v}_{r_1^i}) = k^s_{r_2^i} \wedge \overline{vis^i(\mathtt{v}_{r_2^i})} \tag{4.41}$$

$$\text{fuzzy: } vis^i(\mathtt{v}_{r_1^i}) = k^s_{r_2^i} \wedge \overline{vis^i(\mathtt{v}_{r_2^i})} \vee k^s_{r_2^i} \wedge vis^i(\mathtt{v}_{r_2^i}) \tag{4.42}$$

where a line over a symbol denotes the logical negation.

As already mentioned in section 3.1.2 the visibility function evaluates the (semi local) neighborhood $\mathcal{N}_r^v$ for a particular voxel. Due to the simplicity of both visibility functions it is enough to evaluate only the next voxel on the ray in camera direction together with the regarded one (for each camera).

4.5 Feedbacks from an other module

The probabilistic model defined in chapter 3 shall also be able to receive feedbacks from another module which input is the output of the presented model according to the processing scheme in figure 1.1. Since the structure of this module is unknown its feedback is modeled in the most general way. This is realized with a function Ψ: $\mathcal{K} \mapsto \{0, ..., 1\}$ which takes the current labeling and returns a probability value. Then, the joint model (3.31) is extended to the following form

$$P(x, k) = Z_{\Psi}^{-1} \cdot P(k) \cdot P(x|k) \cdot \Psi(k; \Omega) \tag{4.43}$$

where Ω is a symbolic set of parameters representing any further knowledge and $Z_{\Psi} = \sum_k P(k) \cdot P(x|k) \cdot \Psi(k; \Omega)$ is the normalization factor.

The other module can therefore change the probabilities of labels. More precisely, if the other module decreases the probability of (or simply suppresses) a particular label in node r it might change the probability of other labels as well due to semi-local visibility conditions.

4.6 Summary

It has already been said that the distributions for foreground and background pixels are assumed to differ significantly, otherwise foreground pixels will fit to the background distribution and will thus be selected to be background. The result of the reconstruction would be an (almost) empty scene.

Due to the MRF-approach the chicken-egg visibility problem has been simplified to a semi-local binary visibility function and it is not necessary to estimate the visibility separately from the reconstruction process or to use any heuristics. However, the evaluation (chapter 6) will show that the visibility is still a problem due to the sampling approach.

The main advantage of the proposed model is its flexibility. The a priori model as well as the observation model can be extended or exchanged independently.

summary of assumptions

- no semi-transparent objects in the scene
- no reflections: scene illumination and surface materials similar to the Lambertian radiance model in short, the simplest radiance model with Gaussian noise is assumed.
- the color distribution of background pixels differs from the color distribution of the foreground pixels significantly
- pixel colors do not depend on each other
- locally contiguous object surfaces building compact shapes (if the Potts model is adjusted to favor compact voxel regions)

summary of required parameters

- color map $\mathcal{L}^c$ (may be calculated wrt. the images for a given size $|\mathcal{L}^c|$)
- number of color space partitions b
- Potts model parameter α ($\beta = 1$ without loss of generality)
- voxel grid dimension $V_{width}, V_{height}, V_{depth}$
- learn rate for the foreground and background distribution learning

5 Implementation

The implementation of the probabilistic model is focused on efficiency and flexibility at the same time. Due to encapsulated objects and well defined interfaces, it is easy to exchange parts of the model, e.g. for testing. Besides that, there are many special data structures and algorithms providing an efficient test framework. This chapter briefly discusses the main aspects of the implementation. More detailed information can be found in appendix B.

5.1 Logarithmic Model

There are several reasons to use log-likelihoods instead of original values. One advantage is the efficiency because all products can be replaced by summations which is usually much faster than multiplication. Moreover, this transformation is a lot more computationally secure as it leaves out the multiplication of many small numbers and therefore prevents discretization and underflow errors.

The conditional distribution for the label in a single node r defined in equation (4.39) is equivalent to

$$P(k_r|k_{\mathcal{R}\setminus\{r\}}, x) = \exp\Bigg[\sum_{r'\in\mathcal{N}_r} \ln g^s(k_r^s, k_{r'}^s) + \sum_{i=1}^{n}\Big[\sum_{u\in fg^i(\mathrm{v}_r)} \ln q(x_u^i, k_r^c) + \sum_{u\in BG^i} \ln f^i(x_u^i)\Big] - \ln Z_r\Bigg] \tag{5.1}$$

The definition of the foreground observation models can also be further simplified to

$$\ln q(x_u^i, k_r^c) = \frac{d}{2\left|\mathcal{R}_{fg^iu}\right|} \ln \frac{\left|\mathcal{R}_{fg^iu}\right|}{2\pi\sigma^2} - \frac{\left\|x_u^i - k_r^c\right\|^2}{2\sigma^2} \tag{5.2}$$

Underflow Prevention

The main benefit of this method which significantly improves the accuracy of the calculations is pre-normalization. Since the marginal probability distribution can be calculated up to a common factor they can be multiplied with an arbitrary factor. During the sampling process, the probabilities for each label in node r are calculated:

$$
\begin{aligned}
p_1 &= \ln P(K_r = l_1 | k_{\mathcal{R}\setminus\{r\}}, x) \\
p_2 &= \ln P(K_r = l_2 | k_{\mathcal{R}\setminus\{r\}}, x) \\
&\vdots \\
p_m &= \ln P(K_r = l_m | k_{\mathcal{R}\setminus\{r\}}, x) \qquad\qquad \forall j = 1, \ldots, m : l_j \in \mathcal{L}, m = |\mathcal{L}|
\end{aligned}
$$

Before applying the exp function one could for example subtract the largest log-likelihood (there are all negative!) to prevent underflow errors:

$$p_i^* = p_i - max_{j=1}^{m} p_j \tag{5.3}$$

This does not change the proportions within the likelihood distribution but it improves the accuracy when using finite data types.

5.2 Sampling Process

With respect to the implementation a single sampling step (for a particular node) can be divided into three parts. First, the probabilities for each label are calculated separately. Then, a sample is randomly selected according to the calculated marginal probability distribution. Finally, the label is actually changed (in contrast to the first step).

One disadvantage of the proposed MRF approach is the vast number of calculations needed for the sampling process, especially for large MRFs. Furthermore, the general advantage of MRFs to require only local calculations cannot be exploited due to the semi local visibility neighborhood $\mathcal{N}_r^v$.

Therefore, an essential part of the implementation is the use of appropriate data structures make the calculations on todays computers feasible. It is possible to avoid most of the expensive ray tracing operations during the sampling process. Since it has already been said that visibility is a binary function over depth and it can be represented with a camera aligned depth map, called *visibility map*. However, for the first part of the sampling step one also needs the voxels behind the regarded one for the case it is chosen to be empty. These ray tracing operations can also be pre-calculated and saved in a map which holds the next occupied voxel on the ray, called *nextOccVoxelMap*. The advantage of these pre-calculations is that these data structures only need to be updated, if the label in the voxel actually change (part three of the sampling step). With the knowledge that the sampling process converges and that many nodes in the MRF keep their labels during a sampling cycle this is an enormous reduction of calculations. The important data structures used in the implementation are explained in more detail in appendix B.2.

Voxel State Transitions

Both definitions of visibility functions (section 4.4) are very simple, but on the other hand they are more difficult to implement with respect to the chosen data structures. Then, one also needs to consider the possible visibility states of a voxel, which are: invisible, visible, next visible. Furthermore, one needs to distinguish between foreground and background

voxels if the voxel is empty. For the simple surface visibility function this leads to 7 distinguishable states for the first part of the sampling step and 15 possible transitions for the final part which updates all data structures. Figure 5.1 shows the state transitions for the update step when using the surface visibility function.

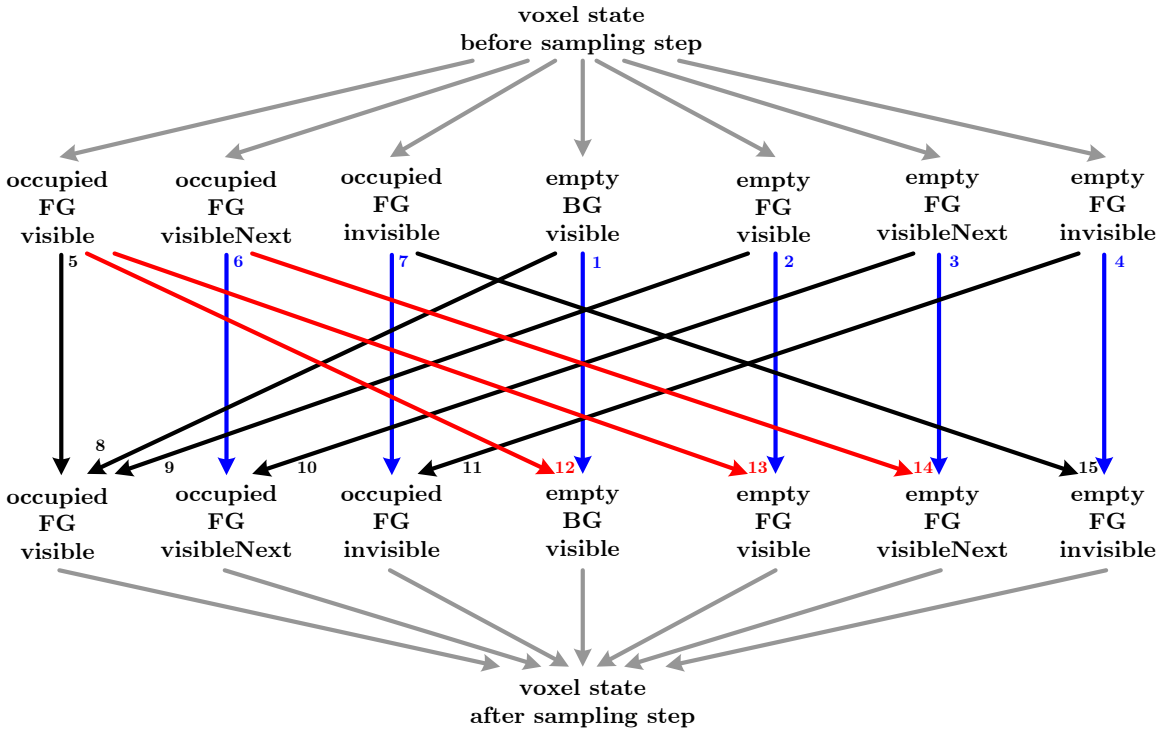


Figure 5.1: Possible voxel state transitions for the surface visibility function.

The most important updates corresponding to figure 5.1 are:

- **1-4, 6, 7**: no updates necessary
- **5**: foreground color distribution learning, if color has changed
- **8**: foreground and background distribution learning,
  update: *visibility map, pixelOccVoxelCounter*
- **9**: foreground color distribution learning,
  update: *visibility map, nextOccVoxelMap, pixelOccVoxelCounter*
- **10**: update: *nextOccVoxelMap, pixelOccVoxelCounter*
- **11**: update: *pixelOccVoxelCounter*
- **12**: background distribution learning, update: *visibility map, pixelOccVoxelCounter* (may need **ray tracing** in some cases)
- **13**: foreground distribution learning, update: *visibility map, nextOccVoxelMap, pixelOccVoxelCounter* (needs **ray tracing**)
- **14**: update: *nextOccVoxelMap, pixelOccVoxelCounter* (needs **ray tracing**)
- **15**: update: *pixelOccVoxelCounter*

It can be seen that only 3 of 15 possible transitions need ray tracing for an update step. Furthermore, transition **12** only needs ray tracing, if the *pixelOccVoxelCounter* which counts the number of occupied voxels on a ray is not consistent. This can happen if the

voxel size is significantly smaller or larger than the pixel span volume (discussed in the next section).

The state transitions for the fuzzy visibility function is even more complex. Due to the possible intervals of *fuzzy* labeled voxels one has to distinguish between 6 different locations (instead of only 3), because a voxel can hide/unhide, split/unify a *fuzzy* interval, can be attached in front or at the end of a *fuzzy* interval and the cases for the next occupied voxel map are similar to the surface visibility function. This leads to 10 possible cases for the first part of the sampling step and 30 different state transitions in which many of them are non-trivial to update (7 of them need ray tracing operations).

5.3 Pixel-Voxel Size Proportion

It has turned out to be difficult to use voxels efficiently and accurately at the same time. For efficiency reasons it has been decided only to use the centers of voxels. This implies the assumption that voxels approximately have the same size as the span volume of pixels with respect to their distance and camera parameters. The use of voxels being significantly smaller leads to inconsistencies in combination with the proposed visibility maps. On the other hand, if voxels are bigger than the pixel span volume they color several voxels and one needs to model the size of voxels to find about which pixels are colored by a voxel.

Therefore, this approach more or less assumes a one-to-one relation between voxels (at particular depth) and pixels. Nevertheless, there are still problems remaining even if this assumption holds. Consider a voxel changing its state from occupied to empty during a sampling step. Then, the next occupied voxel behind gets interesting. This is reached by traversing the ray from the regarded voxel to a camera in backward direction. The ray traversal algorithm (see appendix B.1) returns all voxels being intersected by this ray. If the start point of this ray (the regarded voxel center) is close to the pixel span volume boundaries, the ray traversal algorithm may deliver many voxels which centers do not project to the same pixel. This leads again to inconsistencies with the assumed one-to-one relation. The simplest solution is to ignore such voxels when searching for the next occupied voxel. Depending on the ray angle to the voxel grid one could miss up to 10 voxels. Therefore, all neighbor voxels in perpendicular direction to the camera ray are searched in the case that the traversed voxel center does not project to the regarded pixel. Still, this approach is not perfect but it minimizes the number of errors to a tolerable level.

5.4 Geometry Calculations

5.4.1 Coordinate Systems

Since the number of voxels is an essential parameter the reconstruction volume can have arbitrary location, size and orientation independent from the number of contained voxels, i.e. the voxel length, width and height can be chosen independently. This leads to five coordinate systems used to describe a general scene for stereo reconstruction which are explained in the following list (the letters in parentheses indicate the subscript being used for all variables regarding the coordinate system):

- (W) world coordinate system
- (C) camera coordinate system (camera projection center is the origin)
- (R) reconstruction volume coordinate system
- (P) pixel/image coordinate system
- (V) voxel coordinate system

The coordinate system definitions are illustrated in figure 5.2.

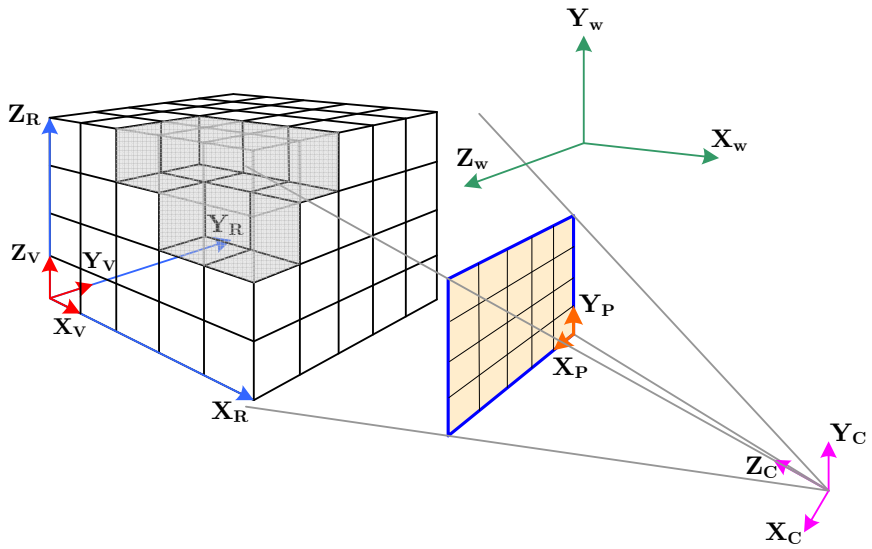


Figure 5.2: Illustration of the used coordinate systems.

The necessary basic transformations together with their matrix names are: translation $\mathbf{T}_{\{t_x,t_y,t_z\}}$, rotation $\mathbf{R}_{\{\gamma_x,\gamma_y,\gamma_z\}}$, scale $\mathbf{S}_{\{s_x,s_y,s_z\}}$ and projection $\mathbf{K}_{\{f,m_x,m_y,p_x,p_y\}}$ which are listed in appendix A.1. A combination of such matrices will be called a transformation matrix and will be denoted by $\mathbf{P}$.

5.4.2 Coordinate Transformations

$\mathbf{P}$ denotes a general transformation matrix where a subscript $\mathbf{P}_{\text{TO FROM}}$ denotes that the matrix transforms from coordinate system FROM to coordinate system TO (written in reverse order according to the order in which matrices are applied to a vector).

The matrix $\mathbf{P}_{RV}$ transforms a voxel index to a world point in the reconstruction volume coordinate system. The voxel grid is defined to be within the unit cube independently from its dimension. To get the voxel center out of the index 0.5 is added.

$$\mathbf{P}_{RV} = \mathbf{S}_{\{V_{width}^{-1}, V_{height}^{-1}, V_{depth}^{-1}\}} \cdot \mathbf{T}_{\{0.5,0.5,0.5\}} \tag{5.4}$$

The reconstruction volume can have an arbitrary position and orientation in the world coordinate system and for more flexibility the voxel size can be changed by a scale factor.

$$\mathbf{P}_{WR} = \mathbf{R}_{\{\gamma_{Rx}, \gamma_{Ry}, \gamma_{Rz}\}} \cdot \mathbf{S}_{\{s_{Rx}, s_{Ry}, s_{Rz}\}} \cdot \mathbf{T}_{\{t_{Rx}, t_{Ry}, t_{Rz}\}} \tag{5.5}$$

where the variables in the subscripts are all parameters which need to be set to define the reconstruction volume position (they are not given!).

The camera coordinate system (for camera i) can be reached by translation and rotation of a given world point:

$$\mathbf{P}_{CW}^{i} = \mathbf{T}_{\{t_{Cx}^{i}, t_{Cy}^{i}, t_{Cz}^{i}\}} \cdot \mathbf{R}_{\{\gamma_{Cx}^{i}, \gamma_{Cy}^{i}, \gamma_{Cz}^{i}\}} \tag{5.6}$$

With the projection, being the last operation, one gets homogeneous pixel coordinates from a point in camera space:

$$\mathbf{P}_{PC}^{i} = \mathbf{K}_{\{f^{i}, m_{x}^{i}, m_{y}^{i}, p_{x}^{i}, p_{y}^{i}\}} \tag{5.7}$$

The parameters used in the last two equations are the given camera parameters, where (5.6) uses the external and (5.7) uses the internal camera parameters.

Summarizing all equations above the transformation of a voxel index into a (homogeneous) pixel coordinate can be expressed by a single matrix:

$$\mathbf{P}_{PV}^{i} = \mathbf{P}_{PC}^{i} \mathbf{P}_{CW}^{i} \mathbf{P}_{WR} \mathbf{P}_{RV} \tag{5.8}$$

Based on this equation the projection function $proj^{i}$ described in section 3.1.1 is the application of the overall transformation matrix $\mathbf{P}_{PV}^{i}$ to the homogeneous vector of $\mathtt{v}_r$ combined with the back transformation to inhomogeneous coordinates:

$$proj^{i}(\mathtt{v}_r) = (\mathbf{x}/\mathbf{z},\ \mathbf{y}/\mathbf{z})^{T} \tag{5.9}$$

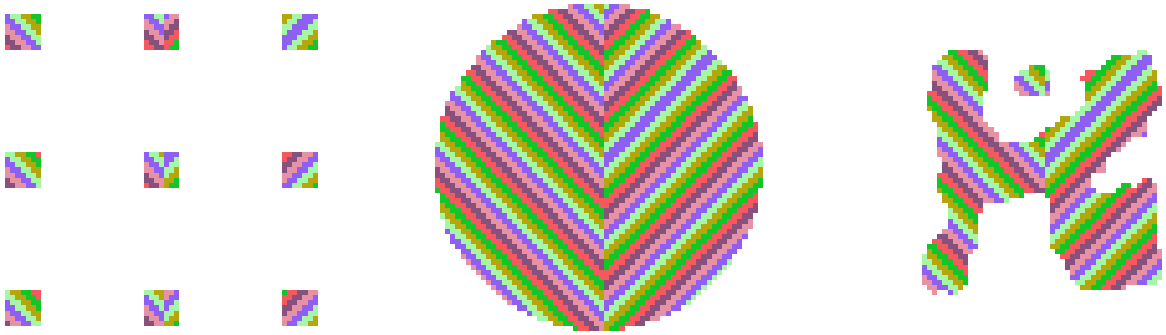
$$\text{with: } (\mathbf{x},\mathbf{y},\mathbf{z})^{T} = \mathbf{P}_{PV}^{i} \cdot (\mathbf{x}_r, \mathbf{y}_r, \mathbf{z}_r, 1)^{T}, \quad (\mathbf{x}_r, \mathbf{y}_r, \mathbf{z}_r)^{T} = \mathtt{v}_r$$

6 Evaluation

6.1 Data Sets

6.1.1 Handmade data sets

For simple tests four 2D (one voxel slice) data sets have been generated with a binary labeling (*occupied*/*empty*). These data sets are then textured by the test framework to be able to influence the texture properties. Figure 6.1 shows an overview of the data sets being textured with stripes of random colors (color map size: 8).



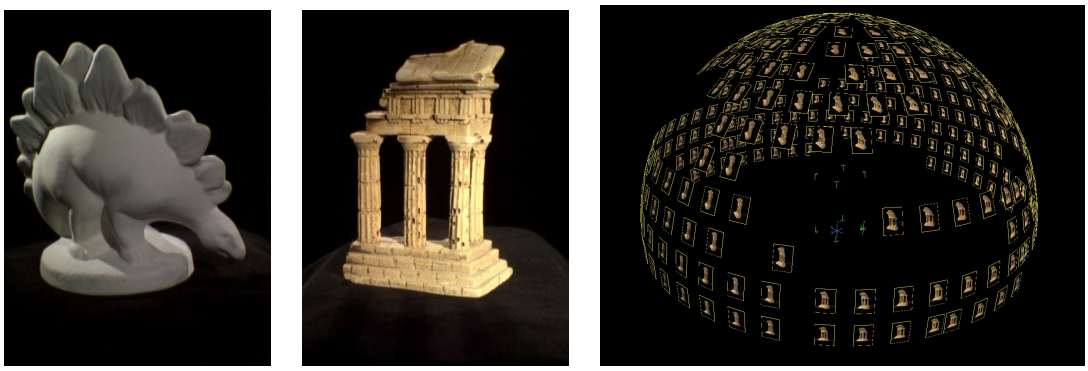
9 pillars circle splotch

Figure 6.1: Generated 2D data sets with a sample texture of 8 colors

The test framework then provides the possibility to align n cameras equidistantly on a circle around the scene. For the most tests 16 cameras have been used if not denoted otherwise. Two resolutions of the same voxel tests scenes have been tested $64 \times 64 \times 1$ and $256 \times 256 \times 1$. For better readability only the results of the small data sets are presented. However, the reconstruction results were in general more accurate on the larger data sets which may be because of smaller discretization errors. The camera images have then been generated with a simple backward ray tracing method. The image resolutions were 60×1 for the small (64^2) data set and 250×1 for the larger data set.

6.1.2 Middlebury data sets

The Middlebury College in Vermont [Col] offers two free data sets for multi-view stereo reconstruction, as well as a possibility to evaluate the results in comparison with the true models. One of the data sets depicts a temple being a plaster reproduction of the "Temple of the Dioskouroi" in Agrigento, Sicily. The second data set shows a plaster stegosaurus. Figure 6.2 (a),(b) shows two sample images of each data set. The images were acquired using the so called Stanford Spherical Gantry which enables moving a camera on a sphere to specified latitude/longitude angles. Both data sets consist of 363



(a) (b) (c)

Figure 6.2: Middlebury data sets (a) dino data set. (b) tempel data set. (c) hemisphere

calibrated viewpoints on a full hemisphere (fig. 6.2 (c)). Furthermore, both data sets are divided into three evaluation sets, being a *sparse ring* (16 cameras) around the object, a *full ring* (48 cameras) and the *full hemisphere* (363 cameras). For testing, only the *sparse ring* data set has been used. Its camera configuration has been shown in figure 2.8.

6.1.3 Image RGB-Clustering

Due to the fact that a color map has been used to reduce the amount of data and to accelerate the sampling process an appropriate color map needs to be calculated from the input images. K-Means clustering is used to find a particular number of (sub)optimal color representatives (cluster centers) in the RGB color space for the observation data. The K-Means algorithm minimizes the distance of points to their representatives by iteratively changing the class membership of points and the position of cluster centers. The algorithm finds a local minimum and is described in [OL06]. Moreover, the associated software has been used to cluster the input images.

All input images of the dino *sparse ring* data set were used as input data for the clustering. Randomly selected RGB-data points were used as initial cluster centers and the euclidean distance has been selected as distance function to generate color maps of size $8, 16, 32, 64, 128$ and 256. This has also been done for the temple data set.

6.2 Functionality of the Model

To be able to test the foreground observation model independently from the background observation model a priori silhouette information has also been used. The background color has also been assumed to be noisy with a Gaussian distribution around a known background color:

$$f^i(x_u^i) = \hat{q}(x_u^i, c_{BG}) \tag{6.1}$$

where c_{BG} is the background color of the images. In the following these background models will be referred to either the histogram background model or the Gaussian background model.

In the following tests the surface visibility function has been selected.

The Gibbs sampler probability estimates will be referred to as relative frequencies (RF) in the following, i.e. the frequency the Gibbs sampler has selected a particular label.

Figure 6.3: Tests with the pillars scene with $\alpha = 10$. The figure depicts a cross correlation between start labelings: true scene (true), visual hull (vh) and the background models: Gaussian, histogram used for the experiments. Furthermore, one picture always shows the labeling after 100 sampling cycles and a second shows the according relative frequency (RF) for label *empty*

Figure 6.3 shows the result of 4 experiments to reconstruct the pillars scene when starting from the true scene / the visual hull and using the histogram background model / Gaussian background model. The figure also depicts the visual hull in which occupied voxels have the color being the closed to the mean color value of all its observations (without taking the visibility into account). It can be seen that the labeling during the sampling process on both background models stays close to the true scene. In contrast, when starting from the visual hull, the histogram background model converges slowly to the fully occupied scene

whereas the Gaussian background model (which still possesses silhouette information) converges slowly to the true scene.

In general, it has turned out that the learning process of the histogram background model is very sensitive to the sampling process, especially if the "distance" of the current labeling is far from the true scene labeling. The sampling process then usually converges to the fully occupied scene.

The choice of the learn rate and the Potts weight does not have any effect on this problem. The use of strong interactions in the Potts model does actually slow down the convergence but does not stop or change it. However, it is still possible to do reconstructions with the histogram background model (which will be seen later), but the learning process is far from being robust and one needs to be lucky to find an appropriate configuration of all parameters. Therefore, the Gaussian background model has been selected for most of the following experiments (if not denoted otherwise).

The main problem for the learning process of the histogram background model is the slow convergence of the sampling process to the true scene. This affects the learning process and therefore changes the probability distribution for background pixels which again affects the sampling process. Due to the semi local visibility constraints of the proposed model the sampling process has very special properties which are discussed in the following.

6.3 Sampling

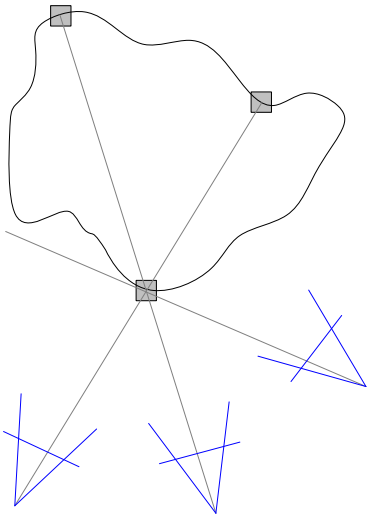


Figure 6.4: Sampling step illustration.

Due to the semi local visibility conditions the sampling process is less flexible to get from one arbitrary labeling to another arbitrary labeling. In general it has turned out that the sampling is more flexible to enlarge the current labeling (i.e. add occupied voxels) than shrinking it. This is due to the fact that the Gibbs sampler is only able to change a single label at the same time.

During a sampling step the probability for a voxel having a particular color is weighted against the colors of the voxels behind to express the probability that the voxel is empty. This is illustrated in figure 6.4. With the use of the Potts Model the area behind reconstructed surfaces is usually filled up with voxels having a random color. Consequently, the color of a single voxel is often weighted against voxels with random colors and it is therefore less likely that several voxels will have the appropriate color with respect to their observation. However, carving incorrect voxels is very important during the search for the most likely set of scenes.

In theory, Gibbs sampling on a MRF can be regarded as a Markov chain in which the whole MRF labeling represent the labels of the chain. Let the superscripted a denote the current sampling step, i.e. sampling in one node r and a superscripted $*$ denote a new, randomly selected label. Then, the Markov chain can be illustrated in the following way:

$$\cdots \longrightarrow \begin{pmatrix} k_1^{a-1} = k_1^{a-2} \\ k_2^{a-1} = k_2^{a-2} \\ \vdots \\ k_{r-1}^{a-1} = k_{r-1}^{*} \\ k_r^{a-1} = k_r^{a-2} \\ k_{r+1}^{a-1} = k_{r+1}^{a-2} \\ \vdots \\ k_{|\mathcal{R}|}^{a-1} = k_{|\mathcal{R}|}^{a-2} \end{pmatrix} \longrightarrow \begin{pmatrix} k_1^{a} = k_1^{a-1} \\ k_2^{a} = k_2^{a-1} \\ \vdots \\ k_{r-1}^{a} = k_{r-1}^{a-1} \\ k_r^{a} = k_r^{*} \\ k_{r+1}^{a} = k_{r+1}^{a-1} \\ \vdots \\ k_{|\mathcal{R}|}^{a} = k_{|\mathcal{R}|}^{a-1} \end{pmatrix} \longrightarrow \begin{pmatrix} k_1^{a+1} = k_1^{a} \\ k_2^{a+1} = k_2^{a} \\ \vdots \\ k_{r-1}^{a+1} = k_{r-1}^{a} \\ k_r^{a+1} = k_r^{a} \\ k_{r+1}^{a+1} = k_{r+1}^{*} \\ \vdots \\ k_{|\mathcal{R}|}^{a+1} = k_{|\mathcal{R}|}^{a} \end{pmatrix} \longrightarrow \cdots$$

step $a-1$ step a step $a+1$

Thus, only one label in a particular node can be changed in one sampling step and it needs $|\mathcal{R}|$ steps for one sampling cycle in which every node has once the chance to change its label. It is easy to imagine that it needs a huge number of sampling cycles to walk from one arbitrary labeling to another arbitrary labeling for large MRFs. Additionally, this number highly depends on the transitions probabilities. If some of these transitions are 0 it may be impossible to get from one labeling to the other via Gibbs sampling, because the set of possible labels is then divided into partitions and the start labeling will define which partition is chosen. This is not the case in the proposed model, since none of the probabilities is 0, but - keeping the idea of a partitioned set of labelings in mind - some probabilities may be very small which makes it less likely to get from one partition to another.

Initial Labelings

The following initial labelings have been regarded:

- **True Scene** only for the 2D data sets available
- **Visual Hull** (section 2.1.1), if background information is available
- **Photo Hull** (section 2.1.6)
  With a simple definition of a consistency measure, the *Photo Hull* seems to be a good initialization for the sampling process because the *Photo Hull* is usually very close to true scene. However, the proposed $Space-Carving-Algorithm$ is extensive to implement and could therefore not been used in this work.
- **Empty Scene**
- **Random Labeling** of the *RSV* volume (with 10%,30%,50%,100% occupancy)

Except for the Photo Hull, all above mentioned initializations have been tested as start labeling for the sampling process. It has turned out that the true scene, the empty scene

and the fully occupied scene are attractors for the sampling process. Due to the problems mentioned above the sampling process does usually not converge to the true scene when starting from the empty, full or randomly labeled scene (for a small number of sampling cycles, i.e. < 5000 cycles). Neither of the proposed background models is able to force the sampling process to converge to the true scene when starting from an arbitrary labeling.

An extended a priori model has been tested which helps to solve the problem of convergence partially. Additional to the state a priori term an a priori value for general occupancy is introduced in the following:

$$P(k) = Z_{g^{sc}}^{-1} \prod_{r \in \mathcal{R}} g^o(k_r^s) \prod_{r' \in \mathcal{N}_r^s} g^s(k_r^s, k_{r'}^s) \tag{6.2}$$

with

$$g^o(k_r^s) = \begin{cases} \kappa & \text{if } k_r^s = \textit{ occupied} \\ 1 - \kappa & \text{if } k_r^s = \textit{ empty} \end{cases} \qquad \textit{with } \; 0 \leq \kappa \leq 1 \tag{6.3}$$

This modified a priori model changes the original probability distribution ($\kappa \neq 0.5$). Especially, for small occupancy probabilities $\kappa < 0.5$ (which is the case for most of the data sets) the a priori model hinders the Potts model to fill the space behind reconstructed occupied voxels and leads to worse reconstruction results. The modified a priori model is therefore only used in the beginning of the sampling process to accelerate the search for the approximate true visibility configuration and is deselected after a certain number of sampling cylces.

6.4 Reconstruction Results

With the modified a priori model it is possible to converge to the true scene during sampling when starting from a random labeling with at least 10% occupancy. The empty scene as start labeling does still not converge to the true scene even with the modified a priori model. Figure 6.5 shows the reconstruction of the pillars scene using the Gaussian background model with a random start labeling (50% occupancy) and a prior occupancy of 5% selected for the first 30 sampling cycles. Figure 6.5 (b) depicts the labeling after 30 cycles and shows that the visibility configuration is close to the true configuration and that the Potts model has not filled up the pillars with occupied voxels, (c) shows the associated probability distribution for empty voxels (white $\hat{=}$ 1, black $\hat{=}$ 0). Then, after deselecting the prior occupancy, the Potts model fills up the pillars (d) and the sampling process converges approximately to the true scene (e), (f).

The small relative frequencies of label *empty* at the border of the reconstruction volume in figure 6.5 (c) are due to the fact that Gibbs sampler immediately selects voxels to have a color close the colors of its projection whereas invisible voxels get a random label. An evaluation of the reconstruction is given in table 6.1.

In general, the reconstruction with the a priori occupancy from a random labeling is not robust and it may need several experiments with different parameter sets to obtain a good reconstruction result.

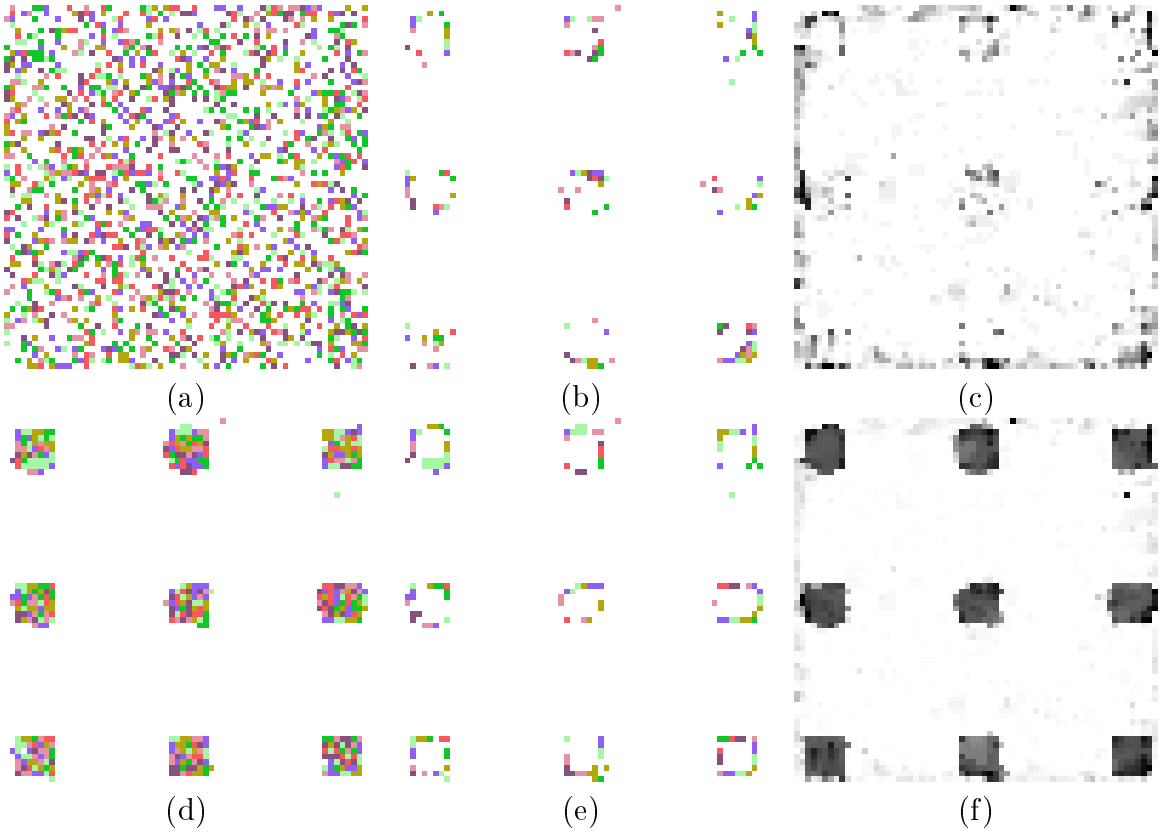


Figure 6.5: Sampling result with 100 cycles, $\alpha = 5$, $\kappa = 0.05$ prior occupancy for 30 cycles. (a) random start labeling (b) labeling after 30 cycles (c) RF label *empty* after 30 cycles (d) labeling after 100 cylces (e) MAP decision after 100 cycles (f) RF label *empty* after 100 cycles

occupied correct	133	3.25%
occupied incorrect	35	0.85%
empty correct	3845	93.87%
empty incorrect	83	2.03%
overall	4096	100 %
hamming distance	118	2.88%

Table 6.1: Evaluation table for the reconstruction shown in figure 6.5

To illustrate the quality of reconstructed surface voxels in a better way, the (invisible) randomly colored voxels in the object centers are filtered by thresholding the relative frequencies with a value of 0.5. This is done for all 2D test images in this chapter showing a map decision.

Figure 6.6 illustrates the reconstruction results for the circle data set which needed more sampling cycles for a good reconstruction result. This time the prior occupancy has been chosen to be $\kappa = 0.1$ and has been used for the first 50 cycles. Fig. 6.6 (a) again shows that the modified prior model hinders the Potts model to fill up the inside of the object, but it helps to find the object boundaries and sets up the correct visibility configuration.

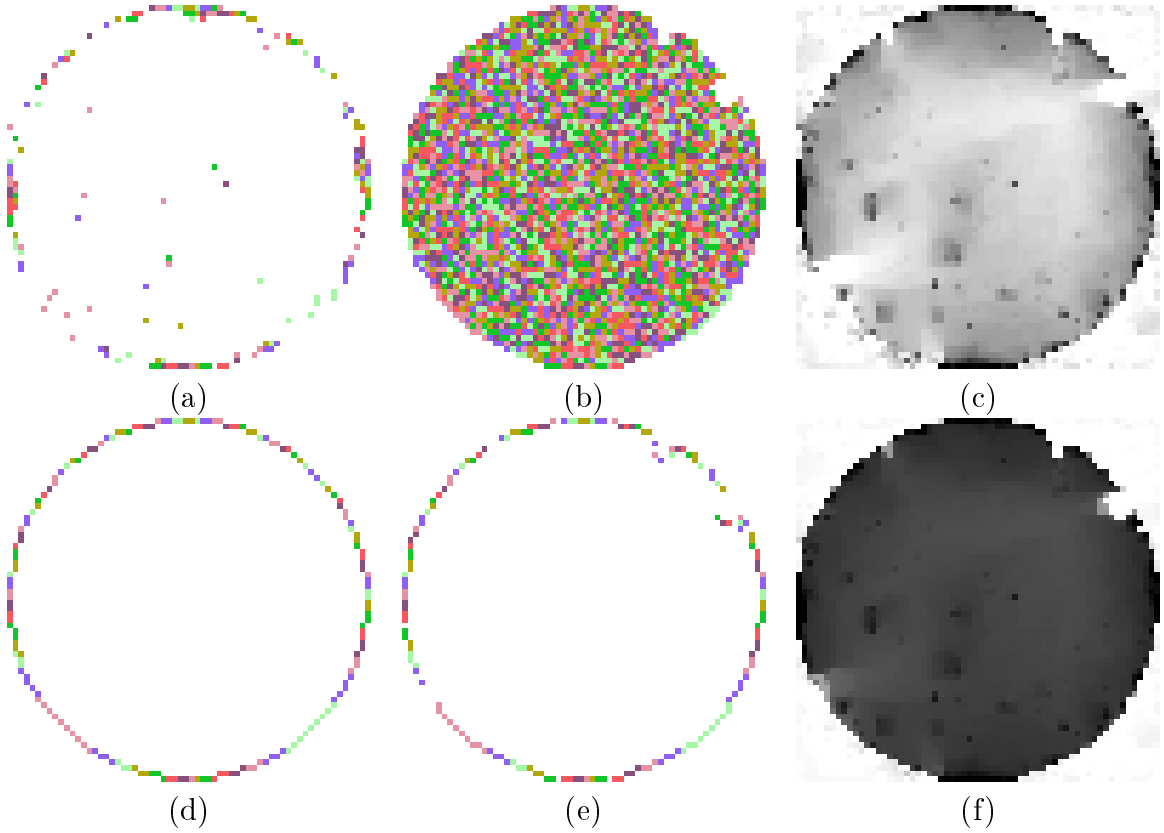


Figure 6.6: Reconstruction result: random start labeling, 300 cycles, $\alpha = 5$, $\kappa = 0.1$ prior occupancy for 50 cycles. (a) labeling after 50 cylces (b) labeling after 300 cylces (c) RF label *empty* after 100 cycles (d) true scene (e) MAP decision after 300 cycles (f) RF label *empty* after 300 cycles

Concavities

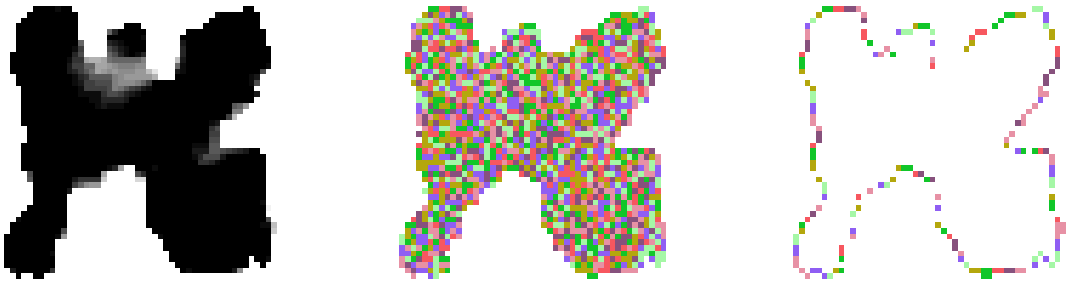


Figure 6.7: Sampling results: true start labeling, 400 cycles, $\alpha = 8$, without prior occupancy. (a) RF label *empty* after 20 cylces (b) labeling after 400 cylces (c) MAP decision after 400 cycles

Concavities are in general more difficult to reconstruct which can be seen in figure 6.7. Even if the start labeling is the true scene the sampling process blurs the surfaces in concave areas. This may be due to sparse observation and homogeneously colored surfaces.

The Potts model suppresses concave surface parts too, but the problem does not disappear if the Potts model is turned off ($\alpha = 1$).

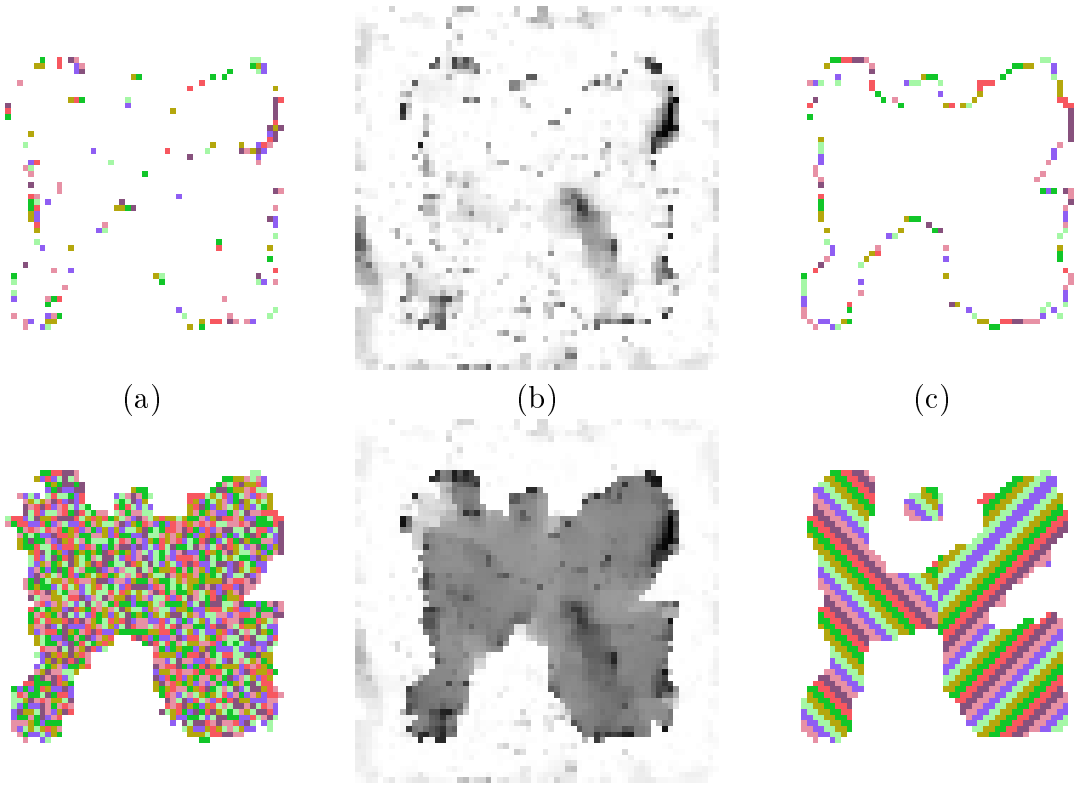


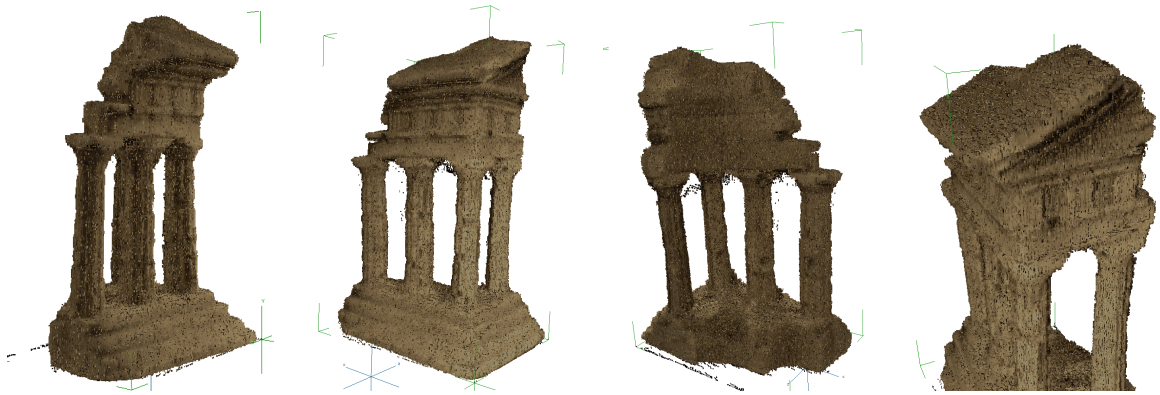
Figure 6.8: sampling result: random start labeling, 400 cycles, $\alpha = 4$, $\kappa = 0.1$ prior occupancy for 100 cycles. (a) labeling after 90 cycles (b) RF label *empty* after 90 cycles (c) MAP decision after 400 cycles (d) labeling after 200 cycles (e) RF label *empty* after 200 cycles (f) true scene

While figure 6.7 has shown that concavities are not always reconstructable figure 6.8 shows that the same results can be achieved with the modified a priori model when starting from a random labeling.

Middlebury Data Sets

The 3D data sets from the Middlebury College contain only images of resolution 640×480 which is compared to size of the tight object bounding volume a high resolution. Due to limited resources the reconstruction could only be realized with a maximum of 256^3 voxels. Using this resolution a voxel is much bigger than a pixel span volume which stays in conflict with the proposed implementation (5.3). Therefore, the images have been scaled down to 320×240 and the calibration data has been adapted appropriately.

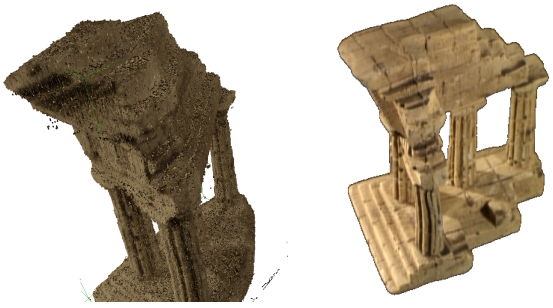
The following figure 6.9 shows the reconstruction result of the temple data set with 256^3 voxels. The figure shows the labeling after 200 sampling cycles. The prior occupancy has been used for 50 cycles but without any noteworthy success. The reconstruction result is still very close to the initial visual hull. Whereas small concavities in the facade beneath the roof could be reconstructed properly the big concavities at the back of the roof and the steps at the backside of the basement could not be carved.



(a) (b) (c) (c)

Figure 6.9: Reconstruction of the temple: 256^3 voxels, 16 colors, 200 sampling cycles, visual hull as start labeling, $\alpha = 8$, prior occupancy $\kappa = 0.05$ for 50 cycles

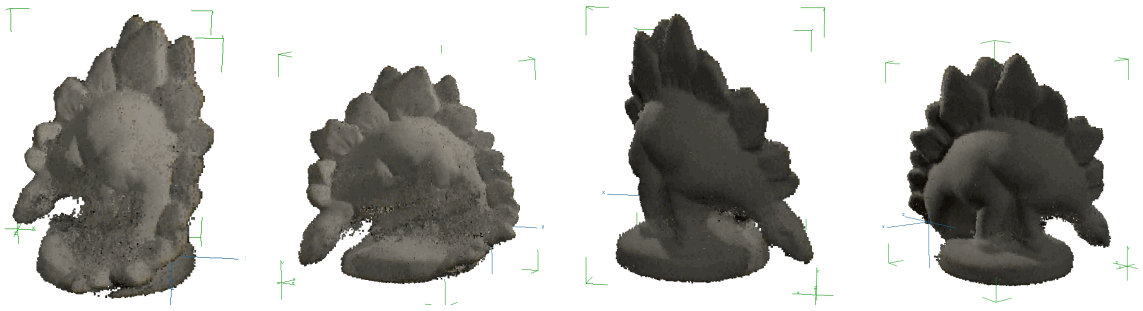
The shortcoming of the method with respect to concavities is especially shown in figure 6.10. The backside of the roof (a) is according to the visual hull still filled with occupied voxels which is a concave area in the original image (b). Note, that the shown bird view of the temple (b) has not been used for the reconstruction and that the input images have been scaled down for the reconstruction. Due to light conditions the color information is poor on the backside of the temple. Fig. 6.9 (c) shows that the roof almost looks like a concave area due to homogeneous coloring.



(a) (b)

Figure 6.10: Temple data set: concavities, same reconstruction as in fig. 6.9 (a) reconstructed model (b) one of the original images

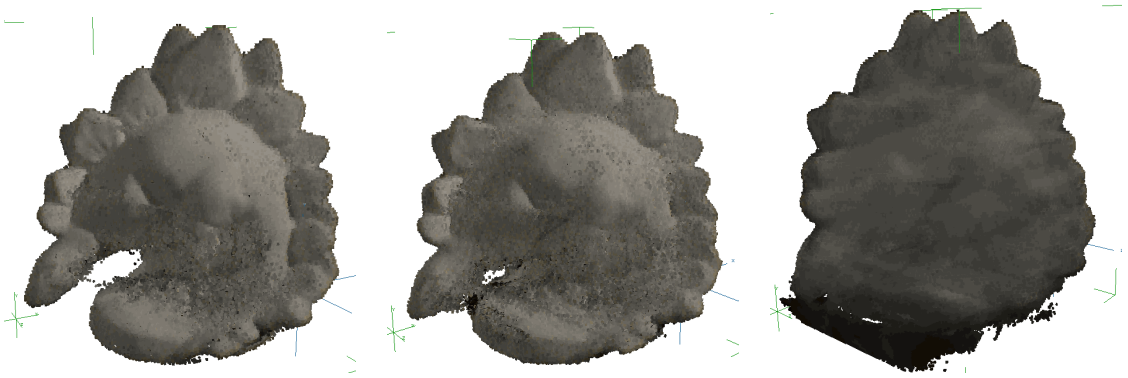
Figure 6.11 gives an overview of the reconstruction results of the dino data sets. Pictures (c) and (d) show that the concave areas between the legs have been well reconstructed. In contrast, the concavities between the legs on the opposite site (a), (b) are not properly reconstructed and may need more sampling cycles to achieve better results.



(a) (b) (c) (d)

Figure 6.11: Dino reconstruction 128^3 voxel, 32 colors, 200 sampling cycles, visual hull as start labeling, $\alpha = 8$, prior occupancy $\kappa = 0.05$ for 80 cycles

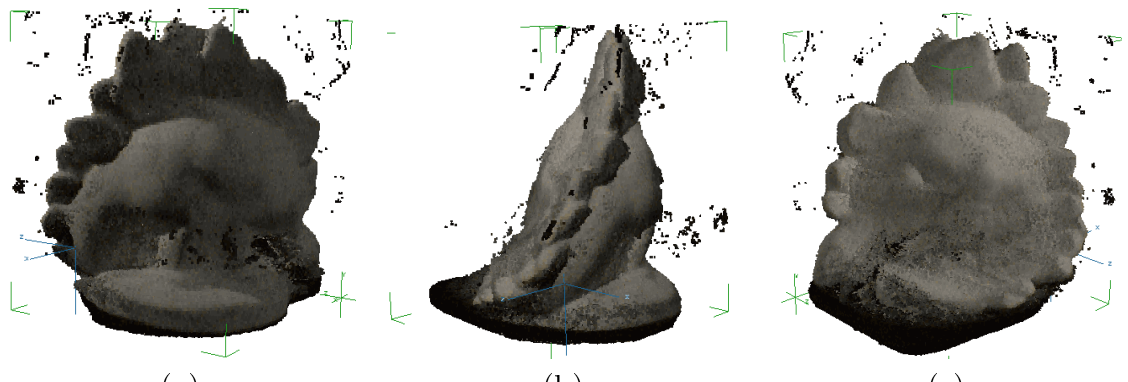
The best reconstruction results have been achieved with the modified a priori model, because this accelerates the carving of voxels which also helps to uncover concavities. Figure 6.12 illustrates the difference between the proposed a priori models. In (a) the concavities around the head of the dino are properly reconstructed whereas in (b) this area is still filled with occupied voxels. For both reconstructions the visual hull (c) has been used as start labeling and except for the different a priori models all parameters were the same.



(a) (b) (c)

Figure 6.12: Difference between the proposed a priori models shown at the dino reconstruction: 128^3 voxel, 32 colors, 200 sampling cycles, visual hull as start labeling, $\alpha = 8$, (a) with prior occupancy $\kappa = 0.05$ for 80 cycles, (b) without prior occupancy (c) visual hull

Figure 6.13 again shows a reconstruction of the dino data set, but this reconstruction has been done with the proposed histogram background model. This shows at least the capabilities of the model. On the other hand, the model has generated artifacts with voxels being colored very close to the background color. This is due to the fact that the background distribution is assumed to be piecewise constant. Then, the background color (black) may be in the same histogram cell as some of the foreground pixel colors (darkgray).



(a) (b) (c)

Figure 6.13: Reconstruction result with the histogram background model and the same parameters as above: 128^3 voxel, 32 colors, 100 sampling cycles, visual hull as start labeling, $\alpha = 8$, without prior occupancy

Some tests have also been made to reconstruct the Middlebury data sets with a random

start labeling in combination with the modified a priori model which led to highly perforated reconstructed objects. Results with a comparable quality to the above outcomes could not be achieved.

6.5 Color A Priori Model

The a priori model defined in chapter 3, equation (3.20) only evaluates the state of voxels. Additionally, another a priori model has been tested which evaluates the color of neighbored voxels too. It is defined as

$$P(k) = Z_{g^{sc}}^{-1} \prod_{r \in \mathcal{R}} g^o(k_r^s) \prod_{r' \in \mathcal{N}_r^s} g^s(k_r^s, k_{r'}^s) \cdot g^c(k_r^c, k_{r'}^c) \tag{6.4}$$

where $g^o(k_r^s)$ is again the prior occupancy term described above and $g^c(k_r^c, k_{r'}^c)$ is chosen to be a Gaussian

$$g^c(k_r^c, k_{r'}^c) = \exp\left[- \frac{\|k_{r'}^c - k_r^c\|^2}{\sigma_c^2} \right] \tag{6.5}$$

and $g^s(k_r^s, k_{r'}^s)$ is the Potts model defined in section 3.2.1.

This a priori model has mainly been tested on the Middlebury data sets which mostly satisfy the assumption that neighbored voxels have similar colors. In the randomly textured 2D data sets the prior color model disturbs the reconstruction and has therefore not been used. The neighborhood structure has been chosen to be the same 6-neighborhood as for the states.

(a) (b) (c)

Figure 6.14: Sampling results of the temple with 128^3 voxels: (a) without a priori color information (b) color a priori model with strong interactions ($\sigma_c = 5$) (c) one of the original images. The brightness differences are due to the color map reduction.

In general, the color a priori model does not significantly change the occupancy but (of

course) the color values of neighbored voxels. Figure 6.14 shows the difference of the colored reconstruction results. The color a priori model leads to a smoother surface coloring compared to the noisy surface colors of the normal a priori model.

6.6 Parameters

Used Parameters

- color map size $|\mathcal{L}^c|$: All color map sizes up to 256 and being a power of 2 have been tested. The Middlebury data sets have only been tested up to 64 due to limited memory resources.
- color space partitions b: For the histogram distribution learning the color space has been divided into 64^3 cells of equal size.
- The number of necessary sampling steps depends on the scene to be reconstructed, but there is usually only little change when the sampling process has reached one of the attracting labelings. Generally, there has not been a big change in any of the tested data sets between 500 and > 1000 sampling cycles.
- learn rate: The foreground parameter learning is very robust and the learn rate has been varied between 0.1 and 0.5 without any significant differences in the reconstruction results. The background distribution learning could not be tested reasonably and values between 0.05 and 0.4 did work depending on the data set. In most cases a learn rate of 0.2 has been selected for both foreground and background distribution learning.
- The reconstruction volume position, size and orientation has been adjusted according to the data sets to fit the object of interest within a small bounding box. For the Middlebury data sets these values are given additionally to the calibration data.

Number of Voxels

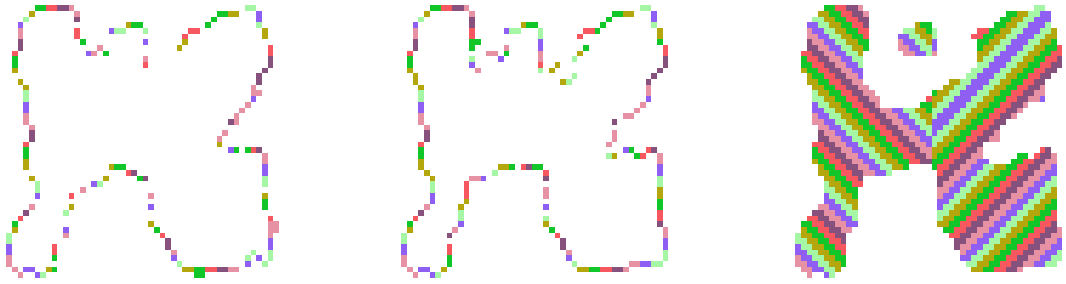
A higher number of voxels usually leads to more accurate reconstruction results, but this value could only be changed within a small interval. First, due to the assumed (approximately) one-to-one relation between voxels and pixels and, second, due to limited resources. For the 2D data sets the voxel grid dimensions have been chosen to be either 64^2 or 256^2. The 3D data sets have been tested with grid dimensions of 128^3 and 256^3.

Number of Cameras

In general the results of a reconstruction become better with a higher number of cameras. Figure 6.15 shows that especially concavities have been reconstructed more accurately.

Parameter Learning

The learning of the foreground color distribution is very robust, even with different learn rate values. The start value does not have a big influence on the learning which is illustrated in figure 6.16 showing 4 different start values for the sampling on the pillars data set (starting from a random labeling). Depending on the choice of the learn rate the standard deviation value converges rapidly. In this case the learn rate has been 0.2. To



(a) (b) (c)

Figure 6.15: Results of the sampling after 400 cycles, true start labeling, $\alpha = 8$. (a) MAP decision, 16 cameras (b) MAP decision, 32 cameras (c) true scene

minimize the number of necessary sampling steps the initial value for σ should be nearby the true value. For the presented experiments values between 20 and 40 have usually been used.

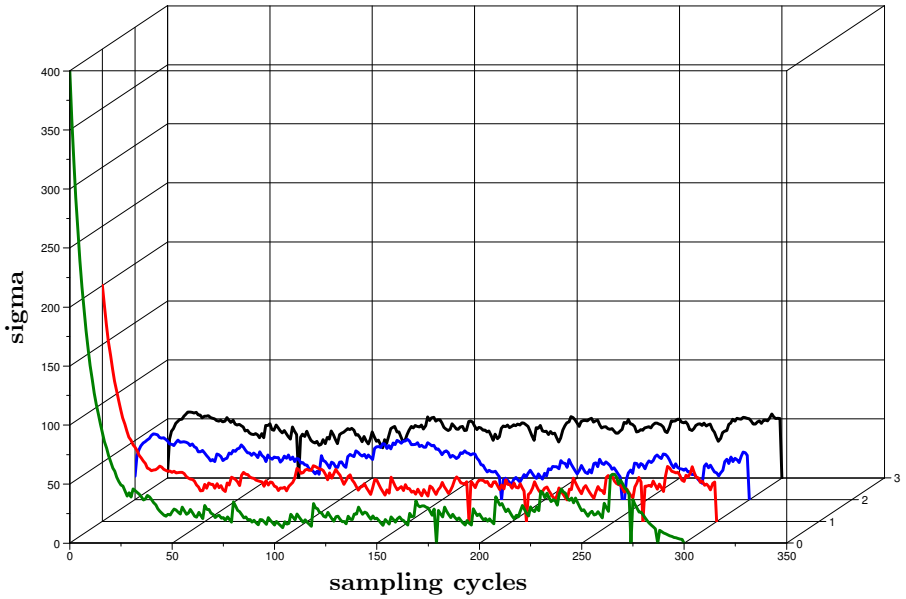


Figure 6.16: Color standard deviation of the foreground model during sampling on the pillars data set with intial σ values: $5, 20, 200, 400$

The learning of the histogram distribution is not robust, because it is negatively affected by the sampling process. In addition, the color space had to be reduced due to performance reasons which leads to a worse approximation of the real background color distribution on the one hand and promotes an overlap of foreground and background color distribution on the other. The initial value of the histogram distribution has always been chosen to be an equal distribution which has then been adapted to the chosen initial labeling. Due to the complexity of the background histogram distribution (3D in the RGB space) it is difficult to visualize the learning process or to find some quantitative measures for evaluation. Moreover, due to the available data sets the background distribution learning could not be properly tested and will need further investigation in future work.

Potts Model Parameter

The Potts model has been chosen to evaluate edges independently of their direction. Table 6.2 justifies this assumption. The evaluation of a dino reconstruction with deselected Potts weights shows that state dependencies are equally distributed over the different dimensions.

	overall	empty		occupied	
number of voxels	2097152	1443543	68.833496%	653609	31.166506%
edge occurences	overall	equal states		non-equal states	
x-direction	2080768	2049795	98.511459%	30973	1.488537%
y-direction	2080768	2046001	98.329124%	34767	1.670873%
z-direction	2080768	2049638	98.503922%	31130	1.496082%
overall	6242304	6145434	98.448166%	96870	1.551831%

Table 6.2: Evaluation result of the a dino reconstruction with deselected Potts model, 128^3 voxels, 100 sampling cycles.

As a result of the evaluation, the Potts model is not necessary for stereo reconstruction but it helps to improve the quality of the results. As assumed in section 3.2.1 it closes gaps in the surface and fills up the interior of models. The model parameter α has been varied between 1 and 50. The best reconstruction results could usually be obtained with a Potts weight arround 10.

6.7 Summary and Discussion

It has turned out that silhouette information (either in background model or in the start labeling) has always been necessary to let the sampling process converge to the true scene. However, the model of the background color distribution and its learning process needs further investigation.

Due to its complexity with respect to the implementation the fuzzy visibility function could not be tested within this thesis. Its detailed investigation will be subject for future work.

Sampling

Experiments have shown that the sampling process is sensitive to the initial labeling and less flexible due to the visibility constraints. This hinders the sampling to uncover concavities and lead to worse reconstruction results. This problem can partially be compensated with the use of a prior occupancy value. In sum, the model needs to be improved with respect to the sampling process. It may be possible to add some kind of color quality measure in order to prefer the *empty* label in case that the voxel color does not fit to its observation with respect to this measure. Another possible solution may be to preselect the color labels of invisible voxels for the case that they become visible. As a result, invisible voxels would have the appropriate color when they are weighted against the neighbor voxels in camera direction. This approach is possible since invisible voxels get a random label and are thus unimportant for the reconstruction result. On the other hand, it may be hard to find a good strategy for the color selection.

7 Summary and Outlook

7.1 Conclusion

In this thesis the elementary problems of stereo reconstruction have been investigated and a world representation which is able to describe a set of possible reconstruction results has been developed. A flexible probabilistic approach for stereo reconstruction has been defined and studied in this thesis. The defined model is able to work as a stand-alone method for stereo reconstruction as well as within a framework of two modules in which information is exchanged iteratively to improve reconstruction results. In addition, some of the model parameters have been proposed to be estimated with optimization techniques (unsupervised learning). A method to handle the problem of the unknown background color distribution has been proposed, but it could not be shown that this method is able to tackle this problem. The problem to describe the background color distribution therefore needs further investigation. The method has proven to be very flexible not only in theory but also in the implementation. Due to the plug-in arrangement of modules it is easy to exchange a priori or observation models. Moreover, the implementation shows that the proposed method can be realized in an efficient way (compared with the average times of other reconstruction methods reported on the Middlebury [Col] evaluation page).

The Gibbs sampling approach stays in conflict with the constraints of visibility in stereo reconstruction and further investigation needs to be done to improve the reconstruction method. With the availability of silhouette information the presented model is able to reconstruct objects from calibrated images. However, the method is not robust to reconstruct concavities accurately which also needs further investigation.

In sum, it has been shown that the model is flexible to use different a priori and observation models and tests have proven that the presented model is able to reconstruct artificial and real world scenes. Furthermore, it is able to keep ambiguities within the reconstruction result to work as pre-calculating module within in an iterative framework stereo reconstruction framework.

7.2 Future Work

Due to simplification of the voxel projection (voxel center) and discretization errors not all available information is used for reconstruction. A. Broadhurst shows in his PhD. Thesis [Bro99] that voxel oversampling is an approach to overcome this problem and may be worth to investigate in combination with the proposed method.

It might be useful to weight the importance of information from different cameras according to their location and orientation, e.g., cameras looking at each other are very unlikely to observe the same surface. This thought is mainly motivated by efficiency reasons but one may consider that the number of cameras observing the same surface will increase

the accuracy of the estimated depth value on the one hand but due to calibration errors the accuracy will decrease from a certain number of cameras on the other hand.

The problem of the pixel-voxel size proportion can be tackled for the case that voxels are larger than the span volume of pixels (which is currently not supported by the implementation). With moderate effort one could modify the projection cache to save the voxel corner projections (instead of the centers). After the projection of all corner points to the projection plane one could use the polygon of their convex hull to identify all pixels which need to be updated.

Moreover, one could extend the model to the Mixture of Gaussian approach in the observation model which as been mentioned in section 3.2.2. While this would need effort for an efficient implementation the model would be more flexible with respect to the voxel resolution, if the color distribution learning and visibility function with their update strategy is adapted appropriately.

The assumption of color constancy (Lambertian model) is a main deficit of the proposed approach. Davis et al.[Wan] proposed an interesting approach to describe the radiance in a scene. Instead of assuming color consistency they define a measure called light transport consistency which is able to handle reflections and is still independent from light source positions. This approach may also be applicable in combination with the presented method.

A very interesting topic could also be the sampling process. The shortcoming of sampling in Markov random fields is the often huge number of nodes which need to be evaluated. On the other hand, in many tasks there are often labels which are more interesting then others. In this thesis this would be the fact that occupied scene parts are the interesting parts, not the empty ones. This reminds of the change over from Markov-Localization to Monte-Carlo-Localization in robotics.

Imagine to split up the process into two parts: a spatial sampling process which selects nodes and the common Gibbs sampling step which is done in the afore selected node. Starting with an equally spatial distribution one could focus the sampling process to areas in which labels of interest are selected with higher probability. This could be done, for example, by using a mixtures of Gaussians which are updated according to the sampling process and which determine the current spatial distribution for the node selection process. This would keep all advantages of a MRF and might not only be applicable to this task. Of course, it is difficult to provide an appropriate update scheme for the node selection process because it highly influences the convergence criteria of the Gibbs sampler, but this investigation might be a step in the right direction.

A Mathematical Issues

A.1 Transformation Matrices

translation matrix

$$\mathbf{T}_{\{t_x,t_y,t_z\}} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_x \\ 0 & 0 & 1 & t_x \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.1)}$$

scale matrix

$$\mathbf{S}_{\{s_x,s_y,s_z\}} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.2)}$$

rotation matrix (x-axis)

$$\mathbf{R}_{x;\gamma_x} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\gamma_x & -\sin\gamma_x & 0 \\ 0 & \sin\gamma_x & \cos\gamma_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.3)}$$

rotation matrix (y-axis)

$$\mathbf{R}_{y;\gamma_y} = \begin{pmatrix} \cos\gamma_y & 0 & \sin\gamma_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\gamma_y & 0 & \cos\gamma_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.5)}$$

rotation matrix (z-axis)

$$\mathbf{R}_{z;\gamma_z} = \begin{pmatrix} \cos\gamma_z & -\sin\gamma_z & 0 & 0 \\ \sin\gamma_z & \cos\gamma_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.4)}$$

projection matrix

$$\mathbf{K}_{\{f,m_x,m_y,p_x,p_y\}} = \begin{pmatrix} f \cdot m_x & 0 & m_x p_x & 0 \\ 0 & f \cdot m_y & m_y p_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{(A.6)}$$

The overall rotation matrix $\mathbf{R}$ is defined as $\mathbf{R}_{\{\gamma_x,\gamma_y,\gamma_z\}} = \mathbf{R}_{x;\gamma_x}\mathbf{R}_{y;\gamma_y}\mathbf{R}_{z;\gamma_z}$.

The shown projection matrix represents a CCD[1] camera model [Har00, page 143] with f being the focal distance, p the principal point and m the scale factor according to the pixel size.

A.2 Derivations and Lemmas

A.2.1 From Bayesian to MAP-Estimation

Bayesian estimation is identical to maximum a posteriori estimation in case of the following locally additive cost function:

[1] CCD = Charge-coupled Device

$$c(k, k') = \sum_{r \in \mathcal{R}} c(k_r, {k_r}') \tag{A.7}$$

with the local cost function

$$c(l, l) = 1 - \delta_{ll'} = \begin{cases} 0 & \text{if } l = l' \\ 1 & \text{otherwise} \end{cases} \tag{A.8}$$

where $\delta_{ll'}$ is the Kroneckersymbol. The Bayesian optimization strategy is then to minimize the expected value of the mean costs which yields together with the additive cost function:

$$k^* = \arg\min_{k'} \sum_{k \in \mathcal{K}} P(k|x) c(k, k') \tag{A.9}$$

$$= \arg\min_{k'} \sum_{r \in \mathcal{R}} \sum_{k \in \mathcal{K}} P(k|x) c(k_r, {k_r}') \tag{A.10}$$

By fixing the label l in random variable $K_r = l$ the sum over all labelings k can also be written as as a sum over all labelings k which have this particular label l at node r. Additionally, one needs to sum up over all labels $l \in \mathcal{L}$. After that, the cost function can be factored out and the resulting sum (in brackets) is equivalent to the marginal $P(K_r = l \mid x)$:

$$k^* = \arg\min_{k'} \sum_{r \in \mathcal{R}} \sum_{l \in \mathcal{L}} \sum_{k \in \mathcal{K}: k_r = l} P(k|x) c(k_r, {k_r}') \tag{A.11}$$

$$k^* = \arg\min_{k'} \sum_{r \in \mathcal{R}} \sum_{l \in \mathcal{L}} \Big[\sum_{k \in \mathcal{K}: k_r = l} P(k|x) \Big] c(l, {k_r}') \tag{A.12}$$

$$k^* = \arg\min_{k'} \sum_{r \in \mathcal{R}} \sum_{l \in \mathcal{L}} P(K_r = l \mid x) c(l, {k_r}') \tag{A.13}$$

The minimization task in equation (A.13) consists of a sum over all nodes $r \in \mathcal{R}$ where each summand does only depend on its corresponding label ${k_r}'$. Hence, the summands can minimized independently and the task decomposes into $|\mathcal{R}|$ independent minimization tasks:

$${k_r}^* = \arg\min_{{k_r}'} \sum_{l \in \mathcal{L}} P(K_r = l \mid x) c(l, {k_r}') \tag{A.14}$$

Applying the definition of the chosen cost function yields

$$k_r{}^* = \arg\min_{k_r{}'} \sum_{l\in\mathcal{L}} P(K_r = l \mid x)(1 - \delta_{lk_r{}'}) \tag{A.15}$$

$$= \arg\min_{k_r{}'} \Big[\sum_{l\in\mathcal{L}} P(K_r = l \mid x) - \sum_{l\in\mathcal{L}} P(K_r = l \mid x) \cdot \delta_{lk_r{}'} \Big] \tag{A.16}$$

$$= \arg\min_{k_r{}'} \Big[\sum_{l\in\mathcal{L}} 1 - P(K_r = l \mid x) \cdot \delta_{lk_r{}'} \Big] \tag{A.17}$$

Since the Kroneckersymbol is only 1 if $l = k_r{}'$ it can be substituted and one gets the task to minimize the negative marginal probability which is equivalent with maximizing the marginal probability:

$$k_r{}^* = \arg\min_{k_r{}'} \Big[1 - P(K_r = k_r{}' \mid x) \Big] \tag{A.18}$$

$$= \arg\max_{k_r{}'} P(K_r = k_r{}' \mid x) \tag{A.19}$$

In short, the optimal Bayesian decision for the chosen costs function is to locally select the label with the largest probability according to the given observation. This is equivalent to the MAP-decision.

A.2.2 Normalization Factor for the Foreground Observation Model

The foreground observation model has been defined (3.23) over pixels and the probability for a single pixel is a product of $\left|\mathcal{R}_{fg^iu}\right|$ Gaussians. The product of these Gaussians is only in case of $\left|\mathcal{R}_{fg^iu}\right| = 1$ a probability distribution

$$\sum_{c\in\mathcal{C}} \prod_{r\in\mathcal{R}_{fg^iu}} \hat{q}(c,\mu) \leq 1 \quad \forall(u \in FG^i, \mu \in \mathcal{C}) \tag{A.20}$$

With the requirement that all visibible voxels observed by a pixel have the same color

$$\forall u \in FG^i, \forall r \in \mathcal{R}_{fg^iu}, \exists c \in \mathcal{C} : \quad k_r^c = c \tag{A.21}$$

one is able to normalize the product above such that

$$\sum_{c\in\mathcal{C}} \prod_{r\in\mathcal{R}_{fg^iu}} Z_u^{-1} \cdot \hat{q}(c,\mu) = 1 \quad \forall(u \in FG^i, \mu \in \mathcal{C}) \tag{A.22}$$

The task is to find the appropriate normalization factor Z_u^{-1}.

In general, a product of m Gaussians is given by

$$g_1(x;\mu) = \prod_{j=1}^{m} \frac{1}{(\sqrt{2\pi}\sigma)^d} \exp\left[-\frac{\|x-\mu_j\|^2}{2\sigma^2} \right] \tag{A.23}$$

$$= \frac{1}{(\sqrt{2\pi}\sigma)^{d\cdot m}} \exp\left[-\frac{\sum_{j=1}^{m} \|x-\mu_j\|^2}{2\sigma^2} \right] \tag{A.24}$$

with the requirement: $\forall j : \mu = \mu_j$ one gets:

$$= \frac{1}{(\sqrt{2\pi}\sigma)^{d\cdot m}} \exp\left[-\frac{m\cdot\|x-\mu\|^2}{2\sigma^2} \right] \tag{A.25}$$

$g_1(x;\mu)$ does not sum up to one over all x (except for $m = 1$) since it is a product of Gaussians (it would sum up to one when summing over all μ's). In contrast $\mathcal{G}_2(x;\mu)$ sums up to one since it is a single Gaussian distribution

$$\mathcal{G}_2(x;\mu) = \frac{1}{(\sqrt{2\pi}\hat{\sigma})^d} \exp\left[-\frac{\|x-\mu\|^2}{2\hat{\sigma}^2} \right] \tag{A.26}$$

with the substition $\sigma = \dfrac{\hat{\sigma}}{\sqrt{m}}$ one gets:

$$= \left(\frac{\sqrt{m}}{\sqrt{2\pi}\sigma} \right)^d \exp\left[-\frac{m\cdot\|x-\mu\|^2}{2\sigma^2} \right] \tag{A.27}$$

which also represents a product of m (unnormalized) Gaussians but it still sums up to one over all x. With the definition of the function

$$g_3(x;\mu) = \left(\frac{\sqrt{m}}{\sqrt{2\pi}\sigma} \right)^{\frac{d}{m}} \exp\left[-\frac{\|x-\mu\|^2}{2\sigma^2} \right] \tag{A.28}$$

one gets the property that this function to the power of m is a single Gaussian:

$$\mathcal{G}_2(x;\mu) = \Big(g_3(x;\mu) \Big)^m \tag{A.29}$$

$$= \left(\frac{\sqrt{m}}{\sqrt{2\pi}\sigma} \right)^d \exp\left[-\frac{m\cdot\|x-\mu\|^2}{2\sigma^2} \right] \tag{A.30}$$

Hence, function $q(x_u^i, k_r^c)$ should have a form similar to equation (A.28) and the observation function can be defined by:

$$q(x_u^i, k_r^c) = \prod_{u\in fg^i(\mathtt{v}_r)} \left(\frac{\sqrt{|\mathcal{R}_{fg^iu}|}}{\sqrt{2\pi}\sigma} \right)^{\frac{d}{|\mathcal{R}_{fg^iu}|}} \exp\left[-\frac{\|x_u^i - k_r^c\|^2}{2\sigma^2} \right] \tag{A.31}$$

A.2.3 Shannon Theorem

Let $\alpha_i, i = 1, \ldots, n$ be positive constants and $x_i, i = 1, \ldots, n$, positive variables for which it holds $\sum_{i=1}^{n} = 1$. In this case the inequality holds

$$\sum_{i=1}^{n} \alpha_i \log x_i \leq \sum_{i=1}^{n} \alpha_i \log \frac{\alpha_i}{\sum_{j=1}^{n} \alpha_j} \tag{A.32}$$

and the equality comes only when $x_i = \alpha_i / \sum_{j=1}^{n} \alpha_j$ for all i.

The theorem and its proof can be found in [Hla02, page 242].

B Implementation Details

B.1 Ray Traversal

The widely known Bresenham (or Midpoint) algorithm is a very fast ray discretization algorithm. Due to tests it has been pointed out that the algorithm is not sufficient for the voxel ray traversal task because the algorithm occasionally omits voxels intersecting the ray which leads to errors in the sampling update calculations.

Instead, a slightly modified algorithm has been implemented based on the algorithm presented by [Woo]. Due to the modifications the algorithm also supports backward iteration and clipping to the voxel volume boundaries. This is necessary as a ray is usually defined by a voxel center and the camera ray. During sampling it is interesting to find out about the next occupied voxel in the backward camera direction. Hence, the clipping is necessary as there is no boundary except of the reconstruction volume boundaries.

B.2 Speed Ups and Memory Savings

B.2.1 Visibility Maps

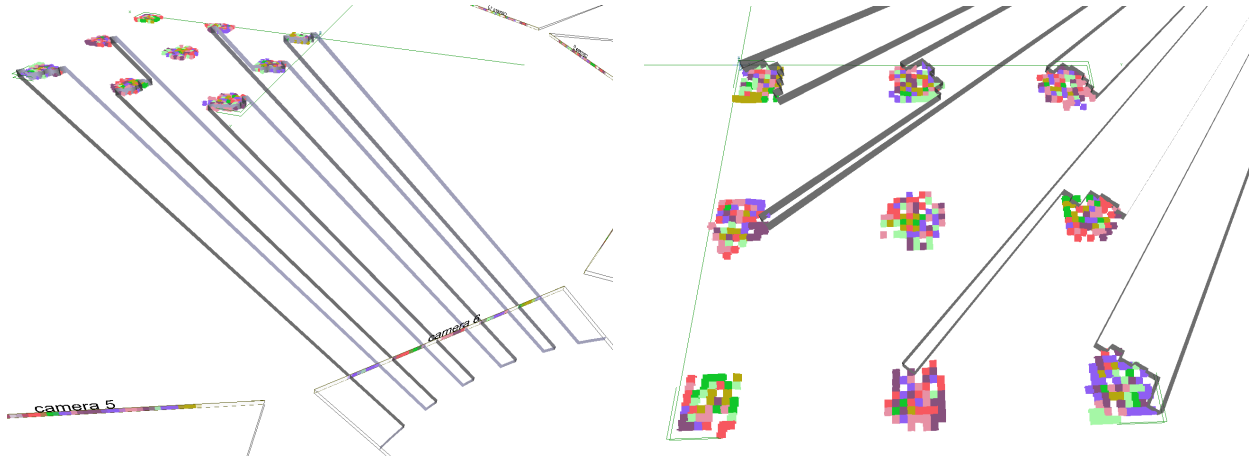
From the viewpoint of a camera the visibility is a binary function of the depth value. Thus, it is possible to use depth maps for cameras which are well-known methods in computer graphics and allow rapid visibility calculations for objects regarding a certain camera. A visibility map simply stores a depth value for each pixel where the visibility ends on this camera ray. During the sampling process simple depth value comparisons are superseding most of the expensive ray tracing operations. Additionally, depth maps do not need much memory and are easy to update at the same time. For flexibility the implemented visibility maps also support resolutions higher than the image resolution.

B.2.2 Next Occupied Voxel Maps

This map consists of two parts: first, the label index to the second occupied voxel (SOV) on the camera ray for each pixel and second the depth value of this particular voxel.

For the sampling process the marginal probability distribution for all labels in one node has to be calculated. To avoid ray tracing the first part stores the next occupied voxel and the probability that one particular voxel is empty can be determined rapidly.

The depth value of the SOV then allows to find out quickly if label index for the SOV needs to be updated in case of a label change in between the first and the second occupied voxel.



(a) (b)

Figure B.1: Visibility maps (the gray bands) showing where the visibility ends in a sample voxel scene. A visibility of zero means that the pixel observes the background.

B.2.3 Projection Cache

The projection cache is a simple array which saves the pixel index for each voxel projection in each camera. This accelerates the sampling process massively as the projection is needed n times for each node in the MRF (with n being the number of cameras) and an expensive matrix multiplication is replaced by two simple table lookups. However, the main drawback of this method is the large memory usage, an approach to overcome this disadvantage is given in the next paragraph.

B.2.4 Bit Data Field

The so called BitDataField is a trade off between memory usage and a little more overhead in index calculations. In programming languages fields (or arrays) of data are bounded to bit sizes which are a power of two. Depending on the number of bits needed and the size of the field this can waste lots of memory. The main motivation to create this class was the usage for the projection cache explained above, but it can be used for any other task with a small effort of code changes. The BitDataField is an array of values having arbitrary bit sizes and their are managed in linear memory space without wasting a single bit. To illustrate its usefulness the following table shows the memory usage of normal arrays and the BitDataField in comparison by using the projection cache example. The chosen test data images have an image size of 640×480 which is a very common graphics format. This means that only 19 bits are needed to store the pixel index for each voxel projection. Thus, 13 bit per value would be wasted when using normal array structures.

Note, that these values are per camera, e.g. using 16 cameras for reconstruction with 256^3 voxels saves 416 MB of memory. Of course, the memory savings have to be paid with additional calculations, but this is still only one multiplication, a few binary shift operations and some conditional branches for the index calculation. Except of the one multiplication, all other operations can be done in one processor clock and are thus very fast computed.

number of voxels	128^3	256^3	512^3
bit size per value	memory usage		
32 bit (normal array)	8 MB	64 MB	512 MB
19 bit (BitDataField)	4.75 MB	38 MB	304 MB
saved memory	3.25 MB	26 MB	208 MB

Table B.1: Memory usage of the projection cache per image for different numbers of voxels.

In sum, the additional index calculation is by far faster than the matrix multiplication for the projection. Since, this task is done n times for each voxel in each sampling step, the impact of this data structure regarding computational time is enormous.

B.2.5 Color Map Index Cache

The color map index cache is another lookup table to accelerate calculations. It stores the nearest neighbor color map value in color space for each pixel. This saves a nearest neighbor search per camera for a sampling step in each node.

B.3 Processing Time

Table B.2 gives a brief overview of the computation times of the proposed implementation. These times may be different for other scenes and the used reconstruction parameter set, because the computation time highly depends on the number of necessary ray tracing operations and the sampling convergence.

number of voxels	64^2	256^2		128^3	256^3
number of cameras	16	16	32	16	16
8 colors	02 sec	33 sec	00:59 min	-	-
16 colors	03 sec	47 sec	01:24 min	00:36 h	03:55 h
32 colors	05 sec	01:13 min	02:13 min	00:56 h	-
64 colors	08 sec	02:16 min	03:47 min	01:20 h	-
128 colors	16 sec	03:52 min	07:14 min	-	-
256 colors	29 sec	07:27 min	13:20 min	-	-

Table B.2: Computation times for 100 sampling cycles on the pillars 2D data set. Measured on a AMD 64 Athlon 3000+, 2GHz, 1.5GB RAM PC with Windows XP.

Note, that sampling process gets faster with the convergence of the sampling process due to fewer ray tracing operations.

B.4 Software Structure

Overview

The following software parts have been implemented:

- 3D test framework
- loading of scenes from the Middlebury multi-view stereo evaluation page
- 2D test framework providing 2D scene generation
- visual hull algorithm if silhouette information is available
- simplified space carving algorithm for data analyzing

The software has been implemented fully object orientated which makes it easy to exchange components rapidly (e.g for testing). Apart from the implementation of the algorithms and methods presented in thesis a visualization module has been created for the evaluation of algorithms and results.

Used Software Packages

The software has been implemented in C++ using Trolltechs Qt Framework [Sof06b] and the SIM Coin3D library [Sof06a] which implements the SGI Open Inventor application programming interface (API) being a VRML (Virtual Reality Modeling Language) based extension to the SGI OpenGL API. Furthermore, Coin3D offers some more functionality with the extension packages SIMVoleon and SIMage. The following enumeration summarizes the used packages and gives a little description:

- **Trolltech Qt** The main functionality of this library is platform independent Window/GUI management.
- **SIM Coin3D** Implements the SGI Open Inventor API and offers the use of VRML techniques to describe three dimensional scenes.
- **SIMVoleon** Systems in Motion (SIM) offers several extensions to the Coin3D library package. SIMVoleon supplies an interface to display and handle volumetric data (e.g. voxel arrays) efficiently.
- **SIMage** Being another extension package from SIM, the library supplies reading and writing functionality for several standard graphic and movie file formats.

All the above mentioned libraries are cross-platform and the whole test framework should work under Windows, Linux and Macintosh Systems. However, it has only been tested under Windows XP Professional.

Notation

a	sampling step
b	number of color space partitions
BG^i	set of background pixel indices regarding camera i
C^i	projection center of camera i (world coordinate system)
C	set of all camera positions
c	superscript to indicate the color of a label
c	color, i.e., an element of the color space $c \in \mathcal{C}$
$\mathcal{C}$	color space (either gray or RGB values)
$c(k, k')$	cost function for BAYESian estimation
d	dimension of the color space (gray: d=1 or RGB: d=3)
$\mathcal{E}$	set of all edges in graph $\mathcal{G}$
$\mathcal{E}^s$	set of edges in graph $\mathcal{G}$ regarding the voxel state
$\mathcal{E}^v$	set of edges in graph $\mathcal{G}$ regarding the voxel visibility
f	background observation function
$fg^i(\mathtt{v}_r)$	set of foreground pixel indices where $\mathtt{v}_r$ projects to
FG^i	set of foreground pixel indices regarding camera i
g	a priori function
$\mathcal{G}$	graph with nodes $\mathcal{R}$ and edges $\mathcal{E}$
i	camera index (always superscripted)
I^i	image of camera i
$I^i_{\mathbf{xy}}$	pixel with index $(x, y)^T$ in image I^i
$\mathcal{IN}^i$	set of ignored nodes
$\mathcal{IP}^i$	set of ignored pixels
k_r	labels connected to node r
k	labeling
$k^s_{\mathcal{N}^s_r}$	set of state labels of all nodes in the neighborhood of node r
K_r	random variable for a hidden state connected to node r
K	set of all random variables for the hidden states
$\mathbf{K}$	projection matrix
l	label
$\mathcal{L}$	set of labels
n	number of cameras in the scene
$\mathcal{N}_r$	neighborhood of node r
$proj^i$	projection function
$\mathbf{P}$	general transformation matrix
q	foreground observation function
ray	ray function, i.e., voxel traversal function
r	node in graph $\mathcal{G}$
$\mathcal{R}$	set of all nodes
$\mathcal{R}^i_u$	subset of all nodes which project to pixel index u in camera i
$\mathcal{R}_{BG^i}$	subset of all nodes which project to background pixels in camera i
$\mathcal{R}_{fg^iu}$	nodes being visibible, occupied and project to foreground pixel index u in i

$\mathcal{R}_{FG^i}$	subset of all nodes which project to foreground pixels in camera i
$\mathbf{R}$	rotation matrix
s	superscript to indicate the state of a label
$\mathbf{S}$	scale matrix
t	time / number of sampling cycles
$\mathbf{T}$	translation matrix
u	pixel index in the grid U
U	pixel grid
x^i	set of pixels from image I^i
x	observation, i.e., set of all pixel sets x^i
X	set of all random variables connected with the observation
X_u^i	random variable for pixel index u related to camera i
vis^i	visibility function
$\mathtt{v}_r$	voxel center connected with node r
V	set of voxel center positions
Z	normalization factor
$\delta_{ll'}$	KRONECKER-symbol
γ	rotation angle
κ	a priori occupancy
θ	set of unknown parameters
σ	standard deviation of the voxel color distribution

List of Abbreviations

API	application programming interface
BG	background
BRDF	bidirectional reflection distribution function
CCD	charged-coupled device
EM-algorithm	expectation maximization algorithm
FG	foreground
GRF	Gibbs random field
MAP	maximum a posteriori
MLE	maximum likelihood estimation
MRF	Markov random field
RF	relative frecuency
RGB	red green blue (color space)
RSV	reconstructable scene volume
VRML	virtual reality modeling language
wrt.	with respect to

Bibliography

[A.S] M.Schlesinger; B.Flach; B.Savchynskyy; D.Bukhantsov; A.Shekhovtsov. Active recognition. Unpublished Report.

[Bro99] T.W. Drummond; R. Cipolla; A. Broadhurst. *A Probabilistic Framework for Space Carving*. Department of Engineering,University of Cambridge,Cambridge, UK CB2 1PZ, 1999.

[Col] Multi-View Stereo Data Calibrated Camera Images; Middlebury College. Multi-view stereo evaluation. http://vision.middlebury.edu/mview/ (visited: 20/05/2007).

[Deu01] Oliver Deussen. *Computer Graphics, Lecture Script*. Dresden University of Technology, 2001.

[Fau] R. Faugeras, O.; Keriven. Variational principles, surface evolution, pdes, level set methods, and the stereo problem. In *IEEE Transactions on Image Processing*, volume Volume 7, Issue 3, March 1998 Page(s):336 - 344. Digital Object Identifier 10.1109/83.661183.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 6:721–741, 1984.

[Har00] Andrew Zisserman; Richard Hartley. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[Hla02] Michail I. Schlesinger; Vaclav Hlavac. *Ten Lectures on Statistical and Structural Pattern Recognition*, volume volmne 24 of Computational Imaging and Vision. Kluwer Academic Press, 2002.

[Kle96] Andreas Koschan; Karsten Schlüns; Reinhard Klette. *Computer Vision - Räumliche Information aus digitalen Bildern*. Vieweg Technik, 1996.

[Kut99] S.M. Seitz; Kiriakos N. Kutulakos. A theory of shape by space carving. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 307 – 314, 1999. 20-27 Sept., Volume 1.

[Lau94] Aldo Laurentini. The visual hull concept for silhouette-based image understanding. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 150 – 162, 1994. Feb., Volume 16, Issue 2.

[Lau03] A. Bottino; L. Jaulin; A. Laurentini. *Reconstructing 3D Objects from Silhouettes with Unknown Viewpoints: The Case of Planar Orthographic Views*, volume Volume 2905/2003. 2003. Book Series: Lecture Notes in Computer Science.

[Li00] Stan Z. Li. *Markov Random Field - Modeling in Computer Vision*. Spinger-Verlag Berlin Heidelberg New York Tokyo, 2000.

[OL06] Martin R. Oswald and Matthias Lange. Common cluster methods used for image clustering and object segmentation. Report: Cluster methods, Dresden University of Technology, 2006.

[Pon06] Yasutaka Furukawa; Jean Ponce. High-fidelity image-based modeling. Technical report, 2006. Technical Report 2006-02, UIUC, http://www-cvr.ai.uiuc.edu/ponce_grp/publication/tr/cvr_tr_2006_02.pdf (visited: 15/01/2007).

[Sla99] W.Bruce Culbertson; Thomas Malzbender; Greg Slabaugh. Generalized voxel coloring. In *Book Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms, Corfu, Greece, September 1999. Proceedings.* Lecture Notes in Computer Science, Volume 1883/2000, 1999.

[Sof06a] Software. Coin3d, sgi open inventor api, vrml api, additional packages: Simvoleon, simage. 2006. http://www.sim.no/, http://www.coin3d.com/ (visited: 15/03/2007).

[Sof06b] Software. Qt framework, trolltech. 2006. http://www.trolltech.com/ (visited: 10/05/2007).

[Str06] R. Fransens; L. Van Gool; C. Strecha. Combined depth and outlier estimation in multi-view stereo. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

[Wan] J.E. Davis; Ruigang Yang; Liang Wang. Brdf invariant stereo using light transport constancy. In *ICCV 2005. Tenth IEEE International Conference on Computer Vision*, volume Volume 1, 17-21 Oct. 2005 Page(s):436 - 443 Vol. 1. Digital Object Identifier 10.1109/ICCV.2005.51.

[Woo] John Amanatides; Andrew Woo. A fast voxel traversal algorithm for ray tracing. Dept. of Computer Science, University of Toronto, Ontario, Canada M5S 1A4 www.cse.yorku.ca/~amana/research/grid.pdf (visited: 17/12/2006).

Declaration

I, Martin Ralf Oswald, hereby declare that the diploma thesis "Concurrent Stereo Reconstruction" was written by myself without any but the resources stated therein.

Dresden, June 4, 2007

Martin R. Oswald