Multi-view Dynamic Scene Modeling

Li Guan

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2010

Approved by:

Marc Pollefeys, Advisor

Jean-Sébastien Franco, Co-principal Reader

Leonard McMillan, Reader

Jan-Michael Frahm, Reader

Svetlana Lazebnik, Reader

Abstract

Li Guan: Multi-view Dynamic Scene Modeling.
(Under the direction of Marc Pollefeys.)

Modeling dynamic scenes/events from multiple fixed-location vision sensors, such as video camcorders, infrared cameras, Time-of-Flight sensors etc, is of broad interest in computer vision, with many applications including 3D TV, virtual reality, medical surgery, marker-less motion capture, video games, and security surveillance. However, most of the existing multi-view systems are set up in strictly controlled indoor environments, with fixed lighting conditions and simple background views. Many complications limit the technology in the natural outdoor environments. These include varying sunlight, shadows, reflections, background motion and visual occlusion *etc.* In this thesis, I address different approaches overcoming all of the aforementioned difficulties, so as to reduce human preparation and manipulation, and to make a robust outdoor system as automatic as possible.

The main novel technical contributions of this thesis are as follows: a generic heterogeneous sensor fusion framework for robust 3D shape estimation; a way to automatically recover 3D shapes of static occluder from dynamic object silhouette cues, which explicitly models the static "visual occluding event" along the viewing rays; a system to model multiple dynamic objects shapes and track their identities simultaneously, which explicitly models the "inter-occluding event" between dynamic objects; and a scheme to recover an object's dense 3D motion flow over time, without assuming any prior knowledge of the underlying structure of the dynamic object being modeled, which helps to enforce temporal consistency of natural motions and initializes more advanced shape learning and motion analysis. A unified automatic calibration algorithm for the heterogeneous network of conventional cameras/camcorders and new Time-of-Flight sensors is also proposed.

To my parents

for letting me pursue my dream

for so long

so far away from home

&

To my wife

for always supporting me

and giving me

new dreams to pursue

Acknowledgments

I would like to thank my advisor, Marc Pollefeys, for supporting me over the years, and for giving me so much freedom to explore and discover new areas of dynamic scene modeling. My other committee members have also been very supportive. Jean-Sébastien Franco has long been an inspiration to me. His precise suggestions and thorough discussions with me are one of my best learning experiences at Chapel Hill. Leonard McMillan brings in-depth experience to my thesis, as a well-known researcher with great contributions in related areas throughout the years. Jan-Michael Frahm proves to be a most considerate mentor and friend who is always supportive with all projects that I participate. Svetlana Lazebnik always give me suggestions right to the point and show me the ray of hope through the darkness of desperations. Also special thanks are to Edmond Boyer who has always been inspiring and generously shared his brilliant ideas with me.

I would like to thank my many friends and colleagues at UNC-Chapel Hill and at ETH-Zürich with whom I have had the pleasure of working over the years. These include Sudipta Sinha, Seon Joo Kim, Christopher Zach, Changchang Wu, David Gallup, Brian Clipp, Megha Pandy, Roland Angst, Friedrich Fraundorfer, Gabriel Brostow and all the members of the Computer Vision Group at UNC-Chapel Hill and Computer Vision and Geometry Lab at ETH-Zürich.

I would like to thank my friends at UNC-Chapel Hill, Chen Wang, Liangjun Zhang, Yuanxin Liu, Xueyi Wang, Jun Huan, Feng Pan, Qi Zhang, Shunping Huang and many others, for giving me memorable experiences outside of my office during my PhD years.

Finally, I would like to thank my wife Xiaoxiao(Sharon) Liu for discussion with me about my problems in the office, for bringing joy and happiness to my life, and for giving me the courage and motivation to finish this thesis.

Table of Contents

List of Figures

List of Symbols

Chapter 1

Introduction

We live in a 3D world and directly perceive 3D information thanks to our pair of eyes. Inspired by this natural endowment, a fundamental problem of computer vision is to obtain 3D geometric information with the help of computers and digital sensing devices. This process is called "3D reconstruction". Due to the rapid advances as well as decreasing cost of computing and sensing hardware, the focus of 3D reconstruction has been gradually shifted from 3D static structure reconstruction to dynamic scene modeling. A "dynamic scene" is a scene containing one or more objects to be modeled, possibly with motion/deformation over time. The main focus of this thesis is the 3D reconstruction of dynamic scenes in a natural environment from multiple viewpoints. The design of robust and efficient algorithms are the major considerations. In addition, emerging new sensing technologies such as the Time-of-Flight cameras are also explored in collaboration with conventional video camcorders as a heterogeneous sensor network for the modeling task.

1.1 Motivation

There are numerous applications of 3D dynamic scene modeling in our multimedia-dominated modern world. In the movie industry, current motion capture systems have to attach trackable markers or sensors to the actors to recreate 3D structures and mo-

tions. They are often inconvenient to put on and uncomfortable to perform with. The ability to create 3D motions using images alone is therefore a very attractive and desirable alternative. In the game industry, new 3D controlling devices such as Nintendo Wii^{TM} LED sensor bar and Xbox Natal^{TM} 2.5D depth camera have already drawn a tremendous amount of attention and opened up possibilities of brand new gaming experiences. The demands for immersive 3D interactive games would be boosted by full 3D human pose real-time modeling. The technology also has applications in the medical field. It can be used to create models of deforming organs, as well as to replay disease development over time. Other application areas include sports broadcasting and commentary, teleconferencing, robot navigation, object recognition, visual surveillance, digital historic archiving *etc.*

The most common setup for a dynamic scene reconstruction consists of multiple cameras mounted on different fixed locations, such as tripods, looking at a common viewing region, as shown in Fig. 1.1. This is because a dynamic scene is theoretically a 4D spatiotemporal continuum, which cannot be captured by a single camera with only 3D sampling ability (2D image frames over time).

Figure 1.1: [1] Multi-camera network setup. Outdoor data capturing with 9 video camcorders behind the Ackland Museum, UNC-Chapel Hill, 2006/8/24.

[1]Fig. 1.1 courtesy of Jae Hak Kim.

Over the past decades, people have proposed effective ways to recover 3D dynamic shapes in indoor laboratory environment with constant lighting, uniform-colored static background, empty viewing space without any visual occluders, etc. However, one cannot extend such systems directly to an uncontrolled natural environment. A natural scene is far more complicated with environmental variations, including sunlight changes, soft shadows casted by clouds, dark shadows cast by trees or dynamic shapes themselves, visual occluders that are common in outdoor scenes, varying background possibly with fluttering tree leaves or passing-by people at distance, reflections on glassy or metallic surfaces, *etc.*

This thesis contributes algorithms to address different aspects of the uncontrolled 3D dynamic scene modeling. Although difficult, the challenges have to be conquered, as it is believed that in order to realize the previously mentioned real-world applications, the system has to work outside of the laboratory ultimately.

1.2 Thesis Statement

3D dynamic scenes in an uncontrolled natural environment can be robustly and efficiently reconstructed with a multi-view sensor network using a probabilistic occupancy model and Bayesian sensor fusion framework, upon which visual occlusion can be explicitly modeled, static occluders can be automatically recovered, individual shapes can be distinctively estimated and tracked, and effective spatiotemporal analysis can be conducted to compute 3D dense motion field and refine the recovered shapes. Moreover, heterogeneous sensors can be easily integrated in the mathematical computation framework.

1.3 Contribution

My thesis makes the following contributions:

1. 3D static occluder automatic recovery.

An algorithm is introduced that robustly estimates the shape of static occluders given only silhouette cues of dynamic shapes. The algorithm is built upon the existing probabilistic occupancy volume algorithm for dynamic shape estimation. In addition to the dynamic shape occupancy probability grid, a second probability grid of the static occluders is introduced and is used to explicitly model the visual occlusion event. With dynamic shapes' arbitrary activity in the scene, the shape of the occluder is automatically accumulated over time.

2. Simultaneous multiple shape estimation and tracking.

An algorithm is presented to reconstruct and track multiple dynamic shapes, such as a group of people. An appearance model of each individual shape is automatically learned when it first enters the scene. Similar to the static occluder recovery, the inter-occlusion event between the dynamic shapes is explicitly modeled. At every time instant, an object's location is also tracked and updated.

3. Dense 3D motion field recovery.

I have developed an algorithm to recover the dense 3D motion field of arbitrary dynamic shapes between two consecutive time instances. Comparing with the previous tracking schemes, this is a more sophisticated way to enforce the temporal consistency. The motion field is also consistent with the probabilistic framework for robustness in the uncontrolled scenes and is computed by maximizing the posterior probability of the occupancy given the image observations from the views and the occupancy estimation from the previous time instance. The recovered dense 3D motion field can be used to refine the estimated shape, as well as generalize the underlying arbitrary rigid skeletal structure of the dynamic shape. Since no prior knowledge of the dynamic shape is required, this algorithm can be used to automatically initialize the underlying skeleton structure, which is used for 3D shape tracking and fitting.

4. Heterogeneous sensor network of video camcorders and Time-of-Flight sensors for dynamic scene reconstruction.

Methods are presented that, as long as the probabilistic sensor model can be defined for shape reconstruction purpose, a sensor fusion framework can seamlessly combine the heterogeneous sensor observations together, which is robust against noise and many challenging scene variations. An example of a network of off-the-shelf video camcorders and technologically-new-but-highly-potential Time-of-Flight(ToF) sensors are tested as an example of the theory. ToF cameras are a relatively new technology that have become widely available in the last decade. One of their features is the ability to directly acquire 2.5D depth images at video frame rates. An automatic scheme to calibrate the extrinsics of such heterogeneous network is also provided, despite the low imaging resolution drawback of the current ToF sensor technologies to the traditional checkerboard or laser-pointer calibration schemes.

1.4 Overview

The remainder of the dissertation is organized as follows.

Chapter 2 presents the background and fundamental technologies required for multi-view reconstruction and an overview of the existing and related terminologies. First, camera network calibration is discussed. Second, two types of reconstruction algorithms, the silhouette-based method and the multi-view stereo are described and compared. Finally, the mathematical foundation for the occupancy estimation, the Bayesian sensor fusion model is described in detail.

Chapter 3 describes a novel method for automatic and robust 3D static occluder reconstruction with only dynamic shapes moving within the scene over time. Applications include automatic 3D scene discovery and automatic camera-view selection.

Chapter 4 introduces a complete framework for simultaneous reconstruction and

tracking of multiple dynamic shapes. Appearance models for each individual shape is automatically learned when it first appears in the scene, and static occluder recovery is also smoothly integrated in the framework.

Chapter 5 presents a new algorithm to compute a dense 3D motion field given the probabilistic occupancy volumes computed at two consecutive time instances. The motion field is computed as a posterior probability maximization problem, which is compatible with the introduced probabilistic sensor fusion framework. The recovered motion field can be used to refine the occupancy estimation and to generalize the underlying skeletal structure of the dynamic shape.

Chapter 6 describes the sensor model of a Time-of-Flight(ToF) camera and how it can be integrated in the Bayesian sensor fusion framework for dynamic scene reconstruction. As a matter of fact, a ToF camera is only one of the many examples of possible vision sensors. Others include infrared cameras, laser scanners, stereo cameras etc. It is in general tempting to exploit different types of sensors, especially new technologies, because one sensor could compensate for another's drawbacks. However, due to the heterogeneity of the sensor output format, the fusion of different types of sensor data is not straightforward. This chapter takes ToF sensor as an example to show the proposed mathematical framework is suitable for not only camcorder network but also such heterogeneous sensor networks. An automatic calibration method is also introduced in this chapter for such heterogeneous sensor network.

Chapter 7 summarizes the contributions, concludes the thesis and discusses potential improvements and directions for future work.

Chapter 2

Background

2.1 Multiple View Calibration

For multi-view 3D dynamic scene reconstruction, the most common set up is to have the cameras looking inward at a common viewing region, where a performance happens. The exact configuration parameters for each camera needs to be known in advance. These parameters include intrinsics, such as focal length and aspect ratio, and extrinsics, such as the camera location and orientation in the world coordinates. This pre-process to the reconstruction is the "camera network geometric calibration". The calibration requires classical tools and concepts of multi-view geometry, including image formation models, epipolar geometry, and projective transformations. In addition, feature detection and matching, as well as estimation algorithms, are required. For all the datasets acquired in this thesis, I use a planar checkerboard pattern as the calibration target. The board is painted with 6×9 $120mm$ by $120mm$ alternating black and white squares, as shown in Fig. 2.1.

One thing to notice is that in Fig. 1.1, instead of digital cameras, camcorders are used. Camcorders output videos (a sequence of frames), and therefore are able to record the dynamic event from a specific point of view. Consumer level camcorders nowadays are able to produce high quality video frames at real-time frame rates with adjustable

Figure 2.1: Camera network calibration session of the SCULPTURE outdoor datasets. This is one pose of the checkerboard calibration target. Four of the eight synchronized camera views are shown.

lenses, shutter speed, aperture, gain, white balance *etc.*, which are nice features to have when shooting in an outdoor environment. They also have more affordable prices.

Although videos are captured from multiple views, the 3D reconstruction is performed with one frame from each view at a time instance, as if one is reconstructing different static objects one at a time. Therefore, the different videos must be synchronized so that one is using the correct set of images for a specific "static object". A clapping of hand before and after the calibration session is used as a visual and acoustic event for synchronization of these camcorders which are set to run at the same frame rate. And then approximately 25 to 30 synchronized frames of different checkerboard poses are taken from all the views. Without confusion, both the cameras taking multiple pictures and the camcorders are called "cameras" in the rest of the thesis.

Camera calibration has been widely studied in [Zhang (2007); Svoboda et al. (2005)],

and the details are not presented here. For the datasets captured in this thesis, I used Jean-Yves Bouguet's checkerboard calibration method[1] based on [Zhang (2007)], followed with a bundle adjustment to recover the cameras' poses in the world coordinate system.

The camera intrinsics (focal length, skew, optical center, radial distortion factors) and the extrinsics (translation and orientation) with respect to the checkerboard pose of the first frame can be recovered. Since this checkerboard pose is the same across the synchronized camera views, it can be treated as the basis of the world coordinate system and the camera network extrinsics in that frame are thus recovered. Since cameras facing opposite to one another would be impossible to see the same checkerboard pose, in practice, three to four cameras with small enough viewing angle difference are calibrated as one group. Multiple calibration groups are carried out to recover the complete camera network, as long as there are overlapping cameras in the groups that can link all groups together. Followed with a final global bundle adjustment of the complete camera network parameters.

A more convenient approach that can calibrate the complete set of cameras all at once is introduced in [Svoboda et al. (2005)], which uses a laser pointer or a diffuse light bulb as a calibration target and moves it around the reconstruction space. The method works well with synchronized cameras indoors. However tracking a laser dot or a light bulb in outdoor sunlight is not very robust. In Chapter 6 an extension of this method using a big spherical calibration target is introduced to calibrate heterogeneous network of camcorders and ToF cameras, which has very low image resolution and only respond to light that is emitted from the sensor itself.

Recently, visual content, such as observations of a dynamic subject's silhouette, have been used to calibrate large-scale camera networks [Sinha et al. (2004)]. This method only requires a dynamic shape, such as a person, moving in the common scene performing

[1] http://www.vision.caltech.edu/bouguetj/calib_doc/

arbitrary actions. An extension of the method does not require that the cameras are synchronized [Sinha and Pollefeys (2004)]. However, a major constraint for the method is the relatively high quality silhouette segmentation in every view, which is hard to guarantee in natural environment as discussed later on in this chapter.

2.2 Dynamic Scene Reconstruction

Given the calibrated camera network, existing methods use various image cues to deduce and construct geometrical object models. The two main cues used frequently are the silhouettes of the object of interest and color consistency information. The former is used to form a 3D approximation shape of the original object, the visual hull. The latter is used to pinpoint and triangulate a surface coherent with the conjunction of observed colors in images. Additional geometrical constraints such as surface smoothness are often used to help deduce the information or fill in the gaps where image data is inconclusive.

All existing methods are generally successful in controlled environments, where the lighting is constrained and the viewing conditions used to obtain images of objects are optimal. They, however, face substantial difficulties when brought outdoors or in generally unconstrained environments.

2.2.1 Silhouette-Based Methods

A common approach to multi-view reconstruction uses the silhouettes of objects as sources of shape information. A 2D silhouette is the set of close contours that outline the projection of the object onto the image plane, as well as the regions inside the contours. A common representation is a binary mask image with black being background and white being foreground, as shown on the left of Fig. 2.2.

The silhouette provides a strong cue for shape understanding. The back-projection of the silhouettes from the camera optical center form generalized "viewing cone"s, as

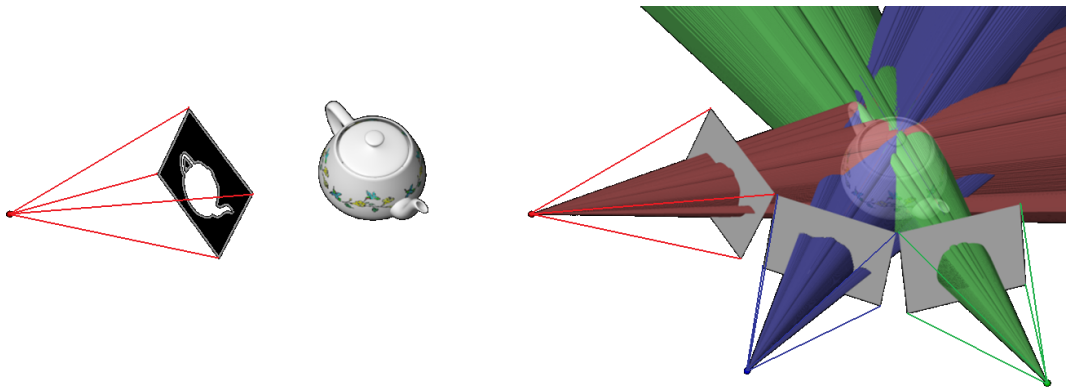

Figure 2.2: [1] Foreground silhouette and visual hull formation. Left: silhouette of the teapot from one view (colored white); Right: back-projection of silhouettes from three views (viewing cones) to intersect the visual hull of the teapot.

shown on the right of Fig. 2.2. The intersection of the viewing cones yields a reasonable approximation of the real object. A "visual hull" is named by [Laurentini (1994)] to describe such intersected volume with infinite number of viewpoints surrounding the actual object.

A visual hull has a few special properties. First of all, it is the maximal object that is consistent with all the silhouettes from the given viewpoints. This property is sometimes called "conservativeness" of the visual hull, because the real shape, which is also consistent with all the silhouettes, is guaranteed to be contained in the visual hull. Secondly, every viewing cone can exclusively eliminate volumes outside of the cone.

However, no matter how many cameras are used, surface concavities are not recovered, due to self-occlusion. Notice the difference between "concavity" and "non-convexity": a visual hull is able to recover a "saddle region", which is non-convex, as shown in the teeth dents in Fig. 2.3, while the concave orbital area is never recovered.

One thing to point out is that although the original visual hull concept is based on an infinite number of views, the above properties still hold for the shape recovered from finite views. In fact, [Baumgart (1974)] originally introduced the visual hull idea with

[1] Fig. 2.2 adapted from [Matusik et al. (2000)].

Figure 2.3: [1] Left: the same skull model as in Fig. 2.5; Right: its visual hull from 24 images. The non-convex teeth dents are recovered, but not the concavities around the orbital and nasal area.

finite views in his PhD thesis. In the rest of this thesis, we only focus on the visual hull reconstruction algorithms with finite number of views. But they are valid with infinite views in theory.

Shape from silhouettes is a particularly good approach if only an approximate model of the real world is required. The methodology is intuitive and easy to implement. With the advances in computing powers, systems generating real-time 3D digital video for dynamic reconstruction in studio-controlled environment are already on the market, such as [$Stage^{TM}$ (2007), $4D\ View^{TM}$(2008)]. Nevertheless, such systems are restricted to a single solid shape within an indoor environment with strictly controlled conditions.

All such systems demand on the delicate process of silhouette extraction. If any part of a single silhouette were corrupted, due to the exclusiveness of the viewing cone carving, the corrupted parts would result in incomplete visual hull (which contradicts the visual hull "conservativeness" property) and would never be recovered even if the silhouettes from all other views are correctly computed. Unfortunately, so far, there is no automatic solution that produces silhouettes with the quality as good as manual segmentation. The technical detail about silhouette extraction is introduced later in

[1] Fig. 2.3 adapted from [Lazebnik et al. (2007)].

Section 2.3.

2.2.2 Multi-view Stereo

Multi-view stereo algorithms recover 3D surface point locations by triangulating corresponding visual features seen from different viewing angles, as shown in Fig. 2.4. Once these sparse critical points are recovered, different kinds of smoothness constraints can be applied to help recover the whole object surface. A common way to draw the feature correspondences between views is to use the photo-consistency measures [Kutulakos and Seitz (2000)]. Simply put, the resemblance between the pixels/patches in one image to those in the other image is evaluated to see how well the two correlate. A thorough survey is given in [Seitz et al. (2006)].

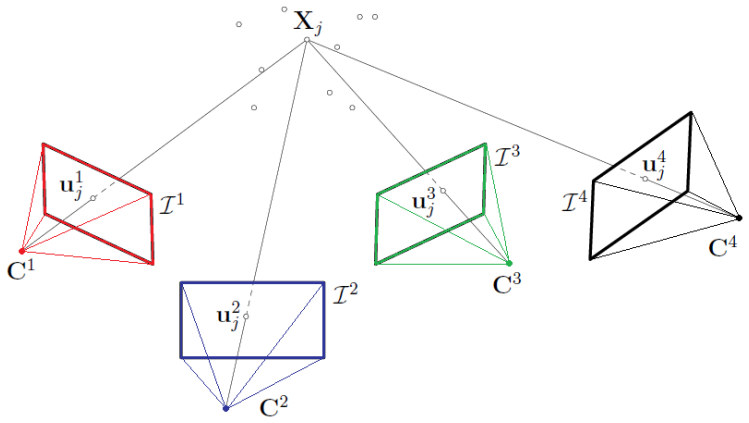

Figure 2.4: [1] Triangulation of 3D point $\mathbf{X}_j$ from four camera views. $\mathbf{u}_j^i$ is the 2D projection on image $\mathcal{I}^i$ with respect to camera $\mathbf{C}^i$, where the camera index $i \in \{1, 2, 3, 4\}$.

Unlike the visual hulls, concave regions can be recovered as long as feature correspondences in those regions can be triangulated from different camera views, Fig. 2.5 shows a few reconstruction examples. Another major advantage of multi-view stereo is that the recovered surfaces are the exact shapes if triangulation is established for every point on the surface. Therefore, even two views may produce some accurate surface

[1] The figure is modified from [Svoboda et al. (2005)].

fragments, while a two-view visual hull is usually not satisfactory.

However, multi-view stereo algorithms generally only work well on highly textured surfaces, where salient feature points are easy to find. In dynamic scene reconstruction applications, especially human modeling, uniformly colored clothes are very common, thus, it is difficult to find salient features on the person. Fig. 2.6 shows one of the most commonly used scale-invariant features in vision community, the SIFT feature detections [Lowe (2004)], in two sample outdoor frames. Most recovered SIFT features are either on the background structures or at the boundary of the silhouette. The latter unfortunately consists of partial foreground and background pixel information, and thus does not correspond to consistent features from a different view. Therefore, most of the recovered feature points cannot be used to perform 3D triangulation. In situations without many robust features, modern stereo methods, such as [Vogiatzis et al. (2007); Sinha et al. (2007)], have to "reduce" to silhouette constraints with varieties of smoothness regularization.

The second concern is the computation complexity. Unlike the straightforward visual hull algorithms which can reach real-time performance, multi-view stereo algorithms usually involves much slower optimization processes to get the detailed surfaces. From [Middlebury (2009)] dataset evaluation, the fastest GPU accelerated algorithms take tens of seconds to output a final shape. It is not generally an issue for static scene modeling, where the reconstruction quality is the main concern. However, for the dynamic scene modeling, these methods are not feasible for real-time applications.

The third concern is the self-occlusion problem. Since the 3D point triangulation requires at least two views of the same surface point from different perspectives, in order to obtain as many corresponding features as possible, neighboring cameras should not be too far apart. In practice, most of the existing successful multi-view stereo techniques exploit tens or hundreds of views of a single static object, such as in [Pollefeys et al. (2004); Seitz et al. (2006)]. This mainly works for static scenes where a camera can be

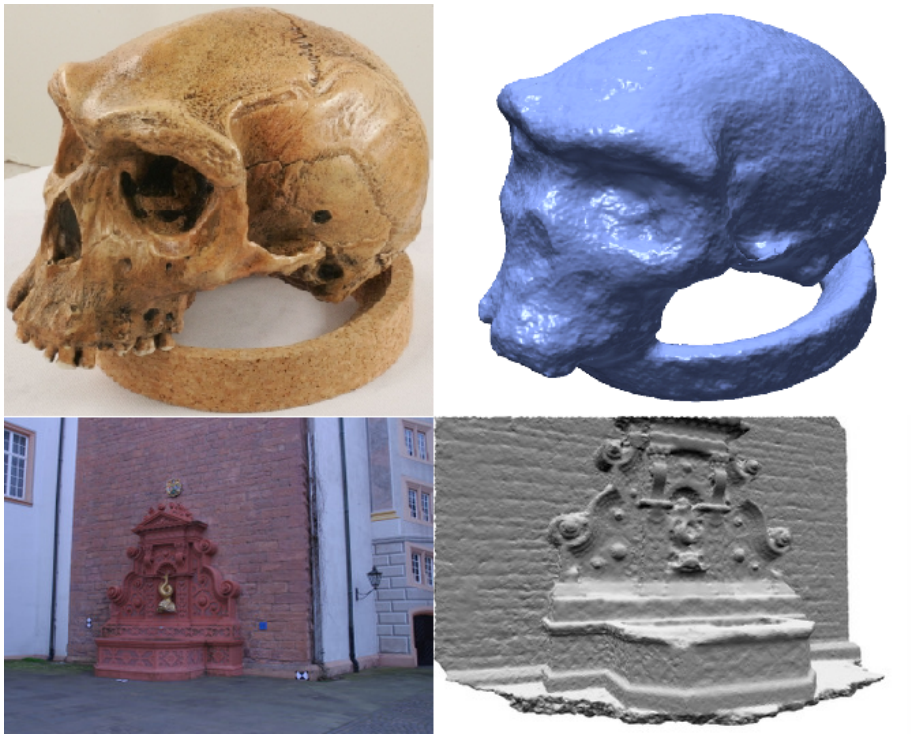

Figure 2.5: [1] The multi-view stereo reconstruction examples. Above: skull dataset(closed surface); Below: fountain dataset(open surface); Left: one of the original images; Right: recovered surfaces. Notice the bottom row, the shape does not require to be closed when using multi-view stereo.

moved around, but is not practical for dynamic scene modeling where different views need to be obtained at the same time. Multiple cameras are thus required to model the latter case.

An additional concern is the color consistency constraint. Besides pixel intensity, pixel color consistency is often used as a multi-view stereo correspondence measure, such as in [Kutulakos and Seitz (2000); Bonet and Viola (1999)]. The multiple color channels contain more information than the pixel intensity and thus produce more accurate 3D point correspondence. However, besides many heuristics discussed in [Larsen (2006)], it requires the cameras from all views should be photometrically calibrated. Although many advance algorithms have been proposed for camera network photometric

[1]The skull and fountain results are from [Sinha et al. (2007)] and [Strecha et al. (2004)] respectively.

calibration as in [Kim (2008)], it is in general a tedious manual task [Ilie and Welch (2005)]. The color inconsistency issue has been further investigated in Fig. 4.4 in Chapter 4, where one can see how different the views of the same object are in a scene without photometric calibration. Such significant difference normally makes multi-view stereo algorithms fail.

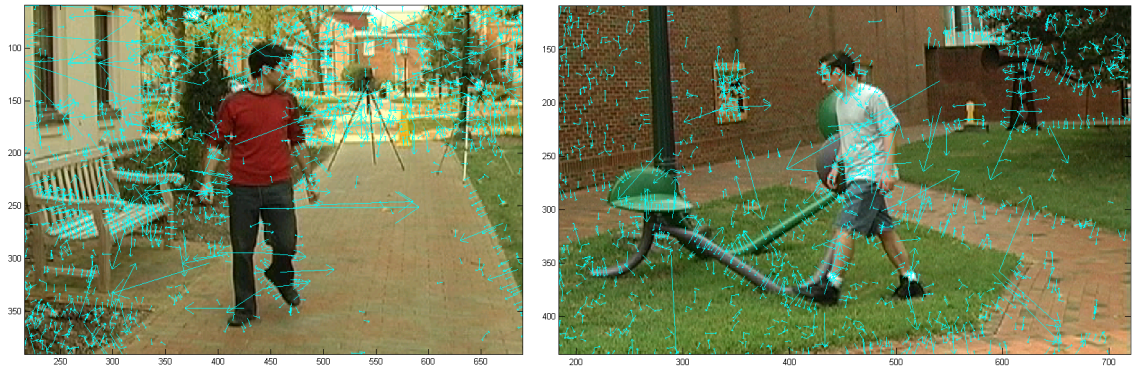

Figure 2.6: The SIFT features extracted for two outdoor frames. Most of the recovered salient features are on the silhouette boundary of the foreground shape or in the background.

2.2.3 Comparison and My Choice

From the previous sections, one can see that both silhouette based and multi-view stereo algorithms have beneficial properties but challenges remain in multi-view dynamic scene modeling, especially in uncontrolled outdoor environments. Comparatively, when modeling dynamic shapes, such as humans, silhouette based methods do give closed shapes, no matter how weak the observation information is. This is not the case for multi-view stereo methods. In fact, as mentioned earlier, some multi-view stereo algorithms rely on a silhouette based visual hull as a coarse approximation of the shape and build a more detailed shape upon that. In terms of computation complexity, given the hardware currently available on the market, silhouette based methods are more practical than multi-view stereo for real-time modeling.

Given the above concerns, the main focus of this thesis is therefore on the challenges of silhouette based methods, especially on how to robustly use silhouette information from multiple views, how to deal with the visual occlusions, how to distinguish multiple dynamic shapes in cluttered environment, and how to effectively make use of temporal consistency given multi-view videos over time.

In the rest of this chapter, I will give an overview of basic automatic silhouette extraction techniques, describing why silhouette extraction is a difficult task in natural scenes. I will also survey silhouette based reconstruction algorithms, including various visual hull representations that researchers have proposed over the years. Among those representations, the probabilistic occupancy grid is the major building block of the algorithms introduced in this thesis.

2.3 Silhouette Extraction

There are mainly two types of algorithms. The first uses appearance models such as the active contour shape [Kass et al. (1987)] to compute the silhouette boundary, and track it between frames. These algorithms do not require the cameras be static. But the computation involves energy minimization. Depending on the choice of the appearance measurement and energy design, the tracking result may be slow and may not be the exact desired solution. Overtime, the tracked silhouettes may drift away from the correct shape boundary, especially in some noisy image sequences.

Another commonly used automatic silhouette extraction algorithm is "background subtraction". It is the basis of the silhouette extraction method used in this thesis. The name comes from the simple technique of subtracting $\mathcal{I}$, an observed image of the shape, from $\mathcal{B}$, an image of the empty scene and thresholding the result to generate the binary silhouette of the shape, as described in [Heikkila and Silven (1999)]. Since an empty background image is required in advance, this algorithm is suitable for fixed camera

views. The algorithm itself usually only involves local pixel color examination, which is much easier than active contour energy minimization and involves intensity evaluation of the whole image. The basic background subtraction algorithm [Heikkila and Silven (1999)] as well as the probabilistic modification are introduced as follows.

A pixel p is labeled as the foreground if

$$\mathcal{I}_p - \mathcal{B}_p > \tau, \tag{2.1}$$

where τ is a predefined threshold. Then the binary image is evaluated with a 3×3 kernel to discard small regions.

This simple subtraction inequality only works with an ideal static background. When sensor noise is taken into account, the pixel color observation of the scene is not a constant, but follows a probabilistic sensor model with a certain noise distribution. The most common noise model is an n-dimensional Gaussian model, where n is the number of color channels of the pixel readout. Suppose the camera output is an RGB color image, for every pixel, the background color model can be written as

$$\mathcal{B}_p \sim \mathcal{N}(\mu_{RGB}, \mathbf{\Sigma}_{RGB}), \tag{2.2}$$

where μ_{RGB} and $\mathbf{\Sigma}_{RGB}$ are respectively the mean vector and covariance matrix of the RGB channels at p, and $\mathcal{N}$ denotes the normal distribution. $\mathcal{B}_p$ explains the probability of a given pixel color to be background, namely $p(\mathcal{I}_p|\mathcal{B}_p)$. The probability distribution can be learned with a few training images in advance. As for the foreground appearance model $\mathcal{F}_p$, namely $p(\mathcal{I}_p|\mathcal{F}_p)$, since the dynamic object can take an arbitrary color, without any prior knowledge, one can assume it to be a uniform distribution. Suppose each RGB channel has been discretized to 256 intensity levels, then theoretically $p(\mathcal{I}_p|\mathcal{F}_p) = 1/256^3$. However, in practice, some colors never show up due to the composition of the actual scene, color space approximation or even the sensing range of the

camera. This implies that the colors actually being observed should have greater chance than $1/256^3$. Therefore, usually a constant value c larger than $1/256^3$ is used for the uniform distribution. Formally put,

$$\mathcal{F}_p = c > 1/256^3. \tag{2.3}$$

A pixel p is labeled as foreground if

$$p(\mathcal{F}_p|\mathcal{I}_p) > \tau. \tag{2.4}$$

Using Bayes rule, one can easily rewrite the left side of Eq. 2.4 as follows.

$$p(\mathcal{F}_p|\mathcal{I}_p) = \frac{p(\mathcal{I}_p|\mathcal{F}_p)p(\mathcal{F}_p)}{p(\mathcal{I}_p)} = \frac{p(\mathcal{I}_p|\mathcal{F}_p)p(\mathcal{F}_p)}{p(\mathcal{I}_p|\mathcal{B}_p)p(\mathcal{B}_p) + p(\mathcal{I}_p|\mathcal{F}_p)p(\mathcal{F}_p)}, \tag{2.5}$$

where $p(\mathcal{B}_p)$ and $p(\mathcal{F}_p)$ are the prior probabilities of an image pixel being labeled as the background and foreground respectively.

Notice the background model described here is a per-pixel model without assuming any spatial consistency between the neighboring pixels. Thus the algorithm is parallelizable, and can gain tremendous speedup when ported to a GPU. This is yet another reason to choose the background subtraction algorithm for silhouette extraction. Details about the GPU acceleration are discussed in Chapter 3. Although one could argue for the use of more advanced mathematical tools such as Markov Random Field (MRF) to model the spatial coherences and eliminate a few falsely classified local pixels, *e.g.* [Ahn et al. (2006)], the general per-pixel model described above is the core for most background subtraction algorithms, and is good enough for many applications. The sub-index p in $\mathcal{I}_p$, $\mathcal{B}_p$ and $\mathcal{F}_p$ are omitted for simplicity afterwards, when not otherwise specified.

So far, background subtraction is based on a static background hypothesis which is

often not the case in real environments. With indoor scenes, reflections or soft shadows lead to background changes. Similarly, the static background assumption has difficulties with outdoor scenes, due to wind, clouds, hard shadows, or background motion (motion in the background that is not of the interest, such as tree leaves flickering, people walking at a distance *etc.*). An example of an outdoor result using Eq. 2.4 is shown in Fig. 2.7, where the shadows and tree leaves are mislabeled as part of the foreground silhouette.

Figure 2.7: The silhouette from background subtraction. Left: an outdoor scene with a person; Middle: the trained mean background; Right: the silhouette(white) by theshholding the per-pixel background probability computed from Eq. 2.4. The shadow is labeled as a part of the silhouette.

The background variation problem can be tackled by background updating algorithms [Toyama et al. (1999)] and with Gaussian Mixed Models (GMM) instead of the naïve normal distribution [Stauffer and Grimson (1999)]. However, complex background models are computationally more expensive. Additionally, the online update of the background model often gets confused with very slow foreground motion. For example, a seated person would gradually fade into the background, which is not desirable for silhouette extraction. An alternative is to find robust features. For example, robust color features alleviate the problem of illumination variations, *e.g.* [Finlayson et al. (1996)], but the trade-off is that a more robust color coordinate often means less discriminative power.

In addition to all of the challenges mentioned above, for silhouette extraction, a very critical problem with background subtraction is the existence of static occluders in

many scenes. An "occluder" in this thesis context is a potential visual obstacle that, from a certain viewing direction, *may* be visually occluding the subject to be modeled. For example, the metallic sculpture in Fig. 1.1. Since the sculpture is not movable in this case, the background model trained in advance would have the sculpture as part of the background. Consequently, when a person goes behind the sculpture, the occluded pixels would still take the color of the sculpture, which is consistent with the trained background model. Such phenomenon together with shadows and reflections are illustrated in Fig. 2.8. As discussed before, such an incomplete silhouette is disastrous for visual hull construction, because every missing bit in any silhouette view will carve away some part belonging to the real shape.

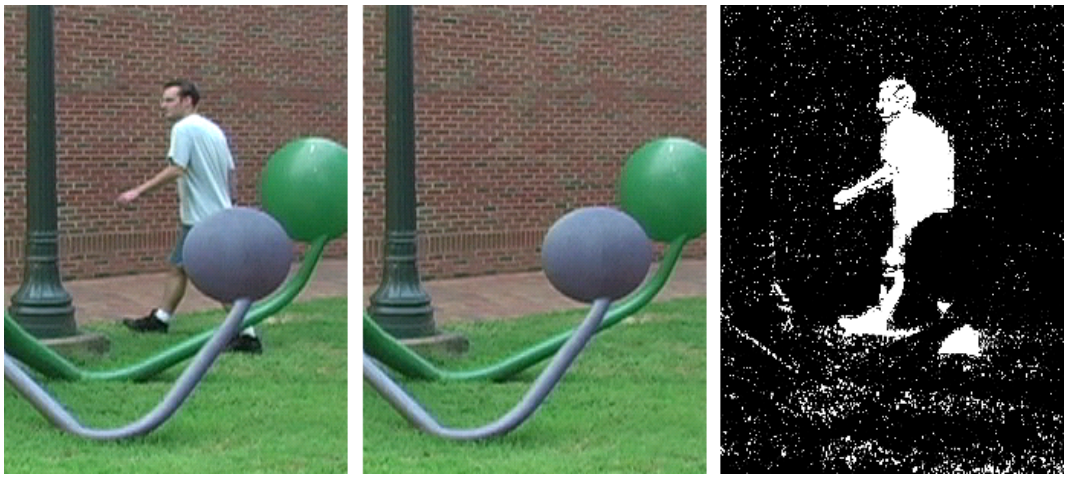

Figure 2.8: Background subtraction results of incomplete silhouettes outdoors. Left: a person behind the metallic sculpture; Middle: the trained mean background; Right: the foreground silhouette probability without thesholding with black and white denoting 0.0 and 1.0 respectively.

Two related problems are also worth noting. First, if a person goes out of a scene (whether partially or entirely) from one camera view, the visual hull would not be correctly constructed. In this case, the camera field-of-view(FOV) can be taken as if it casts a big occluder outside the view. This means the dynamic shape's moving space is constrained by the intersection of the camera network's FOVs. Hence comes the

dilemma: the more cameras used, the better visual hull approximates the real shape, but at the same time, the smaller freedom of motion for the subject. This limits the potential of the multi-view systems in practice. People have come up with algorithms such as camera view selection, which only uses the views that contain the whole subject shape for visual hull construction. But the criterion to determine silhouette completeness is non-trivial. An algorithm that does not require such camera view selection is discussed in Section 2.4.2 and is extended to solve the static occluder problem in Chapter 3.

Second, when the foreground object color happens to be similar to the background color of certain pixels, Eq. 2.4 & 2.5 would classify the pixel as background. In Fig. 2.7, the missing hair at the top of the person is because of this. Given a natural scene, this problem tends to happen only for individual pixels and in individual time instants, but not consistently happen on a specific foreground region (*e.g.* the hair) when the subject moves between locations. Therefore, this problem can be largely overcome by considering spatially and temporally neighboring information. It could also be alleviated by a more specific foreground object appearance model rather than just a uniform foreground model as in Eq. 2.3. The ideas mentioned above have been used in algorithms in Chapter 4 and 5.

2.4 Visual Hull Reconstruction

Visual hull algorithms have a long development history. The original focus was the geometric properties of the reconstruction. Therefore, the majority of the early algorithms are deterministic, *e.g.* they start from perfect binary silhouettes obtained in controlled laboratory scenes or from manual segmentation. Recently, attention has been drawn to the lack of robustness of the algorithms due to the silhouette extraction noise sensitivity. The probabilistic sensor fusion framework is gradually gaining popularity, because it tries to postpone binary object boundary determination to the 3D reconstruction stage.

2.4.1 Deterministic Approaches

Except some hybrid approaches, such as [Boyer and Franco (2003)], most of the deterministic approaches fall into two categories: (1) surface approaches that focus on a surface representation of the visual hull and (2) volumetric approaches that focus on the volume of the visual hull and usually rely on a discretization of 3D space. A third category exists that computes a view dependent visual hull image from an arbitrary viewpoint [Matusik et al. (2000)]. This method does not require recovery of the explicit 3D models, which although is useful in many applications, is not the major concern of this thesis. All deterministic approaches suffer from corrupted silhouette computation.

Surface Representation

The final output of this category of algorithms is the visual hull surface, which is created by analyzing the silhouette boundaries in the images. [Baumgart (1974)] proposed the earliest approach to compute a polyhedral representation of objects from silhouette contours, approximated by polygons. A number of approaches assume the local smoothness of the reconstructed surface [Koenderink (1984); Giblin and Weiss (1987); Cipolla and Blake (1992); Vaillant and Faugeras (1992); Boyer and Berger (1997)], and compute the rim points based on a second-order approximation of the surface, from epipolar correspondences.

More recent methods [Cross and Zisserman (1998); Kang et al. (2001); Brand et al. (2004); Liang and K.Wong (2005)] exploit the duality that exists between points and planes in 3D space, and estimate the dual of the surface tangent planes as defined by silhouette contour points. More recent approaches [Matusik et al. (2001); Shlyakhter et al. (2001); Lazebnik et al. (2007); Franco and Boyer (2009)] extend the image rendering and multi-view geometry concepts to produce view-independent polyhedral models. The most important contribution of these newer algorithms is the reduction of 3D polyhedral intersections to 2D computation. But similar to previous approaches, singularities due

to calibration errors or local surface topology near the frontier points have to be dealt with explicitly and carefully.

Volume Representation

A 3D solid volume is the final output of this category of algorithms, which is usually in the form of discretized unit cells—the voxels. Unlike surface algorithms, the computation is consistent with the image formation procedure. Every voxel is projected from its 3D location to the silhouette images of every camera view. Only the voxels whose projections fall into the silhouette regions in all camera views are considered visual hull voxels and kept. Others are carved away. After all voxels are evaluated, a discretized visual hull approximation is computed.

Various schemes have been proposed to discretize the 3D space, ranging from the basic fixed grid representations with orthogonal axis-aligned voxels [Martin and Aggarwal (1983)], to adaptive or hierarchical decompositions of the scene volume [Chien and Aggarwal (1986); Potmesil (1987); Srivastava (1990); Szeliski (1993)]. Some applications have an extra step to extract the object surface from the computed volume, using for example the Marching Cube algorithm [W. Lorensen (1987)].

The volume based algorithms have inherent disadvantages. Even with very fine discretization of the scene, they may still have aliasing artifacts depending on the camera orientation and volume axes directions. But because these algorithms do not have the numerical difficulties in the surface based algorithms as discussed in the previous section, they are very popular as an initialization for other 3D shape refinement algorithms, such as multi-view stereo. Also because they are simple to implement and easy to parallelize on a GPU, they are often the backbone of many real-time, lab environment dynamic scene modeling systems.

2.4.2 Probabilistic Sensor Fusion

Since deterministic visual hull approaches suffer from unstable silhouette images, a more robust way of 3D shape estimation is needed for noisy environments such as outdoor scenes. Ideas to compute the visibility probability have been used in shape from photo-consistency and multi-view stereo by [Bonet and Viola (1999); Broadhurst et al. (2001); Yao and Calway (2003)]. A 3D spatial probabilistic occupancy grid concept, borrowed from the robotics community to detect obstacles for robot navigation, is introduced by [Franco and Boyer (2005)] into the shape from silhouette problems. Similar to volume based visual hull approaches, the 3D space is discretized into a regular grid. Instead of a hard decision from binary silhouette images, the algorithm computes the probability of a voxel's occupancy by fusing the silhouette information from all camera views.

In terms of probability theory, their goal is to compute the posterior occupancy probability for every voxel given the observed silhouette information in all camera views using Bayes' rule. The silhouette information is modeled as a hidden random variable, which is marginalized in the final formulation. This means no hard threshold is needed to get a binary silhouette before the visual hull is computed, which is one of the main source of instability in deterministic visual hull algorithms. A threshold step can be conducted after the visual hull probability volume is computed, if a deterministic decision is required. But compared to deterministic visual hull algorithms, such a hard decision is postponed from the beginning (the binary silhouette extraction phase) to the very last moment (the 3D surface extraction phase), thus maintaining the maximum robustness. Since this probabilistic framework is the foundation of the algorithms introduced in the later chapters, the variable notations and detailed fusion formulation are given as follows.

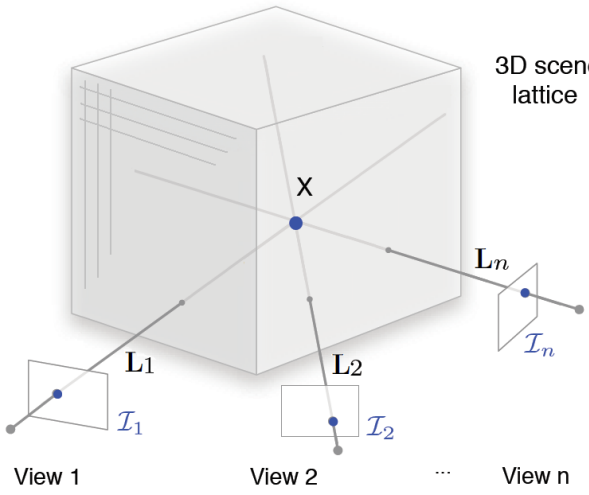

Figure 2.9: Voxel occupancy probability inference overview. Every voxel X is computed from the observations of all camera views using Bayes' rule.

Problem Formulation

Consider a single time instant for now. With an orthogonal axis-aligned equal-sized discretization of the 3D space $\mathcal{X}$, for every 3D location X in $\mathcal{X}$, its probability of being occupied by the dynamic object is computed, given a set of image observations $\mathcal{I}$ from n geometrically calibrated camera views. This occupancy probability is denoted as $p(\mathcal{G}_X)$ with $\mathcal{G}_X$ the binary variable at X. The setup is shown in Fig. 2.9, where $\mathbf{L}_i$, $i \in \{1, ..., n\}$, denotes the n viewing lines going through the camera centers and X.

An intuitive assumption is that different views can be independently rendered without the knowledge of other views. The background model for one view can be independently trained.

A second assumption is that the space occupancy variable $\mathcal{G}_X \in \{0, 1\}$ depends only on the information along the optic rays that go through X, which may include not just the single pixel that the voxel is projected onto, but a 2D neighborhood of pixels around the voxel's projection.

Yet another common occupancy grid assumption is that the voxel probability can be independently inferred just from image observations, no 3D neighboring voxel status

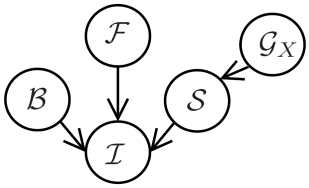

Figure 2.10: Occupancy probability inference dependency graph. An arrow points from a source node to a destination node, indicating the destination is caused by or depended on the source node.

is needed. This assumption allows the tractability of the final probability computation. This assumption is reasonable also because it has been successful in many deterministic volumetric visual hull algorithms, where every voxel's status is evaluated individually against its projections onto the image pixels. Results show that the independent computation, while not as exhaustive as a global search over all voxel configurations, still provides very robust and usable shape estimation, at a much lower cost.

The sensor network relationship is modeled as the joint probability $p(\mathcal{G}_X, \mathcal{F}, \mathcal{B}, \mathcal{I})$. Based on the statistical dependencies expressed in Fig. 2.10, the following decomposition is proposed:

$$p(\mathcal{G}_X, \mathcal{F}, \mathcal{B}, \mathcal{I}) = p(\mathcal{B})p(\mathcal{F})p(\mathcal{G}_X)p(\mathcal{S}|\mathcal{G}_X)p(\mathcal{I}|\mathcal{B}, \mathcal{F}, \mathcal{S}), \tag{2.6}$$

where $\mathcal{S}$ is the binary latent variable introduced to model the silhouette information. Both cases of $\mathcal{S}$—a pixel being the foreground or background, are considered when computing the final occupancy probability, thus maintaining the maximum robustness to silhouette instability.

- $p(\mathcal{B})$, $p(\mathcal{F})$ are the prior probabilities of a pixel to be background and foreground, which can be approximated statistically by the total background or foreground pixels divided by the number of pixels in the video sequence.

- $p(\mathcal{G}_X)$ is the prior likelihood for occupancy. Because this occupancy is at the top of the causality chain in the dependency graph, this term is set to be uniform, without favoring any locations.

- $p(\mathcal{S}|\mathcal{G}_X)$ is the silhouette likelihood term. The dependency reflects that the voxel occupancy in the scene explains the object detection in images.

- $p(\mathcal{I}|\mathcal{B}, \mathcal{F}, \mathcal{S})$ is the image likelihood term. Image colors are conditioned by object detections in the images, and the knowledge of the pre-learned background and uniformly-distributed foreground color models.

Based on the independence assumptions discussed earlier, Eq. 2.6 can be further decomposed as:

$$p(\mathcal{G}_X, \mathcal{F}, \mathcal{B}, \mathcal{I}) = c \prod_{i,p} p(\mathcal{S}_{i,p} \mid \mathcal{G}_X) p(\mathcal{I}_{i,p} \mid \mathcal{F}_{i,p}, \mathcal{B}_{i,p}, \mathcal{S}_{i,p}), \tag{2.7}$$

where c denotes a constant encoding the uniform prior probabilities discussed before. Once the terms in Eq. 2.7 are explained, the voxel occupancy inference can be carried out following Bayes' rule as:

$$\begin{aligned} p(\mathcal{G}_X|\mathcal{F}, \mathcal{B}, \mathcal{I}) &= \frac{\sum_{\mathcal{S}} p(\mathcal{G}_X, \mathcal{S}, \mathcal{F}, \mathcal{B}, \mathcal{I})}{\sum_{\mathcal{G}_X,\ \mathcal{S}} p(\mathcal{G}_X, \mathcal{S}, \mathcal{F}, \mathcal{B}, \mathcal{I})} \\ &= \frac{\prod_{i,p} \sum_{\mathcal{S}_{i,p}} p(\mathcal{S}_{i,p} \mid \mathcal{G}_X) p(\mathcal{I}_{i,p} \mid \mathcal{F}_{i,p}, \mathcal{B}_{i,p}, \mathcal{S}_{i,p})}{\sum_{\mathcal{G}_X} \prod_{i,p} \sum_{\mathcal{S}_{i,p}} p(\mathcal{S}_{i,p} \mid \mathcal{G}_X) p(\mathcal{I}_{i,p} \mid \mathcal{F}_{i,p}, \mathcal{B}_{i,p}, \mathcal{S}_{i,p})}. \end{aligned} \tag{2.8}$$

In Eq. 2.8, $p(\mathcal{I}_{i,p} \mid \mathcal{F}_{i,p}, \mathcal{B}_{i,p}, \mathcal{S}_{i,p})$ is the image formation term. If $\mathcal{S}_{i,p} = 1$, an object detection occurred at pixel (i, p). The pixel color value is explained by the uniform foreground model; if $\mathcal{S}_{i,p} = 0$, the pixel color value is explained by the background model. Both the uniform foreground and Gaussian background models are obtained following Eq. 2.2 & 2.3 respectively.

The only term left in Eq. 2.8 is $p(\mathcal{S}_{i,p} \mid \mathcal{G}_X)$, the silhouette formation term. It models the silhouette detection response of a single pixel sensor (i, p) to the occupancy state of voxel $\mathcal{G}_X$. Two local binary hidden variables—sampling variable $\mathcal{A}$ and external detection cause variable $\mathcal{E}$—need to be introduced to model the uncertainty along the

viewing ray of $\mathcal{G}_X$ that may affect the silhouette status. For example, voxel $\mathcal{G}_X$ may not exactly lie on the viewing line of a pixel, due to potential camera calibration errors, camera mis-synchronization, or simply because a voxel projection is larger than a pixel region. Also, a silhouette may be formed by an object in front of G_X along the viewing ray, or sensor noise variations.

When voxel X is occupied ($\mathcal{G}_X = 1$), the silhouette detection at pixel (i, p) is controlled by the sampling variable $\mathcal{A}$:

$$p(\mathcal{S}|[\mathcal{G}_X = 1]) = p(\mathcal{A} = 0)\ \mathcal{U}(\mathcal{S}_{i,p}) + p(\mathcal{A} = 1)\ P_d(\mathcal{S}_{i,p}). \tag{2.9}$$

By definition, $\mathcal{A} = 0$ if voxel X is not on the viewing line of pixel (i, p). In this case, the knowledge of X's occupancy is irrelevant to the sensor detection at (i, p). Therefore, the uniform distribution $\mathcal{U}(\mathcal{S}_{i,p})$ is used for the silhouette detection in Eq. 2.9. Otherwise, if the voxel is on the viewing line ($\mathcal{A} = 1$), then the detection at the pixel is ruled by the probability distribution $P_d(\mathcal{S}_{i,p})$. In practice, this distribution is set using a constant $P_D \in [0, 1]$, which is a parameter of the system: $P_d([\mathcal{S}_{i,p} = 1]) = P_D$ is the detection rate of a pixel sensor. P_D models the silhouette detection rate, as it happens in practice. The term $p(\mathcal{A})$ is dependent on i, p and X. Both uniform sampling and normal based sampling could be used depending on the required accuracy and computation cost.

When voxel X is empty ($G_X = 0$):

$$p(\mathcal{S}|[\mathcal{G}_X = 0]) = p(\mathcal{A} = 0)\ \mathcal{U}(\mathcal{S}_{i,p}) + p(\mathcal{A} = 1)\ [\ p(\mathcal{E} = 1)P_d(\mathcal{S}_{i,p}) + p(\mathcal{E} = 0)P_f(\mathcal{S}_{i,p})\ ]. \tag{2.10}$$

When the voxel is not on the viewing line of $p(\mathcal{A} = 0)$, no knowledge can be inferred about detection. Therefore the uniform distribution is used here again. When voxel

X is on p's viewing line ($\mathcal{A} = 1$), one needs to also check if there is some other object in front of X too. If yes, then pixel (i, p) is not explained by X. This is a simple model to explain the visual occlusion relationship along a viewing line. By definition, $\mathcal{E} = 1$ accounts for the possibility that some other object is on the same viewing line but in front of X: in this case, the detection is again ruled by the distribution $P_d(\mathcal{S}_{i,p})$. However, when no other object obstructs X on the viewing line ($\mathcal{E} = 0$), the detection is ruled by the distribution $P_f(\mathcal{S}_{i,p})$, which is defined as a constant $P_{FA} \in [0, 1]$, another parameter of the system: $P_f([\mathcal{S}_{i,p} = 1]) = P_{FA}$. It models the false alarm of a pixel sensor, which occurs when the sensor falsely relates the presence of matter on its viewing line, when in fact there is none. $p(\mathcal{E})$ is set to be yet another constant. Because there can be detection anywhere along the viewing line of p, no further assumptions about these causes are made. The constant $p(\mathcal{E})$ means detection is equally likely to be triggered by the voxel occupancy or by the above causes.

Algorithm Experiment and Qualitative Evaluation

In [Franco and Boyer (2005)], an indoor dataset ROND of 8 cameras is tested. A person is walking in the scene captured by cameras at different image resolutions (640×480 and 780×580), but at the same frame rate (15 *fps*). As shown in Fig. 2.11, with $P_D = 0.9, P_{FA} = 0.1$, and a uniform sampling 5×5 window for silhouette formation, the computed occupancy volume itself is a good estimate of the dynamic shape. Although the scene is in a controlled lighting environment, the silhouette segmentations still have artifacts, as shown on the left in Fig. 2.12. But with the introduced probabilistic sensor fusion, most of the system noise does not get much support from another camera at a different viewing angle, and therefore is weakened in the final result. This phenomenon is the key to the robustness of the algorithm.

The computation in the original paper is approximately 13 sec. per volume on a

[1]Fig. 2.11 & Fig. 2.12 courtesy of Jean-Sébastien Franco.

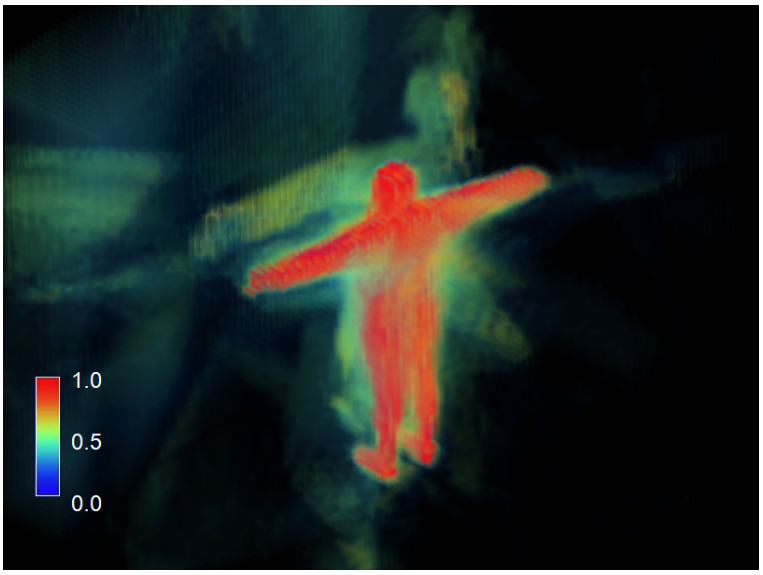

Figure 2.11: [1] Color coded 120^3 occupancy probability volume of ROND sequence, with $P_D = 0.9, P_{FA} = 0.1$, and a uniform sampling 5×5 window for the silhouette formation.

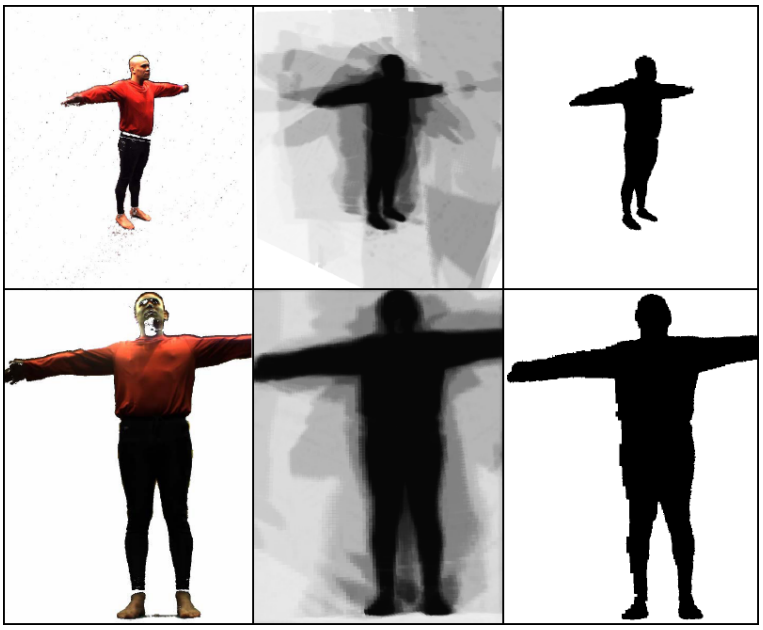

Figure 2.12: [1] Silhouette comparison between deterministic and probabilistic approaches. Left: silhouette extraction of deterministic approaches; Middle: maximum intensity projection rendering of occupancy grid (120^3) probabilities from original viewpoints, with black pixels probability 0 and white pixels 1; Right: thresholded slice of the middle column. Silhouette quality shows drastic improvement.

2.4 GHz PC. Fortunately, similar to volumetric visual hull algorithms, all voxels in the volume go through exactly the same computation procedure. Therefore, the algorithm is generically parallelizable. A GPU acceleration of this algorithm with the nVidia CUDA^{TM} general GPU programming tool is introduced in Chapter 3, which achieves real-time computation with 8 or 9 camera views and a volume size of 128^3 voxels.

Properties of the Probabilistic Sensor Fusion Result

The probabilistic output of the occupancy grid does not strictly follow the "conservativeness" property of the visual hull [Laurentini (1994)]. In other words, after thresholding the probability volume, the output shape is not guaranteed to contain the entire shape. However, the output can be taken as a robust shape estimate of the original object from multiple camera views.

Another property of the occupancy grid is that the recovered shape is not only robust to the sensor noise, but also to the sensor failure or the "out of the field of view" scenario. The second row of Fig. 2.12 indeed shows one example, where the person's arm is out of the camera view. But since most of the cameras see the arm, the final 3D shape estimate still have very high occupancy probability at the arm voxels in Fig. 2.11. Therefore, unlike many multi-view systems, such as [Gupta et al. (2007)], this probabilistic approach does not require any explicit camera selection when the 3D object is out of view. Moreover, this phenomenon gives some intuition into how to automatically deal with visibility occlusions, as discussed in the following chapters.

In this chapter I have reviewed the state-of-the-art in dynamic scene reconstruction. Visual hull, photo-consistency and multi-view stereo approaches are briefly discussed. The most promising framework for the real-time performance appears to be the probabilistic occupancy grid computation originated from the visual hull concept, which pays additional attention on the robustness of the shape estimation.

Chapter 3

Static Occluder (Visibility Obstacle) Inference

In general environments, occlusion is a problem for shape-from-silhouette methods. It can be categorized into (1) static occlusion and (2) dynamic shape inter-occlusion, both of which decrease the reconstruction quality, yet are very common and almost unavoidable in real sequences. Static occlusion is the main focus of this chapter. Dynamic shape inter-occlusion is discussed in Chapter 4.

3.1 Static Occluder Challenge

Static occlusions happen when a static object blocks the view of a dynamic object, such as the sculpture blocking the person in Fig. 2.8 and Fig. 3.1. The static object is called an "occluder". Like the sculpture, occluders cannot always be removed from the scene in advance, so their appearances are learned as a part of the background model if not manually delineated. The problem with static occluder comes when a dynamic object goes behind a static occluder, since the image does not differ from the background model in this occluded region, these pixels still have high background probability. According to background subtraction algorithms discussed as in Eq. 2.4, an incomplete silhouette happens. Consequently, due to the intersection rule, such corrupted silhouettes result

in an incomplete visual hull.

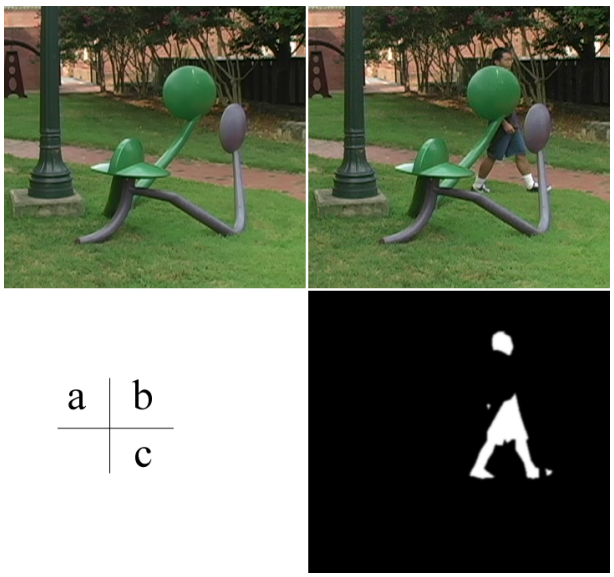

Figure 3.1: Static occlusion problem in silhouette-based reconstruction method. (a) a natural scene with unremovable complicated static occluders; (b) a camera frame during a dynamic scene capturing; (c) manual segmentation of the foreground silhouette.

Generally detecting and accounting for occlusions has attracted the attention of researchers for problems such as structure from motion [Favaro et al. (2003)], motion and occlusion boundary detection [Apostoloff and Fitzgibbon (2005)]. The scope of these works is however limited to extraction of sparse 2D features such as T-junctions or edges to improve robustness of data estimation. Inter-object occlusions were implicitly modeled in the context of voxel coloring approaches, using an iterative scheme with semi-transparent voxels and multiple views of a scene from the same time instant [Bonet and Viola (1999)]. [Brostow and Essa (1999); Zhou and Tao (2003); Guan et al. (2006)] all propose 2D solutions for detecting one or several motion and occlusion layers for a single camera view. [Stein and Hebert (2009)] show that learning motion and appearance cues on super pixels in video frames can also generate more robust estimate of the boundary location. Both 2D occlusion layers and boundary information can be used and accounted for, when building the visual hull of dynamic objects [Guan et al. (2006)]. To the best

of my knowledge, the method introduced in this chapter, based on [Guan et al. (2007)] is the first to address the dense recovery of full 3D occluder shapes from multiple image sequences.

Recently, [Keck and Davis (2008)] also proposed to explicitly model static occluders. They use iterative EM framework that at each frame first solves the voxel occupancy which then feeds back into the system by updating the occlusion model. Hard threshold of silhouette information is required during initialization and the occluder information is maintained in a 4D (a 3D space volume per camera view) state space. Also, the usage of iterative refinement makes it only an off-line solution and hard for real-time accelerations. The advantage of [Keck and Davis (2008)], is that it focuses on systems with fewer cameras. Although three to four cameras may still be feasible, [Guan et al. (2007)] use eight or more cameras simply to produce decent shape estimate.

Since publication, the proposed algorithm has been embedded in a dynamic scene reconstruction system which incorporates multiple dynamic shape estimation and tracking as well as static occluder recovery [Guan et al. (2008b)]. It has been shown that the recovery of occluders does help refine dynamic shapes. More details are discussed in Section 3.4 and Chapter 4.

3.2 Intuition and Solution

Given video sequences from n fully calibrated cameras, observing a scene at discrete time steps $t \in \{1, ..., T\}$ where people, and more generally dynamic moving objects can evolve. A set of background images of the scene, free from any dynamic object, have previously been observed for each camera. Static occluder objects, whose appearance is recorded in the background images of the scene, are present in the interaction space of dynamic objects. They are thus liable to generate partial occlusions of dynamic objects, with respect to one or several cameras.

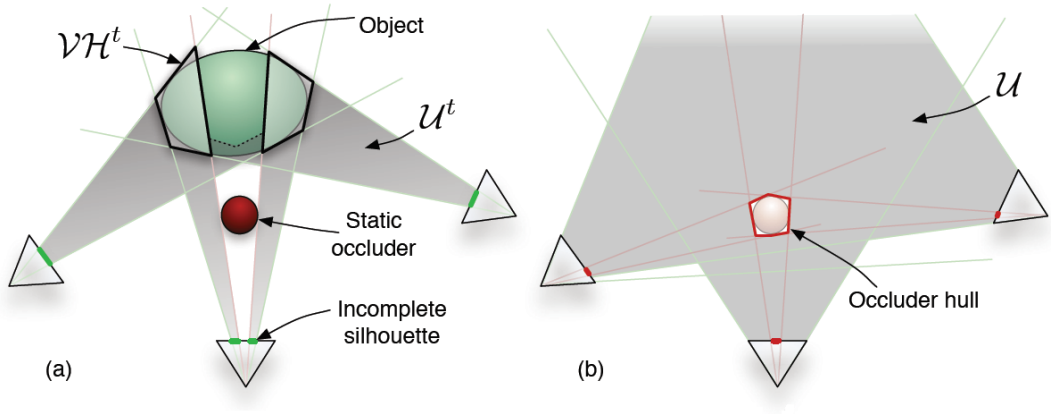

Figure 3.2: Deterministic occlusion reasoning. (a) An occluder-free region $\mathcal{U}^t$ can be deduced from the incomplete visual hull $\mathcal{VH}^t$ at time t. (b) $\mathcal{U}$: occluder-free regions accumulated over time.

Theoretically, occluder shapes can be accessed with careful reasoning about the visual hull of incomplete silhouettes (Fig. 3.2). Let $\mathcal{S}^t$ be the set of incomplete silhouettes obtained at time t, and $\mathcal{VH}^t$ the incomplete visual hull obtained using these silhouettes. These entities are said to be incomplete because the silhouettes used are potentially corrupted by static occluders that mask the silhouette extraction process. However, the incomplete visual hull is a region that is observed by all cameras as being both occupied by an object and unoccluded from any view. Thus an entire region $\mathcal{U}^t$ of points in space can be deduced that are free from any static occluder shape. $\mathcal{U}^t$ is the set of points $X \in \mathbb{R}^3$ for which a view i exists, such that the viewing line of X from view i hits the incomplete visual hull at a first visible point A, and $X \in OA$, with O the optical center of view i (Fig. 3.2 (a)). The latter expresses the condition that X appears in front of the visual hull with respect to view i. The region $\mathcal{U}^t$ varies with t, thus assuming static occluders and broad coverage of the scene by dynamic object motion, the free space in the scene can be deduced as the region $\mathcal{U} = \bigcup_{t=1}^{T} \mathcal{U}^t$. The shape of occluders, including concavities if they were covered by object motion, can be recovered as the complement of $\mathcal{U}$ in the common visibility region of all views (Fig. 3.2 (b)).

However this deterministic approach would yield a non-robust solution, due to the

inherent silhouette sensitivity to noise. It also suffers from the limitation that only portions of objects that are seen by all views can contribute to occlusion reasoning. In addition, this scheme only accumulates negative information, where occluders are certain not to be. However, positive information is also available to the problem: if one had known or could take a good guess at where the object shape was, discrepancies between the object's projection and the actual silhouette recorded would tell where an occlusion is happening. Thanks to the sensor fusion occupancy grid introduced in Chapter 2, it can lift these limitations and provide a robust probabilistic solution.

Recall from Section 2.4.2, that the probabilistic occupancy grid has the property that even if a part of the dynamic shape is out of a certain camera's field of view, (*e.g.* the person's arm on the bottom row of Fig. 2.12), as long as majority of the cameras see the object, the out-of-view parts still have high occupancy probability, as shown in Fig. 2.11. In fact, one can think of the out-of-view scenario as a special case of static occlusion, that the area outside of a camera field of view is equivalent to a big occluder. This robustness is also true for general occlusions. As long as majorities of the camera views can provide correct support, the occupancy grid would have high occupancy probability in the occluded region.

Intuitively, one can project the computed occupancy probability grid to the occluded view, with every pixel storing the maximum probability along the projection ray, similar to the maximum intensity projection rendering result shown in Fig. 2.12. The inconsistency between the "projected probabilistic dynamic shape" and the silhouette information provided by the background model indicates occlusion event has happened here. It tells that there may be occluders along the viewing ray (positive cue for occluder inference). On the other hand, for an occlusion-free view, the occupancy grid projection is consistent with the silhouette cue obtained from the background model. Such consistency indicates the viewing ray from the camera to the dynamic shape is occlusion-free (negative cue for occluder inference).

Given the robust sensor fusion framework for dynamic shape estimation, a second occupancy grid is introduced to model the static occluder probabilistically, in which all negative and positive cues are fused and compete in a complementary way toward occluder shape estimation. Similar to the dynamic shape estimation, this occlusion computation algorithm is also robust to natural scene variations.

3.3 Modeling

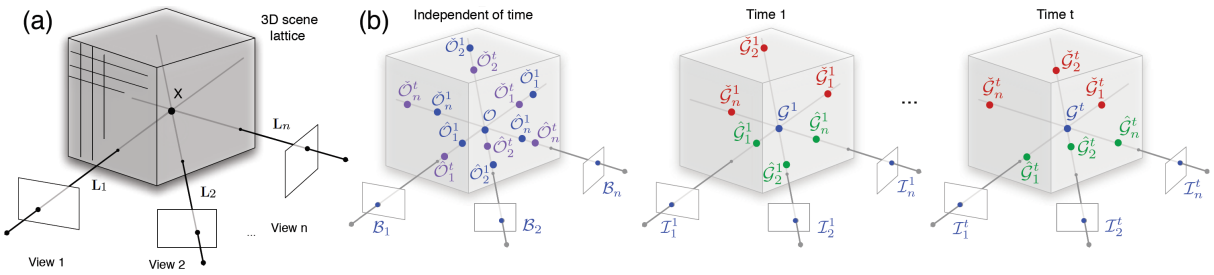

Figure 3.3: Occluder inference problem overview. (a) Geometric context of voxel X. (b) Main statistical variables used to infer the occluder occupancy probability of X. $\mathcal{G}t$,$\hat{\mathcal{G}}_i^t$,$\check{\mathcal{G}}_i^t$: dynamic object occupancies at relevant voxels at, in front of, behind X respectively. $\mathcal{O}$,$\hat{\mathcal{O}}_i^t$,$\check{\mathcal{O}}_i^t$: static occluder occupancies at, in front of, behind X. $\mathcal{I}_i^t$, $\mathcal{B}_i$: colors and background color models observed where X projects in images.

Consider a scene observed by n calibrated cameras. Focus on the case of one scene voxel with 3D position X among the possible coordinates in the lattice chosen for scene discretization. The two possible states of occluder occupancy at this voxel are expressed using a binary variable $\mathcal{O}$. This state is assumed to be fixed over the entire experiment in this setup under the assumption that the occluder is static. Clearly, the regions of importance to infer $\mathcal{O}$ are the n viewing lines $\mathbf{L}_i$, $i \in \{1, ..., n\}$, as shown in Fig. 3.3(a). Scene states are observed for a finite number of time instants $t \in \{1, ..., T\}$. In particular, dynamic shape occupancies of voxel X at time t are expressed by a binary statistical variable $\mathcal{G}^t$, treated as an unobserved variable, which is computed using the formulation introduced in Section 2.4.2. Notice that, the subscript $_X$ at $\mathcal{G}_X^t$ is omitted from now on for clearer readability.

Observed Variables

The voxel X projects to n image pixels x_i, $i \in \{1, ..., n\}$, whose color observed at time t in view i is expressed by the variable $\mathcal{I}_i^t$. Assume that static background images were observed free of dynamic objects, and that the appearance and variability of background colors for pixels x_i was recorded and modeled using a set of parameters $\mathcal{B}_i$. Such observations can be used to infer the probability of dynamic object occupancy in the absence of background occluders. Since the foreground model $\mathcal{F}$ still follows the uniform distribution, it is omitted for readability. The actual dynamic shape computation is exactly the same as in Section 2.4.2. The problem of recovering occluder occupancy is more complex because it requires modeling interactions between voxels on the same viewing lines. Relevant statistical variables are shown in Fig. 3.3(b).

Viewing Line Modeling

Because of potential mutual occlusions, one must account for other occupancies along the viewing lines of X to infer $\mathcal{O}$. These can be either other static occluder states, or dynamic object occupancies that vary across time. Several such occluders or objects can be present along a viewing line, leading to a number of possible occupancy states for voxels on the viewing line of X. Accounting for the combinatorial number of possibilities for voxel states along X's viewing line is neither necessary nor meaningful: first, because occupancies of neighboring voxels are fundamentally correlated with the presence or absence of a single common object; second, because the main useful information one needs to make occlusion decisions about X is whether something is in front of it or behind it, regardless of where the intervening object is along the viewing line.

The image pixel of the viewing line through Xalways falls into one of the following three scenarios: (1) the pixel formation can be explained by the component in front of X, if the voxel(s) are not all empty; (2) it can be explained by X, if all voxels in front of $X$is empty but $X$is not; (3) it can be explained by the component at the back

of $X$with respect to the camera view, if all space from the camera to Xincluding $X$is empty. With this in mind, each viewing line is modeled using three components, the state of X, the state of occlusion of $X$by anything in front and at the back of X. As mentioned before, neighboring voxels' states are highly correlated. Namely if a voxel has a high probability of being occupied by a dynamic object, its neighbor is very likely to have a high probability too. Therefore, the front and back components of $X$are modeled by extracting the two most influential modes in front and behind of X, that are given by two voxels $\hat{X}_i^t$ and $\check{X}_i^t$ with the highest occupancy probability. We select $\hat{X}_i^t$ as the voxel at time t that most contributes to the belief that X is obstructed by a dynamic object along $\mathbf{L}_i$, and $\check{X}_i^t$ as the voxel most likely to be occupied by a dynamic object behind X on $\mathbf{L}_i$ at time t.

Viewing Line Unobserved Variables

With this three component modeling, comes a number of related statistical variables illustrated in Fig. 3.3(b). The occupancy of voxels $\hat{X}_i^t$ and $\check{X}_i^t$ by the visual hull of a dynamic object at time t on $\mathbf{L}_i$ is expressed by two binary state variables, respectively $\hat{\mathcal{G}}_i^t$ and $\check{\mathcal{G}}_i^t$. Two binary state variables $\hat{\mathcal{O}}_i^t$ and $\check{\mathcal{O}}_i^t$ express the presence or absence of an occluder at voxels $\hat{X}_i^t$ and $\check{X}_i^t$ respectively. Note the difference in semantics between the two variable groups $\hat{\mathcal{G}}_i^t$, $\check{\mathcal{G}}_i^t$ and $\hat{\mathcal{O}}_i^t$, $\check{\mathcal{O}}_i^t$. The former designates dynamic visual hull occupancies of different time instants and chosen positions, while the latter expresses static occluder occupancies, whose position only was chosen in relation to t. Both need to be considered because they both influence the occupancy inference and are not independent. For legibility, the conjunction of a group of variables is occasionally referred to without indices and exponents, *e.g.* $\mathcal{G} = \{\mathcal{G}_1, ..., \mathcal{G}_T\}$, $\mathcal{B} = \{\mathcal{B}_1, ..., \mathcal{B}_n\}$.

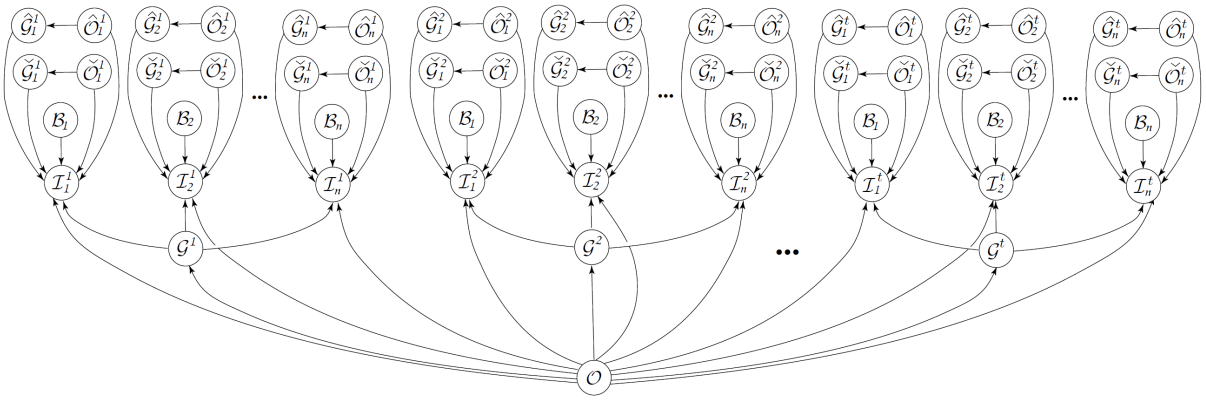

Figure 3.4: The dependency graph for the static occluder inference at voxel X, assuming the probability for X to be $\mathcal{G}$ is known. Notice that the background model for each view $\mathcal{B}_i$ does not change with time, but just drawn duplicatedly for the clarity of the graph.

3.3.1 Joint Distribution

As a further step toward offering a tractable solution to occlusion occupancy inference, the noisy interactions between the considered variables are described through the decomposition of their joint probability distribution $p(\mathcal{O}, \mathcal{G}, \hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t, \mathcal{I}, \mathcal{B})$. According to the dependency graph shown in Fig. 3.4 the following decomposition of the joint probability is proposed:

$$
\begin{aligned}
&p(\mathcal{O}, \mathcal{G}, \hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t, \mathcal{I}, \mathcal{B}) = \\
&p(\mathcal{O}) \prod_{t=1}^{T} p(\mathcal{G}^t|\mathcal{O}) \prod_{i=1}^{n} p(\hat{\mathcal{O}}_i^t) p(\hat{\mathcal{G}}_i^t|\hat{\mathcal{O}}_i^t) p(\check{\mathcal{O}}_i^t) p(\check{\mathcal{G}}_i^t|\check{\mathcal{O}}_i^t) p(\mathcal{I}_i^t|\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t, \mathcal{B}_i),
\end{aligned}
\tag{3.1}
$$

where $p(\mathcal{O})$, $p(\hat{\mathcal{O}}_i^t)$, $p(\check{\mathcal{O}}_i^t)$ are priors of occluder occupancy. They are set to a single constant distribution P_O which reflects the expected ratio between occluder and non-occluder voxels in a scene. No particular region of space is to be favored *a priori*.

3.3.2 Dynamic Occupancy Priors

Next, let us discuss $p(\mathcal{G}^t|\mathcal{O})$, $p(\hat{\mathcal{G}}_i^t|\hat{\mathcal{O}}_i^t)$ and $p(\check{\mathcal{G}}_i^t|\check{\mathcal{O}}_i^t)$. They are priors of dynamic visual hull occupancy with identical semantics. This choice of terms reflects the following modeling decisions. First, the dynamic visual hull occupancies involved are considered independent of one another, as they synthesize the information of three distinct regions for each viewing line. However, they depend upon the knowledge of occluder occupancy at the corresponding voxel position, because occluder and dynamic object occupancies are mutually exclusive at a given scene location. Importantly, one has no direct access to dynamic object occupancies but to the occupancies of its *visual hull*. Fortunately, this ambiguity can be adequately modeled in a Bayesian framework by introducing a local hidden variable $\mathcal{C}$ expressing the correlation between dynamic and occluder occupancy:

$$p(\mathcal{G}^t|\mathcal{O}) = \sum_{\mathcal{C}} p(\mathcal{C})p(\mathcal{G}^t|\mathcal{C},\mathcal{O}). \tag{3.2}$$

One can set $p(\mathcal{C} = 1) = \mathcal{P}_c$, a constant, expressing the prior belief about the correlation between visual hull and occluder occupancy. The prior $p(\mathcal{G}^t|\mathcal{C},\mathcal{O})$ explains what is expected to be known about $\mathcal{G}^t$ given the state of $\mathcal{C}$ and $\mathcal{O}$:

$$p(\mathcal{G}^t = 1|\mathcal{C} = 0, \mathcal{O} = \omega) = \mathcal{P}_{\mathcal{G}^t} \quad \forall\omega \tag{3.3}$$

$$p(\mathcal{G}^t = 1|\mathcal{C} = 1, \mathcal{O} = 0) = \mathcal{P}_{\mathcal{G}^t} \tag{3.4}$$

$$p(\mathcal{G}^t = 1|\mathcal{C} = 1, \mathcal{O} = 1) = \mathcal{P}_{go} \ , \tag{3.5}$$

with $\mathcal{P}_{\mathcal{G}^t}$ the prior dynamic object occupancy probability as computed independently of occlusions as in Section 2.4.2, and $\mathcal{P}_{go}$ set close to 0, expressing that it is unlikely that the voxel is occupied by dynamic object visual hulls when the voxel is known to be occupied by an occluder and both dynamic and occluder occupancy are known to be strongly correlated (Eq. 3.5). The probability of visual hull occupancy is given by the

previously computed occupancy prior, in case of non-correlation (Eq. 3.3), or when the states are correlated but occluder occupancy is known to be empty (Eq. 3.4).

3.3.3 Image Sensor Model

The sensor model $p(\mathcal{I}_i^t|\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t, \mathcal{B}_i)$ is governed by a hidden local per-pixel process $\mathcal{S}$. Similar to dynamic shape modeling, the binary variable $\mathcal{S}$ represents the hidden silhouette detection state (0 or 1) at this pixel. It is unobserved information and can be marginalized, given an adequate split into two subterms:

$$\begin{aligned} & p(\mathcal{I}_i^t|\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t, \mathcal{B}_i) \\ = & \sum_{\mathcal{S}} p(\mathcal{I}_i^t|\mathcal{S}, \mathcal{B}_i) p(\mathcal{S}|\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t). \end{aligned} \tag{3.6}$$

The first term $p(\mathcal{I}_i^t|\mathcal{S}, \mathcal{B}_i)$ indicates what color distribution is expected given the knowledge of silhouette detection, trained background color model and uniform foreground model at this pixel.

The second part of the sensor model $p(\mathcal{S}|\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t)$ specifies what silhouette state is expected to be observed given the three dominant occupancy state variables of the corresponding viewing line. Since these are encountered in the order of visibility $\hat{X}_i^t$, X, $\check{X}_i^t$, the following relations hold:

$$\begin{aligned} & p(\mathcal{S}|\{\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t\} = \{o, g, \mathrm{k}, \mathrm{l}, \mathrm{m}, \mathrm{n}\}, \mathcal{B}_i) \\ = & p(\mathcal{S}|\{\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t\} = \{0, 0, o, g, \mathrm{p}, \mathrm{q}\}, \mathcal{B}_i) \\ = & p(\mathcal{S}|\{\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t\} = \{0, 0, 0, 0, o, g\}, \mathcal{B}_i) \\ = & P_S(\mathcal{S}|o, g) \qquad \forall (o, g) \neq (0, 0) \;\; \forall (\mathrm{k}, \mathrm{l}, \mathrm{m}, \mathrm{n}, \mathrm{p}, \mathrm{q}). \end{aligned} \tag{3.7}$$

These expressions convey two characteristics. First, that the form of this distribution

is given by the first non-empty occupancy component in the order of visibility, regardless of what is behind this component on the viewing line. Second, that the form of the first non-empty component is given by an identical sensor prior $P_S(\mathcal{S}|o,g)$. The four parametric distributions of $P_S(\mathcal{S}|o,g)$ are set as following:

$$P_S(\mathcal{S}=1|0,0)=\mathcal{P}_{fa} \qquad P_S(\mathcal{S}=1|1,0)=\mathcal{P}_{fa} \tag{3.8}$$

$$P_S(\mathcal{S}=1|0,1)=\mathcal{P}_{d} \qquad P_S(\mathcal{S}=1|1,1)=0.5, \tag{3.9}$$

$\mathcal{P}_{fa} \in [0,1]$ and $\mathcal{P}_{d} \in [0,1]$ are constants expressing the prior probability of *false alarm* and the probability of *detection*, respectively. They can be chosen once for all datasets as the method is not sensitive to the exact value of these priors. Meaningful values for $\mathcal{P}_{fa}$ are close to 0, while $\mathcal{P}_{d}$ is generally close to 1. Eq. 3.8 expresses the cases where no silhouette is expected to be detected in images, *i.e.* either when there are no objects at all on the viewing line, or when the first encountered object is a static occluder, respectively. Eq. 3.9 expresses two distinct cases. The first case is where a dynamic object's visual hull is encountered on the viewing line, in which case we expect to detect a silhouette at the matching pixel. The second case is where both an occluder and dynamic visual hull are present at the first non-free voxel. This is perfectly possible, because the visual hull is an overestimate of the true dynamic object shape. While the true shapes of objects and occluders are naturally mutually exclusive, the *visual hull* of dynamic objects can overlap with occluder voxels. In this case the distribution is set to uniform, because the silhouette detection state cannot be predicted: it can be caused by shadows cast by dynamic objects on occluders in the scene, and noise.

3.3.4 Inference

Estimating the occluder occupancy at a voxel translates to estimating $p(\mathcal{O}|\mathcal{I},\mathcal{B})$ in Bayesian terms. Applying Bayes' rule to the modeled joint probability (Eq. 3.1) leads

to the following expression, once hidden variable sums are decomposed to factor out terms not required at each level of the sum:

$$p(\mathcal{O}|\mathcal{I},\mathcal{B}) = \frac{1}{c}\, p(\mathcal{O}) \prod_{t=1}^{T} \left(\sum_{\mathcal{G}^t} p(\mathcal{G}^t|\mathcal{O}) \left(\prod_{i=1}^{n} \mathcal{P}_i^t \right) \right) \tag{3.10}$$

where

$$\mathcal{P}_i^t = \sum_{\check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t} p(\check{\mathcal{O}}_i^t) p(\check{\mathcal{G}}_i^t|\check{\mathcal{O}}_i^t) \sum_{\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t} p(\hat{\mathcal{O}}_i^t) p(\hat{\mathcal{G}}_i^t|\hat{\mathcal{O}}_i^t)\; p(\mathcal{I}_i^t|\hat{\mathcal{O}}_i^t, \hat{\mathcal{G}}_i^t, \mathcal{O}, \mathcal{G}^t, \check{\mathcal{O}}_i^t, \check{\mathcal{G}}_i^t, \mathcal{B}_i). \tag{3.11}$$

$\mathcal{P}_i^t$ expresses the contribution of view i at a time t. The formulation of Eq. 3.10 therefore expresses Bayesian fusion over the various observed time instants and available views, with marginalization over unknown viewing line states. The normalization constant c is easily obtained by ensuring that the distribution sums to 1.

3.3.5 Online Incremental Computation

Occlusion information is not gathered from all views at the same time. As the dynamic shape moves around in the scene, different viewing angles may have gathered occlusion cues at different times. Due to calibration errors and other sources of noise, one should not trust the observation of a voxel from just one camera view. Only when occlusion cues of a voxel have been gathered from all camera views, can one reliably compute the voxel's occluder probability. Another critical reason to have such a reliability term is because if information is obtained from a single view, one only knows that the occlusion is happening somewhere along the viewing line, but has no idea of where exactly in 3D until a second view gathers enough knowledge so that 3D triangulation can be performed.

To determine the reliability of voxels, one needs to model the intuition that voxels whose occlusion cues arise from an abnormally low number of views should not be

trusted. Since this involves all cameras and their observations jointly, the inclusion of this constraint in the initial model would break the symmetry in the inference formulated in Eq. 3.10 and defeat the possibility for online updates. Instead, a second criterion is used in the form of a reliability measure $R \in [0, 1]$. Small values indicate poor coverage of dynamic objects, while large values indicate sufficient cue accumulation. One can define reliability using the following expression:

$$R = \frac{1}{n} \sum_{i=1}^{n} \max_{t} (1 - \mathcal{P}_{\hat{\mathcal{G}}_i^t}) \mathcal{P}_{\check{\mathcal{G}}_i^t}, \tag{3.12}$$

where $\mathcal{P}_{\hat{\mathcal{G}}_i^t}$ and $\mathcal{P}_{\check{\mathcal{G}}_i^t}$ the prior probabilities of dynamic visual hull occupancy. R examines, for each camera i, the maximum occurrence of X across the complete video sequence duration to be both unoccluded and in front of a dynamic object. This determines how well a given view i was able to contribute to the estimation across the sequence. R then averages these values across views, to measure the overall quality of observation, and underlying coverage of dynamic object motion for the purpose of occlusion inference.

The reliability R can be used online in conjunction to the occlusion probability estimation to evaluate a conservative occluder shape at all times, by only considering voxels for which R exceeds a certain quality threshold. As shown in Section 3.4, it can be used to reduce the sensitivity to noise in regions of space that have only been observed marginally.

3.3.6 Accounting for Occlusion in Dynamic Shape Estimation

As more data becomes available and reliable, the results of occluder estimation can be accounted for when inferring the occupancies of dynamic objects. This translates to the evaluation of $p(\mathcal{G}^\tau | \mathcal{I}^\tau \mathcal{B})$ for a given voxel X and time τ. The difference with the single-frame formulation of dynamic object occupancy in Section 2.4.2 is that a prior over the occlusions at every voxel in the grid is now known. For this inference, $\mathcal{G}^\tau$ is

considered independent of $\mathcal{G}^t$ $\forall t \neq \tau$, leading to the following simplified joint probability distribution:

$$p(\mathcal{O})p(\mathcal{G}^\tau|\mathcal{O})\prod_{i=1}^{n} p(\hat{\mathcal{O}}_i^\tau)p(\hat{\mathcal{G}}_i^\tau|\hat{\mathcal{O}}_i^\tau)p(\mathcal{I}_i^\tau|\hat{\mathcal{O}}_i^\tau,\hat{\mathcal{G}}_i^\tau,\mathcal{O},\mathcal{G}\tau,\mathcal{B}_i),$$

where $\mathcal{G}^\tau$ and $\mathcal{O}$ are the dynamic and occluder occupancy at the inferred voxel, $\hat{\mathcal{O}}_i^\tau$, $\hat{\mathcal{G}}_i^\tau$ the variables matching the most influential component along $\mathbf{L}_i$, *in front of* X. This component is selected as the voxel whose prior of being occupied is maximal, as computed to date by occlusion inference. In this inference, there is no need to consider voxels behind X, because knowledge about their occlusion occupancy has no influence on X's state.

The parametric forms of this distribution have identical semantics as in Section 3.3.1 but different assignments because of the nature of the inference. Naturally, no prior information about dynamic occupancy is assumed here. $p(\mathcal{O})$ and $p(\hat{\mathcal{O}}_i^\tau)$ are set using the result to date of Eq. 3.10 at their respective voxels, as prior. $p(\mathcal{G}^\tau|\mathcal{O})$ and $p(\hat{\mathcal{G}}_i^\tau|\hat{\mathcal{O}}_i^\tau)$ are constant: $p(\mathcal{G}^\tau = 1|\mathcal{O} = 0) = 0.5$ expresses a uniform prior for dynamic objects when the voxel is known to be occluder free. $p(\mathcal{G}^\tau = 1|\mathcal{O} = 1) = \mathcal{P}_{go}$ expresses a low prior of dynamic visual hull occupancy given the knowledge of occluder occupancy, as in Eq. 3.5. The term $p(\mathcal{I}_i^\tau|\hat{\mathcal{O}}_i^\tau,\hat{\mathcal{G}}_i^\tau,\mathcal{O},\mathcal{G}^\tau,\mathcal{B}_i)$ is set same as Eq. 3.7, only stripped of the influence of $\check{\mathcal{O}}_i^\tau$, $\check{\mathcal{G}}_i^\tau$.

Notice that the formulation introduced here is an extension of that in Section 2.4.2. In order to make such a model feasible, one can assume that $\mathcal{O}$ follows a uniform occluder distribution at first, and change it to the reliable occluder distribution after the dynamic shape has explored the scene well enough, so that the dynamic computation can be refined with the accumulated knowledge about the static environment.

3.4 Results and Evaluation

3.4.1 Sensor Model Summary

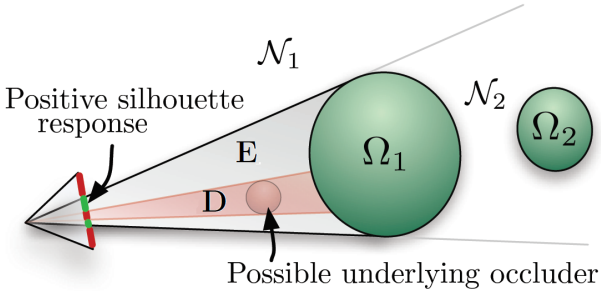

Figure 3.5: Regions of influence of a mono-camera sensor among the various voxels of a scene, as described by the proposed model.

The core of the occlusion formulation is controlled by five parameters $\mathcal{P}_o$, $\mathcal{P}_{go}$, $\mathcal{P}_c$, $\mathcal{P}_d$ and $\mathcal{P}_{fa}$. If two dynamic objects are *perfectly* known to occupy space in regions Ω_1 and Ω_2 (Fig. 3.5), various regions of importance appear in the occlusion inference, for a given camera and time instant. $\mathcal{N}_1$ and $\mathcal{N}_2$ are regions where the current view does not contribute and the inference reverts to the prior $\mathcal{P}_o$: $\mathcal{N}_1$ because it is outside of the viewing cone of dynamic objects, $\mathcal{N}_2$ because it is obstructed by an actual dynamic object Ω_1. **E** projects to a positive silhouette response area in the image and the probability of occluder occupancy is thus deduced to be low. **D** projects to an image area with low silhouette response, despite being in front of Ω_1, thus it is deduced that an occluder is probably in this region. The strength of the contribution in these regions depends on the confidence in observations, as expressed by $\mathcal{P}_d$ and $\mathcal{P}_{fa}$. Finally, Ω_1 and Ω_2 also contribute directly to the estimation through $\mathcal{P}_c$ and $\mathcal{P}_{go}$: a higher $\mathcal{P}_c$ and lower $\mathcal{P}_{go}$ give more weight to the mutual exclusivity constraint between occluders and dynamic objects and thus lead to lower occluder probabilities in these regions.

Depending on the actual probabilities of silhouette response and on the prior probabilities of the dynamic occupancy in regions Ω_1 and Ω_2, actual voxel contributions exhibit a mixture of these different behaviors in practice, which the model automati-

cally combines. Values given to the model parameters could be learned using training data. Nevertheless, the inference has low sensitivity to small changes of these parameters, and they are sufficiently generic and intuitive that setting them manually for a large number of different sequences is possible. Throughout the experiments described in this section, a single set of parameters is used: $\mathcal{P}_o = 0.15$, $\mathcal{P}_{go} = 0.001$, $\mathcal{P}_c = 0.5$, $\mathcal{P}_d = 0.8$ and $\mathcal{P}_{fa} = 0.1$.

3.4.2 Occlusion Inference Results

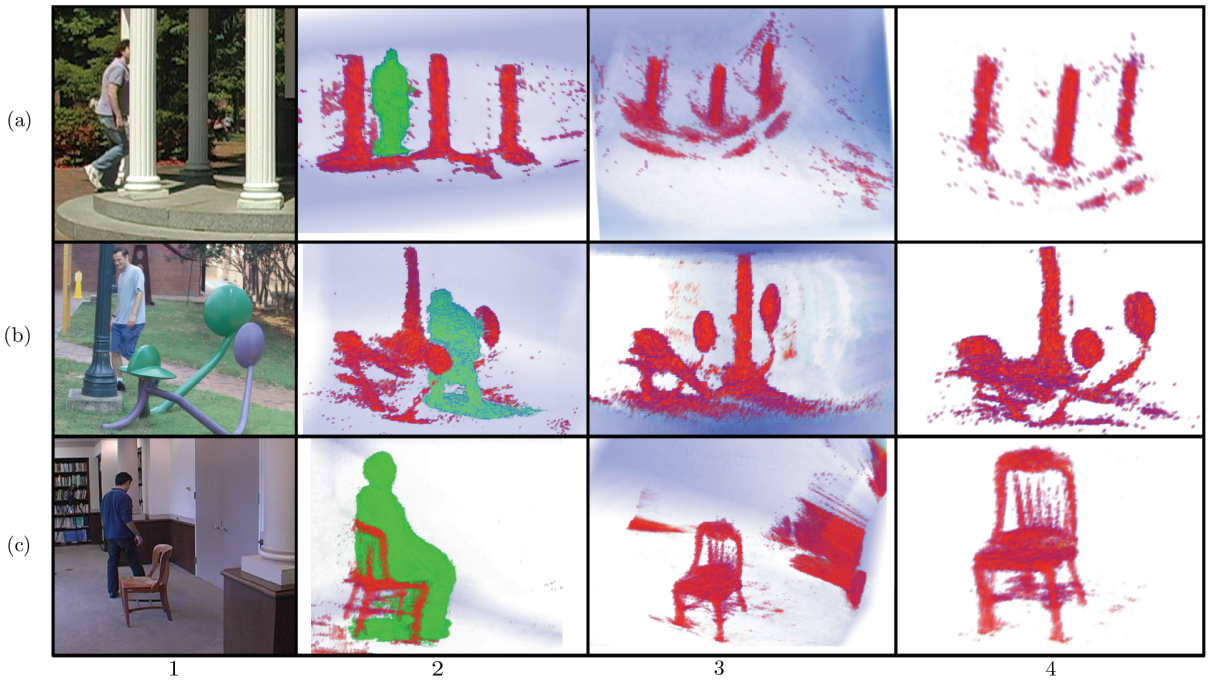

Figure 3.6: Occluder shape retrieval results (**best viewed in color**). Sequences: (a) PILLAR , (b) SCULPTURE , (c) CHAIR . 1) Scene overview. Note the harsh light, difficult backgrounds for (a) and (b), and specularity of the sculpture, causing no significant modeling failure. 2-3) Occluder inference according to Eq. 3.10. Blue: neutral regions (prior $\mathcal{P}_o$), red: high probability regions. Brighter/clear regions indicate the inferred absence of occluders. Fine levels of detail are modeled, sometimes lost—mostly to calibration. In (a) the structure's steps are also detected. 4) Same inference with additional exclusion of zones with reliability R under 0.8. Peripheral noise and marginally observed regions are eliminated. The background protruding shape in (c3) is due to a single occlusion from view (c1), thus yielding the viewing cone of the occluder as expected. This shows why the reliability term is important.

The approach is tested on several multi-view sequences: the PILLAR and SCULPTURE sequences which are acquired outdoors, and the CHAIR sequence, acquired indoors, with combined artificial and natural light from large bay windows. In all sequences 9 DV cameras surround the scene of interest, background models are learned in the absence of moving objects. One or several people then walk around and through the occluder in each scene. The shape of the people is estimated at each time step and used as prior to occlusion inference. The data is used to compute an estimate of the occluder's shape using Eq. 3.10. Results are presented in Fig. 3.6.

All cameras are recording at 30Hz. Color calibration is unnecessary because the model uses silhouette information only. The background model is learned per-view using a single Gaussian color model per pixel, and training images. Although simple, the model proves sufficient, even in outdoor sequences subject to background motion, foreground object shadows, and substantial illumination changes, illustrating the strong robustness of the method to difficult real conditions. The method can cope well with background mis-classifications that do not lead to large coherent false positive dynamic object estimations: pedestrians are routinely seen in the background for the SCULPTURE and PILLAR sequences (*e.g.* Fig. 3.6(a1)), without any significant corruption of the inference.

Adjacent frames in the input videos contain largely redundant information for occluder modeling, thus videos can safely be subsampled. PILLAR was processed using 50% of the frames (1053 frames processed), SCULPTURE and CHAIR with 10% (160 and 168 processed frames respectively). Processing of both dynamic and occluder occupancy was handled on a 2.8 GHz PC at approximately 1 timestep per minute. The very strong locality inherent to the algorithm and preliminary benchmarks suggest that real-time performance could be achieved using a GPU implementation. Occluder information does not need to be processed for every frame because of adjacent frame redundancy, opening the possibility for online, asynchronous cooperative computation of occluder

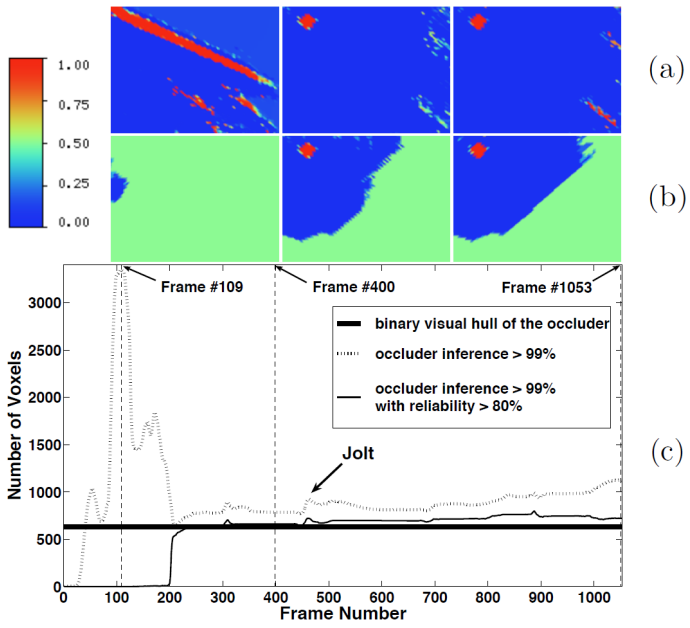

Figure 3.7: Online inference analysis and ground truth visual hull comparison, using PILLAR dataset, focusing on a slice including the middle pillar (**best viewed in color**). (a) Frames 109, 400 and 1053, inferred using Eq. 3.10. (b) Same frames, this time excluding zones with reliability under 0.8 (reverted here to 0.5). (c) Number of voxels compared to ground truth visual hull across time.

and dynamic objects at interactive frame rates.

3.4.3 Online Computation Results

All experiments can be computed using incremental inference updates. Fig. 3.7 depicts the inference's progression, using the sensor fusion formulation alone or in combination with the reliability criterion. For the purpose of this experiment, the PILLAR sequence is used and the occluder is manually segmented in each view for a ground truth comparison, and a subregion of the scene is analyzed in which the expected behaviors are well isolated. Fig. 3.7 shows that both schemes converge reasonably close to the visual hull of the considered pillar. In scenes with concave parts accessible to dynamic objects, the estimation would carve into concavities and reach a better estimate than the occluder's visual hull. A somewhat larger volume is reached with both schemes in this example. This is attributable to calibration errors which over-tightens the visual hull with respect

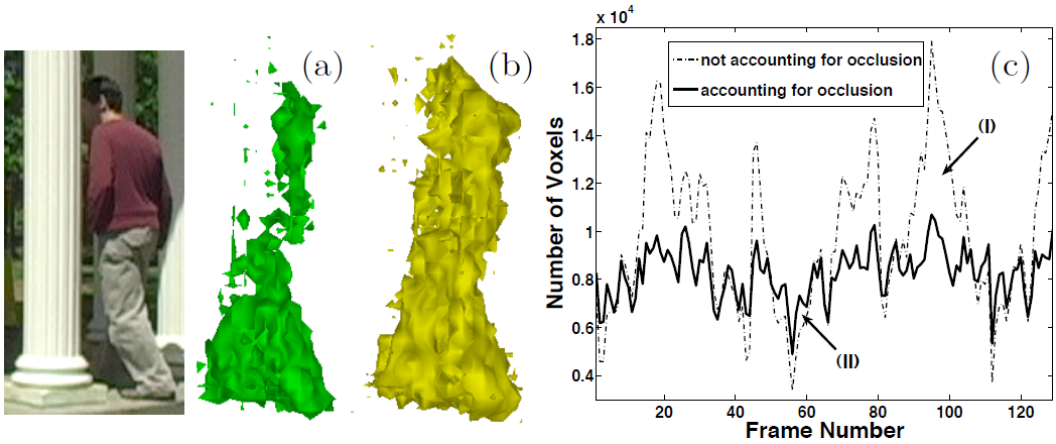

Figure 3.8: (a) Person shape estimate from PILLAR sequence, as occluded by the rightmost pillar and computed without accounting for occlusion. (b) Same situation accounting for occlusion, showing better completeness of the estimate. (c) Volume plot in both cases. Accounting for occlusion leads to more stable estimates across time, decreases false positives and overestimates due to shadows cast on occluders (I), increases estimation probabilities in case of occlusion (II).

to the true silhouettes, and accumulation of errors in both schemes toward the end of the sequence. Those are traced to the redundant, periodical poses contained in the video, that sustain consistent noise (*e.g.* the person periodically walks close to the pillar). This suggests the existence of an optimal finite number of frames to be used for processing. Jolts can be observed in both volumes corresponding to instants where the person walks behind the pillar, thereby adding positive contributions to the inference. Use of the reliability criterion contributes lowers the sensitivity to noise, and gives a conservative estimate of the occluder volume as the curves show in frames 100-200. Raw inference Eq. 3.10 momentarily yields large hypothetical occluder volumes when data is biased toward contributions of an abnormally low subset of views (frame 109).

3.4.4 Accounting for Occlusion in Dynamic Shape Inference

The formulation of Section 3.3.6 can be used to account for the accumulated occluder information in dynamic shape inference. Only occlusion cues from reliable voxels ($R > 0.8$) are used to minimize false positive occluder estimates, whose excessive presence would

lead to sustained errors. While in many cases the original dynamic object formulation from Section 2.4.2 performs robustly, a number of situations benefit from the additional occlusion knowledge (Fig. 3.8). Person volume estimates can be obtained when accounting for occluders. These estimates appear on average to be a stable multiple of the real volume of the person, which depends mainly on camera configuration. This suggests a possible biometrics application of the method, for disambiguation of person recognition based on computed volumes.

3.5 GPU Acceleration

Since the scene involves dynamic activities, ideally the algorithm should be real-time. But it is not achieved on CPU implementation. The optimized CPU version of the dynamic shape volume computation algorithm runs 11.1317 sec per time instant for a grid size of 128^3, and 9 camera views of 720 by 480 RGB images on a AMD Athlon^{TM} 64x2 Dual Core 4800, 2.41GHz, 2.0G RAM machine.

However, one may have noticed that most of the computation process is the same for every voxel. Intuitively, this means the algorithm can be parallelized. In this section, a GPU implementation based on CUDA unified pipeline version on nVIDIA's G80 graphics hardware is introduced.

3.5.1 Algorithm Analysis

Foreground Inference

Background color RGB Gaussian model has to be trained for every pixel in a camera view in advance. Assume that the background stays the same during the capturing, for every time instant, given the background model and a new image frame, the silhouette probability of a pixel is computed. Using this information from all views, the posterior probability of a 3D space voxel to be occupied by a dynamic object can be inferred. In

fact, voxel occupancy probability can be reliably computed from its corresponding pixels along the camera-viewing ray. Therefore, assuming neighboring voxel occupancies are independent, the inference procedure is the same for every voxel.

Occluder Inference

After the dynamic shape is computed, the occlusion events at every voxel position are examined by looking at inconsistency between the computed dynamic shape volume and the silhouette information from the background models at image views. As discussed before, this inconsistency happens when the dynamic objects has been occluded in the same view, while some other views still give positive information for dynamic shape occupancy. For a voxel in the occluder volume, again, only the corresponding camera-viewing rays need to be examined. The major difference is that one needs to know the maximum values of dynamic shape occupancy probability along the viewing ray in the direction forwards and backwards from the voxel being examined. The examination requires view-dependent ordering, which is the most challenging part for parallelization. Once the peak information is computed, the rest is almost the same information fusion process as dynamic shape volume inference. A merging of the accumulated occluder computed at each time instant is needed to get a final grid, which is again very easy to parallelize.

Algorithm Complexity Analysis

In Section 3.3.5, a term R is introduced for every occluder voxel to model how reliable its value already is, given the inference up to the current time instant. The CPU implementation complexity chart is given in Fig. 3.9. For the current GPU version this term is not implemented yet, and according to the chart, it does not affect the total complexity of the algorithm and can be added easily too.

The CPU version is bounded by $O(fnN^3)$, where f is the number of frames, n is the

3D OCCLUSION INFERENCE ALGORITHM:			**O($3fnN^3$)**
SET UP CAMERAS (projection matrix and background model)			**O(n)**
For every time instant			**O(f)**
For every voxel		**G** Grid	**O(nN^3+N^3)**
For every camera view			**O(n)**
Compute the probability inference from image and background			**O(1)**
Compute the Bayesian posterior probability of the voxel existence			**O(1)**
For every camera view		**O** Grid	**O($3nN^3$)**
Compute the peak-in-front volume & the peak-behind volume			**O($2N^3$)**
For every occluder voxel			**O(N^3)**
Compute the inference of occlusion for current frame			**O(1)**
Compute the temporary value for reliability of occluder for current voxel			**O(1)**
For every voxel			**O($2N^3$)**
Compute the inference of occlusion over the complete stack of past frames			**O(1)**
Compute the reliability of occluder for current voxel			**O(1)**
Display the Volumes			**O(1)**

Figure 3.9: CPU occupancy grid algorithm complexity analysis, including both dynamic shape and static occluder computation.

number of cameras, N is the side length. Most of the computations are on the voxels, which makes GPU parallelization feasible. It is unlikely that all temporary volumes can be stored in memory, which means one might need to re-design the data flow for GPU implementation. The most time-consuming process is the "peak-finding" in the occluder grid computation step, which takes $O(2nN^3)$ time complexity for every time instant.

Peak Finding—Brute force method

For every voxel, the brute-force algorithm would traverse the viewing ray for all n camera views, which takes nN times, therefore the whole algorithm takes $O(nN^4)$. This algorithm is very slow, but because it is implemented on a voxel basis, it takes the advantage of parallelization. One definitely can make this implementation together with occluder probability inference in a single function, thus reduce the data transfer time between the CPU and GPU. This algorithm takes the camera projection matrices, foreground volume, pre-computed background probability images from all cameras as input and computes two accumulating values, namely Eq. 3.10, and the final occluder

probability as output. However, the implementation shows that it takes more than 4 minutes to compute one time instant, which is much slower than optimized CPU version. Therefore, it is definitely not acceptable for a real-time solution.

Peak Finding—Divide and conquer method

What is actually implemented in the final GPU version is splitting the peak finding process and the occluder inference process. More specifically, one can pre-compute two volumes storing "peak-in-the-front" and "peak-behind" values for each voxel from one camera direction, compute the intermediate marginalization probability result in a temporary volume, and move on to next camera direction. For each direction, a 2D image is used to store the maximum value along the viewing ray so far has been swept, and sequentially test 2D slices along the direction in the 3D volume against this 2D image. While this reduces the time complexity to $O(N^3)$, two 2D images have to be kept to store the current "peak-in-the-front" and "peak-behind" values when the sweeping plan traveling the 3D volume in the "front-to-back" and "back-to-front" order with respect to the camera direction. Four more volumes are also needed to store the temporary probability result, since the algorithm is computing every camera view separately first and merging them in the final step.

Peak Finding—Cache-friendly divide and conquer method

Since the plane sweeping direction depends on the camera view orientation, for a certain camera view, the plane sequential value access may be not local at all, for which the operating system may be constantly transferring data pieces in and out of the cache. This actually has a huge impact on the speed of the peak finding. From the "Peak finding analysis" in Section 3.5.3, one can see that it might take about 2 times more to complete the peak finding process for a cache-unfriendly direction than a cache-friendly one as CPU implementation. However, since the cache-friendly CPU version requires ordered

traversal, which prevents parallelization, the GPU version cannot really benefit from it. Therefore, the final GPU implementation goes with the cache-unfriendly version as described in the previous section. However, there might still have room in this direction for speedup.

3.5.2 GPGP Solution

nVIDIA's G80 and above graphics cards plus CUDA provides a programming environment similar to traditional C programming language to code algorithms for execution on the GPU. The divide-and-conquer method data flow is implemented as in Fig. 3.10.

Input data as texture

In CUDA, various types of memories can be accessed by the hardware with different access time. Texture memory is cached, so a texture sampling costs one memory read from device only on a cache miss. However, texture memory is read only. Therefore, it should be used to store constant values that are used frequently. Therefore the camera projection matrices are assigned into textures, because these parameters are used for every voxel location, and their values remain un-changed during the whole computation. This immediately doubles the speed of dynamic shape computation.

The most recommended feature of CUDA is its shared memory mechanism. Shared memory can be accessed much faster than global memory. However, in terms of data size and thread interaction, no place in the implementation would possibly benefit from this mechanism. This may be a direction of further improvement though.

Intermediate result handling

The function "Peak Finding" assigns values for a 2D slice of 3D volume, and is called N times to complete the peak volume for a certain camera direction. The computed slices are stored on GPU and are not read back to CPU. For later "After Peak Finding"

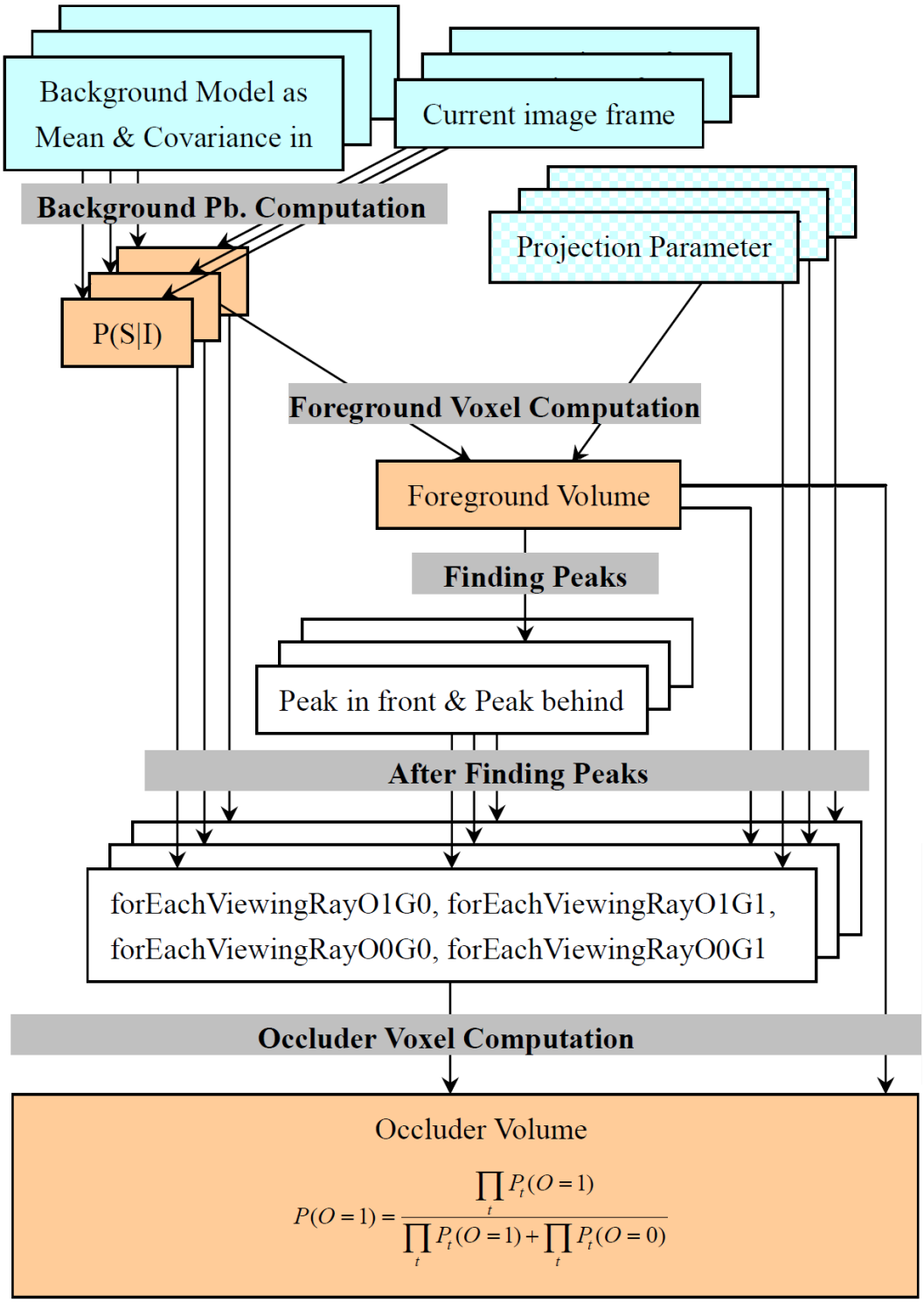

Figure 3.10: The data flow for GPGP CUDA version, including foreground and occlusion inference. The light blue color indicates inputs, of which the checker box pattern indicates inputs as textures. The pink color indicates outputs. Gray boxes are function names, and the white boxes are intermediate results. **Best viewed in color.**

function and "Occluder Voxel Computation" function, the results are directly read from GPU and thus do not send the peak volumes from CPU to GPU again. This also gains a speedup from 2.5 second per time instant to 1.8 second per time instant.

3.5.3 Result and Comparison

Throughput and Timing

The optimized CPU version of the algorithm runs 29.1317 second per time instant for a grid size of 128^3, and 9 camera views of 720 by 480 RGB images on an AMD Athlon 64x2 Dual Core 4800, 2.41GHz, 2.0GB RAM machine on single thread.

CPU read time 0.48722 sec. Camera Num*height*width*RGB*byte =9*480*720*3*1=9331200 Bytes CPU write time 1.31966 sec. Volume*4bytes = 128*128*128*4 = 8388608 Bytes Total CPU read & write time 1.8 sec. CPU throughput 17719808 Bytes / 1.368382 sec. = 12949460 Bytes / sec. = 12.94 MB/sec.	
GPU read time 0.010261 sec. Camera Num*height*width*RGB*byte =9*480*720*3*1=9331200 Bytes GPU write time 0.009859 sec. Volume*4bytes = 128*128*128*4 = 8388608 Bytes Total GPU read & write time 0.02012 sec. GPU throughput 17719808 Bytes / 0.02012 sec. = 880706163 Bytes / sec. = 880.70 MB/sec.	
Grid size (480/20, 720*9/20), Block size (20,20)	**Background prb** computation 0.020970 sec.
Grid size (128/8, 128*128/8), Block size (8,8)	**Foreground volume** computation 0.051701 sec.
Grid size (128/8, 128*128/8), Block size (8,8)	**After peak finding** computation 0.025066 sec.
Grid size (128/8, 128*128/8), Block size (8,8)	**Occluder volume** computation 0.011499 sec.

Figure 3.11: Chart of throughput and timing.

The chart of Fig. 3.11 shows the statistics of throughput and function computation time. The CUDA version is tested on an Intel Core2 6600, 2.40GHz, 2.0GB RAM. The CPU writing to the hard disc is one of the main bottlenecks in data transferring for now. fread/fwrite in C are used to perform the file reading/saving. It takes almost 2 times more for saving than reading. More analysis should be addressed for this issue. For improvements, one can do a cached streaming for reading and a separate CPU thread

for compressing and output. The GPU data bandwidth is around 70 times faster than CPU. The CPU/GPU data transfer only takes 0.02 second to complete. Therefore, one can ignore this part of the overhead for now. Deeper analysis should be on how to achieve the best result by changing block/grid sizes [nVidia (2009)]. Currently, for silhouette probability image pre-computation, the block size cannot be set over 30 by 30, and since the image size is 720 by 480, the block size is set to be 20. For dynamic shape volume computation, the card will not load the executable file if the block size is set to 16 by 16, so 8 by 8 is used for the 128^3 volume.

Foreground computation on different hardware architectures

The maximum performances of the pure foreground computation are listed as follows. In this part, the CPU version and CUDA version are compared, together with a traditional GPU version (vertex-fragment-shader style) of foreground computation. The tests are all performed on an AMD Athlon 64x2 Dual Core 4800, 2.41GHz, 2.0GB RAM machine. This has included the CPU to GPU and GPU to CPU data transfer time. From Fig. 3.12, one can see that CUDA version for the foreground computation is the best among the three.

CPU Version	11.132 sec./frame	-
Traditional GPU version	0.263 sec./frame	43.327 times as CPU version
CUDA version (Block size 8 by 8)	0.124 sec./frame	89.774 times as CPU version

Figure 3.12: Performance chart of dynamic shape computation with CPU and GPU versions.

The reason that CUDA version is twice as fast as the traditional GPU version is not fully investigated. However, the unified shader pipeline tends to simplify a lot of the redundancy in setting up the input and output formats and the function calls.

Peak finding analysis

The brute-force algorithm takes around 4 minutes to finish the peak finding for 9 cameras, so no further discussion is addressed on this $O(nN^4)$ algorithm. The divide-and-conquer method reduces the time to around 1.8 second per time instant on an Intel Core2 6600, 2.40GHz, 2.0GB RAM machine. However, as shown in Fig. 3.13 of the 9 camera views, depending on the specific camera orientation, it is almost 3.5 times to compute a z direction sweeping path than an x direction one. This has everything to do with the caching missing in z direction sweeping on GPU. In other words, depending on the camera orientations, the method might take 0.9 seconds to 3.15 seconds to run the peak finding for 9 cameras.

direction	-x	-x	-z	-z	-z	-z	x	x	x
Time (ms)	96.57	102.68	341.59	337.01	325.63	329.36	104.64	98.56	98.41

Figure 3.13: GPU occluder computation peak finding analysis.

3.5.4 GPU implementation summary

Dynamic shape volume inference from multiple camera videos has been implemented based on the new unified shader nVIDIA G80 pipeline. The complete algorithm gets a speedup of around 15 times. The dynamic shape computation alone reaches a speed-up of more than 80 times and a frame-rate of 0.2 second per frame on the test-machines, which is already satisfactory for real-time applications.

Although the static occluder computation is far from real time yet (0.9 seconds to 3.15 seconds per time instant), as a complete dynamic scene modeling system, this speedup is already feasible. This is because the static occluder inference is based on dynamic shape occluding cues. Within the time of seconds, the dynamic shape has not changed its location much yet, therefore the occluding cue is redundant for occluder computation anyway.

Future works mainly follow the direction of CUDA architecture in a more cache-friendly manner. Grid-size/block-size influence and shared memory usage for peak finding in the occlude volume computation stage also require further analysis. Structures that are more delicate can be used for background probability computation. For example, one really does not need to pre-compute the complete images because the 3D volume may not reach all parts of the image. Another direction is to make more use of the cached texture as the foreground volume, this might alleviate the bottleneck of peak finding, although the cache misses still exist. In short, as the development of the new parallel processing hardware, such as true 3D grid computation, double precision accuracy, etc. the real-time implementation of the algorithm will finally be achieved. After integration with volume rendering and video capturing pipeline, the ultimate goal of an automatic/semi-automatic real-time dynamic scene analysis system can be achieved.

3.6 Further Discussion

Properties of Static Occluder Shape

The computed occluder shape is in a probabilistic form. Its counterpart in the deterministic representation is given in Fig. 3.2. Since it is formed by carving away dynamic shapes, it has some unique properties that are different from the traditional visual hull. Consider a dynamic shape D with infinitesimal volume. I define a "visibility Hull" as an approximation volume to a static occluder recovered with a infinitesimal volume dynamic shape D moving randomly about for long enough time that the recovered shape does not change anymore. It can be shown that the visibility hull of the occluder is *the combination of regions that only one camera or no camera can see, where the dynamic object D's visual hull cannot be recovered.* The actual occluder region also belongs to the "no camera visible region". In Fig. 3.14 (a) and (b), the thick black lines delineate the visibility hull. In comparison, in Fig. 3.14 (c) and (d), the thick black lines delineate

the visual hull of the occluder, supposing the silhouette of the object is known. The visual hull is the region formed by intersecting the back-projections of all the cameras' silhouettes.

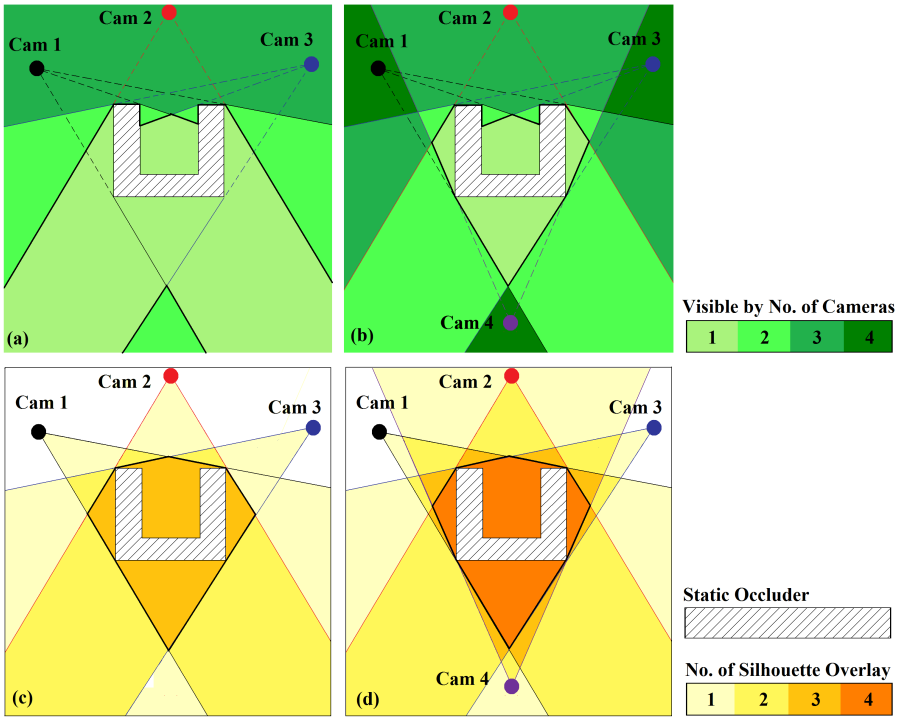

Figure 3.14: 2D theoretical visibility hull and visual hull. (a) 3 camera visibility hull; (b) 4 camera visibility hull; (c) 3 camera visual hull; (d) 4 camera visual hull. Concavities can be recovered by visibility hull. **Best viewed in color.**

Fig. 3.14 shows that unlike the visual hull, the visibility hull can recover concavities. In fact, when cameras are distributed all over space, the actual shape of an arbitrary static occluder can be recovered. (In fact, it is a sufficient but not necessary condition, finite number of cameras sometimes may also be capable of this.) However, the visibility hull shape is highly dependent on the camera placement. As (a) shows, the visibility hull may even not be closed. For visibility hull, there is no lower bound number to guarantee the closed shape. Although the visibility hull in (b) is closed, if the fourth camera changes its orientation or position, this may be open again. On the contrary, only two silhouettes from different views can guarantee a closed visual hull, which is the minimum number of cameras required for a visual hull.

Given the above analysis, some empirical requirements for good quality occluder estimation are summarized as follows:

- There is no guarantee that how many cameras would produce closed occluder shape. But when the size of the occluder is small relative to the camera focal length, or the occluder position is far enough from the cameras, so that the region where only one camera can see the dynamic shape is limited, a closed occluder shape can usually be recovered through the algorithm in this chapter.

- For a region behind the occluder, where no camera view has sampled, the algorithm cannot infer any information. For example, the algorithm does not recover the wall, for a person is hiding completely behind it. In this case, the person's occupancy is not recovered in the first place. One solution may be to add more camera views behind the wall.

- Since the closed dynamic shape is required (needs at least two camera views), plus an occluded view for the occluding incidence, at least three cameras are required for the occluder inference in theory.

What Happens with Multiple Dynamic Shapes?

In the formulation, all dynamic objects are treated as a single object as being occupied by $\mathcal{G}$. This may introduce ambiguity when objects go too close to one another. But even with such additional noise, the algorithm is still robust enough to recover the occluder probability volume correctly. This is mainly due to the automatic correction of information with accumulation over time. Fig. 3.15 shows a two people result of the SCULPTURE dataset. In the next chapter, an algorithm is proposed to refine the dynamic shape estimation with multiple people scenario.

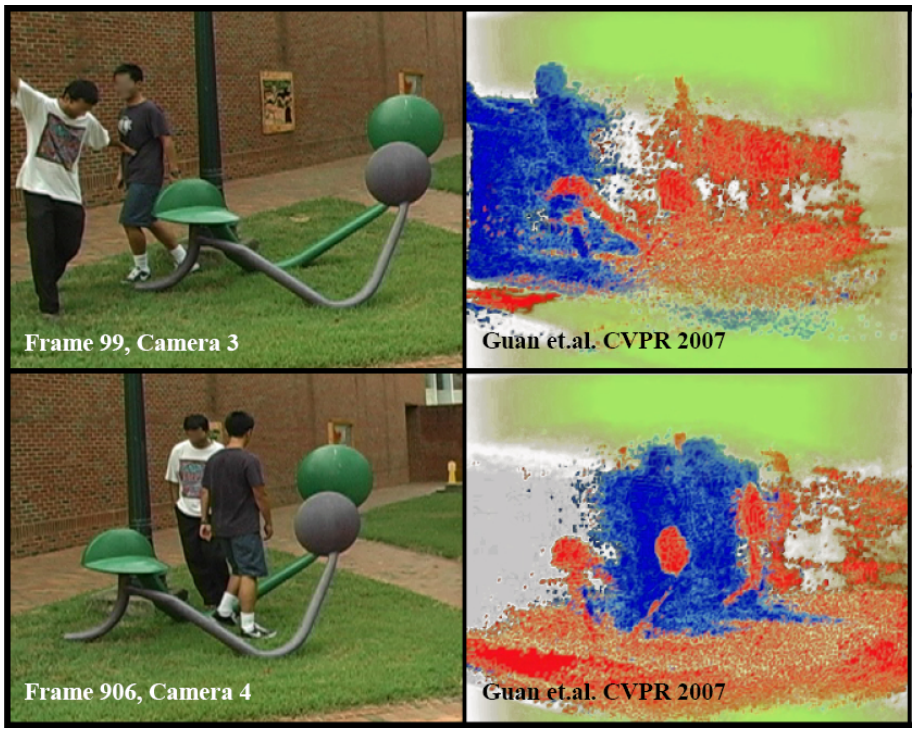

Figure 3.15: SCULPTURE occluder recovered from two people sequence.

3.6.1 Conclusion

In this chapter, a method has been proposed to detect and build the dense 3D shape of occluders indirectly observed through the motion of dynamic objects in a scene, in calibrated videos obtained from multiple views. The proposed Bayesian sensor formulation provides a useful probabilistic occluder representation, enabling detection and online accumulation of occluder information, and cooperative estimation of occluder and dynamic object shapes. The prposed framework is robust to noise and avoids hard decisions about scene state. This new approach can lead to promising applications. Shape-from-occlusion could prove useful in conditions where segmenting objects is difficult or does not make sense, and using a moving object is easier, when all cameras do not have a complete view of the occluder for example. Visual media such as infrared images exhibiting cold static objects of interest, inseparable from a broader cold background, could be used for modeling using a third, warm moving object. Detecting occlusions

using this method can be helpful for a number of vision problems related to modeling, not limited to silhouette-based approaches. Many extensions are possible, such as automatic detection of changes in occluder configuration, cooperative background color model updates and occlusion estimation, and integration of other cues such as color and texture.

Chapter 4

Multiple Dynamic Shape Modeling and Tracking

In the previous chapter, we introduced an algorithm to recover a static occluder given an occluding event with dynamic object(s). Occlusions may also occur between two or more dynamic objects, as shown in Fig. 3.15. With the increase of such "inter-occlusions", the discriminatory power of the silhouettes decreases, resulting in the reconstructed shapes much larger in volume than the real objects. In fact, when multiple dynamic objects clutter the scene, the visibility ambiguity in general increases, no matter if two dynamic objects are occluding each other or if they are well-separated.

Most shape-from-silhouette techniques use a bianry-classification of space occupancy and silhouettes, based on image regions that match or disagree with a static background appearance model. Binary silhouette information becomes insufficient to unambiguously carve 3D space regions as the number and density of dynamic objects increases. In such difficult scenes, multi-view stereo methods suffer from visibility problems, and rely on color calibration procedures difficult to apply outdoors. In this chapter a new algorithm is proposed to automatically detect and reconstruct scenes with a variable number of dynamic objects. The formulation distinguishes between m different shapes in the scene by using automatically learnt view-specific appearance models, eliminating the color calibration requirement. Bayesian reasoning is then applied to solve the m-shape

occupancy problem, with m updated as objects enter or leave the scene. Results show that this method yields multiple silhouette-based estimates that drastically improve scene reconstructions over traditional two label silhouette scene analysis. This enables the method to also efficiently deal with multi-person tracking problems.

4.1 Intuition and Related Works

The ability of visual hull algorithms to capture the dynamic scenes degrades as the number of objects in the scene increases. In such cases the binary silhouettes are ambiguous in distinguishing between regions actually occupied by objects and unfortunate silhouette-consistent "ghost" regions. Such regions have been analyzed in the context of tracking applications to avoid committing to a "ghost" track [Otsuka and Mukawa (2004)]. The method proposed in this chapter casts the problem of silhouette modeling at the multi-object level, where ghosts can naturally be eliminated based on per-object silhouette consistency. Multi-object silhouette reasoning has been applied in the context of multi-object tracking [Mittal and Davis (2003); Fleuret et al. (2007)]. The reconstruction and occlusion problem has also been studied for the specific case of transparent objects [Broadhurst et al. (2001)]. Recent tracking efforts also use 2D probabilistic occlusion reasoning to improve object localization [Gupta et al. (2007)]. The algorithm introduced in this chapter based on [Guan et al. (2008b)] is more general as it estimates full 3D shapes and copes with 3D occlusions, both dynamic and static.

Perhaps the closest related work is the approach of [Ziegler et al. (2003)], which builds 3D models deterministically from multi-label user-provided silhouette segmentations. The approach discussed in this chapter produces a more general probabilistic model that accounts for process noise and requires little or no user intervention.

The ghost phenomenon occurs when the configuration of the scene is such that regions of space occupied by objects of interest cannot be disambiguated from free-space

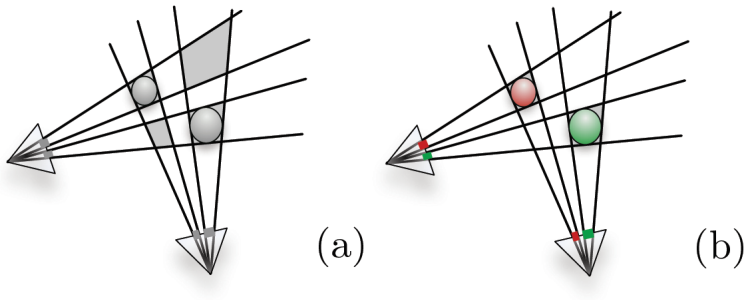

Figure 4.1: The principle of multi-object silhouette reasoning for shape modeling disambiguation. **Best viewed in color.**

regions that also happen to project inside all silhouettes, as the polygonal gray region in Fig. 4.1(a). Ghosts are increasingly likely as the number of observed objects rises, because it then becomes more difficult to find views that visually separate objects in the scene and carve out unoccupied regions of space. This problem is even aggravated for robust probabilistic occupancy scheme as described in Section 2.4.2 [Franco and Boyer (2005)], which do not strictly require silhouettes to be observed in every view. To address this problem, a set of view-specific appearance models associated to m objects in the scene is initialized and learned. The intuition is then that the probability of confusing ambiguous regions with real objects decreases, because the silhouette set corresponding to ghosts is then drawn from non-object-consistent appearance model sets, as in Fig. 4.1(b).

It is possible to process multiple silhouette labels in a deterministic, purely geometric fashion [Ziegler et al. (2003)], but this comes at the expense of an arbitrary hard threshold for the number of views that define consistency. Silhouettes are then also assumed to be manually given and noiseless, which is not realistic for automatic processing. Using a volume representation of the 3D scene, multi-object sequences are thus processed by examining each voxel in the scene using a Bayesian formulation (Section 4.2), which encodes the noisy causal relationship between the voxel and the pixels that observe it in a generative sensor model. In particular, given the knowledge that a voxel is occupied by a certain object among m possible in the scene, the sensor model defines

the appearance distribution corresponding to that object. It also encodes state information about the viewing line and potential obstructions from other objects, as well as a localization prior used to enforce the compactness of objects, which can be used to refine the estimate for a given instant of the sequence. The proposed method can be seen as a multi-object generalization of previous probabilistic approaches focused on 2-label silhouette modeling in Section 2.4.2 and Chapter 3.

This scheme enables silhouette inference in Section 4.2.3 in a way that reinforces regions of space which are drawn from the same conjunction of color distributions, corresponding to one object, and penalizes inconsistent regions, while accounting for object visibility. An algorithm in Section 4.3 is then proposed to integrate the inference framework in a fully automatic system. Because they are mutually dependent, specific steps are proposed for the problems of initialization, appearance model estimation, multi-object and occluder shape recovery.

4.2 Formulation

Consider a single time instant in this section. With a scene observed by n calibrated cameras, a maximum of m dynamic objects of interest can be present. Let us focus on the state of one voxel at position X chosen among the positions of the 3D lattice used to discretize the scene. Assuming that a static appearance model for the background has previously been observed, one can model how knowledge about the occupancy state of voxel X influences image formation. Because of occlusion relationships arising between objects, the zones of interest to infer the state of voxel X are its n viewing lines $\mathbf{L}_i$, $i \in \{1, ..., n\}$, with respect the different views. Assume that some prior knowledge about scene state is available for each voxel X in the lattice and can be used in the inference. Various uses of this assumption will be demonstrated in Section 4.3. A number of statistical variables are used to model the state of the scene, the image generation

process and to infer $\mathcal{G}$, as in figure Fig. 4.2.

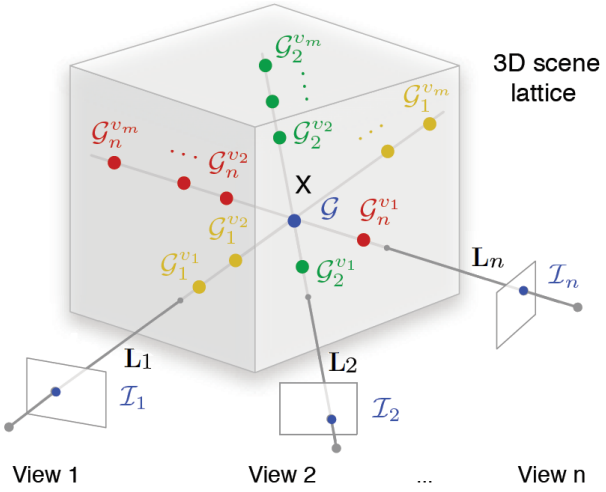

Figure 4.2: Overview of main statistical variables and geometry. $\mathcal{G}$ is the occupancy at voxel X and lives in a state space $\mathcal{L}$ of object labels. $\{\mathcal{I}_i\}$ are the color states observed at the n pixels where X projects. $\{\mathcal{G}_i^{v_j}\}$ are the states in $\mathcal{L}$ of the most likely obstructing voxels on the viewing line, for each of the m objects, enumerated in their order of visibility $\{v_j\}_i$.

4.2.1 Statistical Variables

Scene voxel state space

The occupancy state of X is represented by the variable $\mathcal{G}$. The difference of the modeling from the previous chapter lies in the multi-labeling characteristic of $\mathcal{G} \in \mathcal{L}$, where $\mathcal{L}$ is a set of labels $\{\emptyset, 1, ..., m, u\}$. A voxel is either empty ($\emptyset$), one of m objects the model is keeping track of (numerical labels), or occupied by an unidentified object (u). u acts as a default label capturing all objects that are detected as different than background but not explicitly modeled by other labels, which proves useful for automatic detection of new objects (Section 4.3.3). The notation $\mathcal{G}$ is an extension of the previous chapters.

Observed appearance

The voxel X projects to n camera views, with the projected pixel color denoted by $\mathcal{I}_i$ $i \in \{1, ..., n\}$. Assume these colors are drawn from a set of object-and-view-specific color models whose parameters noted as $\mathcal{F}_i^l$, where $l \in \mathcal{L}$. More complex appearance models are possible using gradient or texture information, without loss of generality.

Latent viewing line variables

To account for inter-object occlusion, it is necessary to model the contents of viewing lines and how they contribute to image formation. Assuming some *a priori* knowledge about where objects lie in the scene is known, the presence of such objects can have an impact on the inference of $\mathcal{G}$ because of the visibility of objects and how they affect $\mathcal{G}$. Intuitively, conclusive information about $\mathcal{G}$ cannot be obtained from a view i if a voxel in front of $\mathcal{G}$ with respect to view i is occupied by another object, for example. However, $\mathcal{G}$ directly influences the color observed if it is unoccluded and occupied by one of the objects. But if $\mathcal{G}$ is known to be empty, then the color observed at pixel $\mathcal{I}_i$ reflects the appearance of objects behind X in image i, if any. These visibility intuitions are similar to the ones in Chapter 3, and are formalized in detail below (Section 4.2.2).

It is not meaningful to account for the combinatorial number of occupancy possibilities along the viewing rays of X. This is because neighboring voxel occupancies on the viewing line usually reflect the presence of the same object and are therefore correlated. In fact, assuming no more than one instance of every one of the m objects along the viewing line is witnessed, the fundamental information that is required to reason about X is the knowledge of presence and ordering of the objects along this line. To represent this knowledge, as depicted in Fig. 4.2, assuming prior information about occupancies is already available at each voxel, for each label $l \in \mathcal{L}$ and each viewing line $i \in \{1, ..., n\}$, one can extract the voxel whose probability of occupancy is dominant for that label on the viewing line. This corresponds to electing the voxels which best represent the

m objects and have the most influence on the inference of $\mathcal{G}$, and generalizes the peak influence idea of Chapter 3. To account for this knowledge in the problem of inferring X, a set of statistical occupancy variables $\mathcal{G}_i^l \in \mathcal{L}$ is introduced, corresponding to these extracted voxels. Since only one time instant is considered, the superscript of $\mathcal{G}_i^l$ does not have the same temporal meaning as in Chapter 3.

4.2.2 Dependencies

Several simplifications can be considered in the joint probability distribution of the set of variables, that reflect the prior knowledge about the problem. To simplify the writing, the conjunction of a set of variables is noted as following: $\mathcal{G}_{1:n}^{1:m} = \{\mathcal{G}_i^l\}_{i\in\{1,\ldots,n\},l\in\{1,\ldots,m\}}$. The following decomposition for the joint probability distribution $p(\mathcal{G}, \mathcal{G}_{1:n}^{1:m}, \mathcal{I}_{1:n}, \mathcal{F}_{1:n}^{1:m})$ is proposed:

$$p(\mathcal{G}) \prod_{l\in\mathcal{L}} p(\mathcal{F}_{1:n}^l) \prod_{i,l\in\mathcal{L}} p(\mathcal{G}_i^l|\mathcal{G}) \prod_i p(\mathcal{I}_i|\mathcal{G}, \mathcal{G}_i^{1:m}, \mathcal{F}_i^{1:m}). \tag{4.1}$$

The following is a detailed explanation of the above expression:

Prior terms $p(\mathcal{G})$ carries prior information about the current voxel. This prior can reflect different types of knowledge and constraints already acquired about $\mathcal{G}$, *e.g.* localization information to guide the inference (Section 4.3). $p(\mathcal{F}_{1:n}^l)$ is the prior over the view-specific appearance models of a given object l. The prior, as written over the conjunction of these parameters, could express expected relationships between the appearance models of different views, even if the cameras are not color-calibrated. Since the focus in this chapter is on the learning of voxel X, this capability is not used here and we assume $p(\mathcal{F}_{1:n}^l)$ to be uniform.

Viewing line dependency terms The prior information along each viewing line is represented using the m voxels most representative of the m objects, so as to model

inter-object occlusion phenomena. However, when examining a particular label $\mathcal{G} = l$, keeping the occupancy information about $\mathcal{G}_i^l$ would lead to account for intra-object occlusion phenomena, which in effect would lead the inference to favor mostly voxels from the front visible surface of the object l. Because it is intended to model the *volume* of object l, the influence of $\mathcal{G}_i^l$ is discarded, when $\mathcal{G} = l$:

$$p(\mathcal{G}_i^k|\{\mathcal{G} = l\}) = \mathcal{P}(\mathcal{G}_i^k) \qquad \text{when } k \neq l \tag{4.2}$$

$$p(\mathcal{G}_i^l|\{\mathcal{G} = l\}) = \delta_\emptyset(\mathcal{G}_i^l) \qquad \forall l \in \mathcal{L}, \tag{4.3}$$

where $\mathcal{P}(\mathcal{G}_i^k)$ is a distribution reflecting the prior knowledge about $\mathcal{G}_i^k$, and $\delta_\emptyset(\mathcal{G}_i^k)$ is the distribution giving all the weight to label $\emptyset$. In Eq. 4.3, $p(\mathcal{G}_i^l|\{\mathcal{G} = l\})$ is thus enforced to be empty when $\mathcal{G}$ is known to be representing label l, which ensures that the same object is represented only once on the viewing line.

Image formation terms The image formation term $p(\mathcal{I}_i|\mathcal{G}, \mathcal{G}_i^{1:m}, \mathcal{F}_i^{1:m})$ explains what color one expects to observe given the knowledge of viewing line states and per-object color models. Each such term is decomposed into two sub-terms by introducing a local latent variable $\mathcal{S} \in \mathcal{L}$ representing the hidden silhouette state:

$$p(\mathcal{I}_i|\mathcal{G}, \mathcal{G}_i^{1:m}, \mathcal{F}_i^{1:m}) = \sum_{\mathcal{S}} p(\mathcal{I}_i|\mathcal{S}, \mathcal{F}_i^{1:m}) p(\mathcal{S}|\mathcal{G}, \mathcal{G}_i^{1:m}). \tag{4.4}$$

The term $p(\mathcal{I}_i|\mathcal{S}, \mathcal{F}_i^{1:m})$ simply describes what color is likely to be observed in the image given the knowledge of the silhouette state and the appearance models corresponding to each object. It generalizes the $p(\mathcal{I}|\mathcal{S}, \mathcal{B}, \mathcal{F})$ term of Section 2.4.2 for multiple objects. $\mathcal{S}$ acts as a mixture label: if $\{\mathcal{S} = l\}$ then $\mathcal{I}_i$ is drawn from the color model $\mathcal{F}_i^l$. For objects ($l \in \{1, ..., m\}$) Gaussian Mixture Models (GMM) [Stauffer and Grimson (1999)] is typically used to efficiently summarize the appearance information of dynamic object silhouettes. For background ($l = \emptyset$), per-pixel Gaussians are used as learned from

pre-observed sequences, although other models are possible. When $l = u$ the color is drawn from the uniform distribution, as no assumption about the color of previously unobserved objects is known.

Defining the silhouette formation term $p(\mathcal{S}|\mathcal{G}, \mathcal{G}_i^{1:m})$ requires that the variables be considered in their visibility order, to model the occlusion possibilities. Note that this order can be different from $1, ..., m$. $\{\mathcal{G}_i^{v_j}\}_{j\in\{1,...,m\}}$ denotes the variables $\mathcal{G}_i^{1:m}$ as enumerated in the permuted order $\{v_j\}_i$ reflecting their visibility ordering on $\mathbf{L}_i$. If $\{g\}_i$ denotes the particular index after which the voxel X itself appears on $\mathbf{L}_i$, then the silhouette formation term can be re-written as $p(\mathcal{S}|\mathcal{G}_i^{v_1} \cdots \mathcal{G}_i^{v_g}\ , \mathcal{G},\ \mathcal{G}_i^{v_{g+1}} \cdots \mathcal{G}_i^{v_m})$. A distribution of the following form can then be assigned to this term:

$$p(\mathcal{S}|\emptyset \cdots \emptyset\ l\ *\cdots*) = d_l(\mathcal{S}) \quad \text{with } l \neq \emptyset \tag{4.5}$$

$$p(\mathcal{S}|\emptyset\ \cdots\cdots\ \emptyset) = d_\emptyset(\mathcal{S}),, \tag{4.6}$$

where $d_k(\mathcal{S}), k \in \mathcal{L}$ is a family of distributions giving strong weight to label k and lower equal weight to others, determined by a constant probability of detection $P_d \in [0, 1]$: $d_k(\mathcal{S} = k) = P_d$ and $d_k(\mathcal{S} \neq k) = \frac{1-P_d}{|\mathcal{L}|-1}$ to ensure summation to 1. Eq. 4.5 thus expresses that the silhouette pixel state reflects the state of the first visible non-empty voxel on the viewing line, regardless of the state of voxels behind it ("*"). Eq. 4.6 expresses the particular case where no occupied voxel lies on the viewing line, the only case where the state of $\mathcal{S}$ should be background: $d_\emptyset(\mathcal{S})$ ensures that $\mathcal{I}_i$ is mostly drawn from the background appearance model in Section 4.4.1.

4.2.3 Inference

Estimating the occupancy at voxel X translates to estimating $p(\mathcal{G}|\mathcal{I}_{1:n},\ \mathcal{F}_{1:n}^{1:m})$ in Bayesian terms. Bayes' rule is applied using the joint probability distribution, marginalizing out

the unobserved variables $\mathcal{G}_{1:n}^{1:m}$:

$$p(\mathcal{G}|\mathcal{I}_{1:n}\ \mathcal{F}_{1:n}^{1:m}) = \frac{1}{c}\sum_{\mathcal{G}_{1:n}^{1:m}} p(\mathcal{G},\ \mathcal{G}_{1:n}^{1:m},\ \mathcal{I}_{1:n},\ \mathcal{F}_{1:n}^{1:m}) \tag{4.7}$$

$$= \frac{1}{c}p(\mathcal{G})\prod_{i=1}^{n} f_i^1 \tag{4.8}$$

$$\text{where } f_i^k = \sum_{\mathcal{G}_i^{v_k}} p(\mathcal{G}_i^{v_k}|\mathcal{G}) f_i^{k+1} \quad \text{for } k < m \tag{4.9}$$

$$\text{and } f_i^m = \sum_{\mathcal{G}_i^{v_m}} p(\mathcal{G}_i^{v_m}|\mathcal{G}) p(\mathcal{I}_i|\mathcal{G},\ \mathcal{G}_i^{1:m},\ \mathcal{F}_i^{1:m}). \tag{4.10}$$

The normalization constant c is easily obtained by ensuring that the distribution sums to 1: $c = \sum_{\mathcal{G},\mathcal{G}_{1:n}^{1:m}} p(\mathcal{G},\ \mathcal{G}_{1:n}^{1:m},\ \mathcal{I}_{1:n},\ \mathcal{F}_{1:n}^{1:m})$. Eq. 4.7 is the direct application of Bayes rule, with the marginalization of latent variables. The sum in this form is intractable, thus the sum in Eq. 4.8 is factorized. The sequence of m functions f_i^k specify how to recursively compute the marginalization with the sums of individual $\mathcal{G}_i^k$ variables appropriately subsumed, so as to factor out terms not required at each level of the sum. Because of the particular form of silhouette terms in Eq. 4.5, this sum can be efficiently computed by noting that all terms after a first occupied voxel of the same visibility rank k share a term of identical value in $p(\mathcal{I}_i|\emptyset\cdots\emptyset\ \{\mathcal{G}_i^{v_k} = l\}\ *\cdots*) = \mathcal{P}_l(\mathcal{I}_i)$. They can be factored out of the remaining sum, which sums to 1 being a sum of terms of a probability distribution, leading to the following simplification of Eq. 4.9, $\forall k \in \{1, \cdots, m-1\}$:

$$f_i^k = p(\mathcal{G}_i^{v_k} = \emptyset|\mathcal{G}) f_i^{k+1} + \sum_{l\neq\emptyset} p(\mathcal{G}_i^{v_k} = l|\mathcal{G})\mathcal{P}_l(\mathcal{I}_i). \tag{4.11}$$

4.3 3D Modeling and Localization Algorithm

Section 4.2 presents a generic framework to infer the occupancy probability of a voxel X and thus deduce how likely it is for X to belong to one of m objects. Some additional work is required to use it to model objects in practice. The formulation explains how to

compute the occupancy of X if some occupancy information about the viewing lines is already known. Thus the algorithm needs to be initialized with a coarse shape estimate, whose computation is discussed in Section 4.3.1. Intuitively, object shape estimation and tracking are complementary and mutually helpful tasks. Section 4.3.2 explains how object localization information is computed and used in the modeling. To be fully automatic, our method uses the inference label u to detect objects not yet assigned to a given label and learn their appearance models (Section 4.3.3). Finally, it has been shown that static occluders can be computed using silhouette occlusion reasoning as discussed in Chapter 3 [Guan et al. (2007)]. This reasoning can easily be integrated in the current approach and help the inference be robust to static occluders (Section 4.3.4). The algorithm at every time instant is summarized in Fig. 4.3.

Algorithm 1: Dynamic Scene Reconstruction

Input: Frames at a new time instance for all views
Output: 3D object shapes in the scene

1 **Coarse Inference**;
2 **if** *new object enters the scene* **then**
3 add a label for the new object;
4 initialize foreground appearance model;
5 go to step 1;
6 **Refined Inference**;
7 static occluder inference;
8 update object location and prior;
9 **return**

Figure 4.3: Multi-shape dynamic scene reconstruction algorithm.

4.3.1 Shape Initialization and Refinement

The proposed formulation relies on some available prior knowledge about the scene occupancies and dynamic object ordering. Thus, part of the occupancy problem must be solved to bootstrap the algorithm. Fortunately, using multi-label silhouette inference

with no prior knowledge about occupancies or consideration for inter-object occlusions provides a decent initial m-occupancy estimate. This simpler inference case can easily be formulated by simplifying occlusion related variables from Eq. 4.8:

$$p(\mathcal{G}|\mathcal{I}_{1:n},\ \mathcal{F}_{1:n}^{1:m}) = \frac{1}{c}p(\mathcal{G})\prod_{i=1}^{n} p(\mathcal{I}_i|\mathcal{G},\ \mathcal{F}_i^{1:m}). \tag{4.12}$$

This *coarse inference* can then be used to initialize a second, *refined inference*, this time accounting for viewing line obstructions, given the voxel priors $p(\mathcal{G})$ and $\mathcal{P}(\mathcal{G}_i^j)$ of Eq. 4.2 computed from the coarse inference. The prior $p(\mathcal{G})$ is then used to introduce soft constraints to the inference. This is possible by using the coarse inference result as the input of a simple localization scheme, and using the localization information in $p(\mathcal{G})$ to enforce a compactness prior over the m objects, as discussed in Section 4.3.2.

4.3.2 Object Localization

The localization prior can be used to enforce the compactness of objects in the inference steps. For the particular case where walking people represent the dynamic objects, one can take advantage of the underlying structure of the dataset, by projecting the maximum probability over a vertical voxel column on the horizontal reference plane. Then the most likely position of objects is localized by sliding a fixed-size window over the resulting 2D probability map for each object. The resulting center is subsequently used to initialize $p(\mathcal{G})$, using a cylindrical spatial prior. This favors objects localized in one and only one portion of the scene and is intended as a soft guide to the inference. Although simple, this tracking scheme is shown to outperform state of the art methods (Section 4.4.2), thanks to the rich shape and occlusion information modeled.

4.3.3 Automatic Detection of New Objects

The main information about objects used by the proposed method is their set of appearances in the different views. These sets can be learned offline by segmenting each observed object alone in a clear, uncluttered scene before processing multi-objects scenes. More generally, one can initialize object color models in the scene automatically. To detect new objects, label u's object location and volume size can be computed during the coarse inference, and is used to track the unknown volume just like other objects as described in Section 4.3.2. A new dynamic object inference label is created (and m incremented), if all of the following criteria are satisfied:

- The entrance is only at the scene boundaries;
- Label u's volume size is larger than a threshold;
- Subsequent updates of $\mathcal{U}$'s track are bounded.

To build the color model of the new object, the maximum voxel probability along the viewing ray is projected to the camera view, which is then thresholded to form a "silhouette mask". Pixels within the mask are chosen as training samples for a GMM appearance model. Samples are only collected from unoccluded silhouette portions of the object, which can be verified from the inference. Because the cameras may be badly color-calibrated, an appearance model is trained for each camera view separately. This approach is fully evaluated in Section 4.4.1.

4.3.4 Occluder computation

The algorithm introduced in Chapter 3 [Guan et al. (2007)] computes dynamic object binary occupancy distributions at every voxel. It then analyzes the presence of dynamic object dominant probabilities of occupancy in front and behind of the voxel on its viewing lines, for every view and passed time instant of the sequence. Such dominant occupancies are then used to accumulate cues about occluder occupancy at the current

inferred voxel. The same formulation can easily be used and extended with the analysis presented in this chapter. At every time instant the dominant occupancy probabilities of m objects are already extracted; the two dominant occupancies in front and behind the current voxel X can be used in the occupancy inference formulation of Chapter 3. The occlusion occupancy inference then benefits from the disambiguation inherent to multi-silhouette reasoning.

4.4 Result and Evaluation

Four multi-view sequences are used to validate the approach. Eight 30Hz 720 by 480 DV cameras surrounding the scene in a semi-circle are used for the CLUSTER, SCULPTURE and BENCH sequences. LAB dataset is from [Gupta et al. (2007)].

	Cam. No.	Dynamic Obj. No.	Occluder
CLUSTER (outdoor)	8	5	no
BENCH (outdoor)	8	0 - 3	yes
LAB (indoor)	15	4	no
SCULPTURE (outdoor)	9	2	yes

Cameras in each data sequence are geometrically calibrated but not color calibrated. The background model is learned per-view using a single Gaussian color model at every pixel, with training images. Although simple, the model proves sufficient, even in outdoor sequences subject to background motion, foreground object shadows, window reflections and substantial illumination changes, showing the robustness of the method to difficult real conditions.

For dynamic object appearance models of the CLUSTER, LAB and SCULPTURE data sets, an RGB GMM model is trained for each person in each view with manually segmented foreground images. This is done offline. For the BENCH sequence however, appearance models are initialized online automatically.

The time complexity is $O(nmN^3)$, with n the number of cameras, m the number of objects in the scene, and N^3 the scene volume resolution. The data sets are processed on a 2.4 GHz Core Quad PC with computation times varying of $1-4$ min per time step. The very strong locality inherent to the algorithm and preliminary benchmarks suggest that around 10 times faster performance could be achieved using a GPU implementation, similar to Section 3.5.

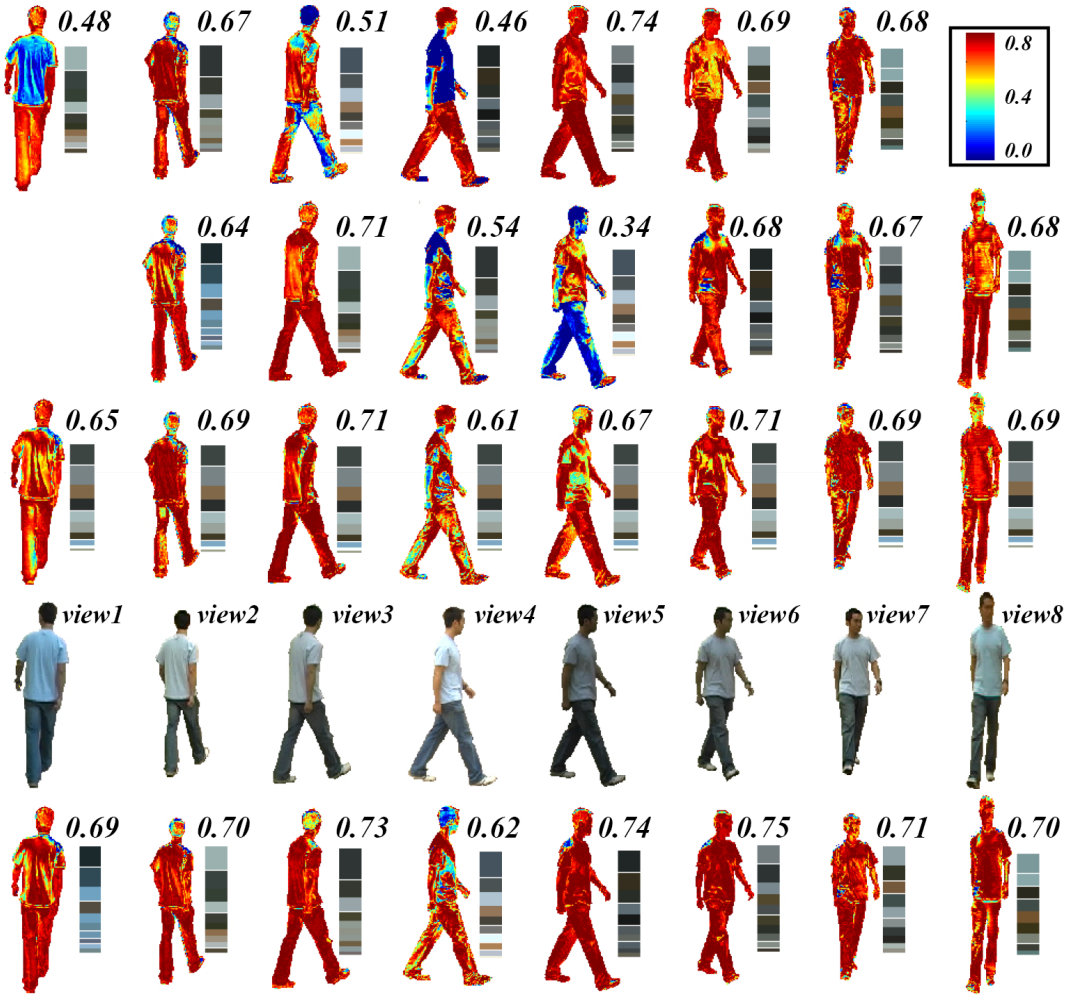

Figure 4.4: Appearance model analysis. A person in eight views is displayed in row 4. A GMM model $\mathcal{F}_i$ is trained for view $i \in [1, 8]$. A global GMM model $\mathcal{F}_0$ over all views is also trained. Row 1, 2, 3 and 5 compute $p(\mathcal{S}|\mathcal{I}_I, \mathcal{B}, \mathcal{F}_{i+1})$, $p(\mathcal{S}|\mathcal{I}, \mathcal{B}, \mathcal{F}_{i-1})$, $p(\mathcal{S}|\mathcal{I}, \mathcal{B}, \mathcal{F}_0)$ and $p(\mathcal{S}|\mathcal{I}, \mathcal{B}, \mathcal{F}_i)$ for view i respectively. The probability is displayed according to the color scheme at the top right corner. The average probability over all pixels in the silhouette region and the mean color modes of the applied GMM model are shown for each figure. **Best viewed in color.**

4.4.1 Appearance Modeling Validation

It is extremely hard to color-calibrate a large number of cameras, not to mention under varying lighting conditions, as in a natural environment. To show this, different appearance modeling schemes are compared in Fig. 4.4, for a frame of the outdoor BENCH dataset. Without loss of generality, GMMs are used. The first two rows compare silhouette extraction probabilities using the color models of spatially neighboring views. These indicate that stereo approaches which heavily depend on color correspondence between neighboring views are very likely to fail in the natural scenarios, especially when the cameras have dramatic color variations, such as in view 4 and 5. The global appearance model in row 3 performs better than the models in row 1 and 2, but this is mainly due to its compensation between large color variations across camera views, which at the same time, decreases the model's discriminability. The last row, where a color appearance model is independently maintained for every camera view, has the best performance. Therefore, the last scheme is used in the final system. Once the model is trained, it is currently not updated as time goes by. But the online updating could be an easy extension for robustness.

4.4.2 Densely Populated Scene

The CLUSTER sequence is a particularly challenging configuration: five people are in a circle of less than 3m in diameter, yielding an extremely ambiguous and occluded situation at the circle center. Despite the fact that none of them are being observed in all views, the proposed algorithm is still able to recover the people's label and shape. Images and results are shown in Fig. 4.5. The naïve 2-label reconstruction from Section 2.4.2 yields large volumes with little separation between objects, because the entire scene configuration is too ambiguous. Adding tracking prior information estimates the most probable compact regions and eliminates large errors, at the expense of dilation and lower precision. Accounting for viewing line occlusions enables the model to recover

more detailed information, such as the limbs.

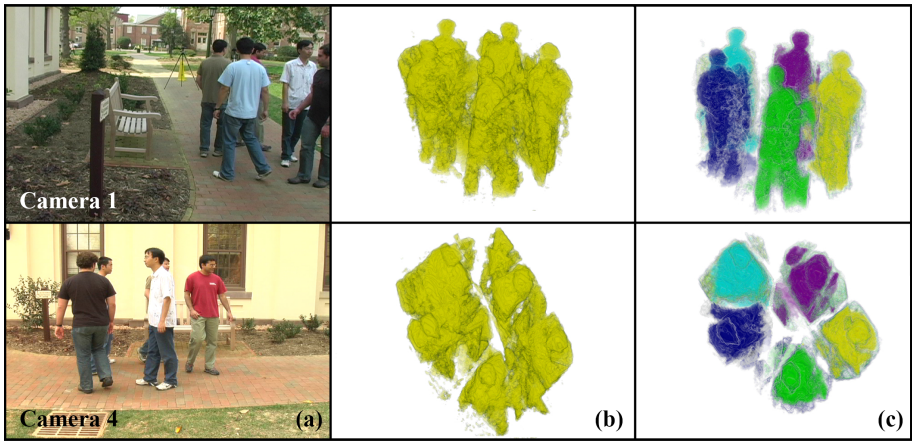

Figure 4.5: Result from 8-view CLUSTER dataset. (a) Two views at frame 0. (b) Respective 2-labeled reconstruction. (c) More accurate shape estimation using our algorithm. **Best viewed in color.**

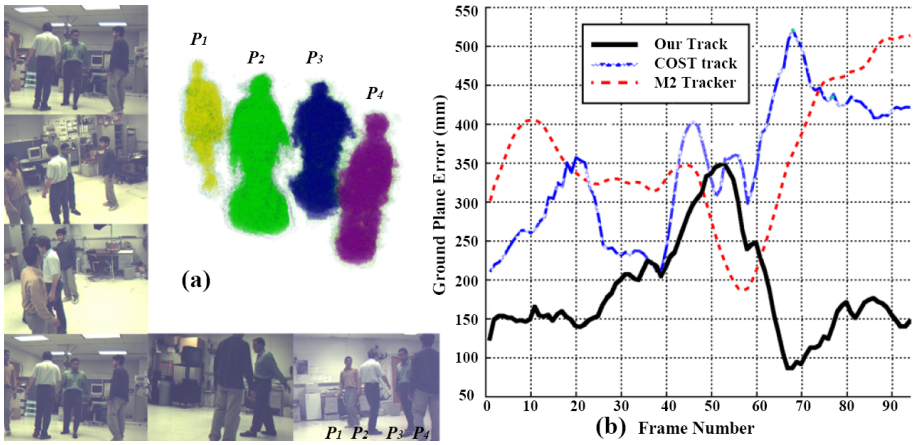

Figure 4.6: LAB dataset result from [Gupta et al. (2007)]. (a) 3D reconstruction with 15 views at frame 199 (b) 8-view tracking result comparison with methods in [Gupta et al. (2007)], [Mittal and Davis (2003)] and the ground truth data. Mean error in ground plane estimate in *mm* is plotted. **Best viewed in color.**

The LAB sequence [Gupta et al. (2007)] with poor image contrast is also processed. The reconstruction result from all 15 cameras is shown in Fig. 4.6. Moreover, in order to evaluate our localization prior, we compare our tracking method (Section 4.3.2) with the ground truth data, the result of [Gupta et al. (2007)] and [Mittal and Davis (2003)]. We use the same 8 cameras as in [Mittal and Davis (2003)] for the comparison, shown

in Fig. 4.6(b). Although slower in its current implementation (2 min. per time step) our method is generally more robust in tracking, and also builds 3D shape information. Most existing tracking methods only focus on a tracking envelope and do not compute precise 3D shapes, such as [Fleuret et al. (2007)]. This shape information is what enables our method to achieve comparable or better precision.

4.4.3 Automatic Appearance Model Initialization

The automatic dynamic object appearance model initialization has been tested using the BENCH sequence. Three people are walking into the empty scene one after another. By examining the unidentified label u, object appearance models are initialized and used for shape estimation in subsequent frames. Volume size evolution of all labels are shown in Fig. 4.7 and the reconstructions at two time instants are shown in Fig. 4.8.

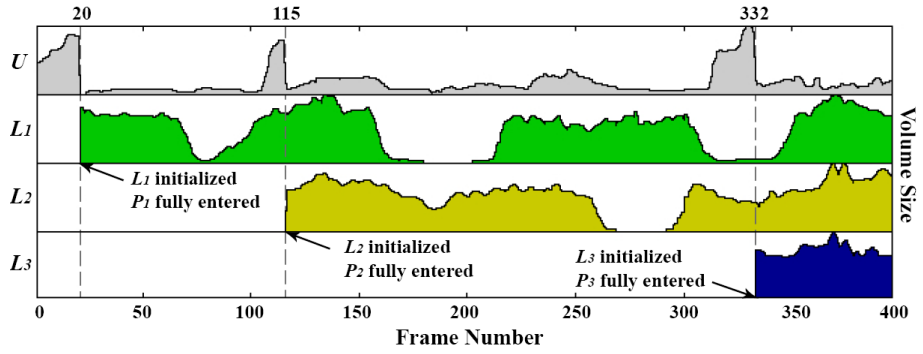

Figure 4.7: Appearance model automatic initialization with the BENCH sequence. The volume of u increases if a new person enters the scene. When an appearance model is learned, a new label is initialized. During the sequence, L_1 and L_2 volumes drop to near zero because they walk out of the scene on those occasions.

During the sequence, u has three major volume peaks due to three new persons entering the scene. Some smaller perturbations are due to shadows on the bench or the ground. Besides automatic object appearance model initialization, the system robustly re-detects and tracks the person who leaves and re-enters the scene. This is because once the label is initialized, it is evaluated for every time instant, even if the person is out of the scene. The algorithm can easily be improved to handle leaving/reentering

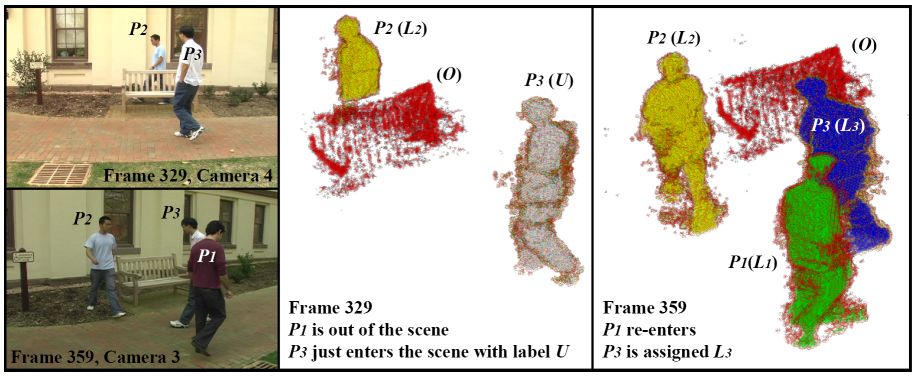

Figure 4.8: BENCH result. Person numbers are assigned according to the order their appearance models are initialized. At frame 329, P_3 is entering the scene. Since it's P_3's first time into the scene, he is captured by label u (gray color). P_1 is out of the scene at the moment. At frame 359, P_1 has re-entered the scene. P_3 has its GMM model already trained and label L_3 assigned. The bench as a static occluder is being recovered. **Best viewed in color.**

labels transparently.

4.4.4 Dynamic Object & Occluder Inference

The BENCH sequence demonstrates the power of our automatic appearance model initialization as well as the integrated occluder inference of the *"bench"* as shown in Fig. 4.8 between frame 329 and 359. Refer to Fig. 4.7 about the scene configuration during that period.

We have also processed the SCULPTURE sequence from the end of Chapter 3 with two persons walking in the scene, as shown in Fig. 4.9. For the dynamic objects, much cleaner shapes are obtained when the two persons are close to each other, and more detailed shapes such as extended arms. For the occluder, the finer shape is also recovered, while the computation using uniform foreground appearance model has a lot of noise, due to the occluder inference using ambiguous regions when people are clustered.

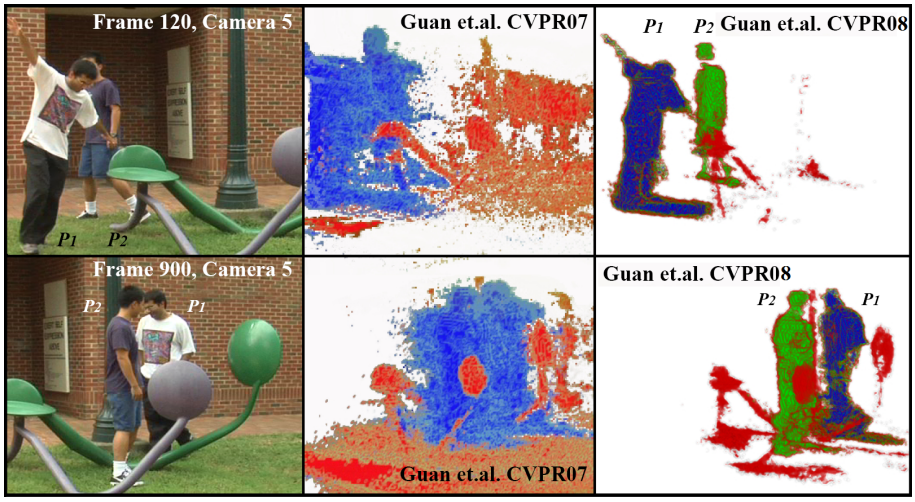

Figure 4.9: SCULPTURE data set comparison. While both methods from the previous chapter and this one recover the static sculpture, the current method (the right column) resolves inter-occlusion ambiguities, and estimates much better dynamic object shapes. **Best viewed in color.**

4.5 Further Discussion

A Bayesian method to build 3D shapes from multi-object silhouette cues is proposed in this chapter. The appearances of objects are used to disambiguate free regions of space that project inside silhouettes, and occlusion information and object localization priors are used to update the representation iteratively so as to refine the resulting shapes. Our results show that the shapes obtained using this approach yield significantly better results than pure silhouette reasoning, which makes no distinction between different objects. This new multi-silhouette inference algorithm is robust to very difficult conditions, and can prove very useful for various vision tasks such as tracking, localization and 3D reconstruction, in highly cluttered scenes with densely packed dynamic object groups. A large number of extensions can be tested on the basis of the framework provided, including more general and complex appearance modeling, different enforcements of the compactness of objects, a more general management of objects entering and leaving the scene. It is possible to analyze object label transition, for example a static object in the scene might be moved to a different place, and a person might come and sit statically on

the bench. Temporal consistency constraints could also be included in stronger forms, to enforce temporal continuity of the reconstruction and smoothness of the flow in the scene.

4.5.1 Limitations and Extensions

The algorithm proposed in this chapter uses image observation information for occupancy inference. If a dynamic shape is not observed from more than one camera view, in theory, the visual hull occupancy would not be closed. Therefore in an extremely densely populated scene, the algorithm may fail. One solution is to add more camera views so as to increase the spatial sampling power of the system.

Both the static occluder inference and multiple dynamic shape inference benefit from explicitly modeling the occluding event. The difference is that for static occluder, temporal accumulation is possible while for dynamic shapes, the decision must be made within the time instant. Therefore, the occlusion modeling for multiple dynamic shapes also tends to be less robust than static occluder accumulation.

One direction to go is to explore the temporal consistency of the dynamic scene, based on the observation that the video frame rates are so high that consecutive frames are almost the same. In this chapter, the temporal link is only exploited in the form of the cylindrical shape location prior, which is a very naïve enforcement. In the next chapter, a method to recover the dense 3D motion field between consecutive time instances is introduced, with the help of which the ultimate temporal consistency is constrained, and the 3D dynamic shape occupancy probability is refined. More interesting applications can also be computed from this dense field, such as the automatic generalization of the rigid motion skeleton of the dynamic shape.

Chapter 5

Dense Occupancy Flow Estimation

It has been shown in the last chapter that a simple tracking scheme can improve the 3D shape estimation quality as a way to constrain the temporal consistency. To fully explore this direction, this chapter presents a new framework to analyze 3D motions of dynamic subjects with a calibrated multi-view setup. Inspired by 2D optical flow motion analysis methods [Sun et al. (2008)] and the seminal 3D scene flow work from [Vedula et al. (2005)], the proposed framework allows to estimate 3D motion flow between two subsequently acquired frame sets of the scene for arbitrary moving objects. Unlike existing approaches however, the motion flow computed in this chapter is volumetrically dense in 3D space and does not rely on any explicit boundary representation of the scene subjects. It only relies on (1) implicit silhouette cues with no binary segmentation decision, and (2) the assumption of spatial continuity of the motion field. The proposed framework thus explores what minimal constraints and data can be used for 3D motion analysis. The motivation is to better exploit raw observations from multiple views without premature assumptions, while being robust to typical sources of noise in images that affect 3D reconstruction methods.

Building 4D space-time representations of scenes observed from multiple calibrated views is a major challenge in computer vision. Such representations are often sought, to track and build time-coherent 3D shape geometry and analyze 3D motion of subjects in

the scene. They are relevant to many fields in research and industry, for free viewpoint video acquisition, automatic 3D shape and human performance acquisition, virtual reality and HCI applications, 3D shape matching and recognition. An overview of the main related works and problems are given below.

5.1 Related works

The problem of building geometric 3D representations across time was first approached in a purely frame-by-frame manner without any temporal consistency constraint, using photometric stereo information [Kutulakos and Seitz (2000)], sparse feature matches, or silhouettes [Laurentini (1994)]. While photometric stereo and match information can provide precise surface information for reconstructed models, it does so mainly in highly constrained setups with good light control, high resolution inputs, and an implicitly assumption that observed objects surfaces are largely textured. This is why silhouette-based methods [Laurentini (1994)] have gained popularity for shape acquisition tasks [Lazebnik et al. (2007); Franco and Boyer (2009)] as they are robust with color-inconsistent or weakly textured data, on top of being generally more time-efficient in terms of computation. Most aforementioned shape modeling approaches focus on surface representations, yet alternative representations, such as volumetric probability grids, have emerged to improve robustness to noise of various methods including photometric space carving[Broadhurst et al. (2001)]. The use of such representations with latent silhouette data has been demonstrated to be particularly robust for difficult, natural environments as discussed in Chapter 2, 3, and 4 [Franco and Boyer (2005); Guan et al. (2007, 2008b)], a property also leveraged for silhouette-based 3D motion analysis in this chapter.

For long scientists have wanted to exploit the redundancy of information across time in the acquired videos to build time-consistent representations of arbitrary objects,

refine shape information, and capture the dynamics of recorded 3D scenes. Lately, mesh-tracking methods have been proved to be successful solutions for time-consistent dynamic acquisition. Their common goal is to retrieve a geometrically and temporally consistent surface state from the video sequences. Many such methods fit existing, fixed-topology mesh/skeletal model templates as in [de Aguiar et al. (2007); Ballan and Cortelazzo (2008)] to image data. These methods are however often particularized for the case of a specific shape, usually human [Vlasic et al. (2008)], by underlying geometric or kinematic assumptions and user manipulation. Other methods aim for more general surfaces and can sometimes deal with surface topology changes [Varanasi et al. (2008). To constrain surface construction, the methods use a variety of image cues, for example dense optical flow [de Aguiar et al. (2007)], or sparse feature matches [Varanasi et al. (2008)]. Such inputs can be difficult to obtain in uncontrolled and poorly textured general environments. Other 4D analysis methods exist to build alternative time-consistent shapes representations. [Cheung et al. (2003)] combines voxel-based representations with silhouette inputs, albeit in the more particular cases of rigid or articulated objects.

Quite remarkably, a large majority of time-consistent shape modeling methods use silhouette-based constraints in one form or another to stabilize estimation, since most real-life scenes with human subjects tend to be poorly textured or simply hard to find effective 3D feature correspondences as shown in Fig. 2.6. In fact, some mesh tracking methods actually use silhouette data alone [Vlasic et al. (2008)]. This shows the constraining power of the visual cue, which the proposed method in this chapter wishes to exploit in full generality. Also, the vast majority of 4D shape building methods have only been tested in completely controlled environments, and are prone to drifting and failure in the presence of noisy, uncontrolled scenes. As such, these methods could benefit from a new, general 3D motion analysis step such as the one proposed in this chapter, which makes no geometrical assumptions about the scene, and could be used as additional

input to constrain mesh motion.

Among all 3D motion analysis methods, the proposed method most closely relates to scene flow concepts. Scene flow algorithms produce dense motion vector associated to various visible surface representations including voxels [Vedula et al. (2005)], implicit representations [Pons et al. (2007)], stereo disparity maps [Wedel et al. (2008)] or meshes [de Aguiar et al. (2007)]. Importantly most scene flow approaches assume an underlying surface is already available [Vedula et al. (2005); de Aguiar et al. (2007)] or simultaneously built [Pons et al. (2007)]. Scene flow methods rely on the estimation of spatial derivatives of the image signal, sometimes delegated to existing 2D optical flow estimation [Vedula et al. (2005); de Aguiar et al. (2007)]. As noted in [Starck and Hilton (2007)], flow-based approaches are generally limited to small displacements, as a consequence of finite difference approximations of derivatives. The 3D dense motion flow analysis proposed in this chapter relies on silhouette cues rather than photometric cues, and can be computed before any surface construction, making it a complementary approach to existing scene flow methods. Issues with large displacements is addressed using a multi-scale approach to 3D flow estimation.

5.2 Overview

A different and complementary approach from most methods above is proposed in this chapter, which abandon surface representations altogether and avoid their caveats, yet proves useful for retrieval of 3D shape and motion cues as shown in later experiments. Indeed, no surface initialization, surface matching steps, or underlying kinematic control structure of any type are needed with the proposed method. Moreover, only volumetric continuity of motion is used as a constraint, which enables the inference to extrapolate dense 3D motion estimations from latent silhouette cues, much in the way 2D optical flow methods [Sun et al. (2008)] propagate motion information from edge motion boundaries.

As such the method provides a new tool for 3D motion analysis, which could be used as input to shape analysis and reconstruction, kinematic chain recognition or motion segmentation, using truly minimal scene assumptions and input constraints.

A probabilistic modeling of the scene shape and motion is used in the form of statistic variables attached to voxels in a volumetric grid, described in Section 5.3. The resulting estimation problem translates to an Expectation Maximization algorithm, which is fully described in Section 5.4. It alternates between estimating voxel occupancy probabilities in the E-step (Section 5.4.2), and finding a motion field best estimate in the M-step (Section 5.4.3). The M-step can be cast as a multi-label MRF by discretizing the motion field space, which is efficiently solved using a coarse-to-fine approach as described in Section 5.5. The method is validated on several synthetic, indoor, and outdoor datasets (Section 5.6).

5.3 Problem Formulation

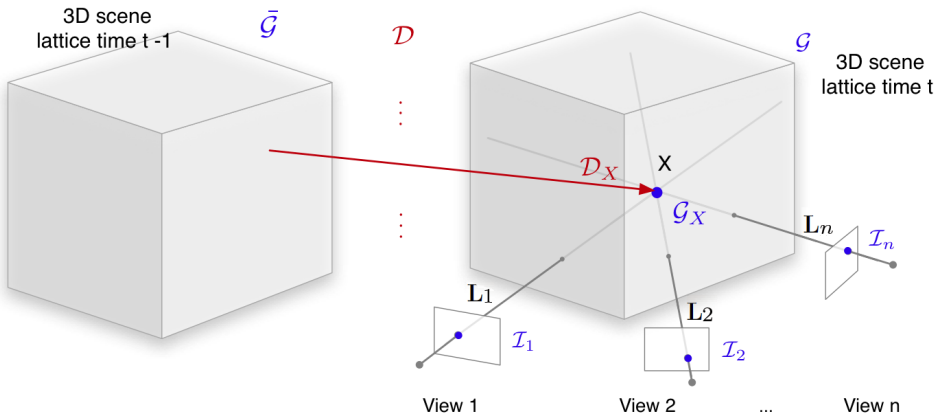

Figure 5.1: Overview of main statistical variables and geometry of the problem. $\mathcal{G}_X$ is the occupancy at voxel X.

The scene is represented with a 3D lattice of points in space (Fig. 5.1), denoted as $\mathcal{X}$. At time t, a set of images $\mathcal{I}$, specifically $\mathcal{I}_1, \mathcal{I}_2 \cdots \mathcal{I}_n$ from n camera views are observed with known geometrical calibration information. Each point $X \in \mathcal{X}$ is associated to a

binary occupancy state, empty or occupied, denoted by $\mathcal{G}_X \in \{0, 1\}$. The conjunction of all grid states is noted $\mathcal{G}$. The algorithm is interested in using information from time $t-1$ given a certain time discretization, where the previous grid state is noted $\bar{\mathcal{G}}$. The motion of matter from $t-1$ to t is represented by a displacement vector field $\mathcal{D}$. Specifically, each point X is associated to the vector $\mathcal{D}_X$ that displaces matter from location $X - \mathcal{D}_X$ to X between times $t-1$ and t. Since no surface representation is used, it is assumed that the motion field is defined everywhere in space and continuous, thus treating space as an image-constrained fluid, where the probabilistic occupancy and motion representation indifferently embeds actual matter or air.

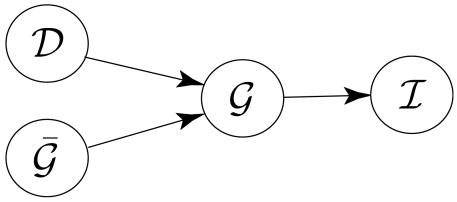

Figure 5.2: System variable dependency graph.

The relationship between the different factors of the system can be modeled by the joint probability of the variables: $p(\bar{\mathcal{G}}, \mathcal{G}, \mathcal{D}, \mathcal{I})$. Based on the dependency graph shown in Fig. 5.2, a logical decomposition of the joint probability is stated in Eq. 5.1: to predict occupancies $\mathcal{G}$, only the displacements $\mathcal{D}$ and previous occupancies $\bar{\mathcal{G}}$ are needed; to predict images $\mathcal{I}$, only the occupancies $\mathcal{G}$ at time t needs to be known.

$$p(\bar{\mathcal{G}}, \mathcal{G}, \mathcal{D}, \mathcal{I}) = p(\mathcal{I}|\mathcal{G})p(\mathcal{G}|\mathcal{D}, \bar{\mathcal{G}})p(\bar{\mathcal{G}})p(\mathcal{D}) \tag{5.1}$$

Assume voxel occupancies $\mathcal{G}_X$ are mutually independent and assume that pixels on which a voxel center X projected are measurements exclusively associated to that voxel. This is a typical silhouette-based method simplification, which enables breaking the dependencies between voxels of the same viewing line. Additionally, assume conditional

independence of a voxel X's pixel measurements given the knowledge of the voxel's state $\mathcal{G}_X$, one can express Eq. 5.1 as a product over voxels, image views and pixels:

$$p(\bar{\mathcal{G}}, \mathcal{G}, \mathcal{D}, \mathcal{I}) = p(\mathcal{D}) \prod_X \left(p(\bar{\mathcal{G}}_X) p(\mathcal{G}_X | \mathcal{D}, \bar{\mathcal{G}}) \prod_i p(\mathcal{I}_i | \mathcal{G}_X) \right) \tag{5.2}$$

where i is the camera image view index. $\mathcal{I}_i$ is the pixel color at the projection of X in image i. For readability, the measurement term at time t is denoted $\mathbf{\Phi}(\mathcal{G}_X) = \prod_i p(\mathcal{I}_i | \mathcal{G}_X)$.

In Eq. 5.2, the term $p(\bar{\mathcal{G}}_X), p(\mathcal{G}_X | \mathcal{D}, \bar{\mathcal{G}})$ models the information obtained from the previous time instant. As a first degree approximation, only one of the displacements, $\mathcal{D}_X$, influences X at time t. Thus only $\bar{\mathcal{G}}_{X-\mathcal{D}_X}$ influences $\mathcal{G}_X$:

$$p(\bar{\mathcal{G}}, \mathcal{G}, \mathcal{D}, \mathcal{I}) = p(\mathcal{D}) \prod_X \left(p(\bar{\mathcal{G}}_{X-\mathcal{D}_X}) p(\ \mathcal{G}_X | \bar{\mathcal{G}}_{X-\mathcal{D}_X}) \cdot \mathbf{\Phi}(\mathcal{G}_X) \right). \tag{5.3}$$

The term $p(\mathcal{D})$ models the prior over the 3D motion field. The first order continuity properties of the fields will be used, as described in Section 5.4.3. As probabilistic inference information is assumed to be already available for the previous time step for $\bar{\mathcal{G}}_X$, $\bar{\mathcal{G}}_X$ is treated as a latent variable. This allows to retain the probabilistic information from $t-1$ by marginalizing out $\bar{\mathcal{G}}_X$ in all subsequent inferences.

5.4 Estimating 3D Motion and Occupancy

Given the identified dependencies, ideally the goal is to compute the full distribution of $p(\mathcal{D}, \mathcal{G} | \mathcal{I})$ since the primary interest is to estimate the displacement field and occupancy probabilities at time t given image observations. Unfortunately this inference is intractable given the huge state spaces of $\mathcal{D}$ and $\mathcal{G}$. However the problem is well suited for an EM formulation [Dempster et al. (1977)] with some simplifications. By

focusing on estimating the Maximum A Posteriori (MAP) of $p(\mathcal{D}|\mathcal{I})$ and treating $\mathcal{G}$ as a latent, unobserved variable set, an EM procedure can be constructed that iteratively refines estimates $d^0, d^1 \cdots, d^*$ of the motion field $\mathcal{D}$ (M-step), while providing with each new iteration $k+1$ an estimate of $p(\mathcal{G}|\mathcal{I}, d^k)$ (E-step). The latter term corresponds to estimating voxel occupancy probabilities given the image data at time t and using the previous iteration's computed motion field, thus providing a probabilistic shape estimate representation analog to previous probabilistic methods [Broadhurst et al. (2001); Franco and Boyer (2005)]. A MAP-EM formulation is chosen also because the traditional EM formulation does not consider the prior $p(\mathcal{D})$ on motion fields, which is used to enforce spatial continuity.

5.4.1 Expectation Maximization Formulation

Here is a MAP-EM formulation, whose goal is to find the optimal d^* such that:

$$d^* = \text{argmax}_{\mathcal{D}}\, F(\mathcal{D}) \quad \text{with} \;\; F(\mathcal{D}) = p(\mathcal{D}|\mathcal{I}). \tag{5.4}$$

This goal is achieved iteratively starting from an initial guess d^0, by building a sequence of motion field estimates $d^0, d^1, \cdots, d^*$ which increase the log-posterior objective function $F(\mathcal{D})$, *i.e.* $F(d^0) \leq \cdots \leq F(d^*)$. This is usually achieved at each step by maximizing a lower bound of $F(\mathcal{D})$, whose maximum coincides with an analytically simpler function $Q(\mathcal{D}|d^k)$. For a good choice of lower bound and $Q(\mathcal{D}|d^k)$, the maximization guarantees an increase of the log-posterior, such that d^{k+1} can be defined as follows:

$$\text{The } \textbf{M-Step:} \qquad d^{k+1} = \text{argmax}_{\mathcal{D}}\, Q(\mathcal{D}|d^k). \tag{5.5}$$

It can be shown that a $Q(\mathcal{D}|d^k)$ verifying these properties is obtained in the case of

our problem using the following conditional expectation:

$$\begin{aligned} Q(\mathcal{D}|d^k) &= E_{\mathcal{G}|\mathcal{I},d^k}\{\ln p(\mathcal{I},\mathcal{G},\mathcal{D})\} \\ &= \sum_{\mathcal{G}} p(\mathcal{G}|\mathcal{I},d^k)\ln p(\mathcal{I},\mathcal{G},\mathcal{D}). \end{aligned} \tag{5.6}$$

The **E-step** evaluates $p(\mathcal{G}|\mathcal{I},d^k)$ of Eq. 5.6, i.e. the grid occupancy probabilities given images $\mathcal{I}$and the previously predicted displacement d^k. Given this distribution, the log-expectation of $p(\mathcal{I},\mathcal{G},\mathcal{D})$ can be computed for all possible $\mathcal{D}$as in Eq. 5.6.

The **E-step** of the algorithm then consists in evaluating the $p(\mathcal{G}|\mathcal{I},d^k)$ term of the expression, *i.e.* the grid occupancy probabilities given images and the previously predicted displacement. Both probability distributions can be obtained from Eq. 5.3. Each step is developed subsequently in further detail.

5.4.2 E-step: Occupancy Probability Update

In order to compute voxel probabilities at a given EM iteration, one needs to express $p(\mathcal{G}|\mathcal{I},d^k)$ in terms of the joint probability distribution (5.3). Bayes' rule is used (5.7) and the summations is re-factored (5.8) to depending terms, with $\propto$ denoting proportionality up to a unit normalization factor:

$$p(\mathcal{G}|\mathcal{I},d^k) \propto \sum_{\bar{\mathcal{G}}} p(\bar{\mathcal{G}}\mathcal{G}d^k\mathcal{I}) \tag{5.7}$$

$$\propto \prod_X \mathbf{\Phi}(\mathcal{G}_X)\cdot\sum_{\bar{\mathcal{G}}_{X-d_X^k}} p(\bar{\mathcal{G}}_{X-d_X^k})p(\mathcal{G}_X|\bar{\mathcal{G}}_{X-d_X^k}), \tag{5.8}$$

where $\sum_{\bar{\mathcal{G}}_{X-d_X^k}} p(\bar{\mathcal{G}}_{X-d_X^k})p(\mathcal{G}_X|\bar{\mathcal{G}}_{X-d_X^k})$ sums possibilities over occupancy states of the voxel $X-d_X^k$ that has been mapped to Xthrough displacement d_X^k. For simplicity,

$p(\mathcal{G}_X|\bar{\mathcal{G}}_{X-d_X^k})$ is set deterministically: if the previous voxel $X - d_X^k$ was occupied (resp. empty), then once displaced to Xit is still occupied (resp. empty) with probability 1. Expression (5.8) then becomes:

$$p(\mathcal{G}|\mathcal{I}, d^k) \propto \prod_X \left(\mathbf{\Phi}(\mathcal{G}_X) \cdot p([\bar{\mathcal{G}}_{X-d_X^k} = \mathcal{G}_X]) \right). \tag{5.9}$$

For the purpose of providing a probabilistic shape estimate, each voxel X's probability after an E-step can thus be identified as $p(\mathcal{G}_X|\mathcal{I}, d^k) \propto \mathbf{\Phi}(\mathcal{G}_X) \cdot p([\bar{\mathcal{G}}_{X-d_X^k} = \mathcal{G}_X])$, the product of current observation terms at time t, with the probability of the voxel mapped to $X$from $t-1$.

$\mathbf{\Phi}(\mathcal{G}_X)$ can be computed by expliciting the image formation terms $p(\mathcal{I}_i|\mathcal{G}_X)$. For every pixel x in every image, it is assumed that the parameters $\mathcal{B}$ of a background model have been learned offline from images of a quasi-static scene with no object of interest. Similar to Section 2.4.2, the image term $p(\mathcal{I}_i|\mathcal{G}_X)$ can then be set as following [Franco and Boyer (2005)]:

$$p(\mathcal{I}_i|\mathcal{G}_X) = p(\mathcal{G}_X{=}0)p(\mathcal{I}_i|\mathcal{B}) + p(\mathcal{G}_X{=}1)\mathcal{U}(\mathcal{I}_i),$$

where $\mathcal{U}(\mathcal{I}_i)$ is the uniform distribution over pixel color space, used to model the aspect of objects of interest since no information about it is used, and $p(\mathcal{I}_i|\mathcal{B})$ is the probability of $\mathcal{I}_i$ to be drawn from the background model $\mathcal{B}$. $p(\mathcal{I}_i|\mathcal{B})$ can be, for instance, a Normal or Gaussian Mixture Model distribution. Here the voxel X's occupancy $\mathcal{G}_X$ serves as a mixture label to draw $\mathcal{I}_i$ from foreground or background aspect distributions. The silhouette information is latent in this representation and does not require any binary segmentation decision.

5.4.3 M-step: 3D Motion Field Update

To optimize Eq. 5.5, one still needs to expand the expression of $Q(\mathcal{D}|d^k)$ in Eq. 5.6. The distribution $p(\mathcal{I}, \mathcal{G}, \mathcal{D})$ can be computed by marginalizing Eq. 5.3 over $\bar{\mathcal{G}}$, and simplified similarly to Eq. 5.8:

$$p(\mathcal{I}, \mathcal{G}, \mathcal{D}) \propto p(\mathcal{D}) \prod_X \left(\mathbf{\Phi}(\mathcal{G}_X) \cdot p([\bar{\mathcal{G}}_{X-\mathcal{D}_X} = \mathcal{G}_X]) \right) \tag{5.10}$$

Taking the logarithm of Eq. 5.10 to compute $Q(\mathcal{D}|d^k)$, and noting that the $\mathbf{\Phi}(\mathcal{G}_X)$term does not depend on $\mathcal{D}$, the M-step becomes:

$$\begin{aligned} d^{k+1} = & \operatorname{argmax}_{\mathcal{D}} \ \ln(p(\mathcal{D})) \\ & + \sum_X \sum_{\mathcal{G}_X} p(\mathcal{G}_X|\mathcal{I}, d^k) \cdot \ln p([\bar{\mathcal{G}}_{X-\mathcal{D}_X} = \mathcal{G}_X]) \end{aligned} \tag{5.11}$$

where $p(\mathcal{G}_X|\mathcal{I}, d^k)$ is computed in the E-step (Eq. 5.9).

To model 3D motion field continuity, $p(\mathcal{D})$ is chosen to be a Markov Random Field. The M-step in the EM framework becomes a standard first-order MRF MAP problem. From time t to $t+1$, if the displacement possibilities at every point is *quantized* to n displacement options denoted as a label set $\mathcal{L} = \{l^1, \cdots, l^n\}$, then Eq. 5.11 can be represented as standard graph optimization pairwise and unary energy terms:

$$E_{MRF} = \sum_X \sum_{Y \in \mathcal{N}(X)} E_{XY}(l_X, l_Y) + \sum_X E_X(l_X), \tag{5.12}$$

where $\mathcal{N}(X)$ is the neighborhood system of point Xin the 3D graph. In Eq. 5.12, $\sum_X \sum_{Y \in \mathcal{N}(X)} E_{XY}(l_X, l_Y)$ are the binary terms, and $\sum_X E_X(l_X)$ are the unary terms.

They correspond to the *negative* of $\ln(p(\mathcal{D}))$ and $\sum_X \sum_{\mathcal{G}_X} p(\mathcal{G}_X|\mathcal{I}, d^k) \cdot \ln p([\bar{\mathcal{G}}_{X-\mathcal{D}_X} = \mathcal{G}_X])$ in Eq. 5.11 respectively.

The M-step in this EM framework thus becomes a discrete multi-labeling problem, with the go$\sum_X \sum_{Y \in \mathcal{N}(X)} E_{XY}(l_X, l_Y)$ are the binary terms, and $\sum_X E_X(l_X)$ are the unary terms. They correspond to the *negative* of $\ln(p(\mathcal{D}))$ and $\sum_X \sum_{\mathcal{G}_X} p(\mathcal{G}_X|\mathcal{I}, d^k) \cdot \ln p([\bar{\mathcal{G}}_{X-\mathcal{D}_X} = \mathcal{G}_X])$ in Eq. 5.11 respectively.

The M-step in our EM framework becomes a discrete multi-labeling problem, with the goal of computing a labeling $L \in \mathcal{L}^{|\mathcal{X}|}$, which assigns each grid node $X \in \mathcal{X}$ a label from $\mathcal{L}$such that the energy E_{MRF} is minimized. Thus

$$d^{k+1} = \text{argmin}_L E_{MRF}. \tag{5.13}$$

The solution to this MRF thus provides the updated displacement field in the EM iteration. Further details on MRF implementation is given in the sections below.

5.5 Motion Field Optimization

The EM previously developed provides the framework and justification for shape and motion estimation. Because of the large state space and ill-posed nature of the problem, and because EM has the potential to converge to unwanted local minima, additional steps must be taken to ensure convergence. First what field assumptions are to be used in the framework (Section 5.5.1) are reviewed. The resulting energy function can be optimized using Fast-PD approaches [Komodakis et al. (2007)] (details in Section 5.5.2). Fast-PD can be applied in a coarse-to-fine approach for method stability, efficiency and convergence (Section 5.5.3).

5.5.1 Motion Field Properties

Only assume that motions are bounded and locally smooth. Since no hard surface assumption is used and to store only probabilistic shape information, the motion field covers the whole grid space. Thus matter and empty space (air) are embedded in the same motion field indifferently, while being image-constrained to map probably empty voxels (resp. occupied) to probably empty (resp. occupied) voxels, as modeled in Eq. 5.9. A simple motion field smoothness can be defined in the pairwise energy function E_{XY} in Eq. 5.12 as a distance function computing the magnitude of vector differences [Glocker et al. (2008)]:

$$E_{XY}(l_X, l_Y) = \lambda_{XY} \left| \boldsymbol{d}(l_X) - \boldsymbol{d}(l_Y) \right|^{0.8} \tag{5.14}$$

where λ_{XY} is a weighting factor, $\boldsymbol{d}(l)$ is the motion vector that label l represents and the index 0.8 is specially chosen to be less than one, motivated by statistics of velocity difference distribution studied for optical flow constraints [Sun et al. (2008)].

However, as pointed out in [Glocker et al. (2008)], a more desirable pairwise energy term can be defined specifically to avoid overly fluid-like deformations in the case of iterative, coarse-to-fine approches:

$$E_{XY}(l_X, l_Y) = \lambda_{XY} \left| \boldsymbol{D}_X + \boldsymbol{d}(l_X) - \boldsymbol{D}_Y - \boldsymbol{d}(l_Y) \right|^{0.8} \tag{5.15}$$

where $\boldsymbol{D}_X$ and $\boldsymbol{D}_Y$ are the motions that have been recovered at location X and Y from previous iterations.

5.5.2 Fast-PD Optimization

Given the form of the energy terms Eq. 5.12, the M-step can be solved by discrete graph optimization schemes. The Fast-PD approach is selected [Komodakis et al. (2007)],

which builds upon principles drawn from the duality theory of linear programming in order to efficiently derive almost optimal solutions for a very wide class of NP-hard MRFs [Komodakis and Tziritas (2007)]. Indeed this approach has several advantages: it is faster than state-of-the-art graph cut α-expansion methods [Boykov et al. (2001)] and guarantees an optimality bound. In addition, it handles cost functions with arbitrary pair-wise potentials, lifting the *submodularity* constraint of previous approaches [Kolmogorov and Zabih (2004)]. This provides freedom to use the more elaborate forms of motion field local properties $E_{XY}(l_X, l_Y)$, such as the one used in Eq. 5.15.

5.5.3 Coarse-to-Fine Approach

To avoid local optima, speed up each EM iteration, and break the problem into memory efficient steps, a coarse-to-fine approach is chosen and parametrization is used for 3D volumetric registration in medical imaging [Glocker et al. (2008)]. The EM is initialized with a coarse global translation registration of grids. Then the 3D space is embedded in a 3D B-Spline free form deformation (FFD) controlled by a uniform grid of sparse control points. At each chosen scale, the MRF previously defined is solved for the control points, and initialization for finer scales obtained by interpolation of the coarser scale. Control point spacing d_s at each scale is set relative to scene scale. The label set $\mathcal{L}$ of a control point's motion is constrained to be within half of the control point spacing, $d_s/2$. This avoids self-folding and constrains the FFD to be a diffeomorphism over the volume. Specifically, the cubic range defined by the maximum displacement $[-d_s/2, d_s/2]$ is sparsely sampled along the three axis. Additionally the deformation optimization is repeated at each control scale until the FastPD stabilizes to null displacements (in practice 4 to 8 times depending on the dataset) which proves more stable. As shown in the result section, this also allows the recovery of motions even larger than the maximal allowed in the coarsest scale.

5.6 Results

The proposed algorithm has been tested on a synthetic dataset with two moving boxes, and several real world datasets, also shown in the supplemental video. The results in this section validate the correctness and feasibility of the algorithm. All the datasets are challenging for state-of-the-art general 3D motion computation algorithms because of texture-less surfaces, different types of noise, and lack of photometric calibration.

In all experiments, a 128^3 occupancy grid and three levels of control grid with 5, 3, and 1 voxel as largest motion magnitude respectively are used. The control points in the three levels are 11, 7 and 1 voxels apart respectively. The EM converges in less than five iterations for all datasets. Each maximization step is repeated, and 4 to 8 times per scale. The preliminary implementation takes several minutes per frame for most datasets. Most of the time is spent building the graph and computing weights, which could be optimized and parallelized.

5.6.1 Synthetic Dataset

Two cubes of different sizes are flying in elliptic orbits in 3D space. They are observed by 9 virtual cameras surrounding the scene as shown in Fig. 5.3(b). The probabilistic occupancy grid of 9 time steps is computed from the synthetic camera frames added with random noise. 8 flow fields are then computed using the proposed method. Then, selected grid point above 0.98 probability in the first time step's occupancy grid can be traced through the sequence using the computed flow field, and accumulated as in Fig. 5.3(a). The ground truth tracks of the orbiting cubes are drawn with thick curves for comparison.

The absolute 3D angular error (AAE) is computed between the ground truth cube motion vector and the obtained motion vectors of every grid point above 0.98 probability. The statistics of each cube at every time instant is plotted as in Fig. 5.5. The small

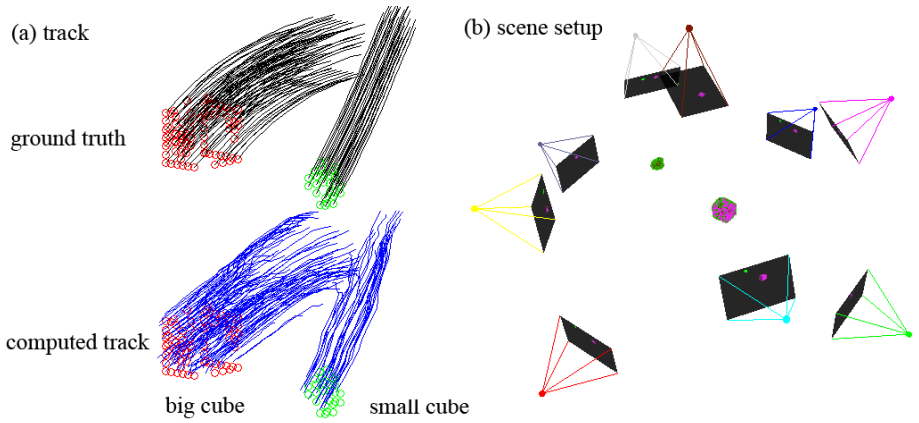

Figure 5.3: Left: the computed dense motion tracks of 2 orbiting cubes. The ground truth trajectories are shifted above for clearer comparison. Right: virtual setup for this synthetic 3D scene capturing. **Best viewed in color.**

cube's AAE against the ground truth is much smaller than the big cube. It is true for the following reasons: (1) As opposed to stereo methods, the proposed method does not have access to tangential motions of the surface. Thus rotations of objects, particularly near an axis of symmetry, are more difficult to estimate by the method. This can be seen in Fig. 5.5, where the motion tracks inside the big cube are shorter than the ground truth trajectory, which is locally a valid translation solution. (2) Voxels of larger shapes tend to rely more on interpolation and less on image information. Conversely, the motion and amount of silhouette information of the smaller cube is large compared with its own size, creating less motion ambiguity for those voxels: this suggests the cases where the algorithm works best, when a thin part of an observed shape is observed moving by several silhouettes. This includes translations, and many of the limb joint-rotations observable from silhouettes, as shown in Fig. 5.4.

5.6.2 Indoor ROND Dataset

This indoor sequence is processed from [Franco and Boyer (2005)], including walking and hand waving motion patterns. It is captured using 8 video camcorders at 15Hz. Due to this relatively low frame rate, motions between frames are relatively large (some larger

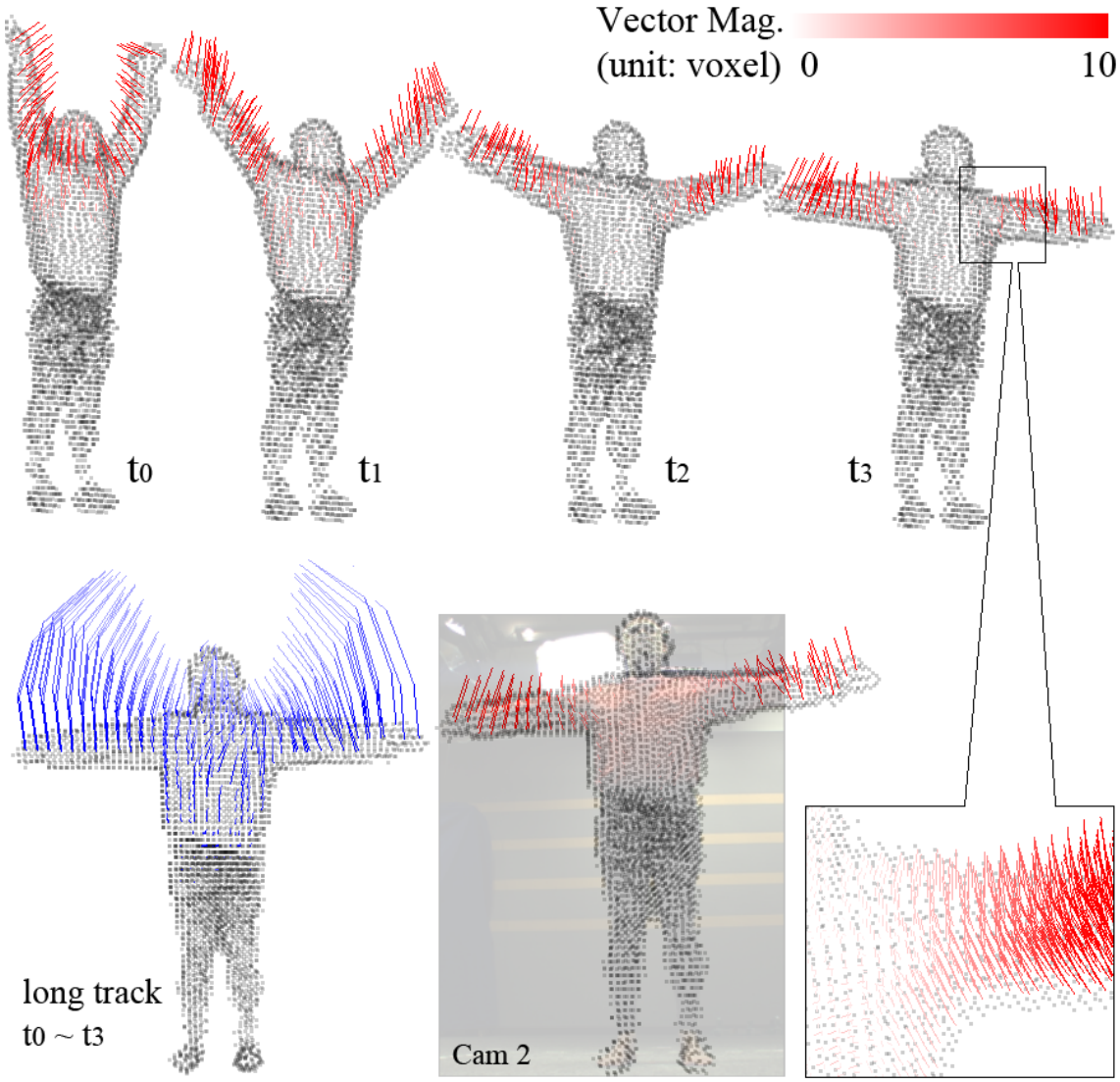

Figure 5.4: Recovered dense flow and the 3D probabilistic occupancy grid. For visualization, we threshold the grid at probability 0.98 to roughly get the person's surface, and only the vectors on this surface are sparsely displayed. The motion vectors are in red, with their magnitudes color coded as top right. The cumulated point tracks over a long time are in blue as bottom left. One of the 8 views is overlaid on t_3's occupancy grid and flow. Part of t_3 is magnified at bottom right. **Best viewed in color.**

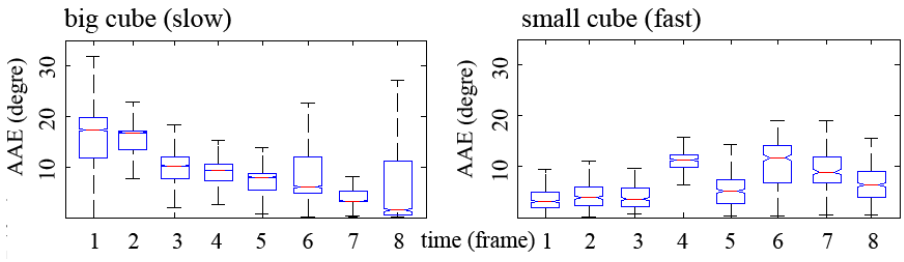

Figure 5.5: The absolute angular error (AAE) of the computed motion vector against the ground truth. The statistics is computed for all points above occupancy probability 0.98. The small and faster cube has more accurate estimated flow in general.

than 5 voxels), but the proposed iterative multi-scale solution recovers large motions correctly. Fig. 5.4 and Fig. 5.6 show the analysis of waving and walking motions in the sequence respectively. The motion tracks are computed by following the computed pairwise motion fields to track the history of some final voxels. The piece-wise linear effect of the motion track is not an artifact of our computation, but shows actual steps between frames, resulting from the combination of strong arm motion and relatively low video capture frame rate.

Fig. 5.7 shows two slices of the occupancy grid at two time instants in the waving sub-sequence. The motion fields in red are overlaid on the probability slices, with higher probability being darker. This shows that the motion fields are computed in all the occupancy grid point locations and probabilistic nature of underlying shape information, which is the biggest difference between our method and the existing 3D surface motion field literatures. Such a representation maintains maximum robustness of the estimations against shadows, occlusion, etc. An example of occlusion case is shown in Fig. 5.4, where the person's arm is outside of the second view, yet the method still successfully recovers arm shape and motion information.

5.6.3 Outdoor SCULPTURE Dataset

This outdoor sequence is processed from [Guan et al. (2007)] with 6 video camcorders running at $30fps$. The cameras are not color calibrated. There are sun light changes,

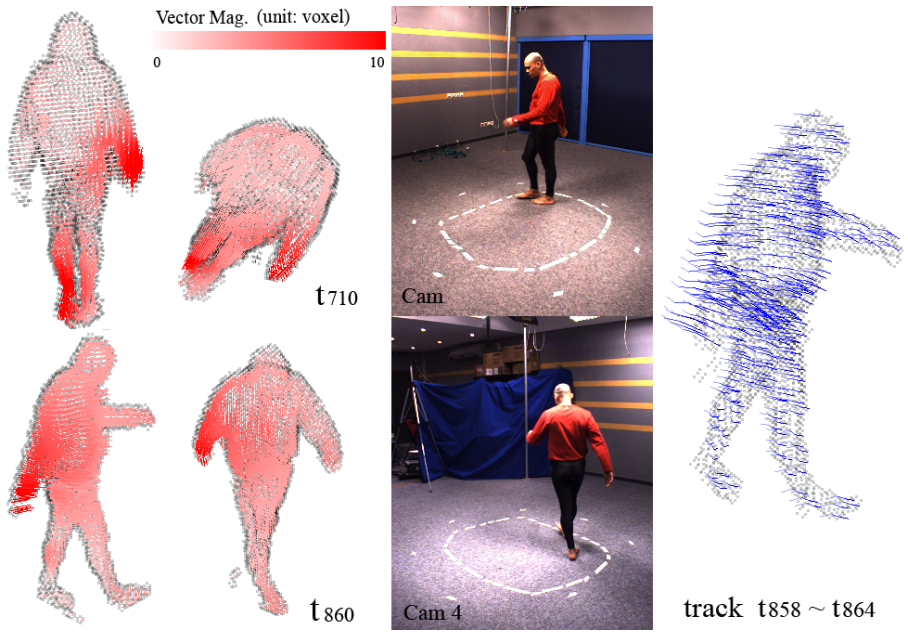

Figure 5.6: Left: two different viewing angles of dense motion field overlaid on top of thresholded (> 0.98) occupancy grid. The two rows are at t_{710} and t_{860} respectively. The motion vector magnitude is color coded. The largest motion happens at the swinging hand and the opposite leg, while walking. Middle: camera images at corresponding time instant. Right: accumulated motion track from t_{858} to t_{864}. The right hand motion is the largest during the interval. **Best viewed in color.**

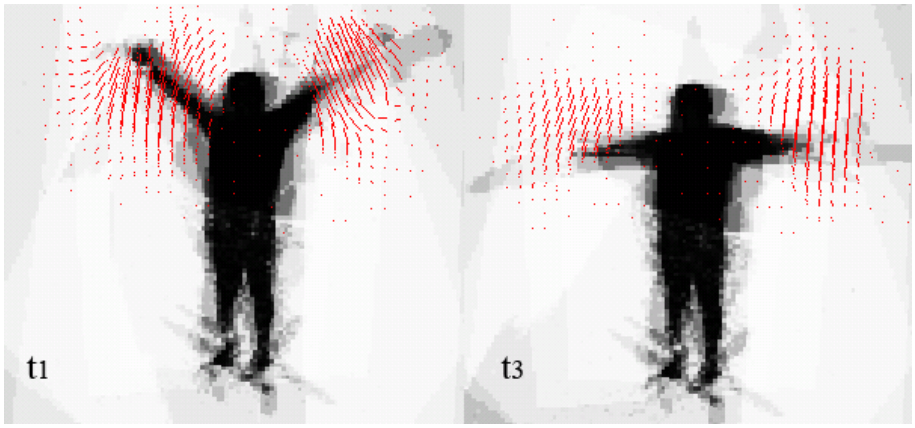

Figure 5.7: Left: occupancy slices at the same position in the volume at two time instants t_1 and t_3. The field is computed on the entire volume. The hands are waving down during the interval. 3D views are shown in Fig. 5.4.

shadows, reflections on the metallic sculpture. Given the noisy data and static background color models used, the computed occupancy grid shown includes high voxel probabilities for unwanted shapes, such as a shadow volume on the ground or sculpture, as visible in Fig. 5.8(a). Nevertheless, the proposed framework is able to recover a coherent shape and dense flow estimate in spite of these underlying geometric incoherences. Such perturbations are likely to lead to failure of boundary-based methods such as mesh tracking approaches.

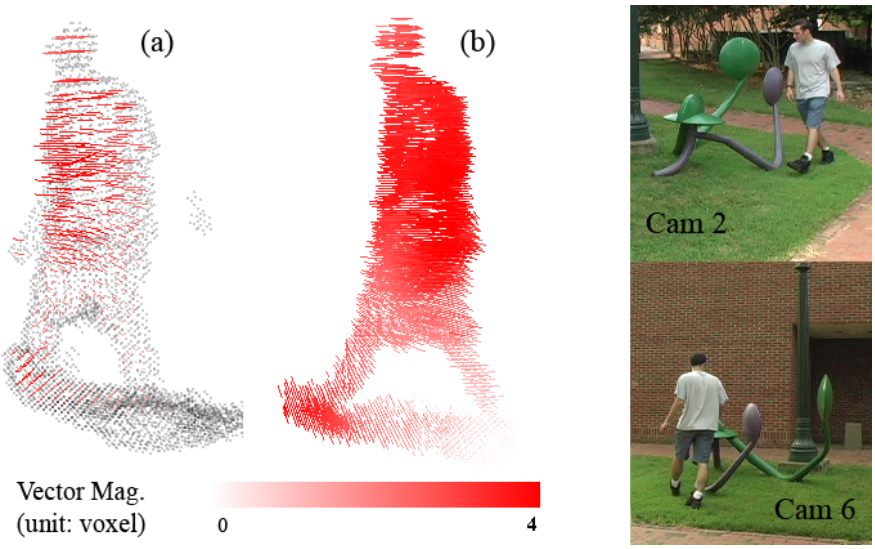

Figure 5.8: Left: estimated occupancy at t_{259} and motion between t_{258} and t_{259}. (a) demonstrates the direction. (b) shows the magnitude. Right: two views at t_{259}. Since the dataset has double frame rates, the color scheme of the velocity magnitude is different from the ROND dataset. **Best viewed in color.**

Fig. 5.9 shows the potential benefits of jointly estimating shape and motion to improve shape estimates. If perfect silhouettes are assumed to be vailable at time t_{258} (manually segmented for the purpose of the experiment), one can help the occupancy estimation at t_{259} and further. By using the proposed method to simultaneously estimate dense motion from t_{258} to t_{259} and occupancy probabilities from t_{259}, occupancy probabilities at t_{258} act as a per-voxel prior, and clean the shadow region and reflection for t_{259} and later frames without using additional appearance models. This suggests the potential for shape refinement across time of the proposed method, and the possibility of

tracking and refining a probabilistic shape template while estimating dense 3D motions.

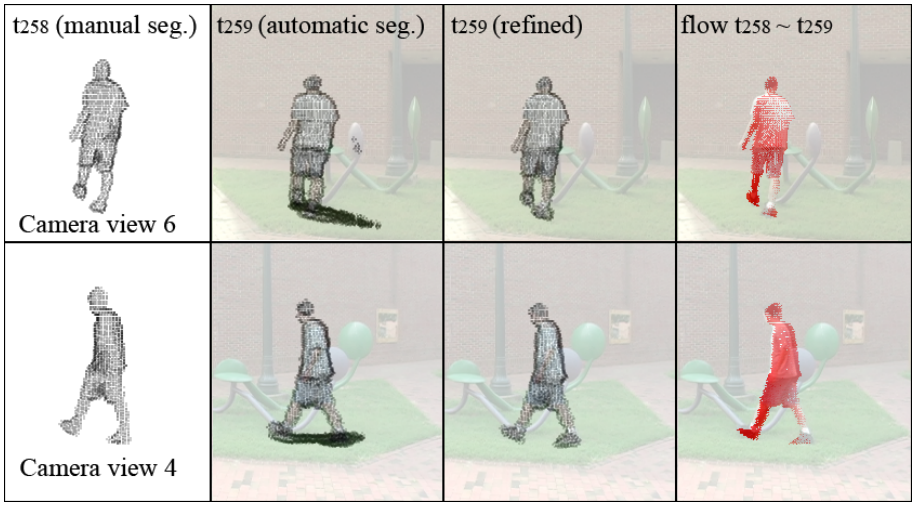

Figure 5.9: Occupancy refinement application. Column 1: the clean t_{258} grid computed from manually segmented silhouettes. Column 2: automatically estimated occupancy at t_{259}. Shadows and reflections are erroneously included due to naive automatically trained appearance models. Column 3: refined occupancy at t_{259} using the clean grid at t_{258} and the computed motion field between t_{258} and t_{259} of Column 4. All the occupancy and motion vectors are only plotted for points above probability 0.98. Column 2, 3 and 4 are overlaid with images at t_{259}. **Best viewed in color.**

5.7 Discussion

A new direction in dense geometric and temporal 3D analysis has been explored, and a novel low-level approach has been proposed for simultaneously estimating 3D dense motion fields and probabilistic shape estimates between two consecutive calibrated view sets, using only silhouette cues. Experiments show the viability and robustness of the approach with various real datasets, and outdoor conditions challenging for photometric and surface-based methods. The method is promising and opens new possibilities and applications. As it relies on no explicit boundary modeling, it can be used as input to a variety of scene analysis tasks, such as motion segmentation with no geometric model or prior, or 3D tracking, kinematic structure inference, shape estimation. Existing shape modeling and tracking methods could use the resulting fields as a cue to replace current

2D optical flow or sparse match inputs without having to explicitly deal with occlusion or premature assumptions associated to an explicit boundary model. Other cues could be included in the temporal analysis, as the Bayesian framework proposed easily allows to perform fusion of heterogeneous cues, such as depth, from stereo or z-cameras, or using other sensor modalities. New temporal shape refinement schemes could be explored by using soft shape priors or using more past observations.

5.7.1 Motion Flow Ambiguity

The computed flow in this chapter is called "3D occupancy flow" instead of general 3D motion flow, not only because it is computed from probabilistic occupancy information inferred from silhouette cues. It is also because some specific motions may not be recovered from this inference, for example self-rotation of a 3D sphere, where the occupancy of the shape is static. This drawback may be alleviated if the surface texture information can be combined in the motion field estimation. But with a uniform colored sphere, it is still going to be a problem. Such degeneracy is also intractable for any vision only based approaches, such as 2D optical flow and 3D scene flow cases.

However, to combine surface textures together with the 3D volumetric motion field estimation is not straightforward, since the proposed approach deliberately assumes no explicit surface during the computation for robustness, and it would therefore require a probabilistic representation of where the surface, hence the surface texture is in the 3D volume. This would further introduce visibility ordering along viewing lines, which complicates the voxel independence assumption, *i.e.* every voxel's state can be independently inferred from its camera view projection appearance. This volume-surface information combination will be a future direction to go.

5.7.2 Motion Discontinuity

There are two aspects of the problem. First of all, it has been assumed that the motion field in the 3D volume is spatially smooth in local regions. Dynamic shapes, especially articulated shapes may have drastically different motion at different parts. When these parts happen to be spatially very close, the above assumption is only valid when the volume resolution is at a even finer granularity. Otherwise, the recovered motion would be "over-smoothed" in the questionable region. Fig. 5.8 is likely to be affected by such issue, where the arm and the torso are close together but with different motion directions and magnitudes.

Instead of a uniform sampling of the 3D space with a regular grid representation, the aforementioned problem may be solved by occupancy-probability-guided sampling scheme, which densely samples the regions more likely to have cluttered different motion parts, while sparsely samples more uniform regions. This would certainly lose the convenient ordered indexing of the regular grid, but helps overcome the "over-smooth" problem.

Second, the temporal motion smoothness is not taken for granted. Although a lot of the motions daily observed are temporally smooth, it is very common in reality, to have sudden motion changes, such as in a hand-waving motion, which the temporal motion smoothness assumption would in fact cause more problems than to be helpful. Therefore, this still remains an open question whether and how to effectively use the temporal motion smoothness constraint in the formulation.

5.7.3 Skeleton Generalization

One interesting and important application is that skeleton model of an arbitrary piecewise rigid dynamic shape (such as a person or a spider) can be generalized given a few such motion fields. With a second EM framework, the parameters of the rigid skeletal parts can be estimated, given the fact that voxels in the rigid body parts should follow

the same motion pattern described by the set of parameters.

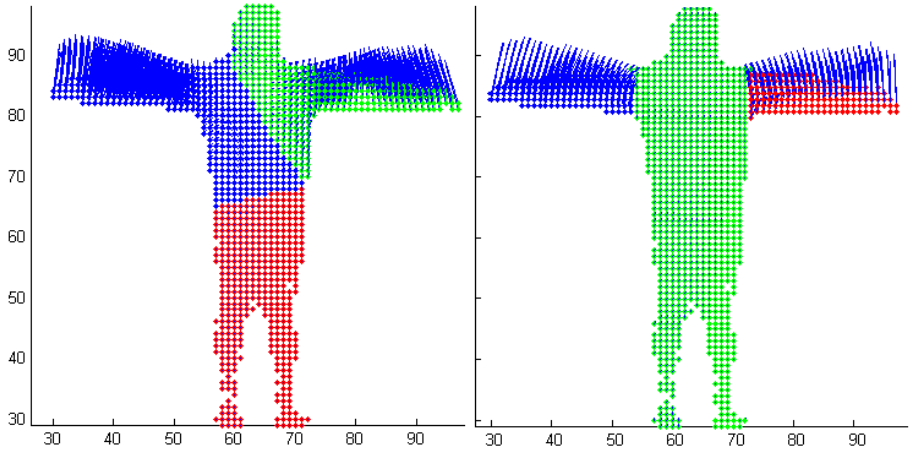

Figure 5.10: Skeleton generalization from dense motion field. Left: initialization of the three-part cluster of the human volume, thresholded at 0.85 probability; Right: the converged three rigid parts and the motion of these three parts.**Best viewed in color.**

As an example shown in Fig. 5.10, the motion field is computed frame 740 to 742 of the ROND hand waving sequence. The field is first thresholded at 0.95 probability, and shown in the left plot of Fig. 5.10. The arms are waving upward, and the motion is shown in blue vectors as well as one of the three body parts. The system is initiated with three rigid body parts, and use MATLAB K-mean clustering to get the three parts (red, green and blue). The dense flow is shown in blue vector fields. The parameters to be estimated for each rigid parts are R the 3D rotation matrix and the T the 3D translation. After 2 iterations of using R and T computation plus Fast-PD label assignment taking into account spatial continuity, the refined three rigid parts are shown in the right plot of Fig. 5.10. The computed Rs and Ts for each of the three parts are shown in blue vector fields.

The rigid parts can be linked together as a complete skeleton model of the person, given more motion fields informations in the following sequences. More detailed rigid part decomposition may also be feasible when enough motion information is acquired. After the full skeleton model is recovered, the dense motion field computation can be only applied on recovered rigid parts, with normally a much smaller parameter space,

thus drastically speeds up the motion field computation. However, the dense motion field computation proposed by this chapter is the key to the novel arbitrary skeleton model initialization.

Chapter 6

Heterogeneous Network for Dynamic Shape Estimation

6.1 Time-of-Flight Sensor

In the previous chapters, the main focus is on the video camcorder network. It has been shown that 3D dynamic shape reconstruction from camcorder networks has many applications such as virtual reality, vision-guided surgeries, medical studies and simulations, video games, architectural design, etc.

However, people have never slowed down the exploration of new sensing technologies. Within the past five years, a promising new technology, Range Imaging (RIM) cameras based on Time of Flight (ToF) principles are coming to market. Swiss Ranger 3000 as shown in Fig. 6.1 is a typical model. 2.5D range images combined with 2D intensity images can be directly read out up to 50 fps. Although most of these ToF cameras do not have high image resolution (*e.g.* 176 x 144 for Swiss Ranger 3000), their measurement throughput is still far beyond the traditional depth sensors, such as LIDAR. This opens enormous potential in a wide range of application areas, including action recognition and tracking, object pose recognition, obstacle detection and so on. Unfortunately, few literatures have explored its potential in 3D object reconstruction. The main challenges are (1) the range images are noisy and not always accurate enough for

3D reconstruction purposes. In fact, the RIM camera depth calibration itself remains a new and active research topic [Kahlmann (2007)]; (2) the relatively low image resolution prohibits detailed reconstruction.

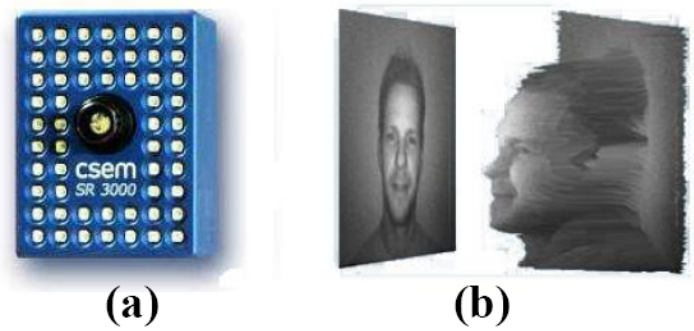

Figure 6.1: (a) ToF camera, Swiss Range 3000; (b) a typical output from the sensor.

This chapter explores the reconstruction potential of the ToF cameras by introducing a heterogeneous sensor network of ToF cameras and high-res camcorder. A unified approach to automatically calibrate such sensor network is first introduced [Guan and Pollefeys (2008)]. Then the Bayesian sensor fusion framework is extended to solve the dynamic shape estimation problem [Guan et al. (2008a)].

6.2 Heterogeneous Sensor Network Calibration

Typically, a ToF camera emits a modulated optical radiation field in the infra-red spectrum. This signal is diffusely backscattered by the scene and detected by the camera. Every CMOS/CCD pixel is able to demodulate the signal and detect its phase, which is proportional to the distance of the reflecting object. Although, currently most of these ToF cameras do not have high image resolution (e.g. 176×144 for Swiss Ranger 3000, as shown in Fig. 6.1), they can generate a 2.5D *depth image* together with an *intensity image* (an *amplitude image* in some literatures) at a frame rate up to $50fps$, which is far beyond the throughput of traditional depth sensors, such as LIDAR.

For such a powerful multi-modal sensor network to work for 3D reconstruction, the first requirement is the geometric calibration of the system. All the literatures related to ToF camera calibration are focused on the single camera extrinsics calibration or depth calibration [Kahlmann (2007); Lindner et al. (2008)]. For camera/camcoder, contrarily, multi-view calibration has been well studied over the past decades.

Thanks to the intensity image generated by the ToF cameras, the traditional checkerboard calibration of the network as introduced in Chapter 2 [Zhang (2007)] would work for such system. But due to the extremely low imaging resolution of the ToF cameras, the checkerboard corner points cannot be robustly extracted, which is a major source of computation noise. In addition, the checkerboard calibration itself is very tedious because not all sensors can see the board at the same time, if they are placed in a circular fashion, hence the complete calibration of the sensor network would require merging all camera views together and extra bundle adjustment to minimize the computation errors.

An alternative solution for multi-view camera/camcorder calibration is to use a single 3D laser point [Svoboda et al. (2005)] to recover the parameters up to the global scale ambiguity, or a rigid grid of 3D points [Uematsu et al. (2007)] to recover the full Euclidean space. In these cases, all the cameras can see the calibration target simultaneously, thus solve the second problem of the previous planar calibration target approach. But because of the low image resolution, the detection of this type of calibration targets is not always robust in the ToF camera images. What is worse, since most of the active sensing ToF cameras are filtering out light frequencies that were not sent out from the cameras themselves, it might be very possible that the ToF cameras cannot detect any laser point dots at all.

In this section, a new calibration approach is proposed [Guan and Pollefeys (2008)], which follow the laser point global calibration scheme, but instead of using a moving laser dot or a grid of rigidly connected points, a moving sphere of an unknown ra-

dius is used as the calibration target. Literatures [Agrawal and Davis (2003); Ying and Zha (2006); Zhang et al. (2007)] also propose to use a spherical calibration target for a camera/camcorder network. The main difference is that they use the complete sphere contour information constrained by the absolute conic. However, due to the low resolution of the ToF camera images, the sphere contour extraction is risky for the heterogeneous setup discussed here.

Whereas it is shown later in the section that the sphere center can be robustly extracted not only from the high-res camcorder frames, but also the low-res ToF frames, based on the original observation that *in a ToF camera intensity image, a sphere center is always highlighted*. The highlight is due to the ToF camera active sensing mechanism, the surface reflectivity of the sphere and the spherical surface normal direction property. Real images from an SR3100 ToF camera are shown in Fig. 6.3 and Fig. 6.6. After the sphere center locations are extracted, an automatic bundle adjustment similar to Svoboda et al. (2005) can be performed recovering the global extrinsic camera poses and sphere center 3D locations. Given the extra depth information from the ToF cameras, the global scale and the full Euclidean space can be recovered.

6.2.1 Unified calibration scheme Overview

A sphere with an unknown radius is moved around in the common viewing space of the sensor network. Assume the intrinsic parameters of the sensors are known, either from factory specification of the cameras or through a pre-calibration [Zhang (2007)]. Thus the image radial distortion can be removed. By extracting the sphere center from the synchronized video frames from all sensors, a bundle adjustment over sensor extrinsic parameters and sphere 3D locations can be performed to solve the system up to a scale factor. Then, the global scale ratio can be recovered using the ToF sensor depth measurement and the already-computed 3D sphere centers locations in the similarity space. Notice that this scale ratio is not taken into account during the geometry bun-

dle adjustment, because the ToF camera depth measurement sometimes can be very noisy. Therefore, it is not recovered during the bundle adjustment optimization, but is recovered in a separate step.

6.2.2 Sphere center detection

Camera/camcorder

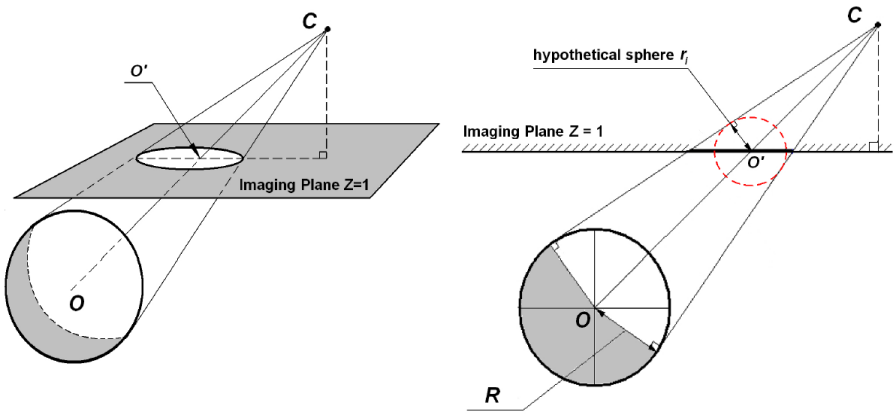

Figure 6.2: Left: A sphere projection on to the image of a camera centered at C is in general an ellipse, due to the projective distortion. The sphere center's projection O' is also not at the ellipse center. Right: a 2D side view of the same configuration. The "hypothetical sphere" located at $(x_i, y_i, 1)^\mathsf{T}$ is shown as the red dotted circle with radius r_i.

After the radial distortion is removed, the 2D image of a sphere is an ellipse [Agrawal and Davis (2003); Ying and Zha (2006); Zhang et al. (2007)], as illustrated in Fig. 6.2. Hough transform is applied for robust ellipse detection. In general, an ellipse is defined by five parameters. So the Hough space is 5D. However, since the camera principal point is known, one can unambiguously define an ellipse in an image i by the sphere radius R and the viewing ray from the camera optical center to the sphere center, namely vector $\langle x_i, y_i, 1\rangle$. But since R is unknown, an alternative way is needed to describe the radius. Given that the intrinsics are known, for every image i, a "*hypothetical sphere*" can be introduced located at $(x_i, y_i, 1)^\mathsf{T}$ – one can think of a sphere located on the $Z = 1$ plane in the 3D space – with a varying radius r_i, such that this new sphere results in the

same elliptical image projection as our *real* radius R sphere, as shown in the right plot of Fig. 6.2. Therefore, the problem actually has only an (x_i, y_i, r_i) 3D Hough space, which makes the computation easier. Practically, the "edgeness" cue and the color of the sphere are used to guide the Hough transform. To further exploit the temporal consistency between neighboring image frames, given $(x_{i-1}, y_{i-1}, r_{i-1})$ in frame $i-1$, a simple tracking scheme is applied to constrain the local search window for detection in frame i.

ToF camera

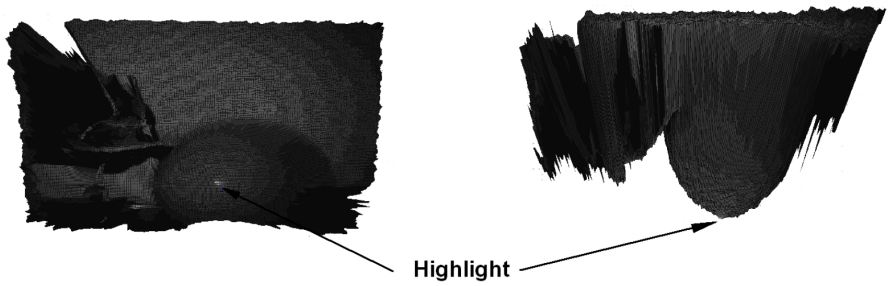

Figure 6.3: Left: A typical intensity image of a Swiss Ranger 3100 mapped on to its depth mesh. The highlight is due to most of active light reflection in a local region. Right: A top view of the same depth mesh. The highlight is along the viewing ray through the camera optical center and sphere center.

A sphere center in a ToF camera image should not be extracted in the same way as a camera/camcorder, because of the extremely low image resolution and relatively bad sphere edge contrast. However, thanks to the ToF camera active sensing mechanism, an even simpler way is available to find a sphere center.

Most of the current ToF cameras (e.g. Swiss Rangers, PMD sensors and Canesta cameras) have LEDs evenly distributed around the camera lens. The active light can thus be think of as from a single virtual "*point light source*" located at the camera optical center. Therefore, assume a Lambertian sphere surface, which has the property that *the witnessed brightness in an image depends only on the angle between the surface normal and the light source*, the brightest pixel in the intensity image is the one that

lies on the line connecting the camera optical center (the virtual "*point light source*" position) and the sphere center, which by definition is the sphere center's projection in the image, as shown in Fig. 6.4. In fact, the Lambertian assumption is not necessary, since the specularity of the surface even strengthens the highlight, due to the overlapping virtual light source with the camera optical center. Both Fig. 6.3 and Fig. 6.6 show this phenomenon in a real SR 3100 image.

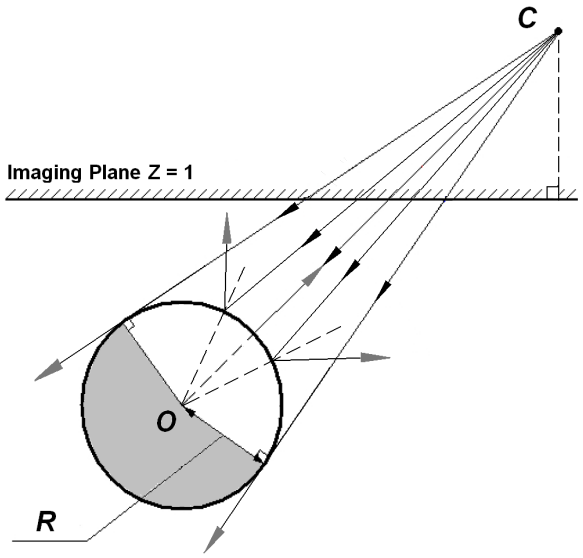

Figure 6.4: Explanation of sphere center coincide the ToF image highlight. The highlight locates at the normal direction to the camera, which is the same as the direction going through the sphere center.

Some might point out that strictly speaking the assumption that it is a virtual "*point light source*" does not hold, because the light source is not a point (but a network of LEDs), the non-perfect shape and texture of an actual spherical object, and aliasing effects. But instead of blame the low resolution, one should thank that the observed highlight blob is so small that only with in the size of a pixel and the questioned assumption still holds, as long as the sphere is far away enough from the imaging plane and the relative sphere radius is much bigger than the dimension of the light source array.

Also some might have noticed that the depth image from the ToF cameras as well

gives some hint to the sphere center's image location, but there are at least two reasons that it is not computed from the depth cue. First of all, the closest point on the sphere depth mesh is usually NOT the sphere center location, unless the sphere center goes through the principal axis. Thus the computation would not be as easy and direct as the following proposal. Secondly, the depth image is noisy, as shown in Fig. 6.3.

Here comes the actual proposal: To recover the sphere center's image location in each ToF frame, one can detect and track the highlight in the intensity image only. A paraboloid is fitted to achieve sub-pixel accuracy, given the intensity values around the found maximum location.

6.2.3 Recover the extrinsics via bundle adjustment

Given the intrinsics and sphere center's image locations in the synchronized frames from all the views, one can now perform a global bundle adjustment similar to [Svoboda et al. (2005)] to recover the camera poses and sphere 3D locations. The intrinsics of the video camcorders are pre-computed using [Zhang (2007)]'s method. The intrinsics of the ToF cameras are obtained from the factory manual. But the intrinsics can be also put into the bundle for a further refinement. Due to the heterogeneity of the sensors, namely the image resolutions are very different and the sphere center extraction methods as just described are very different (The uncertainty of the camcorder Hough transform is related with the sphere boundary extraction, intrinsic optical center computation, radial distortion correction and Hough space resolution; The uncertainty of the ToF camera sphere center extraction is related with image noise and motion blur.), to minimize the algebraic error (pixel re-projection error) is meaningless. Therefore, the bundle adjustment error metric is defined to be the angular re-projection error [Oliensis (2002); Hartley and Schaffalitzky (2004); Ke and Kanade (2005)], *i.e.* the angle θ between the observed ray $\boldsymbol{x}$ and the re-projection ray $\boldsymbol{r}$:

$$f(\boldsymbol{X}) = |\tan(\theta)| = \left| \frac{\boldsymbol{x} \times \boldsymbol{r}}{\boldsymbol{x}^\top \boldsymbol{r}} \right|. \tag{6.1}$$

This overcomes the image resolution difference. Since it is hard to model the methodological distinction between the two sphere center extraction approaches, for now assume the two methods have equal uncertainties, thus this issue is ignored in the rest of the chapter.

6.2.4 Recover the sphere radius R and global scale S

After the bundle adjustment, one can compute the relative distance from each sphere center 3D location to the camcorder optical centers. And since r_i is known from the Hough transform, for each camcorder image i, the 3D sphere radius R_i can be computed by similar triangle analysis, as shown in the right plot of Fig. 6.2. For the real sphere radius R, it just takes the mean of all R_i s. Notice that R is still in the similarity space, but not the metric radius of our calibration target.

To recover the global scale S, the depth measurements D_i at the sphere center's pixel position is read out from the ToF depth images. And suppose the relative distance from the sphere center to the ToF camera optical center is d_i, the expression below can be derived. Detailed implementation is described in Section 6.2.5 with a real dataset.

$$D_i = (d_i - R) \cdot S. \tag{6.2}$$

6.2.5 Calibration Result and Evaluation

Experiment setup

In this section, a real dataset consists of two Canon HG10 camcorders and two Swiss Ranger 3100 ToF cameras is evaluated. The camcorders are set to run at 25 fps with an image resolution of 1920×1080 pixels. The Swiss Rangers are set to run at 20 fps

with an image resolution of 176 by 144 pixels. Although many delicate approaches could be applied, the four views are synchronized by temporally sub-sampling the frames at 5 fps. The calibration target is a yellow gymnastic ball. The four sensors locate on a rough circle, looking inward at a common free space. The two ToF cameras are set at different modulation frequencies, *i.e.* 19 MHz and 20 MHz, so that the active lights are not interfering with each other. And this gives a minimum unambiguous depth range of 7.1 meters, which well satisfies the current scene modeling application.

Sphere center extraction

The sphere centers are extracted using the described methods in the previous section. The camcorder image Hough transform is illustrated in Fig. 6.5. And the ToF camera sphere center highlight is shown in Fig. 6.6. The extracted sphere center locations for all four views are shown as green dots in Fig. 6.7. In order to get an unbiased and robust calibration, the sphere is intensionally move to sample the 3D space as uniformly as possible.

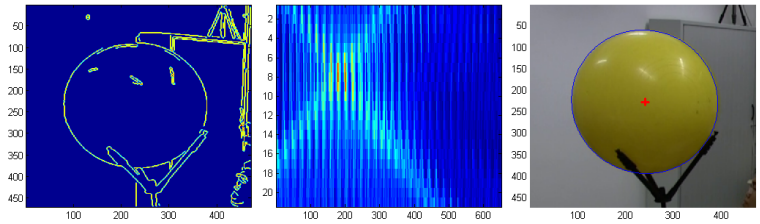

Figure 6.5: Camcorder ellipse extraction. Only cropped images are shown. **Best viewed in color.** Left: A thresholded gradient magnitude image. Middle: 2D projection of the (x_i, y_i, r_i) 3D Hough space. A single solution is found, at the crossing. Right: The recovered ellipse and the sphere center overlaid on the original image.

Bundle adjustment evaluation

After the bundle adjustment, the recovered camera configuration and 3D sphere center locations are shown in the left plot of Fig. 6.8. The plot on the right is the re-projection error statistics in the box-and-whisker diagram from our bundle adjustment result. A

Figure 6.6: ToF camera Swiss Ranger 3100 intensity image sphere center highlights. To detect the highlight robustly and automatically, a simple Region of Interest (ROI) tracking method is applied.

different way to evaluate our recovered poses is to project the camera centers to different camera views, as shown in the yellow crosses in Fig. 6.7. One can see that the projected camera centers well overlay their images from a different view as expected.

For completeness, the physical projection matrices of the four cameras in the setup are listed in Tab. 6.1 (sensor internal and external parameters) and the full dataset used for reconstruction in Section 6.3 is available upon email request.

Table 6.1: Recovered camera projection matrices with our method.

Cam #1	-69.098	123.3147	152.3223	340.4835
	-176.7834	-83.9725	46.8359	103.5043
	-0.0888	-0.3609	0.9284	1.2298
Cam #2	-799.49	1401.3	286.95	2048.2
	-1418.8	-360.37	-123.77	416.72
	-0.56033	0.18502	0.80743	1.7704
Cam #3	380.59	-1900.0	115.72	-1140.6
	-1716.4	-704.98	-151.45	296.97
	-0.14273	-0.54063	-0.82907	-1.0773
Cam #4	25.9357	-129.1248	-167.5592	-304.4062
	-209.1743	-37.0983	-45.5719	31.5774
	-0.3153	0.42896	-0.84651	-0.42014

Attempts also have been made to actually calibrate the system with a planar checkerboard pattern for comparison, but fail to link the camera pairs (#1, #2) and (#3,#4) together. Because as seen in Fig. 6.8, view #1 is almost opposite to view #3, so as #2 to #4, which unfortunately is one of the extreme cases having been discussed before.

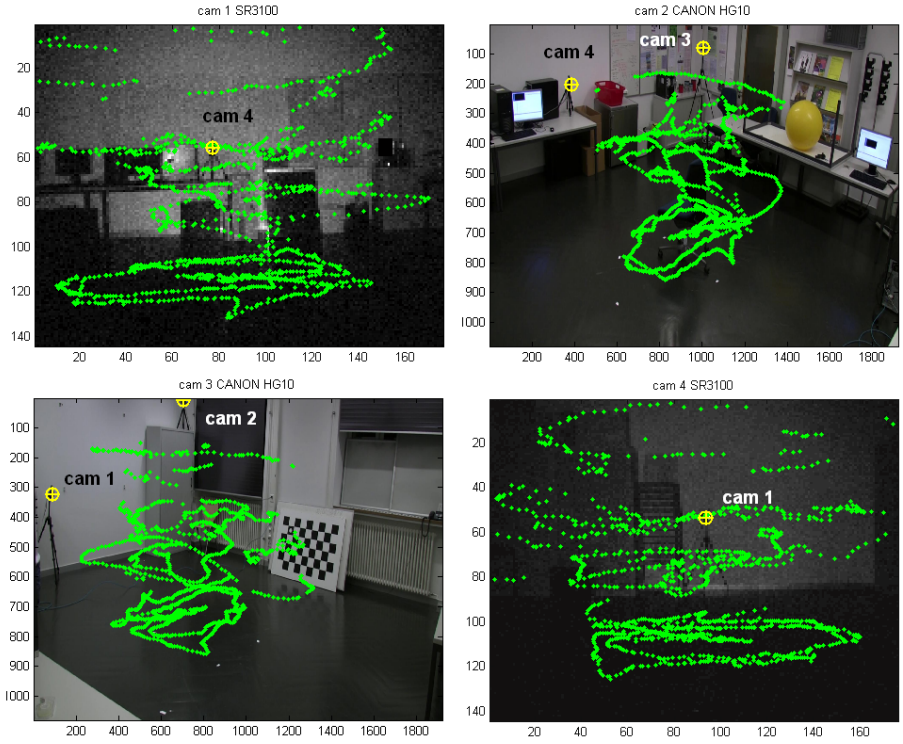

Figure 6.7: The extracted sphere center's image locations are shown in green dots. The sphere samples cover the 3D space as much as possible. After the bundle adjustment, the recovered camera centers are re-projected to the images as yellow crosses. They overlap very well with the camera image, showing that the system is well calibrated.

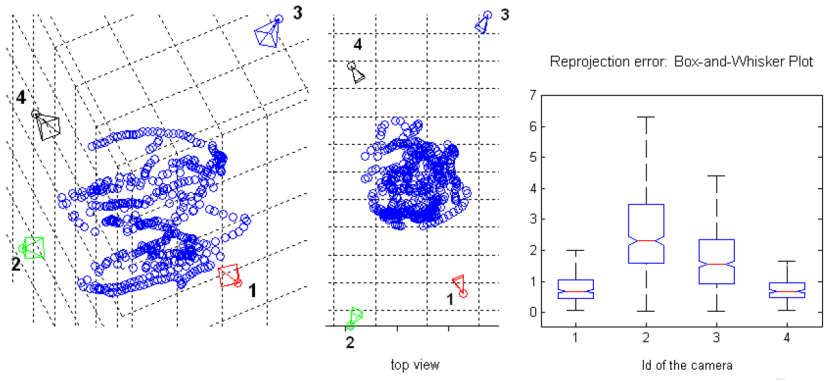

Figure 6.8: Left: The recovered sphere center's image locations for all four views. Right: Image re-projection error statistics. Cam #1 and #4 are ToF cameras. Cam #2 and #3 are video camcorders. The re-projection errors for the ToF cameras are significantly smaller than those of the video camcorders, whose image resolution is much higher. This shows our bundle adjustment does not have a bias to the resolution difference.

This again shows the advantage of the proposed calibration approach.

Sphere Radius and Global Scale Recovery

Given the recovered 3D structure and camera poses, and the hypothetical sphere radius r_is, the mean radius can be computed $R = 0.0248$, simply by similar triangle inference, as shown in the right plot of Fig. 6.2. Note again that R is not the true sphere radius, but in the recovered similarity space.

To recover the absolute scale S, one can re-write Eq. 6.2 as a minimization problem:

$$arg\min_{S} \sum_{i} |D_i + R \cdot S - d_i \cdot S| \,. \tag{6.3}$$

Given the dataset, one can solve $S = 11.3886$, and the absolute sphere radius $R' = R \cdot S = 0.2824\ m$. The actual sphere circumference is measured to be $1.7925\ m$, namely the measured sphere radius is $0.2853\ m$, which is very close to the above computation.

6.3 Heterogeneous Sensor Network Fusion

The depth information and silhouette cues have been explored separately for 3D reconstruction purpose. Both have their own advantages and drawbacks. For the depth information, it can give actual object surface patches. But due to self-occlusion, individual patches only provide a partial model of the object surface, so one of the many challenges is to deal with missing patches and fill up the holes so as to get a topologically correct object shape [Bajaj et al. (1995); Curless and Levoy (1996); Hilton et al. (1998); Davis et al. (2001); Casciola et al. (2005)]. On the other hand, reconstruction from silhouette cues [Baumgart (1974); Laurentini (1994); Szeliski (1993); Lazebnik et al. (2007); Franco and Boyer (2009)] are praised for a closed shape estimate of the object. And recently even no hard decision binary silhouette images are required for a robust probabilistic visual hull reconstruction [Franco and Boyer (2005)], introduced in Section

2.4.2. As discussed, an inherent drawback of a visual hull is that it cannot recover object concavities no matter how many views of silhouettes are provided. However, this can be directly compensated by the depth information. In fact, object depth and silhouette are quite complementary information in nature: the former encodes lights bouncing back from the frontal surfaces; and the latter is tangent to the object. So in theory, these two could be combined to improve the reconstruction quality. Additionally, in the heterogeneous sensor network, the shape details can be recovered with the high-resolution camcorder frames to compensate the low-res ToF camera images.

However, silhouette and depth integration is not straightforward due to the heterogeneity of the information. [Li et al. (2002)] try to tackle the problem with pure surface representation, which requires a lot of delicate handling of geometry computation errors. As an alternative, volumetric fusion can be favored to avoid topological problems [Kampel et al. (2002); Sablatnig et al. (2002); Yemez and Wetherilta (2007)], but these methods are all based on deterministic criteria, which have to specifically deal with sensor noise perturbations.

In order to achieve a more robust but also more general solution to the fusion problem, similar to [Franco and Boyer (2005)], the framework discussed in this section [Guan et al. (2008a)] borrows the concept of a space occupancy grid from the robotics literature [Elfes (1989); Margaritis and Thrun (1998); Pathak et al. (2007)] as the representation of 3D scenes. After defining the probabilistic sensing models for each type of sensors, the reconstruction simply becomes a Bayesian inference. The reconstruction result is a posterior probability volume given sensor observations. It is inherently robust and requires no special treatment regarding sensor noise, because the noise and variation is already part of the probabilistic sensing models. One thing to note is that the proposed framework is not limited to the fusion between silhouette cues and depth maps, but any type of sensor observations such as point clouds and disparity maps, as long as the sensing model can be properly defined.

6.3.1 Problem Formulation

After the sensor network calibration as discussed in Section 6.2, with the following notations, the problem formally can be defined formally: given a set of synchronized observations $\mathcal{V}$ from n sensors at a specific time instant, one can infer for every 3D location X in an occupancy grid $\mathcal{G}$ expanding the 3D space its probability of being occupied or not by the dynamic object being modeled. And this probability is denoted as $p(\mathcal{G}_X)$ with $\mathcal{G}_X$ the binary variable at X. Because of the heterogeneity of the sensors in the network, the sensing models are denoted as $\mathcal{M}$.

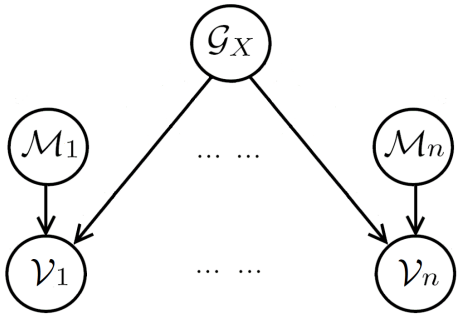

Figure 6.9: General sensor network system dependency.

The space occupancy variable $\mathcal{G}X \in \{0, 1\}$ depends only on the information along optic rays that go through X. However, anti-aliasing effects need to be considered. the same sampling window strategy introduced in Section 2.4.2 [Franco and Boyer (2005)] is used, where a certain 3D voxel affects the formation of pixels within the sampling window similar to a point spread function. Another common occupancy grid assumption is the statistical independence between voxel occupancies. Each voxel occupancy likelihood is computed independently for tractability. Therefore, the sensor network relationships are modeled as computing the joint probability of these variables, $p(\mathcal{G}_X, \mathcal{V}_{1,\ldots,n}, \mathcal{M}_{1,\ldots,n})$, and the following decomposition is proposed, based on the statistical dependencies expressed in Fig. 6.9:

$$p(\mathcal{G}_X, \mathcal{V}_{1,\ldots,n}, \mathcal{M}_{1,\ldots,n}) = p(\mathcal{G}_X) \prod_{i=1}^{n} p(\mathcal{M}_i) p(\mathcal{V}_i | \mathcal{G}_X, \mathcal{M}_i) \tag{6.4}$$

- $p(\mathcal{G}_X)$ is the prior likelihood for occupancy, which is independent of all other variables except τ. It is chosen not favor any voxel location and is set to uniform.

- $p(\mathcal{V}_i|\mathcal{G}_X, \mathcal{M}_i)$, or more specifically, given the aforementioned viewing-ray independence assumption, $p(\mathcal{V}_i^p|\mathcal{G}_X, \mathcal{M}_i^p)$ represents the sensor observation probability.

Once the joint probability distribution has been fully determined, it is possible to use Bayes rule to infer the probability distributions of our searched variable $\mathcal{G}_X$, given the sensor models $\mathcal{M}$ and their observations $\mathcal{V}$.

$$p(\mathcal{G}_X|\mathcal{V}_{1,\ldots,n}, \mathcal{M}_{1,\ldots,n}) = \frac{\prod_{i=1}^{n} p(\mathcal{V}_i^p|\mathcal{G}_X, \mathcal{M}_i^p)}{\sum_{\mathcal{G}_X} \prod_{i=1}^{n} p(\mathcal{V}_i^p|\mathcal{G}_X, \mathcal{M}_i^p)} \tag{6.5}$$

If Eq. 6.5 is applied for all locations and obtain this probabilistic volume $\mathcal{G}$, the 3D objects can be simply reconstructed by extracting iso-probability surfaces, or more robustly using state-of-the-art techniques, such as Graphcut/Levelset algorithms [Snow et al. (2000); Whitaker (2004)]. The remaining problem is to define the proper sensor models $\mathcal{M}$ so that the observation formation $p(\mathcal{V}_i|\mathcal{G}X, \mathcal{M}_i)$ in Eq. 6.5 is reasonable. But so far, a very general sensor fusion framework is introduced, which has no constraints on the sensor type nor data type.

ToF Camera Sensor Model

The probabilistic camcorder background model $\mathcal{B}$ is introduced in Section 2.4.2. For a ToF camera, the observation $\mathcal{V}_i^p$ is the depth measurement. Similar to the silhouette variable $\mathcal{S}_i^p$ in camcorder sensor, here a latent variable $\mathcal{T}_i^p$ is also introduced, to model the front most surface of the object with respect to the ToF camera. The relationship between sensor variables is shown in Fig. 6.10. Basically, the existence of an object at $\mathcal{G}_X$ affects the front most surface location $\mathcal{T}_i^p$ to a certain ToF camera i. And $\mathcal{T}_i^p$ affects the depth measurement directly.

Because $\mathcal{T}_i^p$ is a latent variable, it needs to be marginalized. However, $\mathcal{T}_i^p$ is not a

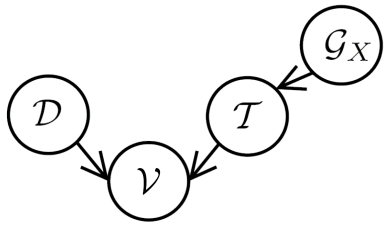

Figure 6.10: ToF camera dependency.

binary variable as its counterpart — the silhouette $\mathcal{S}_i^p$ for a camcorder, but its range expands all possible locations along the viewing direction. Namely, $\mathcal{T}_i^p \in [0, d_{max}]$, with 0 being the ToF camera optical center, and d_{max} the largest detectable distance of the ToF camera.

$$p(\mathcal{V}_i^p|\mathcal{G}_X, \mathcal{M}_i) \tag{6.6}$$

$$= \int_0^{d_{max}} p(\mathcal{V}_i^p|\mathcal{T}_i^p, \mathcal{D}_i^p) p(\mathcal{T}_i^p|\mathcal{G}_X) d\mathcal{T}_i^p \tag{6.7}$$

- $p(\mathcal{V}_i^p|\mathcal{T}_i^p, \mathcal{D}_i^p)$ is the **depth measurement term**. It depicts how precise the ToF camera depth measure is. A normal distribution $\mathcal{N}(\mathcal{T}_i^p, \sigma)$ is used to model it, where σ is trained from depth calibration process or obtained from the camera manual.

- $p(\mathcal{T}_i^p|\mathcal{G}_X)$ is the **surface formation term**. Assume every voxel is independent along the viewing direction of length d_{max}, and any place on the viewing ray has an equal chance of $1/d_{max}$ to be the front most point. Now, if $\mathcal{G}_X = 1$, the front most surface position $\mathcal{T}_i^p$ still has a chance of $1/d_{max}$ to be at any position in front of X, namely $\mathcal{T}_i^p < d_X - \epsilon$, where $\epsilon \to 0$. But this is not the case for the positions behind X, because X is already blocking the viewing ray. Eq. Eq. 6.8 & Eq. 6.9 shows the complete scenario, with d_X being the distance from X to the RIM camera. Both distributions of $p(\mathcal{T}_i^p|[\mathcal{G}_X = 1])$ and $p(\mathcal{T}_i^p|[\mathcal{G}_X = 0])$ must sum up to 1.

$$p(\mathcal{T}_i^p|[\mathcal{G}_X = 1]) \tag{6.8}$$
$$= \begin{cases} 1/d_{max} & \text{if } \mathcal{T}_i^p < d_X - \epsilon \\ (1 - d_X/d_{max})/\epsilon & \text{if } d_X - \epsilon \leq \mathcal{T}_i^p \leq d_X \\ 0 & \text{if } \mathcal{T}_i^p > d_X \end{cases}$$

$$p(\mathcal{T}_i^p|[\mathcal{G}X = 0]) = 1/d_{max} \tag{6.9}$$

To get an intuitive idea of the ToF camera model, imagine a single pixel ToF camera, with the depth detection standard deviation $\sigma = 0.3m$ and maximum detection range of $8m$. If the current sensor readout is $5.0m$, according to the RIM sensor model, one can plot out the space occupancy probability $p(\mathcal{G}_X|\mathcal{V}, \mathcal{D})$ along the viewing ray as in Fig. 6.11, given Eq. 6.4, 6.5 & 6.6-6.9. This means the object is most likely existing at $5m$, the observed depth region. Regions in front of it should be free of any object and visible up to the camera. Regions behind $5m$ remains total uncertainty, 0.5, because one has no clue whether there is matter behind the surface or not. The peak falls smoothly on both directions, because of the limited sensor precision. This plot is consistent with the depth sensor models described in other literatures such as [Coué (2003); Pathak et al. (2007)].

6.3.2 Sensor Fusion Experiment and Result

Two sets of data are acquired to test the proposed calibration and heterogeneous sensor framework. Without losing generality, for the camcorders and ToF cameras, we use Canon HG10 and Swiss Ranger 3100 respectively. Canon HG10 DV camcorders are set to run at 25 fps with an image resolution of 1920 by 1080 pixels. Swiss Ranger 3100

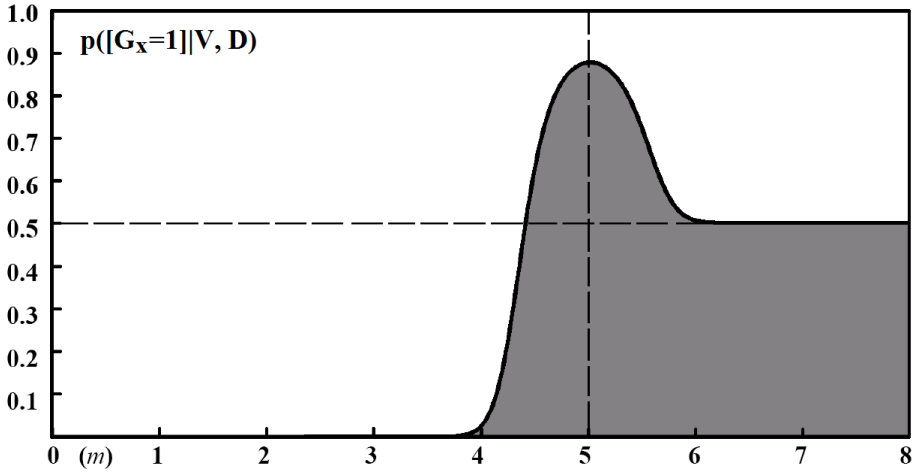

Figure 6.11: Space occupancy probability $p(\mathcal{G}_X|\mathcal{V},\mathcal{D})$ at certain distances given the ToF camera readout of $5.0m$. It is a longitudinal cut of what probabilities look like on one viewing ray in the grid.

are set to run at 5 fps with an image resolution of 176 by 144 pixels. The dataset specifications are listed below. For SENSOR NETWORK 2, in order to prevent the interference between multiple ToF cameras, their modulation frequency are manually set at 19MHz, 20MHz and 21MHz respectively. Although this setting will affect the maximum detection depth of each camera, the minimal range $7.1m$ [Imaging (2007)] is still beyond the reconstruction volume range, $6m$. Both datasets use an occupancy volume.

	Canon HG10	SR 3100	volume size
Sensor Network I	3	1	$128 \times 256 \times 128$
Sensor Network II	6	3	$128 \times 128 \times 128$

SENSOR NETWORK I result

Two static shapes are reconstructed from this 4 sensor setup: an office chair with two boxes and a sitting person. The output of the alrogithm is a probabilistic volume, for visualization purpose, the volume surfaces are extracted at an arbitrary iso-probability of 87%, and the results are shown in Fig. 6.12 and Fig. 6.13. The reconstructions from the proposed framework preserve detailed concavity and significantly improve the

quality of the result from the 3 camcorder only (the probabilistic visual hull). More delicate surface extraction schemes can be applied to get better object shapes, but this is beyond the scope of this paper.

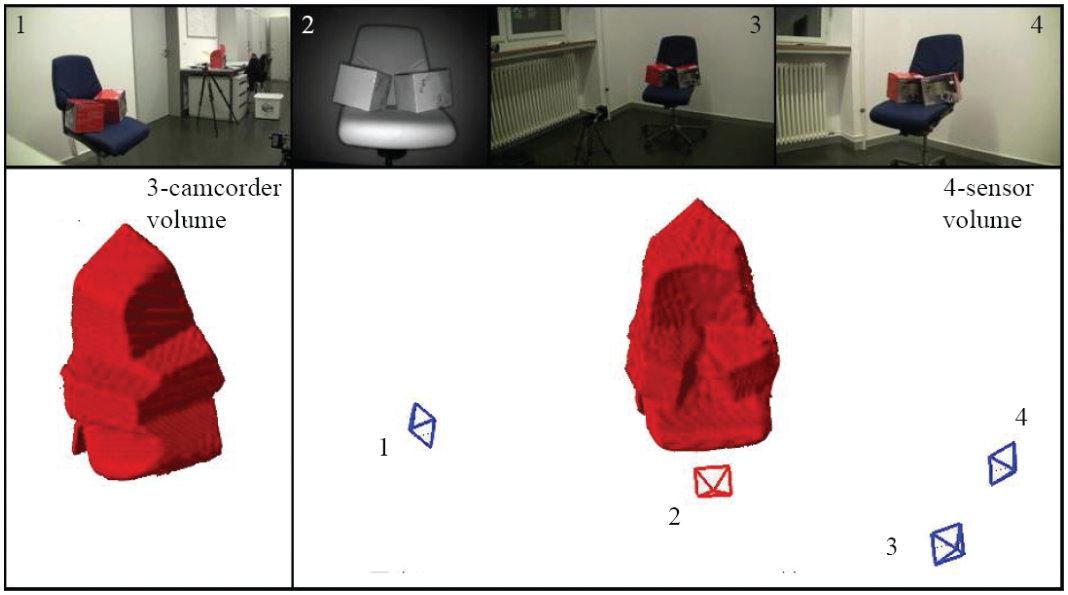

Figure 6.12: An office chair with two boxes. Top: the four camera views Bottom: 3-camcorder probabilistic visual hull and 4-camera fusion result with our proposed algorithm. The calibrated camera configuration is also shown here, with #2 the SR3100, and #1, 3 and 4 the Canon HG10.

SENSOR NETWORK II result

This 9 camera network generates two reconstructions: a person with a rubber ball, and a crowd of 5 people. The number of cameras in use is not designed on purpose, instead is based on the number of sensors available. Admittedly though, more detailed information can be obtained with more sensors, and it really helps in challenging cases such as very cluttered scenes. The results are shown in Fig. 6.14 and Fig. 6.15 respectively. The camera calibration procedures are the same as the previous dataset. And the recovered camera poses are shown in Fig. 6.14, with red cones denoting three SR3100. The reconstructed ball in Fig. 6.14 has a diameter of 60cm, which is pretty close to the actual value is 57.06cm given the low volume resolution. This again shows the power

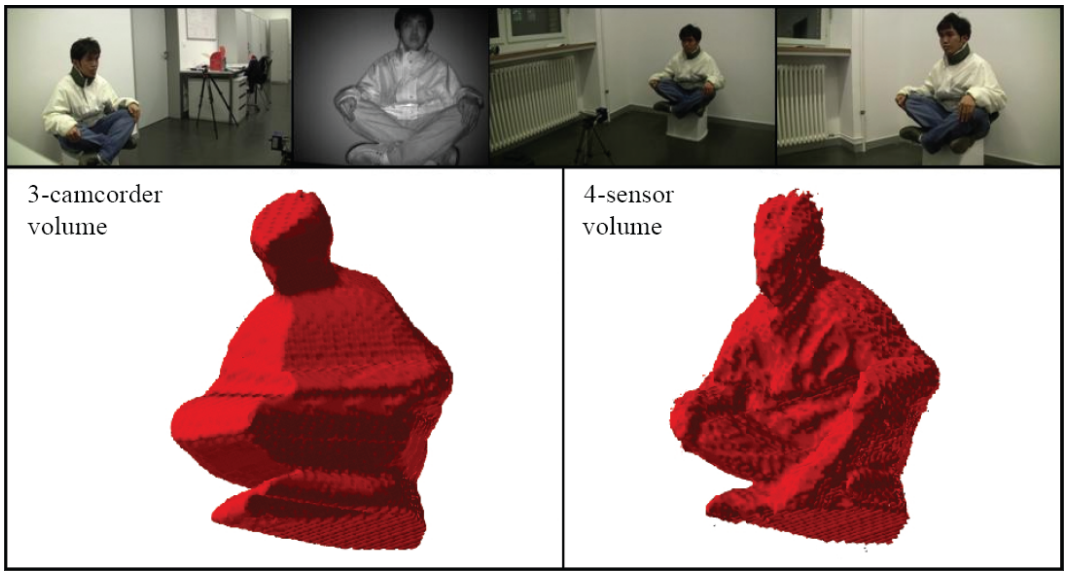

Figure 6.13: A sitting person. The same configuration with Fig. 6.12.

of our depth calibration. A more challenging example is Fig. 6.15, where 5 people are highly clustered in the space. Without the depth information to recover the concavities the visual hull would fail the reconstruction task. One thing to note is that the missing forearms are sub-voxel size. They can be recovered if the volume resolution is increased at those places.

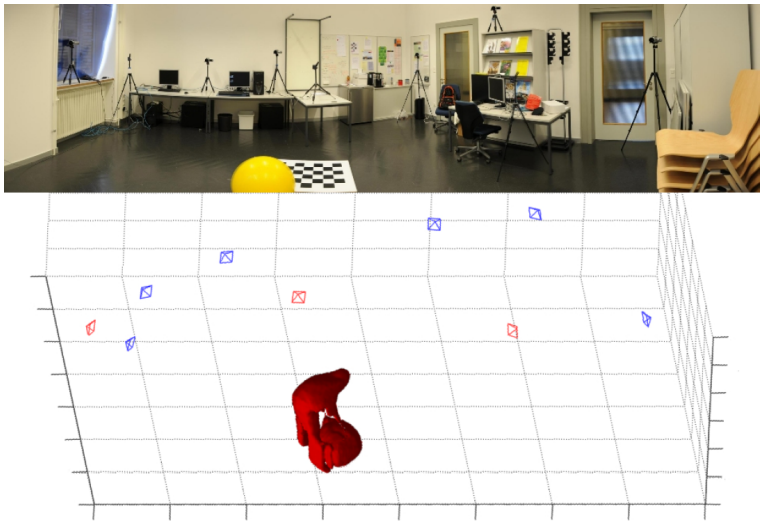

Figure 6.14: Top: the camera settings. Bottom: the reconstruction of a person with a rubber ball. Best viewed in color.

Figure 6.15: The reconstruction of the densely populated scene from all 9 sensors with concavity and details. Visual hull fails in this case, resulting an indistinguishable blob.

6.4 Summary of ToF Camera

In this chapter, a new heterogeneous sensor network of camcorders and ToF cameras in multi-view 3D object reconstruction is proposed. To achieve more accurate geometric measurements, a new unified calibration method with a spherical calibration target is carried out despite the heterogeneity of the network. It overcomes the low resolution ToF camera image issue, and is almost automatic to recover the Euclidean sensor configuration. Both statistical evaluation and real dataset verify the feasibility and power of this calibration approach and this multi-modal sensor network setup.

With the calibrated four-sensor setup, it demonstrates the possibility to use a minimum of two opposite-posing ToF cameras for detailed geometry reconstruction and other two opposite-posing video cameras for extra guidance and more importantly complete texture maps. Another interesting idea worth exploring is the relationship between the ToF camera depth measurement against the active light incident angle to the reflecting surface, a relationship pointed out by [Lindner et al. (2008)]. The fact is that after the geometric calibration of the system, the ToF camera images captured during the calibra-

tion process should be analyzed further, because of the nice surface normal properties of the spherical calibration target. Future works also include depth calibration refinement, more dataset acquisition and temporal synchronization analysis.

After the system is calibrated, a novel probabilistic sensor fusion framework is proposed as an extension to Section 2.4.2 to robustly relate camcorder silhouette cues and ToF camera depth images together, and improve the reconstruction quality significantly comparing with the result using either type of sensor alone. ToF cameras are thus shown for the first time to be a very promising new type of sensor for accurate multi-view 3D reconstruction, besides its proposed usage in object detection, tracking etc. More importantly, the proposed sensor fusion framework is general enough and not limited to a silhouette cues or depth images, but also to disparity maps of stereo camera pairs or 3D point clouds of LIDAR sensors etc., as long as the proper sensor model is provided. Finally, consider computation time to our volume framework, most of the computation can be parallelized on GPU, similar to Section 3.5. Also given the high frame rate of both the camcorders and ToF cameras, dynamic scenes can be recovered in real-time.

Chapter 7

Conclusion and Future Work

The thesis conclusion is stated before opportunities for further investigation are discussed.

7.1 Conclusion

The growing computation power and reducing hardware price have enabled many practical computer vision applications including multi-view 3D dynamic scene modeling. However, to date, most methods have only achieved good reconstruction quality in strictly controlled scene settings with fixed lighting and relatively simple background. The result is far from satisfactory in natural environment. The most promising category of algorithms [Bonet and Viola (1999); Broadhurst et al. (2001); Franco and Boyer (2005)] all taking into account noise issue and model the scene probabilistically.

Following the direction of Franco and Boyer (2005), which recovers the dynamic shapes despite of the lighting variations, shadows and reflections, I have introduced in this thesis an extension to overcome static occlusion issue related with the drawback of traditional background modeling. This greatly extends the system's ability in cluttered indoor/outdoor environment. Combined with the camera network calibration algorithm from only silhouette information [Sinha et al. (2004)], one can imagine a multi-view system being set up in an arbitrary environment, and after a couple of minutes of

recording, automatically learns the background models and computes the camera poses and recovers static occlusions in the scene, and then robustly models dynamic scene in real-time taking into account the recovered static occlusion information. Due to the sensor fusion principle, this system does require majority of the sensors observing the dynamic shape without visual obstacles in the way.

The idea of explicitly modeling the static occlusion can be further extended to infer the inter-occlusion between dynamic shapes, so as to significantly reduce the ambiguity of silhouette cues. A nice thing is that this framework can naturally combine the previously introduced static occluder recovery algorithm. Although the shape estimation quality is not good enough for video conferencing so far, it is already good enough for outdoor surveillance purposes. And as the dataset evaluation shows, by explicitly modeling the shapes of the dynamic objects, the 3D object tracking result gets improved from simple rectangle shape model.

Dynamic event modeling is more constrained than static object reconstruction, I have discussed in the thesis how to enforce temporal consistency between consecutive time step imageries to further refine the reconstruction result. The method is also formulated in the probabilistic framework to model the uncertainty of the real environment and achieve maximum robustness.

I have also shown that such sensor fusion framework is general enough to incorporate heterogeneous sensing modalities as long as the probabilistic imaging sensing model can be well-defined. An example of a video camcorder and ToF camera network is proposed for dynamic scene modeling task.

7.2 Future Work

7.2.1 Combination of Multi-view Stereo

It is tempting to combine the silhouette information with surface correspondences recovered with multi-view stereo technique. Because similar to ToF camera's depth map, multi-view stereo also recovers surface information compensating the silhouette information. However, in order to exploit the advantages of both methods in a robust probabilistic framework, one needs to either convert surface information to the occupancy probability with a hidden variable of the frontal surface of the occupancy, similar to the ToF camera and camcorder fusion in Chapter 6; or one can formulate to compute the "object surface probability" from the silhouette information.

The latter is straightforward because within the visual hull, the surface of the real shape could literally take any form, which means given the shape of the visual hull, the object surface probability is 0.5 within the visual hull and 0.0 outside of the visual hull. Although this does not give any positive information (probability greater than 0.5) alone, when combined with the surface information from multi-view stereo, one would definitely have a refined probabilistic shape estimation, as long as the stereo information is easily transformed into the same probabilistic form.

7.2.2 Dense Motion Flow and Motion Segmentation

Dense 3D motion computation is ill-posed. As I discussed in Chapter 5, many heuristic assumptions are made. The local motion smoothness is one of them. On the one hand, it helps to recover the motion inside the shape where no visual observation is available. On the other hand, it smooths out the motion discontinuities where two different motions occur very closely together. For example, during the normal walking sequence, one of the two arms aside the torso has a backward swing motion, while the torso is moving forward.

One solution is to track body parts instead of voxels. Because now voxels within the same body part are spatially connected together and share the same parameterized motion, it is not necessary to have the strong assumption of "motion smoothness" anymore. However, the algorithm introduced in Chapter 5 is important in the sense of a bootstrapping of what body parts are in the scene and which voxels belong to them. The bootstrapping can be done following the direction of motion segmentation in the discussion section of Chapter 5. And only with such initialization, can arbitrary dynamic shapes be modeled and tracked in the 4D spatiotemporal domain.

A second issue is when large motions are at presence in the scene. Intuitively, the larger the motion, the more likely the optimization gets stuck in a local optima. For the specific framework proposed in Chapter 5 though, given a fixed video frame rate, there is always a chance that some parts of the motion are too large, *i.e.* the magnitude of certain motion vectors are larger than that of the coarsest level motion labels. The current framework iteratively computes the motion field to a locally optimal solution. This may result in fluid-like motion, which is usually not a desirable solution. One improvement is to incorporate visual feature correspondence information from the dynamic shape surfaces if there are any, so as to drag the solution out of local optima.

Finally, although for such wide baseline camera setup, to find very dense surface correspondences across views are impractical, the optical flow over time within the same view may provide extra information for the 3D motion field computation. It also does not need the photo-consistency requirement across views as for the photometric stereo algorithms. But since the corresponding features and optical flows only appear on the actual object surface or its 2D projection, one must be careful how to use these constraints in a probabilistic volume, where no explicit surfaces are extracted.

Bibliography

Agrawal, M. and Davis, L. (2003). Complete camera calibration using spheres: A dual-space approach. *International Conference on Computer Vision*, pages 782–790. 116, 117

Ahn, J., Kim, K., and Byun, H. (2006). Robust object segmentation using graph cut with object and background seed estimation. *International Conference on Pattern Recognition*, 2:361–364. 19

Apostoloff, N. and Fitzgibbon, A. (2005). Learning spatiotemporal T-junctions for occlusion detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2:553–559. 34

Bajaj, C., Bernardini, F., and Xu, G. (1995). Automatic reconstruction of surfaces and scalar fields from 3D scans. *Proceedings of SIGGRAPH*, pages 109–118. 125

Ballan, L. and Cortelazzo, G. M. (2008). Marker-less motion capture of skinned models in a four camera set-up using optical flow and silhouettes. *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*. 90

Baumgart, B. (1974). Geometric modeling for computer vision. *PhD thesis-University of Stanford*. 11, 23, 125

Bonet, J. D. and Viola, P. (1999). Roxels: Responsibility weighted 3D volume reconstruction. *International Conference on Computer Vision*, 1:418–425. 15, 25, 34, 136

Boyer, E. and Berger, M.-O. (1997). 3D surface reconstruction using occluding contours. *International Journal of Computer Vision*, 22(3):219–233. 23

Boyer, E. and Franco, J.-S. (2003). A hybrid approach for computing visual hulls of complex objects. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:695–701. 23

Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *PAMI*. 101

Brand, M., Kang, K., and Cooper, D. B. (2004). Algebraic solution for the visual hull. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:30–35. 23

Broadhurst, A., Drummond, T., and Cipolla, R. (2001). A probabilistic framework for the space carving algorithm. *International Conference on Computer Vision*, pages 388–393. 25, 68, 89, 95, 136

Brostow, G. and Essa, I. (1999). Motion based decompositing of video. *International Conference on Computer Vision*, 1:8–13. 34

Casciola, G., Lazzaro, D., Montefusco, L., and Morigi, S. (2005). Fast surface reconstruction and hole filling using positive definite radial basis functions. *Numerical Algorithms*, 39(1-3):289–305. 125

Cheung, G., Baker, S., and Kanade, T. (2003). Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. *Computer Vision and Pattern Recognition*, 1:77–84. 90

Chien, C. and Aggarwal, J. (1986). Volume/surface octress for the representation of three-dimensional objects. *Computer Vision, Graphics and Image Processing*, 36(1):100–113. 24

Cipolla, R. and Blake, A. (1992). Surface shape from the deformation of apparent contours. *International Journal of Computer Vision*, 9:83–112. 23

Coué, C. (2003). Modèle bayésien pour l'analyse multimodale d'environnements dynamiques et encombrés : Application à l'assistance à la conduite en milieu urbain. *Dissertation to doctor of Sciences Institut National Polytechnique De Grenoble*. 130

Cross, G. and Zisserman, A. (1998). Quadric reconstruction from dual-space geometry. *International Conference on Computer Vision*, 0:25. 23

Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. *Proceedings of SIGGRAPH*, pages 303–312. 125

Davis, J., Marshner, S., Garr, M., and Levoy, M. (2001). Filling holes in complex surfaces using volumetric diffusion. *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, pages 428–438. 125

de Aguiar, E., Theobalt, C., Stoll, C., and Seidel, H.-P. (2007). Marker-less deformable mesh tracking for human shape and motion capture. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. 90, 91

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorith. *Journal of the Royal Statistical Society: Series B*. 94

Elfes, A. (1989). Occupancy grids: a probabilistic framework for robot perception and navigation. *Dissertation to doctor of Sciences CMU*. 126

Favaro, P., Duci, A., Ma, Y., and Soatto, S. (2003). On exploiting occlusions in multiple-view geometry. *International Conference on Computer Vision*, 0:479–486. 34

Finlayson, G., Chatterjee, S., and Funt, B. (1996). Color angular indexing. *European Conference on Computer Vision*, 2:16–27. 20

Fleuret, F., Berclaz, J., Lengagne, R., and Fua, P. (2007). Multi-camera people tracking with a probabilistic occupancy map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):267–282. 68, 84

Franco, J.-S. and Boyer, E. (2005). Fusion of multi-view silhouette cues using a space occupancy grid. *International Conference on Computer Vision*, 2:1747–1753. 25, 30, 69, 89, 95, 97, 103, 125, 126, 127, 136

Franco, J.-S. and Boyer, E. (2009). Efficient polyhedral modeling from silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):414–427. 23, 89, 125

Giblin, P. and Weiss, R. (1987). Reconstruction of surfaces from profiles. *International Conference on Computer Vision*, 9:136–144. 23

Glocker, B., Komodakis, N., Tziritas, G., Navab, N., and Paragios, N. (2008). Dense image registration through MRFs and efficient linear programming. *Medical Image Analysis*. 100, 101

Guan, L., Franco, J.-S., and Pollefeys, M. (2007). 3D occlusion inference from silhouette cues. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. 35, 77, 79, 89, 105

Guan, L., Franco, J.-S., and Pollefeys, M. (2008a). 3D object reconstruction with heterogeneous sensor data. *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, pages 1–8. 114, 126

Guan, L., Franco, J.-S., and Pollefeys, M. (2008b). Multi-object shape estimation and tracking from silhouette cues. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. 35, 68, 89

Guan, L. and Pollefeys, M. (2008). A unified approach to calibrate a network of camcorders and ToF cameras. *IEEE workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications (M2SFA2)*, pages 1–8. 114, 115

Guan, L., Sinha, S., Franco, J.-S., and Pollefeys, M. (2006). Visual hull construction in the presence of partial occlusion. *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, pages 413–420. 34

Gupta, A., Mittal, A., and Davis, L. (2007). COST*: An approach for camera selection and multi-object inference ordering in dynamic scenes. *International Conference on Computer Vision*, 1:1–8. 32, 68, 80, 83

Hartley, R. I. and Schaffalitzky, F. (2004). l_∞ minimization in geometric reconstruction problems. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:504–509. 120

Heikkila, J. and Silven, O. (1999). A real-time system for monitoring of cyclists and pedestrians. *Second IEEE Workshop on Visual Surveillance*, pages 74–81. 17, 18

Hilton, A., Stoddart, A., Illingworth, J., and Winderatt, T. (1998). Implicit surface based geometric fusion. *Computer Vision Image Understanding*, 69(3):273–291. 125

Ilie, A. and Welch, G. (2005). Ensuring color consistency across multiple cameras. *International Conference on Computer Vision*, 2:1268–1275. 16

Imaging, M. (2007). Swiss Ranger 3000 help document, 1.0.8.x, miniature 3D time of flight camera. *support@swissranger.ch*. 131

Kahlmann, T. (2007). Range imaging metrology: Investigation, calibration and development. *Dissertation to doctor of Sciences ETH Zürich*. 114, 115

Kampel, M., Sablatnig, R., and Tosovic, S. (2002). Fusion of surface and volume data. *Vision with Non-Traditional Sensors, Proc. of the 26th Workshop of the Austrian Association for Pattern Recognition (ÖAGM)*, 160:21–28. 126

Kang, K., Tarel, J., Fishman, R., and Cooper, D. (2001). A linear dual-space approach to 3D surface reconstruction from occluding contours. *International Conference on Computer Vision*, 1:198. 23

Kass, M., Witkin, A., and Terzopoulos, D. (1987). Snakes: active contour models. *International Journal of Computer Vision*, 1(4):321–331. 17

Ke, Q. and Kanade, T. (2005). Quasiconvex optimization for robust geometric reconstruction. *International Journal of Computer Vision*, 2:986–993. 120

Keck, M. and Davis, J. (2008). 3D occlusion recovery using few cameras. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. 35

Kim, S. (2008). Radiometric calibration methods from image sequences. *PhD thesis-University of North Carolina at Chapel Hill*. 16

Koenderink, J. (1984). What does the occluding contour tell us about solid shape? *Perception*, 13:321–330. 23

Kolmogorov, V. and Zabih, R. (2004). What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 101

Komodakis, N. and Tziritas, G. (2007). Approximate labeling via graph-cuts based on linear programming. *IEEE Transactions on pattern analysis and machine intelligence*. 101

Komodakis, N., Tziritas, G., and Paragios, N. (2007). Fast, approximately optimal solutions for single and dynamic MRFs. *IEEE Conference on Computer Vision and Pattern Recognition*. 99, 100

Kutulakos, K. and Seitz, S. (2000). A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218. 13, 15, 89

Larsen, S. (2006). Multiview belief propagation for reconstruction of office environments. *PhD thesis-University of North Carolina at Chapel Hill*. 15

Laurentini, A. (1994). The visual hull concept for silhouettes-based image understanding. *IEEE Transactions on pattern analysis and machine intelligence*, 16(2):150–162. 11, 32, 89, 125

Lazebnik, S., Furukawa, Y., and Ponce, J. (2007). Projective visual hulls. *International Journal of Computer Vision*, 74(2):137–165. 12, 23, 89, 125

Li, M., Schirmacher, H., Magnor, M., and Seidel, H. (2002). Combining stereo and visual hull information for online reconstruction and rendering of dynamic scenes. *IEEE Workshop on Multimedia Signal Processing*, pages 9–12. 126

Liang, C. and K.Wong, K.-Y. (2005). Complex 3D shape recovery using a dual-space approach. *IEEE Conference on Computer Vision and Pattern Recognition*, 1:878–884. 23

Lindner, M., Kolb, A., and Ringbeck, T. (2008). New insights into the calibration of ToF sensors. *CVPR Workshop On Time of Flight Camera based Computer Vision (TOF-CV)*. 115, 134

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110. 14

Margaritis, D. and Thrun, S. (1998). Learning to locate an object in 3D space froma sequence of camera images. *Proceeding of International Conference on Machine Learning*, pages 332–340. 126

Martin, W. and Aggarwal, J. (1983). Volumetric description of objects from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):150–158. 24

Matusik, W., Buehler, C., and McMillan, L. (2001). Polyhedral visual hulls for real-time rendering. *Eurographics Workshop on Rendering*, pages 115–126. 23

Matusik, W., Buehler, C., Raskar, R., McMillan, L., and Gortler, S. (2000). Image-based visual hulls. *Proceedings of SIGGRAPH*, pages 369–374. 11, 23

Middlebury (2009). Middlebury public multi-view stereo dataset runtime evaluation. http://vision.middlebury.edu/mview/eval/runtimeTable.php. 14

Mittal, A. and Davis, L. S. (2003). M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. *International Journal of Computer Vision*, 51(3):189–203. 68, 83

nVidia (2009). CUDA programming guide (version 2.3.1). 60

Oliensis, J. (2002). Exact two-image structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1618–1633. 120

Otsuka, K. and Mukawa, N. (2004). Multiview occlusion analysis for tracking densely populated objects based on 2D visual angles. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:90–97. 68

Pathak, K., Birk, A., Poppinga, J., and Schwertfeger, S. (2007). 3D forward sensor modeling and application to occupancy grid based sensor fusion. *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2059–2064. 126, 130

Pollefeys, M., Gool, L. V., Vergauwen, M., Verbiest, F., Cornelis, K., and Tops, J. (2004). Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(12):207–232. 14

Pons, J.-P., Keriven, R., and Faugeras, O. (2007). Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2):179–193. 91

Potmesil, M. (1987). Generating octree models of 3d objects from their silhouettes in a sequence of images. *Computer Vision, Graphics and Image Processing*, 40(1):1–29. 24

Sablatnig, R., Tosovic, S., and Kampel, M. (2002). Combining shape from silhouette and shape from structured light for volume estimation of archaeological vessels. *International Conference on Pattern Recognition*, 1:364–367. 126

Seitz, S., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:519–526. 13, 14

Shlyakhter, I., Rozenoer, M., Dorsey, J., and Teller, S. (2001). Reconstructing 3D tree models from intrumented photographs. *IEEE Computer Graphics and Applications*, 21(3):53–61. 23

Sinha, S., Mordohai, P., and Pollefeys, M. (2007). Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. *International Conference on Computer Vision*, 0:1–8. 14, 15

Sinha, S. and Pollefeys, M. (2004). Synchronization and calibration of camera networks from silhouettes. *International Conference on Pattern Recognition*, 1:116–119. 10

Sinha, S., Pollefeys, M., and McMillan, L. (2004). Camera network calibration from dynamic silhouettes. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 0:1–8. 9, 136

Snow, D., Viola, P., and Zabih, R. (2000). Exact voxel occupancy with graph cuts. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 345–353. 128

Srivastava, S. K. (1990). Octree generation from object silhouettes in perspective views. *Computer Vision, Graphics and Image Processing*, 49(1):68–84. 24

Stage^{TM} (2007). Organic Motion. http://www.organicmotion.com/technology. 12

Starck, J. and Hilton, A. (2007). Correspondence labelling for wide-timeframe free-form surface matching. *International Conference on Computer Vision*, pages 1–8. 91

Stauffer, C. and Grimson, W. (1999). Adaptive background mixture models for real-time tracking. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:2246. 20, 74

Stein, A. and Hebert, M. (2009). Occlusion boundaries from motion: Low-level detection and mid-level reasoning. *International Journal of Computer Vision*, 82:325–357. 34

Strecha, C., Fransens, R., and Gool, L. V. (2004). Wide-baseline stereo from multiple views: a probabilistic account. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:552–559. 15

Sun, D., Roth, S., Lewis, J., and Black, M. (2008). Learning optical flow. *European Conference on Computer Vision*. 88, 91, 100

Svoboda, T., Martinec, D., and Pajdla, T. (2005). A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422. 8, 9, 13, 115, 116, 120

Szeliski, R. (1993). Rapid octree construction from image sequences. *Computer Vision, Graphics and Image Processing*, 58(1):23–32. 24, 125

Toyama, K., Krumm, J., Brumitt, B., and Meyers, B. (1999). Wallflower: Principles and practice of background maintenance. *International Conference on Computer Vision*, 1:255–261. 20

Uematsu, Y., Teshima, T., Saito, H., and Cao, H. (2007). D-calib: Calibration software for multiple cameras system. *International Conference on Image Analysis and Processing*, pages 285 – 290. 115

Vaillant, R. and Faugeras, O. (1992). Using extremal boundaries for 3D object modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):157–173. 23

Varanasi, K., Zaharescu, A., Boyer, E., and Horaud, R. (2008). Temporal surface tracking using mesh evolution. *European Conference on Computer Vision*. 90

Vedula, S., Baker, S., Rander, P., Collins, R., and Kanade, T. (2005). Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):475–480. 88, 91

Vlasic, D., Baran, I., Matusik, W., and Popovic, J. (2008). Articulated mesh animation from multi-view silhouettes. *Proceedings of SIGGRAPH*, 27(3). 90

Vogiatzis, G., Esteban, C. H., Torr, P., and Cipolla, R. (2007). Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2241–2246. 14

W. Lorensen, H. C. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169. 24

Wedel, A., Rabe, C., Vaudrey, T., Brox, T., and Cremers, D. (2008). Efficient dense 3D scene flow from sparse or dense stereo data. *European Conference on Computer Vision*. 91

Whitaker, R. (2004). A level-set approach to 3D reconstruction from range data. *International Journal of Computer Vision*, 29(3). 128

Yao, A. and Calway, A. (2003). Dense 3d structure from image sequences using probabilistic depth. *In Proc. British Machine Vision Conference*, pages 211–220. 25

Yemez, Y. and Wetherilta, C. (2007). A volumetric fusion technique for surface reconstruction from silhouettes and range data. *Computer Vision and Image Understanding*, 105(1):30–41. 126

Ying, X. and Zha, H. (2006). A novel linear approach to camera calibration from sphere images. *International Conference on Pattern Recognition*, pages 535–538. 116, 117

Zhang, H., Wong, K., and Zhang, G. (2007). Camera calibration from images of spheres. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):499–502. 116, 117

Zhang, Z. (2007). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334. 8, 9, 115, 116, 120

Zhou, Y. and Tao, H. (2003). A background layer model for object tracking through occlusion. *International Conference on Computer Vision*, 2:1079–1085. 34

Ziegler, R., Matusik, W., Pfister, H., and McMillan, L. (2003). 3D reconstruction using labeled image regions. *In EuroGraphics symposium on Geometry processing*, pages 248–259. 68, 69