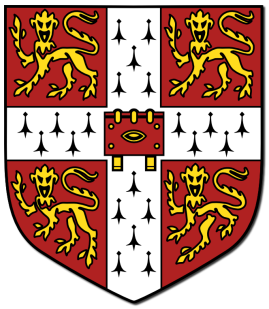
Automatic 3D Model Acquisition from Uncalibrated Images

Neill D.F. Campbell

Gonville & Caius College

University of Cambridge



This dissertation is submitted for the degree of

Doctor of Philosophy

September 2010

To my parents, Lucy and Lorne,
and my brother Robbie

Abstract

The recovery of shape from images has received much attention as a core research topic in Computer Vision, however, these algorithms often require specialist equipment, expert knowledge or large numbers of images to obtain good results. These act as a barrier to the adoption of these technologies by other fields where we wish to allow systems to process images from the 'real world' without Computer Vision experts to operate them. The desire to implement a practical reconstruction system that is useful to and usable by the people who may benefit from these technologies motivates the contributions of this thesis.

This work addresses all three of the stages required to take a sequence of images of an object and recover a 3D model in order to produce a system that maximises automation and minimises the demands placed on the user. To that end we present a practical implementation of an automatic method for recovering the positions and properties of the cameras used to take a series of images using a textured ground-plane.

We then offer two contributions to simplify the task of segmenting an object observed in multiple images. The first, applicable to more simple scenes, automatically segments the object fixated upon by the camera. We achieve this by exploiting the rigid structure of the scene, to perform the segmentation in 3D across all the images simultaneously, and the consistent appearance of the object in an iterative method. For more complex scenes we move to our second algorithm that allows the user to select the required object in an interactive manner whilst minimising demands on their time. We combine the different appearance and spatial constraints to produce a clustering problem to group regions across images that allows the user to label many images at the same time.

Finally we present an automatic reconstruction algorithm that improves the performance of existing state-of-the-art methods to allow accurate models to be obtained from smaller image sequences. This takes the form of a filtering process that rejects erroneous depth estimates by considering multiple depth hypotheses and identifying the true depth or an unknown state using a 2D Markov Random Field framework. We provide experimental validation for all the individual contributions, demonstrate the practical system working as a whole and conclude by discussing the merits of the system and avenues for future work.

Acknowledgements

I must begin by expressing my deepest thanks to my supervisor Prof. Roberto Cipolla and my co-supervisors George Vogiatzis and Carlos Hernández who have been an invaluable source of inspiration, guidance, reassurance and support for which I will always be grateful and I will miss the ever fruitful and insightful discussions and debates.

It has been a great privilege to work in the Machine Intelligence Laboratory, and at Toshiba Research, with such an amazing combination of intellectual stimulation, camaraderie and above all friendship that anyone could wish for. With so many to thank I risk the peril of omission but particular mention must go to my lifelong lab-mates (and in the case of Fabio my Italian shoe guru): Fabio, Tom, Gabe, Björn, Jamie, Matt, Giovanni, Julia, Ignas, Vijay, Fabio G, Kris, Frank, Atsuto, T-K, Yu, Rob, T-H and honorary member John W.

Away from the lab, my time in Cambridge would have been far less enjoyable without the wonderful members of Gilbert & Sullivan Society and my amazing house-mates Miranda, Miri, Kerrie and Lucy (and honorary members Catherine, Charlie, Richard, Mike, Jenny and Rob) who have provided joyous memories for a lifetime and never fail to tolerate my eccentricities. Special mention must of course go to my perpetual lab partner Henry who, since such a fateful meeting at our first practical in the EIETL, has never failed to be a source of friendship, encouragement and particularly good fajitas.

I wish to thank the Schiff Foundation and Toshiba Research for their financial support and additionally Gonville & Caius College and the Engineering Department for grants for travel.

I remain indebted to Dr. Allwood and Prof. Smith at Caius for their encouragement and guidance, above and beyond the call, for all of my time in Cambridge.

Finally, and above all, I must express my undying gratitude to my parents, Lucy and Lorne, and my brother Robbie for their never-faltering belief, encouragement and love that has supported me for all my endeavours.

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text. The work presented in Chapter 5 appears in the conference article [Campbell et al. 2007] and the journal article [Campbell et al. 2010] and the work of Chapter 7 appears in the conference article [Campbell et al. 2008]. I hereby declare that my thesis does not exceed the limit of length prescribed in the regulations of the degree committee of the Department of Engineering. This dissertation contains no more than 40,000 words and 63 figures.

Contents

List of Figures

Notation

The following notation is adopted unless otherwise stated:

$\mathbf{u} = [u, v]^T$	A pixel location of a point within an image
$\tilde{\mathbf{u}} = [u, v, w]^T$	The same point in homogeneous co-ordinates
$\mathbf{X} = [X, Y, Z]^T$	A 3D world position of a point
$\tilde{\mathbf{X}} = [X, Y, Z, W]^T$	The same point in homogeneous co-ordinates
$\mathbf{X}_c = [X_c, Y_c, Z_c]^T$	A 3D position of a point in camera centred co-ordinates
$\tilde{\mathbf{X}}_c = [X_c, Y_c, Z_c, w]^T$	The same point in homogeneous co-ordinates
R	A 3D rotation matrix
$\mathbf{t}$	A 3D translation vector
$K = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$	A camera intrinsic parameters matrix
$P = K[R\|\mathbf{t}]$	A 3×4 projection matrix
H	A 3×3 projective image homography
E	A 3×3 essential matrix
$\mathbf{c}$	A vector in colour space, e.g. the RGB unit cube

Chapter 1

Introduction

> *"Vision is a process that produces from images of the external world a description that is useful to the viewer and not cluttered with irrelevant information."* [Marr 1982]

As proposed by Marr, one of the subject's early pioneers, *computer vision* addresses the problem of inferring useful information from images and videos of the world around us. A statement such as this gives rise to a plethora of questions as we attempt to delve deeper into the aims and motivations of the study of vision. Principally among these might be the question of what do we mean by useful information. Over the intervening years researchers have proposed a variety of tasks that a vision system, whether human or machine, might wish to perform and each task has corresponding requirements on particular pieces of information that may prove useful. At the broadest level these tasks include:

- **Recovering the structure of the observed scene.** This includes forming an understanding of the 3D world from visual observations, including an analysis of its physical shape and the motions of the viewer and the scene.
- **Interpreting the contents of the observed scene.** This includes the tasks of discovering the presence of objects and identifying them in a hierarchy from classes of object down to specific instances.

The first of these tasks forms the area of interest for this thesis, in particular the goal of understanding and recovering the shape of the 3D world from visual information.

1.1 Motivation

Motivation for research in this area may be provided on two levels. Early investigations into vision attempted to break down biological systems into their constituent parts and hence

build a model for the process as a whole. This approach fell down due to the complexity of the systems and the lack of understanding of the information processing tasks being performed; endeavouring to understand a particular implementation (the human visual system) without understanding the underlying principles involved. The insight offered by Marr and his contemporaries was the suggestion that: *"One cannot understand what seeing is and how it works unless one understands the underlying information processing tasks being solved."* [Marr 1982]. Thus, the first motivation to study computer vision is to gain an insight into the fundamentals of information processing in vision. This in turn provides insights into questions about the human visual system and the portions of the brain that perform the corresponding interpretations.

Computational Theory	What is the goal of the computation, why is it appropriate and what is the logic of the strategy by which it can be carried out?
Representation and Algorithm	How can this computational theory be implemented? In particular what is the representation for the input and the output and what is the algorithm for the transformation?
Hardware Implementation	How can the representation and algorithm be realised physically?

Table 1.1: The three levels at which any machine carrying out an information processing task must be understood. *Taken from Figure 1-4, [Marr 1982].*

Marr proposed three distinct levels, given in Table 1.1, upon which an information processing task must be understood. Thus we may conclude that the study of the computational theory, representation and algorithms involved in vision gives us an insight into its challenges and allows us to obtain a deeper understanding whether it be in the context of human or machine vision.

Besides intellectual curiosity, from the viewpoint of an engineer there is a strong motivation for understanding vision in order to create technologies which may useful in their own right. Currently there is a great demand for 3D models of objects in the world; in particular we are noticing the appearance of new 3D display technologies which will create the visualisations and interfaces of the future and change the way we are able to access and interpret information held on computers. Computer vision offers the possibility of providing this 3D information for many applications that in turn give their own motivation. The most significant current and future applications of 3D model acquisition include:

(a) Images taken with a handheld camera

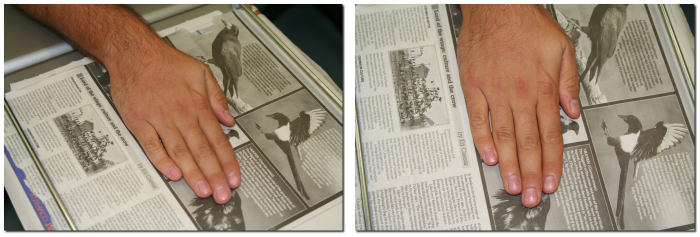
(b) Recovered model

(c) Texture mapped

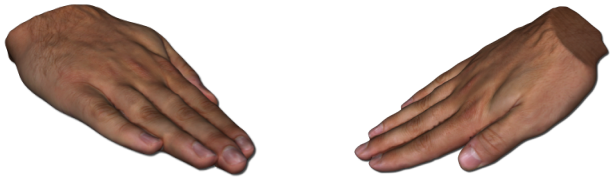
Figure 1.1: Example of 3D digital archiving. *(a) Some of the images of a bronze statue of a young man from the Mimaut Collection (Roman copy of the 1st Century BC after a Greek original) found in Ziphteh, near Tell Atrib (ancient Athribis), Egypt. The images were taken with a hand held camera in the British Museum. The background is extremely cluttered representing a challenging task for automatic segmentation and reconstruction. (b) The 3D model recovered and (c) texture mapped automatically using [Campbell et al. 2007; 2008, Hernández and Schmitt 2004].*

- **Digital archiving and archaeology.** The ability to generate 3D models from images alone is highly attractive to museums and other archival institutions. Photographs are routinely taken for documentation in museums so for very little extra expense we may obtain high resolution models. These models allow interested parties to view pieces from any angle, in an interactive setting, from their own computers and allow museums to provide displays for all the pieces in their collections, not just the ones they have space to display. This is also particularly important for items which must be preserved in a restrictive environment that makes physical display or viewing difficult. Figure 1.1 shows an example of a model produced from a series of images taken with a hand held camera at the British Museum.

- **Medical imaging.** There have been many advances in medical imaging technologies for studying the internal features of the body but there are situations where a cheap method for creating models of the external body could be put to good use. For instance we can remove the need to take plaster casts in order to generate masks for radiotherapy or a brace for orthopaedics by obtaining a 3D model of part of the body from a small number of images. Figure 1.2 demonstrates a simple example: a model



(a) Images taken with a hand held camera



(b) Model shown from novel viewpoints

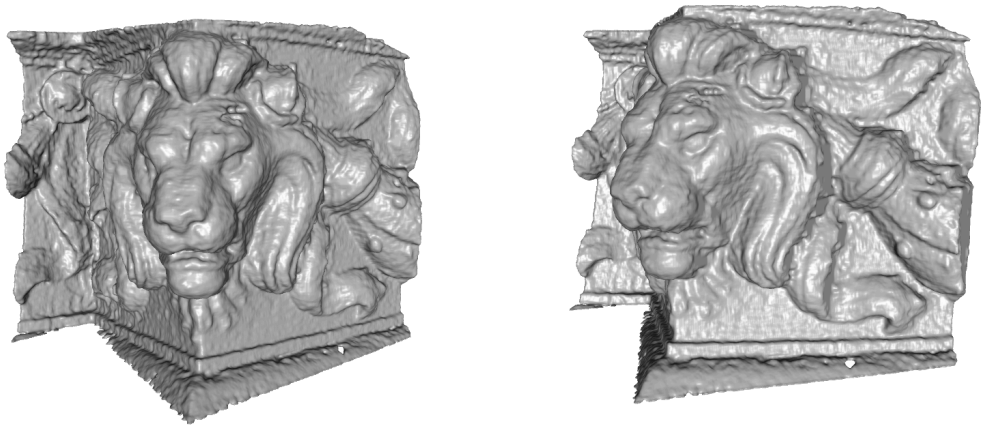
Figure 1.2: Model of a hand obtained with the automatic reconstruction system. *(a) Images taken with a low cost digital camera are the only inputs required to produce (b) a 3D model which physicians may use to construct appliances on the computer, with a high degree of accuracy, and print with 3D printers without any need for plaster casts or discomfort to the patient. The result was obtained using the automatic calibration algorithm of Chapter 3, segmented with [Campbell et al. 2007] and reconstructed with [Campbell et al. 2008].*

of a hand obtained using nothing more that a camera and a newspaper.

- **Entertainment, communication and the media.** The entertainment industries are a huge source of demand for 3D data for film and television or providing interactive experiences in training simulations and games. This demand will only increase with the prevalence of new 3D display technologies that are already available in cinemas and in the home. As the ease of communication found with the Internet is encouraging global collaboration, we face a greater demand for technologies for communication to reduce the need for travel and allow people to collaborate in an interactive setting. Systems to capture 3D content in real time will greatly increase the quality of this experience and allow the technology to blend into the background making communication more natural. We might also like to create our own 3D content in the home: Figure 1.3 shows a model of a sculpture, which was obtained from 8 images taken with a compact digital camera. The processing was performed automatically and required no technical knowledge on the part of the user.

- **Engineering simulation and analysis.** The ability to capture accurate 3D models

(a) Three of the eight input images



(b) 3D model shown from novel viewpoints (provided without texture to emphasise geometry)

(c) The model with texture

Figure 1.3: Model of a lion sculpture on the Ponte Vittorio Emanuele II in Rome. *(a) Eight images (three shown) taken with a low cost digital camera are the only inputs required to produce (b) a 3D model which may also be texture mapped (c). The model was obtained using the system described in [Campbell et al. 2008].*

is very useful to the scientific and engineering communities. An example would be structural analysis of buildings, providing verification of design and looking for wear and fatigue. The ability to perform accurate physical simulations during modern design processes creates a demand for models of existing infrastructure to improve the designs of the future, for example fluid flows around aircraft or buildings, or to assess and protect against earthquakes and hurricanes.

A common factor of these applications is the desire to reconstruct models from images taken in the 'real world', away from the controlled conditions of the laboratory, and the recognition that the end users of these technologies are specialists in their own fields and not experts in computer vision. It is the desire to study vision whilst also making the outputs of this research useful to and usable by the people who may benefit from these technologies that motivates the contributions of this thesis.

1.2 Research Goals

The reconstruction of 3D objects is an active topic of study for the computer vision community and recently there has been a growing interest in the subject, particularly in the development of 'dense multi-view stereo' reconstruction techniques [Seitz et al. 2006]. These techniques focus on producing 3D models from a sequence of calibrated images of an object. Figure 1.4 provides an overview of the pipeline of such processes and Figure 1.5 provides an illustrative result from existing literature [Hernández and Schmitt 2004].

There are a number of automatic algorithms for reconstruction, however there are recurring themes in their requirements and the perhaps rather unrealistic images of Figure 1.5 serve as a good illustration. By and large the existing algorithms share one or more of the following generalised constraints:

- Images are acquired in calibrated rigs or using other specialist equipment such as turntables to help with calibration.
- Large numbers of images are required for reconstruction.
- Images are taken against simple or known backgrounds to provide object silhouettes.
- A degree of specialist user interaction is required, either to control the algorithm during reconstruction or, in the case of the automatic methods, to perform preparatory work on the system input.

The goals of this thesis are to try to extend existing algorithms and develop new approaches to try to overcome some of these limitations which should serve to increase the

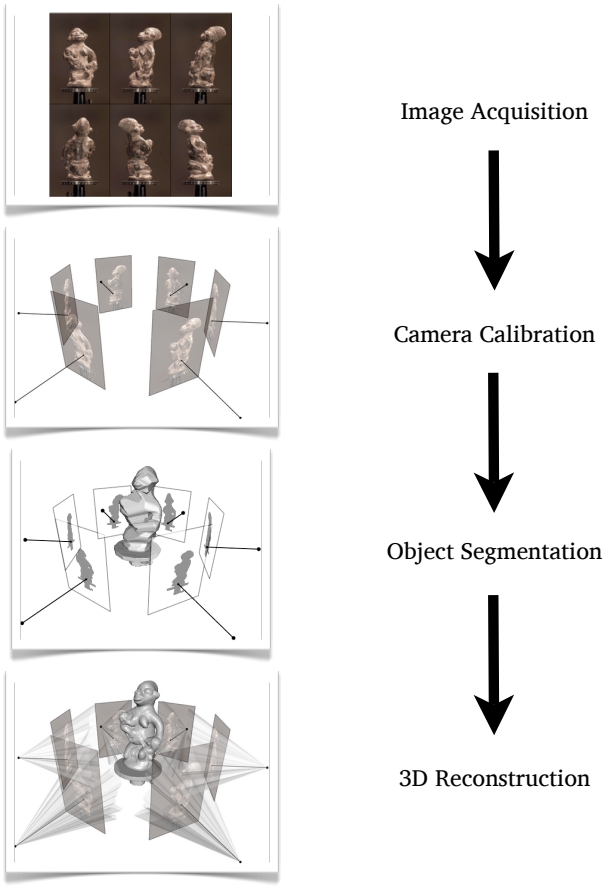


Figure 1.4: The reconstruction pipeline. *Used with permission from [Hernández 2004].*

Figure 1.5: Example of current state-of-the-art reconstruction. *The images of the statue have been taken on a turntable against a fixed background and reconstructed using [Hernández and Schmitt 2004].*

applications of reconstruction techniques as well as making them more accessible to wider audiences. Towards these goals we offer the following contributions:

1. A practical implementation of an undemanding and automatic method for recovering the positions and properties of the camera used to take the images in a sequence.

2. A method to segment automatically an object from a large sequence of images of a relatively simple scene.

3. A method to allow a user to segment interactively objects from a large sequence of images of more complex scenes with the aim of minimising the time and effort required.

4. An automatic algorithm for the recovery of an accurate 3D model from a small number of images of an object built into a practical system.

All these algorithms are intended to operate on images captured on standard digital cameras without any requirements for specialist equipment. The algorithms are intended to be automatic, or at least minimise the amount of input from the user, and thus the greatest demands are nothing more than instructions along the lines of approximate camera positions, guidelines on the number of images to take and in some cases indicating the object they wish to reconstruct. This means that the system may be used without any need for computer vision training or knowledge.

1.3 Contributions

The automatic reconstruction system has required contributions in each of the stages of the pipeline shown in Figure 1.4.

Calibration

The calibration of a set of images involves determining a common set of parameter values, which are intrinsic to the camera used to take the images, as well as the position and orientation of the camera for each of the images taken. There are many camera calibration algorithms that exploit a number of different physical constraints to infer the values of these parameters. We provide a practical implementation of a calibration system that requires no input from the user other than the images themselves and a planar surface, such as the newspaper in Figure 1.2(a), thus presenting a very easy process for the user.

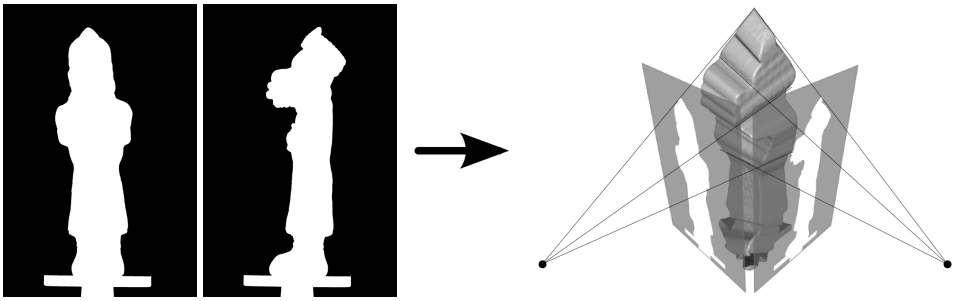


Figure 1.6: Silhouettes and the visual hull. *The intersection of the projected silhouettes of an object form the visual hull which is used in many reconstruction algorithms.*

Segmentation

As previously indicated, many of the existing reconstruction techniques require large numbers of calibrated images to obtain high accuracy reconstructions. In addition to these images, the majority of reconstruction algorithms require the silhouette of the object of interest in each of the images. Collectively these silhouettes form the *visual hull* [Laurentini 1994] of the object as shown in Figure 1.6. This visual hull of the object is often used to provide an approximation for an initial reconstruction, an outer bound on the reconstructed object or perhaps a method of approximate occlusion reasoning to determine which images the surface is visible in.

Unfortunately, images taken outside of the laboratory will often observe the object against a cluttered background, for example the images in Figure 1.1(a). In this situation, the object must be separated from the background (segmented) manually. Even with the latest interactive segmentation software, for example [Rother et al. 2004], this is a sizeable task since every image in the sequence must be segmented individually. In order to simplify this task we have created an automatic segmentation algorithm for whole image sequences [Campbell et al. 2007; 2010]. The algorithm builds probabilistic colour models, for the object and the background, across all the images in the sequence and performs the separation for each image simultaneously in the 3D domain. This is an iterative procedure that is automatically initialised by a fixation constraint; since we are trying to reconstruct a specific object it must be visible in all images in the sequence. Figure 1.7 provides an example of the results obtained.

Whilst it is clearly desirable to have a fully automatic segmentation algorithm, there are a range of images that prove to be very challenging for an automatic process. At the same time, there are many image sequences containing multiple objects and it would be impossible for a segmentation algorithm to pick out the object of choice from the numerous possibilities without input from the user. To extend a system to segment objects to

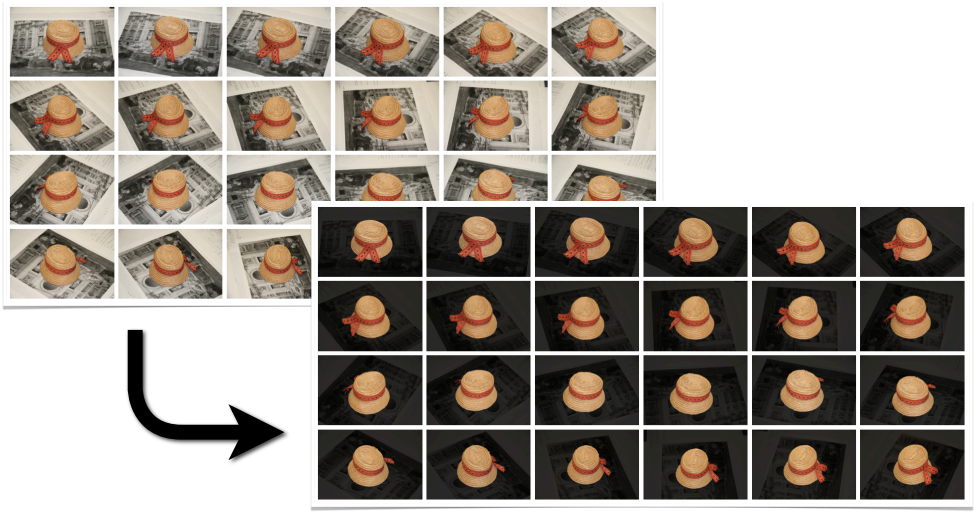


Figure 1.7: Example of automatic segmentation results. *Without any user input the hat is automatically segmented out of all the images.*

more general image sequences we have developed an interactive segmentation algorithm that aims to minimise the demands on the user. To achieve this we simplify the task by performing an initial processing of the images to group regions of the images, corresponding to physical, spatially consistent objects, together as clusters. The user may then label these clusters with the results being applied instantly across the entire image sequence. In the event that the initial clusters are inconsistent with the user's labelling, the result is automatically refined providing immediate feedback to the user and dramatically reducing the time required to segment an entire image sequence (over individual interactive segmentation of each image).

Reconstruction

In computer vision there are a number of stereo algorithms which infer depth in a scene using the parallax between two images [Scharstein and Szeliski 2002b]. Multiple-view (multi-view) stereo algorithms are methods which use stereo techniques over multiple images. The current state of the art algorithms can attain a high degree of accuracy when they have access to a larger number of images of the scene [Seitz et al. 2006]. They achieve this by exploiting the inherent redundancy in the image sequence, i.e. there are many views of the same surface. One strategy is to split the reconstruction process into two stages [Hernández and Schmitt 2004, Vogiatzis et al. 2005; 2007, Hornung and Kobbelt 2006a, Goesele et al. 2006]. The first is to estimate a series of depth-maps using local groups (pairs or triplets) of the input images. The second stage then attempts to combine these into a global surface estimate. This two stage approach is an elegant formulation

that allows different techniques to be chosen independently for the two stages.

The estimation of local depth-maps (images where each pixel represents the distance to the surface) is often performed using window or patch based methods [Scharstein and Szeliski 2002b]. Applying these methods leads to a set of individual depth-maps that are known to contain a large proportion of incorrect depth estimates or outliers. The second stage relies upon redundancy across the many depth-maps to reject these outliers from the true surface. In data-sets containing a large number of images (50-100) this approach performs well, achieving a high degree of accuracy. In so called sparse data-sets (10-20 images) one expects very little redundancy in the reconstructed depth-maps, leading to a drop in reconstruction accuracy.

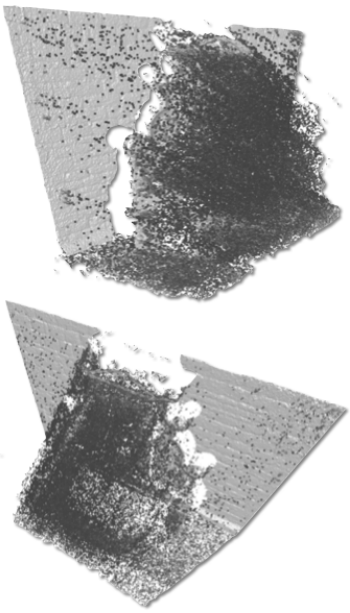
In order to address this we have introduced a new depth-map estimation process [Campbell et al. 2008] and shown that if individual depth-maps are filtered for outliers prior to the fusion stage, good performance can be maintained in sparse data-sets. Figure 1.8 provides an example of the improvement offered by this method. The strategy is to collect a list of good hypotheses for the depth of each pixel and then choose the optimal depth by enforcing consistency between neighbouring pixels in the depth-map. A crucial element of the filtering stage is the introduction of a possible *unknown* depth hypothesis for each pixel that is selected by the algorithm when no consistent depth can be chosen. This pre-processing of the depth-maps allows the second fusion stage to operate subject to fewer outliers and consequently improves the performance. Figure 1.3 shows an example of an accurate 3D model obtained from only 8 images, taken with a compact digital camera, using the depth-map estimation algorithm.

1.4 Why is Vision Challenging?

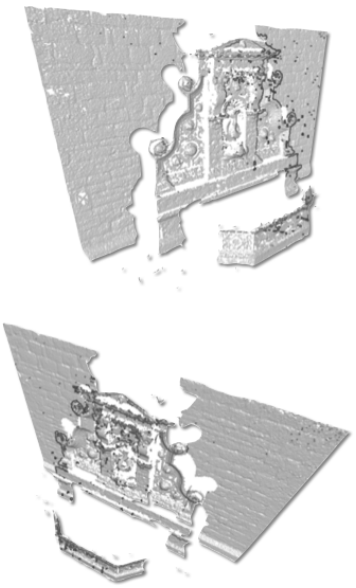
The human visual system has little problem performing the tasks of recovering shape and interpreting scenes and this would lead us to assume that the task should be a relatively simple one for a computer. In fact this was the assumption, or perhaps optimism, of the early computer vision researchers who estimated time frames in the order of months for visual reconstruction tasks. We have yet to produce a fully automated visual system over the intervening years, however we have attained a much greater understanding of the problems involved and are able to explain how complex and challenging the task actually is [Marr 1982].

Figure 1.9 portrays the principal parameters that combine to make up the 2D image captured, at an instant in time, by a camera. Here we have neglected the evolution of time and thus have made an assumption of rigidity, that nothing moves or deforms, which in

(a) The three input images



(b) Depth-map used by [Hernández and Schmitt 2004, Vogiatzis et al. 2007]



(c) Depth-map obtained using [Campbell et al. 2008]

(d) Texture mapped result

Figure 1.8: An example of a result from the depth-map filtering algorithm. *(a) Three input images are used to produce a single depth-map for the central image. (b) The result used by [Hernández and Schmitt 2004, Vogiatzis et al. 2007] contains many outliers which are removed (c) by the filtering algorithm of [Campbell et al. 2008]. (d) The result is also shown from another viewpoint and rendered with the appropriate texture. The data-set is from [Strecha et al. 2008].*

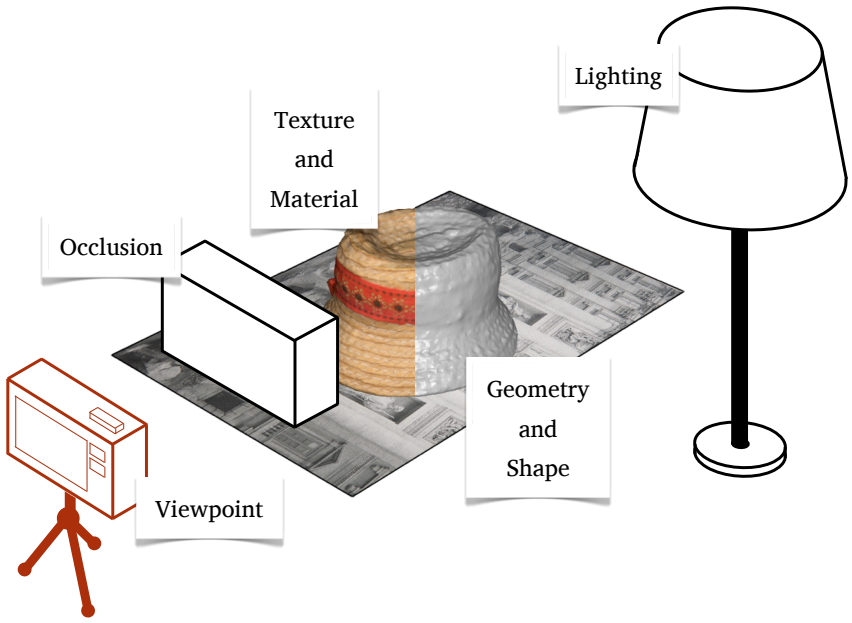


Figure 1.9: The makeup of a visual scene. *Recovering shape from an photograph is a challenging process since the image is obtained from complex interactions between the geometry of the scene, the materials and textures present, the illumination sources and the specific viewpoints and consequently the problem of occlusion.*

turn has already greatly simplified the problem. The probability theorist Jaynes remarked that *"seeing is inference from incomplete information"* [Jaynes 2003], a statement that cuts to the heart of the problem. The combination of all of these complicated factors followed by projection onto a single 2D image plane results in a huge loss of information. When presented with the image alone it is no longer possible to measure any of the aforementioned parameters directly, instead we are left with ambiguities, which at best may place constraints on some of these factors.

The loss of information during image formation may be best illustrated by an example. If we consider a red part of an image then we may place constraints on the possible lighting and texture/colour of the object observed but we cannot measure directly whether a white object is being illuminated by a red light or a red object by a white light. The challenge of recovering 3D shape is then to infer an estimate of the geometry of the object in question such that it would generate the images observed under the same remaining parameters of viewpoint, lighting, material and texture and occlusion.

The study of computer vision looks to attempt the inversion of the imaging process by studying the constraints on the interplay of the factors that make up an image and then to generate (either explicitly or in a learning framework) cues that may be combined with image measurements to resolve the ambiguities and thus recover from the information loss.

For example, a general scene has a very high dimensionality in terms of the freedom in geometry and reflectance, so there will be insufficient information to estimate these values without sufficient constraints, or regularisation, being placed on both the object shape and surface reflectance. We would argue that the analysis of different scene representations and the establishment of techniques which enforce different assumptions lies at the heart of 3D computer vision research.

1.5 Dissertation Outline

As discussed in § 1.2, one of the principle aims of this thesis is to present an automatic reconstruction system suitable for an untrained or inexpert user. The task of acquiring a 3D model from images may be broken down into a four stage pipeline, presented in Figure 1.4. In the context of this pipeline we need to automate as many of the stages as possible in order to automate the overall process. If we look at the first stage, image acquisition, we would like to avoid the need for any specialist equipment so all the algorithms presented are designed to be used with images obtained from a standard digital camera which are now commonplace. This dissertation offers contributions to the remaining three stages of the pipeline and is presented as follows.

Chapter 2 begins by providing a review of the mathematical model used to represent a camera by all stages of the reconstruction process. In order to perform calculations using this model, we must determine the parameters of the model for a particular camera used to take a set of images. This is the camera calibration process and the second half of the chapter provides a background of the established calibration techniques.

After this introduction, Chapter 3 presents our practical implementation of a camera calibration system. Our technique is designed to perform automatic calibration without the need for known calibration objects, instead it uses a textured planar surface to constrain sufficiently the problem. The chapter describes each stage of the algorithm starting with obtaining feature correspondences followed by robust estimation of the planar homographies. Once these have been calculated, linear calibration may be performed as an initialisation for an iterative refinement of the focal length. The final stage is to perform non-linear optimisation, in the form of bundle adjustment, to ensure the calibration is as accurate as possible. We find our system to demonstrate improved performance over an existing state-of-the-art automatic calibration system [Snavely et al. 2006] for planar scenes.

The third stage in the reconstruction pipeline is to separate the foreground object from the background by performing image segmentation and the final stage of the pipeline is

the recovery of shape itself. Chapter 4 begins by presenting an introduction to the segmentation task and outlines the sources of information used by existing algorithms that allow users to perform image segmentation in an interactive process. We then move to review the topic of shape reconstruction with an overview of the range of techniques, followed by focusing on multi-view stereo algorithms. Whilst the top performing algorithms are known to produce accurate results with large number of images, the reconstruction quality is found to decrease substantially with the number of images. This motivates the need for improved sparse reconstruction techniques. The two topics are tied together by an elegant framework, the Markov Random Field, comprising both a mathematical model and a set of optimisation strategies. We present an introduction to Markov Random Fields and review energy minimisation algorithms. The chapter concludes by bringing together the information sources and framework to explain an existing state-of-the-art algorithm.

Chapter 5 details our first contribution to segmentation, an automatic segmentation algorithm for multi-view stereo image sequences [Campbell et al. 2007; 2010]. We motivate the need for an automatic algorithm by explaining that existing systems will require the user to manual segment each image individually. We identify that the image sequence represents many views of the same rigid object and thus the silhouettes of the object are related across images, that is to say they must be coherent. We also propose that the camera will fixate on the object whilst the images are being taken. The chapter then proceeds to demonstrate that we may incorporate these constraints into an iterative algorithm that builds probabilistic colour models for the object and background and uses this information to segment all the images simultaneously in 3D. This enforces the silhouettes to be coherent and the fixation condition allows for automatic initialisation.

We then extend the range of image sequences we can segment through our second contribution to segmentation, the interactive segmentation algorithm presented in Chapter 6. We analyse the limitations of automatic segmentation and propose an interactive approach where the user is asked to label a very small subset of the pixels in the scene. Using the multi-view rigidity constraints we are then able to propagate these manual labels to the entire sequence. The key to our method is the formulation of the segmentation task via a clustering problem. Pixels across the sequence are clustered according to their appearance as well as adherence to the rigidity constraints present in the sequence. The user's input is then used, at interactive speeds, to select which pixel cluster(s) form the object of interest. The proposed method provides an easy to use, interactive segmentation process that minimises the user's input whilst performing segmentations across long sequences.

Chapter 7 presents our contribution to the reconstruction problem, the depth-map estimation algorithm tailored for use with multi-view stereo systems [Campbell et al. 2008].

We show that careful analysis of the matching process used by many multi-view stereo systems reveals its shortcomings and by studying these failure modes we can construct a filtering algorithm that removes outliers. This is performed by obtaining multiple hypotheses for the depth of each pixel and then identifying the true surface location, by looking for spatial support from neighbouring pixels, or returning an unknown state in the event that the true depth cannot be found. We demonstrate that this process improves the performance of multi-view stereo systems under sparse data-sets by comparison with the standard quantitative benchmark [Scharstein and Szeliski 2002a].

The dissertation concludes in Chapter 8 where we present some results from the automatic reconstruction system and find that the combined contributions have produced a system that is capable of recovering 3D shape from images in an automatic fashion without the need for specialist equipment or users and obtain accurate results with fewer images. Finally we identify shortcomings that present avenues for both short term and long term future work.

CHAPTER 2

The Camera Model and Calibration

2.1 The Camera Model

In order to invert the imaging process we must model the image formation process, in particular it is important to model the geometry of forming an image from light rays. To study this we formulate a camera model which details the projection of points in the 3D world into pixel co-ordinates within the image. The camera model used throughout is the central projection camera model (or pinhole camera). A more comprehensive introduction to projective geometry and camera models may be found in [Hartley and Zisserman 2004], here we present an overview of the specific models used by our algorithms. The mathematics of camera projection are expressed most easily in homogeneous co-ordinates (working in a projective space before returning to the standard Euclidean space). Here we offer a brief introduction.

2.1.1 Homogeneous Co-ordinates

The world around us, the scene we wish to reconstruct, is well modelled by 3D Euclidean geometry and its associate concepts of points, lines, distances, orthogonality and parallelism. When looking at visual projection, however, we note that these models fail to describe observed phenomena correctly. For example the parallel lines of a railway track in the real world appear to converge in an image, intersecting in a 'point at infinity' which is not modelled in Euclidean space. To overcome this we move from points in $\mathcal{R}^3$, our Euclidean space, to a projective space $\mathcal{P}^3$ where points are represented in homogeneous co-ordinates as a 4-vector, for example $\tilde{\mathbf{X}} = [X, Y, Z, W]^T$. In this space points are only defined up to scale, therefore $\tilde{\mathbf{X}}$ and $\lambda\tilde{\mathbf{X}}$ represent the same point, and at least one of the 4 co-ordinates must be non-zero. We are now able to model the plane at infinity as any point with $W = 0$ and we may use the remaining co-ordinates to approach infinity in a given

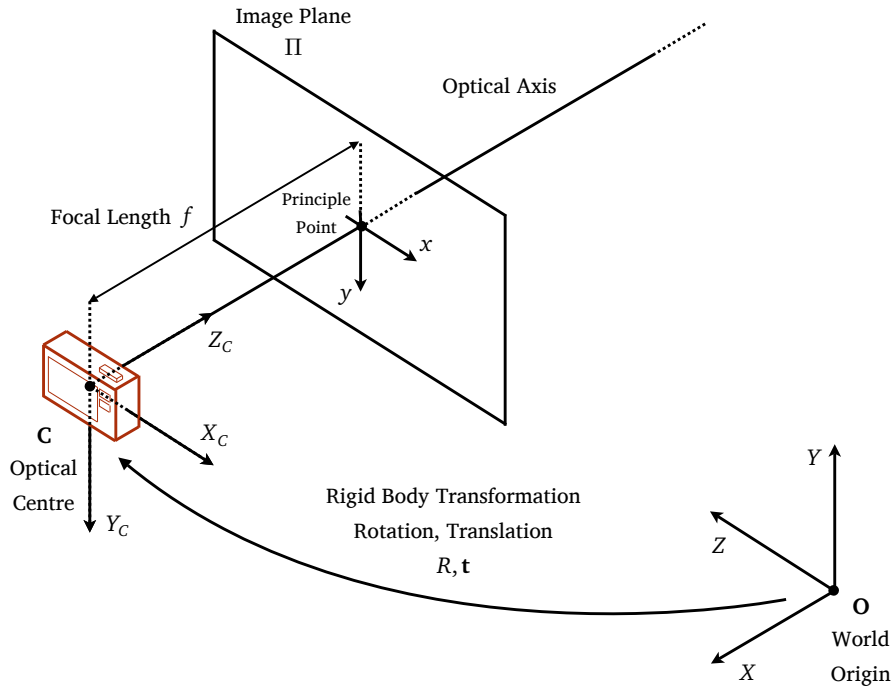


Figure 2.1: The geometry of the central projection camera model.

direction. Thus the vanishing point for the parallel train tracks previously discussed will lie in this plane at infinity. In order to take points (project them) from $\mathcal{P}^3$ to $\mathcal{R}^3$ we divide by the 4^{th} co-ordinate, thus $\mathbf{X} = [X/W, Y/W, Z/W]^T$. We can see that this projection is not valid for points in the plane at infinity which is in correspondence with the fact that such 'ideal' points do not appear in Euclidean space. If we wish to operate in the other direction we observe that setting $W = 1$ and the other co-ordinates to be equal will take a point from $\mathcal{R}^3$ to $\mathcal{P}^3$. The same such reasoning applies to $\mathcal{R}^2$ and $\mathcal{P}^2$ for modelling the image plane.

2.1.2 The Central Projection Camera

The central projection camera models the effect of perspective projection, a topic well studied by Renaissance painters. Figure 2.1 illustrates the geometry of the central projection camera, with respect to the world scene, and Figure 2.2 details the projection of a world point under the camera model. We begin by choosing a projection centre **C** and an image plane Π, located at a distance f. We then define a set of camera centred co-ordinates $\mathbf{X}_C$ with respect to **C** and Π as shown in Figure 2.1. We define a rigid body transformation, consisting of rotation R and translation $\mathbf{t}$, which transform world co-ordinates $\mathbf{X}$ (centred on the origin **O**) to these camera centred co-ordinates as $\mathbf{X}_C = R\mathbf{X} + \mathbf{t}$. Thus we observe

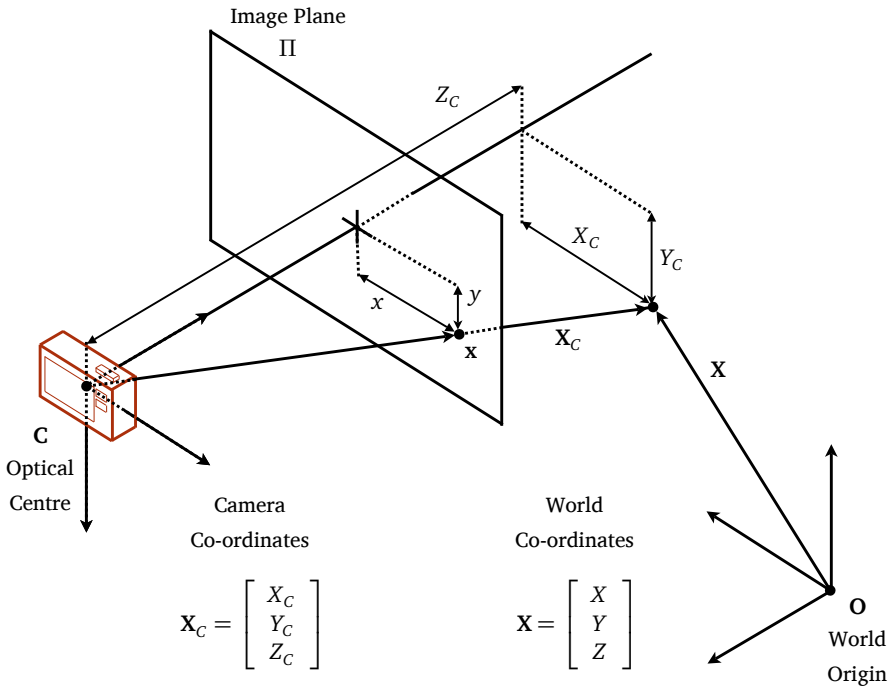


Figure 2.2: The projection of a world point under the central projection camera model.

that $\mathbf{t} = -R^T\mathbf{C}$.

We now consider the projection of a world point onto the image plane in Figure 2.2. We define a point in the image plane $\mathbf{x}$, centred on the principle point (the intersection of the optical axis with the image plane). The projection of the the point at camera centred co-ordinates $\mathbf{X}_C$ is therefore given as

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \left(\frac{f X_C}{Z_C}\right) \\ \left(\frac{f Y_C}{Z_C}\right) \end{bmatrix} . \tag{2.1}$$

If we use homogeneous co-ordinates we may represent this as

$$\tilde{\mathbf{x}} = \begin{bmatrix} f X_C \\ f Y_C \\ Z_C \end{bmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} = \begin{bmatrix} f & & 0 \\ & f & 0 \\ & & 1 \quad 0 \end{bmatrix} \tilde{\mathbf{X}}_C \ . \tag{2.2}$$

Furthermore we have $\tilde{\mathbf{X}}_C = T\,\tilde{\mathbf{X}}$, if we perform our co-ordinate transform in homogeneous co-ordinates as well, where T is the matrix

$$T = \left[\begin{array}{c|c} R & \mathbf{t} \\ \hline \mathbf{0}^T & 1 \end{array}\right] . \tag{2.3}$$

The final step is to map the image plane co-ordinates to pixel co-ordinates. This corresponds to mapping the plane to the CCD array in a digital camera. Note that traditionally the top-left image pixel has the pixel co-ordinate $\mathbf{u} = [0,0]^T$, thus we need to offset the pixel co-ordinates which are centred on the principle point. If we give the principle point the pixel co-ordinate $[u_0, v_0]^T$ and allow different scaling parameters for each axis (in the case of non-square pixels) we have

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} k_u x + u_0 \\ k_v y + v_0 \end{bmatrix} \tag{2.4}$$

which in homogeneous co-ordinates becomes

$$\tilde{\mathbf{u}} = \begin{bmatrix} k_u & & u_0 \\ & k_v & v_0 \\ & & 1 \end{bmatrix} \tilde{\mathbf{x}} \ . \tag{2.5}$$

We combine (2.2) and (2.5) to form the camera calibration matrix

$$K = \begin{bmatrix} k_u & & u_0 \\ & k_v & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

which contains the intrinsic calibration parameters for a camera. This intrinsic calibration matrix has 4 degrees of freedom however we often make the assumption that the aspect ratio is constant, i.e. the pixels are square, and thus we have $\alpha_u = \alpha_v = \alpha$ which reduces to 3 degrees of freedom. The extrinsic camera parameters represent the viewpoint or pose of the camera by the rigid body transformation of R and $\mathbf{t}$. These combine to allow 6 degrees of freedom, 3 for the rotation and 3 for the translation. Combining everything together yields the camera projection matrix $P = K\ [R|\mathbf{t}]$ and thus the whole projection may be written as

$$\tilde{\mathbf{u}} = K\ [R|\mathbf{t}]\ \tilde{\mathbf{X}} = P\tilde{\mathbf{X}} \tag{2.7}$$

or if we make the scale explicit

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\ [R|\mathbf{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \ . \tag{2.8}$$

2.2 Camera Calibration

Given the camera model of § 2.1.2, the process of calibration is to estimate the intrinsic camera parameters of the K matrix, (2.6), as well as the camera pose, the rotation R and

translation $\mathbf{t}$ of (2.7), for all the images. This corresponds to estimating the 10 degrees of freedom expressed in (2.7), or 9 if we assume a constant aspect ratio. We may go further and dictate that the same camera be used to acquire all the images and thus the 4 (or 3) degrees of freedom associated with the intrinsic parameters are common to all the cameras and the remaining 6 for the pose must be determined for each image individually. A detailed analysis of camera calibration techniques and their performance may be found in [Hartley and Zisserman 2004], here we present a basic review of the standard calibration methods used with calibration patterns along the lines of the review in [Hernández 2004]. Since calibration patterns are impractical for many scenes, for example large scenes or scenes where the pattern may obscure an interesting part of the scene, auto-calibration methods have been developed that replace the need for a calibration pattern with constraints such as scene rigidity.

2.2.1 Standard Calibration Methods

The oldest calibration algorithms are from the photogrammetry community. These techniques require a calibration pattern of known geometry to be present in the scene. By identifying 2D image points corresponding to the projection of the known 3D points on the calibration object we may determine the projection matrix P. Thus the input to these algorithms consists of a set of 3D world points with corresponding 2D image points. The algorithms then proceed to estimate the projection matrix via an optimisation procedure. These procedures may be classified into two groups, linear and non-linear methods, based on the nature of the optimisation.

Linear Methods

The first methods were linear methods that allow direct estimation of the projection matrix using a least squares framework. The basic linear method, termed the Direct Linear Transform (DLT), was known to the photogrammetry community for many years, for example [Das 1949], and other fields [Sutherland 1963] before being introduced to the computer vision community [Abdel-Aziz and Karara 1971]. The world to image point correspondences $\tilde{\mathbf{X}}_i \longleftrightarrow \tilde{\mathbf{x}}_i$, in homogeneous co-ordinates, are related by

$$\tilde{\mathbf{x}}_i = \lambda P \tilde{\mathbf{X}}_i \tag{2.9}$$

where we have made the scale explicit as λ. If we let P be represented as

$$P = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{bmatrix} \tag{2.10}$$

we may take the cross-product $\tilde{\mathbf{x}}_i \times P\tilde{\mathbf{X}}_i$ to remove the scale and thus we have

$$\begin{bmatrix} \mathbf{0}^T & -w_i\tilde{\mathbf{X}}_i^T & y_i\tilde{\mathbf{X}}_i^T \\ w_i\tilde{\mathbf{X}}_i^T & \mathbf{0}^T & -x_i\tilde{\mathbf{X}}_i^T \\ -y_i\tilde{\mathbf{X}}_i^T & x_i\tilde{\mathbf{X}}_i^T & \mathbf{0}^T \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = \mathbf{0} \ . \tag{2.11}$$

We only require the first two rows of (2.11) since the 3 equations are linearly dependent. If we take these two rows for each of n point matches we may stack the equations of (2.11) as

$$A\mathbf{p} = \mathbf{0} \tag{2.12}$$

where A is a $2n \times 12$ matrix and $\mathbf{p}$ contains the elements of the projection matrix P. To obtain the 12 entries of P, noting that it is only defined up to scale and therefore has 11 degrees of freedom, we need at least $5\,1/2$ point matches (the half being a single row). In the minimal case we obtain an exact solution to (2.12) since A will have rank 11 and therefore $\mathbf{p}$ will correspond to the null-space of A. The system may also be solved for $n \geq 6$ subject to a constraint on $\mathbf{p}$. If we note the scale invariance we solve (2.12) using the SVD (Singular Value Decomposition) of A to minimise $||A\mathbf{p}||$ such that $||\mathbf{p}|| = 1$. Different constraints have been proposed to solve (2.12) with [Faugeras and Toscani 1986] proposing possibly the most robust as $||\,[p_{31}, p_{32}, p_{33}]^T\,|| = 1$. It should be noted that the numerical properties of the DLT are greatly improved if data normalisation is used [Hartley 1997]. We may decompose the projection matrix as $P = K\ [R\,|\,\mathbf{t}]$ using RQ-decomposition on the first 3 columns of P to produce an upper triangular matrix and an orthogonal matrix, from this we may recover $\mathbf{t}$.

Subsequent work on linear methods has produced algorithms operating on fewer point correspondences when not all the information is required. For example we may estimate camera pose with as few as 3 correspondences [Haralick et al. 1991] although 4 are required for a unique solution [Quan and Lan 1998]. If we restrict some of the internal parameters [Triggs 1999] shows a method for recovering the pose and focal length from 4 points and the principle point as well from 5 points.

Although the linear methods allow for direct calculation of the camera parameters the results are not very accurate due to the fact that optimisation criterion has no valid geometrical meaning in addition to the poor parametrisation of the rotation (3 degrees of freedom are expressed via 9 parameters of the 3×3 matrix rather than a rotation axis and angle). Thus these methods are often used as an initialisation for non-linear optimisation procedures.

Non-Linear Methods

If we allow freedom in the parametrisation of the camera model and its parameters we depart from linear equations but allow the use of cost functions that reflect the quantities which we desire to minimise; for example the reprojection error in the 'Gold Standard' algorithm of [Hartley and Zisserman 2004]. Unfortunately these functions lead to much more complex optimisation procedures and we are often forced to use local optimisation techniques, such as gradient-based methods, that are not guaranteed to converge to a global optimum. In these cases we rely on a linear method to produce an initial estimate which is in the correct cost 'basin' to allow the local optimisation to converge to a meaningful solution. The work of [Tsai 1987] was one of the first computer vision applications to introduce a non-linear calibration process that uses the camera intrinsic parameters directly, along with appropriate parametrisations for the rotations and translations. The cost function minimised reflects the true 2D reprojection error in the images with the estimated camera matrix as

$$\hat{P} = \arg\min_{P} \sum_{i} \left\| \tilde{\mathbf{x}}_i - P\,\tilde{\mathbf{X}}_i \right\|^2 \tag{2.13}$$

which is the maximum likelihood estimate of P under Gaussian errors. Many variants of this technique have followed with perhaps the most interesting being the case where one is unsure of the exact 3D structure and would therefore like to optimise the 3D world positions at the same time as the camera parameters. Methods along these lines were again proposed first in the photogrammetry community [Brown 1976] and have been given the term 'Bundle Adjustment' and received extensive analysis in the computer vision community, for example the survey [Triggs et al. 1999]. The following section provides an overview of the bundle adjustment procedure that may be initialised with linear estimates for the camera parameters and estimates for the world positions.

Bundle Adjustment

The bundle adjustment process uses the Levenberg-Marquardt (LM) algorithm to minimise the reprojection error: the error between the projection of the real world points into the images, under the current calibration parameters, and the observed image points. This is provided in (2.14) where $i = 1 \ldots n$ points are observed in $j = 1 \ldots m$ images.

$$\hat{C} \quad = \quad \min_{\mathbf{a},\mathbf{b}_j} \sum_{i=1}^{n} \sum_{j=1}^{m} d\left(\mathbf{u}_{i,j}\,, Q\left(\mathbf{a}, \mathbf{X}_i, \mathbf{b}_j\right)\right)^2 \tag{2.14}$$

Here we have $d(\cdot,\cdot)$ as the Euclidean distance function used to evaluate the distance between $\mathbf{u}_{i,j}$, the 3D world point $\mathbf{X}_i$ observed in j^{th} image, and the projected position

$Q\left(\mathbf{a}, \mathbf{X}_i, \mathbf{b}_j\right)$ of the same world point under the current calibration parameters. The vector $\mathbf{a}$ encodes the intrinsic parameters of the camera

$$\mathbf{a} = \left[\alpha_u, \alpha_v, u_0, v_0, \kappa_1, \kappa_2\right]^T \tag{2.15}$$

found in (2.6) along with the two radial distortion coefficients κ_1 and κ_2 [Hartley and Zisserman 2004]. The radial distortion parameters intend to capture the disparity between the perspective pin-hole camera model of (2.7) and the lens of the real camera. We therefore insert a term between the conversion to camera co-ordinates, the rigid-body transformation, and the intrinsic matrix, encoding the perspective CCD imaging, as shown in (2.18). The other parameter vector $\mathbf{b}_j$ encodes the extrinsic parameters for the j^{th} image, i.e. $\mathbf{b}_j = \left[\mathbf{q}_j^T \ \mathbf{t}_j\right]^T$ where the rotation matrix R_j is converted into the 4 dimensional quaternion $\mathbf{q}_j$. Therefore the stages encoded by $Q\left(\mathbf{a}, \mathbf{X}_i, \mathbf{b}_j\right)$ for $\mathbf{X} = \mathbf{X}_i$ and $R = R_j, \mathbf{t} = \mathbf{t}_j$ are

$$\begin{aligned}
\mathbf{X}_c &= R\mathbf{X} + \mathbf{t} && (2.16)\\
\mathbf{x}_c &= \begin{bmatrix} \frac{X_c}{Z_c} & \frac{Y_c}{Z_c} \end{bmatrix}^T && (2.17)\\
\mathbf{x}_c' &= \left(1 + \kappa_1 \|\mathbf{x}_c\|^2 + \kappa_2 \|\mathbf{x}_c\|^4\right) \mathbf{x}_c && (2.18)\\
s\,\tilde{\mathbf{u}} &= K\,\tilde{\mathbf{x}}_c' \ . && (2.19)
\end{aligned}$$

The problem is then posed in the standard form for an LM problem. If we have

$$\begin{aligned}
\mathbf{p} &= \left[\mathbf{a}^T, \mathbf{b}_1^T, ..., \mathbf{b}_m^T\right]^T && (2.20)\\
\mathbf{Y} &= \left[\mathbf{u}_{1,1}^T, ..., \mathbf{u}_{1,m}^T, \mathbf{u}_{2,1}^T, ..., \mathbf{u}_{2,m}^T, ..., \mathbf{u}_{n,1}^T, ..., \mathbf{u}_{n,m}^T\right]^T && (2.21)
\end{aligned}$$

and we have

$$\hat{\mathbf{u}}_{i,j} = Q\left(\mathbf{a}, \mathbf{u}_i, \mathbf{b}_j\right) \tag{2.22}$$

then

$$\begin{aligned}
\hat{\mathbf{Y}} &= f(\mathbf{p}) && (2.23)\\
&= \left[\hat{\mathbf{u}}_{1,1}^T, ..., \hat{\mathbf{u}}_{1,m}^T, \hat{\mathbf{u}}_{2,1}^T, ..., \hat{\mathbf{u}}_{2,m}^T, ..., \hat{\mathbf{u}}_{n,1}^T, ..., \hat{\mathbf{u}}_{n,m}^T\right]^T \ . && (2.24)
\end{aligned}$$

The LM algorithm is based around the Taylor expansion

$$f(\mathbf{p} + \delta\mathbf{p}) \approx f(\mathbf{p}) + J\,\delta\mathbf{p} \tag{2.25}$$

where

$$J = \left[\frac{\partial}{\partial \mathbf{p}} f(\mathbf{p})\right] \tag{2.26}$$

is the Jacobian matrix. Provided with an initial parameter estimate $\mathbf{p}_0$, in our case the linear estimate, and a measurement vector $\mathbf{Y}$, the iterative algorithm intends to find the $\delta\mathbf{p}$ that minimises

$$\left\| \mathbf{Y} - f(\mathbf{p}) - J\,\delta\mathbf{p} \right\| \quad = \quad \left\| \epsilon - J\,\delta\mathbf{p} \right\| \tag{2.27}$$

where $\epsilon = \mathbf{Y} - \hat{\mathbf{Y}}$ is the current error. This is the solution to a linear least squares problem with $\delta\mathbf{p}$ being the solution to the normal equations

$$J^T J\,\delta\mathbf{p} \quad = \quad J^T\,\epsilon \ . \tag{2.28}$$

The Newton update, which solves (2.28), assumes a quadratic cost function. Whilst this offers fast convergence properties when near the minimum, it may perform poorly far from the solution when the cost function is not quadratic. LM offers superior convergence properties by actually solving the augmented normal equations

$$N\,\delta\mathbf{p} \quad = \quad J^T\,\epsilon \tag{2.29}$$

$$N \quad = \quad \left[J^T J + \mu I \right] \tag{2.30}$$

where I is the identity matrix and μ is a damping parameter that is adaptively updated to improve the convergence properties as the algorithm runs. When close to the solution, μ takes low values and (2.30) performs a Newton update whereas when far from the solution μ takes a larger value to move (2.30) to a gradient descent update. In the case of Bundle Adjustment the matrices involved will be sparse and efficient methods of solving the update equations may be found [Hartley and Zisserman 2004].

2.2.2 Auto-Calibration

Auto-calibration is the term given to methods that determine the camera parameters for multiple images without the use of calibration patterns. As such, these techniques need to exploit different constraints to make up for the absence of known world points. A variety of constraints have been exploited in the literature [Faugeras and Luong 2001] with most based around detecting correspondences between a constant primitive across multiple images. Perhaps the most typical example of auto-calibration is to constrain the scene to be rigid across multiple camera views, the assumption that all objects in the scene remain static in all the images, and then constrain all the cameras to have the same intrinsic parameters. These two constraints allow a well-posed problem to be formed by matching correspondences between the images rather than correspondences between world points and image points. In general the auto-calibration constraints are imposed in a two step

process. Firstly general projective matrices P_i are obtained: these are general 3×4 matrices which have a projective ambiguity (a collinearity in projective space [Hartley and Zisserman 2004]) which prevents us from direct recovery of K, R and $\mathbf{t}$. This is then followed by a process to update these to the Euclidean camera matrices of (2.7).

The first stage uses point correspondences, usually obtained using feature detection and matching, between images and epipolar geometry to estimate the projective relationship between images. Again a thorough discussion of these processes may be found in [Hartley and Zisserman 2004]. Looking at the second stage, namely updating projective to Euclidean co-ordinates, we may further subdivide the techniques used into either 'direct' or 'stratified' methods. The direct approach is to go straight from the projective estimate to a Euclidean one through an optimisation procedure. The stratified approach makes the transition via an intermediate stage by finding the plane at infinity to produce an affine camera estimate before refining this to a Euclidean one. The literature has shown that stratified methods are in general more robust [Pollefeys and Gool 1997].

To provide an overview of the projective to Euclidean updating procedures we first look at the direct methods. The first auto-calibration method [Faugeras et al. 1992, Maybank and Faugeras 1992] proposed an approach based on the Kruppa equations [Kruppa 1913] establishing the relationship between the intrinsic camera parameters and the absolute conic. We may denote the image of the absolute conic as ω and thus may define the relationship with K as

$$\omega = (K K^T)^{-1} \quad . \tag{2.31}$$

The Kruppa equations have the advantage of not requiring computation of the plane at infinity or the use of a unified co-ordinate system since they use only the epipolar geometry. Unfortunately this comes at the expense of robustness and places a limit on the number of images which may be considered due to the fact that multiple solutions are obtained. This is combined with the problem of degenerate camera motions [Sturm 2000]. The work of [Hartley 1993] uses a QR decomposition to improve robustness over the Kruppa equations and remove restrictions on the number of views considered. Further work has shown that the absolute quadric may be estimated from a set of images [Triggs 1997] using similar equations to [Heyden and Astrom 1996]. These methods, unlike the Kruppa equations, compute the plane at infinity which provides the increased robustness.

Turning our attention to stratified approaches, we note that the main difference is to first compute the plane at infinity, to upgrade to affine, and then continue to estimate the intrinsic parameters in a two step system. The methods of [Hartley 1993, Armstrong et al. 1994, Faugeras 1995] display this idea of separating the computations. Perhaps the top performing approach for these methods is the work of [Pollefeys and Gool 1997] that

proposes a fully stratified approach making use of the 'modulus constraint' to recover the plane at infinity [Pollefeys et al. 1996].

2.2.3 Automatic Calibration

Whilst the term auto-calibration refers directly to the absence of a known calibration object, here we use the term automatic calibration to suggest approaches that require little or no input from the user, they simply provide the photos themselves. The auto-calibration methods are numerical procedures that will infer calibration parameters from image point correspondences and thus to provide an automatic calibration system we need to have an automatic procedure for determining these point correspondences. In general the numerical algorithms expect these point correspondences to have an error distribution that is Gaussian (in the 2D image plane) with respect to the true correspondence location. Therefore the point correspondences must not contain incorrect matches, termed outliers.

The most popular procedure for finding these point correspondences is split into two processes. Firstly a feature detector is used to find locations in the image that correspond to a physical structure in the scene and may thus be found in a manner independent of the viewpoint from which the image was taken. Secondly a feature descriptor is found, usually in the form of a vector in a high dimensional space, that captures a description of the feature point, or a set of sufficient statistics, which is invariant to a variety of transformations, from scale and rotation up to affine and projective deformations. The review of [Mikolajczyk et al. 2005] provides a thorough overview and quantitative analysis of many of the top performing algorithms. In particular the Scale-Invariant Feature Transform (SIFT) algorithm [Lowe 2004] has gained great popularity, comprising both a feature detector and a descriptor.

Once we have a set of feature descriptors corresponding to sets of points in different images, a matching process is required to identify correct correspondences. The feature descriptors are usually matched using a nearest neighbours procedure, in the vector space, to produce a set of putative correspondences. These are then filtered and refined to remove the outliers and retain only true correspondences. The RANSAC algorithm [Fischler and Bolles 1981, Torr and Murray 1993] and its derivatives provide an iterative algorithm to partition the matches into inliers and outliers by selecting matches at random and measuring their support amongst the remaining matches. If a true set of inliers is selected it should have a large support set that may be taken to be the inliers. An example of its use is given in § 3.3.

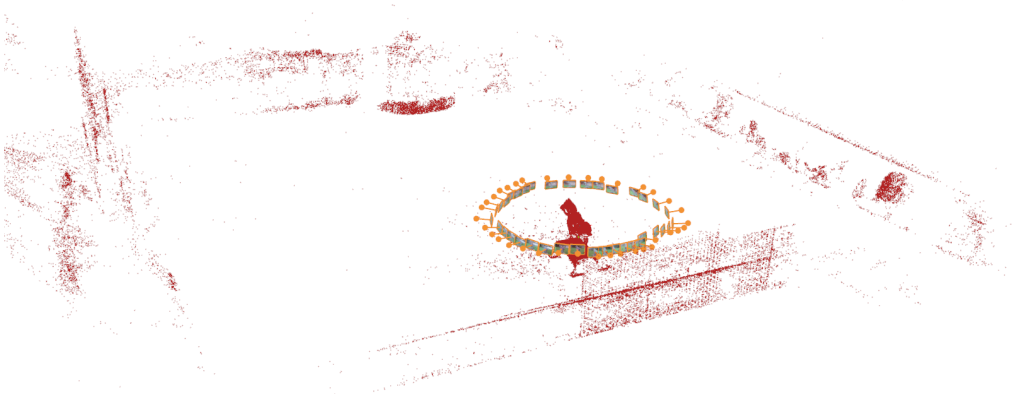
Several popular calibration systems make use of these procedures to calibrate a set of images automatically, including identifying which images correspond to the same scene in

the first place [Brown and Lowe 2005]. Further work has been performed on expanding this to cope with calibrating many hundreds of images from Internet data-sets [Snavely et al. 2006], all in an autonomous fashion. In addition to recovering the camera calibration, the 3D position of the points in the world corresponding to the image points may be recovered. Figure 2.3 shows the results of applying the algorithm of [Snavely et al. 2006] to a set of images of a horse statue. As well as recovering the calibration parameters of the cameras, sets of corresponding image points may be used to estimate the 3D location of the features that give rise to the correspondences by triangulation [Hartley and Zisserman 2004]. These points are shown in red in Figure 2.3.

Whilst this system performs very well for many scenes, it is not capable of performing under all conditions, in particular when the scene is dominated by a single plane. The system makes use of the five-point algorithm of [Nister 2004] to find the relative pose of two intrinsically calibrated cameras using five corresponding image points. Whilst theoretically the algorithm is not degenerate if all the points lie on a single world plane, it has been shown that the algorithm is not robust to noise in terms of inaccuracies in the position of the matches or the calibration [Segvic et al. 2007]. Thus there are a number of applications for model acquisition when the item of interest is located on a planar surface, for example a table top, under which the method of [Snavely et al. 2006] performs poorly. This motivates us to produce an automatic calibration system that can cope reliably with the presence of a dominant world plane. In Chapter 3 we show that, if we are willing to place an extra constraint that the user must take a top down photo of the plane before calibration, it is possible to build an automatic calibration system that will confer greatly improved performance over algorithms similar to [Snavely et al. 2006].

(a) Images of a horse sculpture

(b) Example of the recovered camera calibration and triangulated 3D 'point cloud'



(c) The entire scene structure

Figure 2.3: Calibration results for a sequence of images of a horse statue. *The method of [Snavely et al. 2006] was applied to 36 images of a horse statue. (a) Half of the images used. (b) The recovered camera positions and world points observed from the first image. All the images are shown (rendered transparently) with their poses indicated in orange and the scene structure in red. This scene structure is a 'point cloud' of dots indicating the 3D position of a point that was matched across two or more views from image correspondences and its position subsequently triangulated. (c) The same scene shown from a wider angle showing the extent of the recovered 3D structure (in red) that maps out the four sides of the square court. The positions of the cameras, encircling the recovered points on the horse statue, are shown in orange.*

CHAPTER 3

Automatic Camera Calibration

As shown in Figure 1.4, the first computational task in the reconstruction pipeline is to calibrate the camera used to take the photos as well as estimate the positions the photos were taken from. Since we are attempting to create an autonomous system that is easy to use it is important that we are able to provide the user with an automatic process for calibration that doesn't require any complex or expensive equipment or technical knowledge. In § 2.2.3 we saw that there are existing automatic calibration systems, such as the Bundler system of [Snavely et al. 2006], that perform well for non-planar scenes. For example the images of Figure 2.3. However we also identified that there are a range of scenes, those containing a dominant planar surface, that are not well handled by this method. Here we propose an algorithm that is easy to use and we demonstrate that it outperforms [Snavely et al. 2006] for planar scenes.

3.1 Introduction

There are a number of constraints that may be exploited in order to perform camera calibration as discussed in § 2.2. We wish to obtain a full calibration of the camera that encompasses both the intrinsic parameters of the camera and the pose of the camera for each image taken. We have seen that the general classes of camera calibration may be divided into photogrammetric calibration techniques and auto-calibration techniques. Photogrammetric calibration methods require views of a known calibration object, usually in three dimensions such as three orthogonal planes, that will allow the mapping of image points to the known 3D locations to be determined: this mapping is the camera calibration. On the other hand, auto-calibration techniques rely on the rigidity of a particular scene observed in multiple views. By tracking scene correspondences across these views, the scene rigidity gives rise to two constraints that may be used to calibrate the camera. The approach in

[Zhang 2000] lies between these two classes since it makes use of a known two dimensional calibration pattern. We extend this approach towards the auto-calibration class by removing the requirement for a calibration pattern of known metric, instead we rely on a textured plane to provide the correspondences and make use of the planar constraints to allow for full calibration.

In order to provide an accurate camera calibration it is not always sufficient to adopt simply a 'structure from motion' technique to track features on the object of interest. This would result in feature drift: the fact that each feature is only observed in neighbouring frames results in a propagation of calibration error as the camera views navigate around the object and results in a significant loss of accuracy. A calibration technique based on the estimation of planar homographies protects against this and provides a more stable calibration system. The justification for the use of a such a calibration technique is that the many photos taken will have the object of interest located on a plane (the world plane) and thus if we can take an image of the planar surface prior to putting the object on top we can automatically construct our own calibration pattern and solve for the camera parameters and structures with respect to this pattern, thus avoiding the effects of drift. It is also a reasonable task to ask of a non-technical user of the system, that they simply take an image of the surface they will be placing the object on, rather than expect them to construct a calibration pattern specifically for each object sequence.

In order to perform calibration under these conditions it is necessary to match feature locations from a top-down 'seed' image (an image taken of the planar surface, without the object, looking straight down from above) to their corresponding locations in each of the images in the sequence (accounting for the fact that occlusions from the object will result in only a subset of the features being visible in each view). The standard putative correspondences obtained by comparing localised patches using normalised cross correlation [Hartley and Zisserman 2004] cannot be applied in this situation since there is significant rotation and scaling (wide-baseline matching) between the individual views and the top-down image. Therefore a robust system for affine invariant matching should be used. The SIFT detector of [Lowe 2004] provides such a set of key-points with descriptors that provide a degree of robustness to affine transformations and may therefore be used to identify correspondences. There will be a degradation in performance due to the perspective geometry (rather than simply affine) however since planar correspondences are required this is a reasonably good approximation.

Our technique draws inspiration from the work of [Triggs 1998] which was one of the first auto-calibration techniques for a planar scene. In the discussion of [Triggs 1998] it is noted that the approach aims to perform the calibration based on homographies from

the scene plane to the individual images but that in practice one may only estimate inter-image homographies and thus the choice of a key image may have an undue influence on numerical stability. By dictating that a top-down image be taken we ensure that a suitable key image is made available and also allow the widest possible viewing arc for image correspondences since any homography estimation across a world plane will be limited by the angle subtended between the two optical axes and the plane. The top-down image also allows us to resolve a two-fold ambiguity in the estimated plane normal and set a baseline scale. We then combine this with a simple optimisation scheme and linear estimation phase, which we may then pass on to bundle adjustment, rather than optimise the complex cost function of [Triggs 1998].

The calibration algorithm is given in Algorithm 1. The remaining sections discuss the individual aspects of the method in further detail.

3.2 Obtaining Feature Correspondences

In order to calibrate against a pattern we need to be able to find corresponding locations within the two images, essentially we need to be able to locate the calibration pattern within each of the images to be calibrated. A state-of-the-art system for finding objects in images has recently been developed [Lowe 2004]. The algorithm is termed the Scale-Invariant Feature Transform (SIFT) and intends to detect similar feature points in each of the available images and then describe these points with a feature vector that is independent of image scale and orientation. Thus feature points which correspond to different views of the same object should have similar feature vectors. If this process is successful then we should be able to use a simple algorithm to compare the collected set of feature vectors from one image to another in order to find corresponding feature points in each image.

The SIFT algorithm may be decomposed into four stages:

1. Feature point detection
2. Feature point localisation
3. Orientation assignment
4. Feature descriptor generation

which are detailed in the original paper [Lowe 2004]. The output of the SIFT algorithm is a set of key-point locations $\mathbf{u}_i$ each with a corresponding features descriptor $\mathbf{d}_i$ which is a vector in 128 dimensions.

Algorithm 1: The automatic planar calibration algorithm.

Objective

- Estimate the intrinsic camera parameters and camera poses for an image sequence containing a plane

Input

- A top-down image of a plane I_r
- A sequence of images containing the plane I_m, $m = 1...M$
- Intrinsic camera parameters constant, but unknown, across all images

Algorithm

- **For each image in the sequence**
  1. Locate SIFT features and generate corresponding descriptors
  2. Use the SIFT descriptors to produce a set of putative correspondences from the top-down image to the view
  3. Robustly estimate the homography using the best correspondences and discarding the outliers
- Form a linear estimate of the camera focal length using planar homography constraints
- Perform an optimisation to refine the camera focal length based on the plane normal estimates
- Use the focal length to estimate the camera structure for each view
- Perform bundle adjustment using the matches between each view and the top-down image to optimise the camera parameters, structure and world point positions

Output

- Camera intrinsic matrix K
- Rotations R_m and translations $\mathbf{t}_m$ for each view

3.2.1 SIFT Feature Matching

Since the SIFT descriptor is a vector in 128 dimensions, Lowe proposes a very simple matching scheme based on the nearest neighbour. Essentially a feature descriptor from the query image may be compared to all the descriptors of features in the search image and matched to the feature with the closest descriptor vector. The problem with this approach is that there may be features which are not found in both images, therefore the nearest neighbour scheme enforces that a match is always returned, even if the descriptors themselves are not known. Lowe's solution is to compare the descriptors of the two nearest neighbours found in the search image. If the second nearest descriptor differs significantly from the first nearest neighbour then we assume that the descriptor is isolated in the vector space and may therefore be considered a good match, otherwise the match is rejected. This proceeds along the following lines:

$\mathbf{d}_q$ is the query descriptor

$\mathbf{d}_1$ is the first nearest descriptor in the search image

$\mathbf{d}_2$ is the second nearest descriptor in the search image

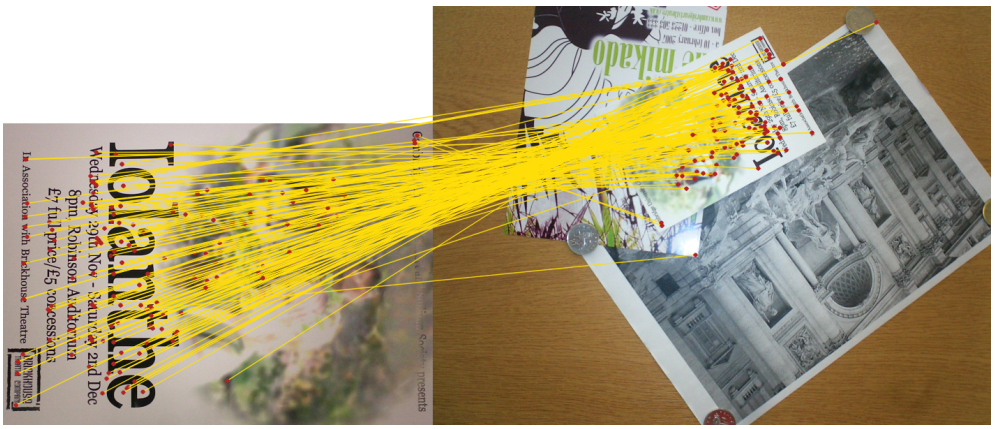
$$\text{reject unless} \quad \frac{\cos^{-1}(\mathbf{d}_q \cdot \mathbf{d}_1)}{\cos^{-1}(\mathbf{d}_q \cdot \mathbf{d}_2)} < r \quad \text{where } r \text{ is the threshold ratio.}$$

Figure 3.1 shows the results of this matching technique applied to a query and a search image with a threshold of $r = 0.6$. The matching results may return erroneous correspondences (outliers) in addition to correct matches (inliers) and therefore further processing stages are required to determine robustly the inlier matches. This allows the outliers to be ignored during the subsequent estimation of correlations between the images.

3.3 Homography Estimation

Two images of the same planar pattern are linked together by a transformation that warps the plane of one image into the other. This transformation is referred to as a (planar) homography. The calibration procedure requires the estimation of a series of homographies between a top-down image and all the images in the sequence to be calibrated. A homography between two images has 8 degrees of freedom, therefore a minimum of 4 corresponding 2D points are required to determine the transformation. The accuracy of the estimation will be increased if a greater number of correspondences are known and an optimisation method used to evaluate the homography.

(a) Detected SIFT features



(b) Matched SIFT features

Figure 3.1: Initial SIFT feature matches. *Sift features for the query (left) and search (right) images are found (a) and the descriptors are matched against one another (b) using a rejection threshold ratio of 60%. The putative correspondences show that there are some erroneous matches (outliers) including matches to the corners of different posters in the search image.*

Matching the SIFT descriptors for the top-down image and a sequence image will present a set of putative point correspondences for the two images. Figure 3.1 indicates that these initial correspondences will contain outliers, therefore a robust estimation scheme is required to select as many correct correspondences as possible and use them to determine the homography. The RANSAC algorithm [Fischler and Bolles 1981] provides such an estimation scheme and may be used to estimate the homography [Torr and Zisserman 1998]. The algorithm proceeds as in Algorithm 2.

The homography estimation follows a least squares formulation when over constrained. If we specify $n \geq 4$ correspondences we proceed with the following. Let $\mathbf{x}_i$ and $\mathbf{x'}_i$, $i = 0 \ldots (n-1)$, be the locations of the i^{th} match in the top-down and sequence images respectively. With $\bar{x}, \bar{y}$ as the means and s is the reciprocal of the standard deviation of $\{\mathbf{x}_i\}$ we define the transformation

$$T = \begin{pmatrix} s & 0 & -s\,\bar{x} \\ 0 & s & -s\,\bar{y} \\ 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

and similarly for T'. Thus we may have $\mathbf{z}_i = T\,\mathbf{x}_i$ and $\mathbf{z'}_i = T'\,\mathbf{x'}_i$ such that $\mathbf{z}_i$ and $\mathbf{z'}_i$ have zero mean and unit variance which greatly improves the numerical conditioning for the problem. This reversible transform may be undone at the end to retrieve the final homography. Posing the problem in the form $A\mathbf{h}_z = \mathbf{0}$, where $\mathbf{h}_z$ is a vector of the elements of H_z, we have

$$A = \begin{pmatrix} \mathbf{z}_i'^T & \mathbf{0}^T & -z_{i,x}\,\mathbf{z}_i'^T \\ \mathbf{0}^T & \mathbf{z}_i'^T & -z_{i,y}\,\mathbf{z}_i'^T \\ \vdots & \vdots & \vdots \end{pmatrix} \qquad \text{where } \mathbf{z}_i = [z_{i,x}, z_{i,y}, 1]^T.$$

Taking the SVD of A, the least squares result for $\mathbf{h}_z$ is given by the unit singular vector corresponding to the smallest singular value of A. Finally we remove the normalisation to recover the homography $H = T'^{-1} H_z\, T$.

Figure 3.2 shows the estimation of a homography between a typical top-down and sequence image. The outliers present in the initial SIFT correspondences of Figure 3.2(b) have been removed by the RANSAC stage and subsequent refinement in Figure 3.2(d).

3.4 Linear Camera Calibration

We make use of the central projection camera model described in § 2.1 and follow the process of [Zhang 2000]. The extrinsic camera parameters describe the 3D position of the camera, one set describes a single view, and consist of a 3×4 rigid-body transformation

Algorithm 2: Homography estimation using RANSAC.

Objective

- Estimate the projective homography between the top-down and a sequence image

Input

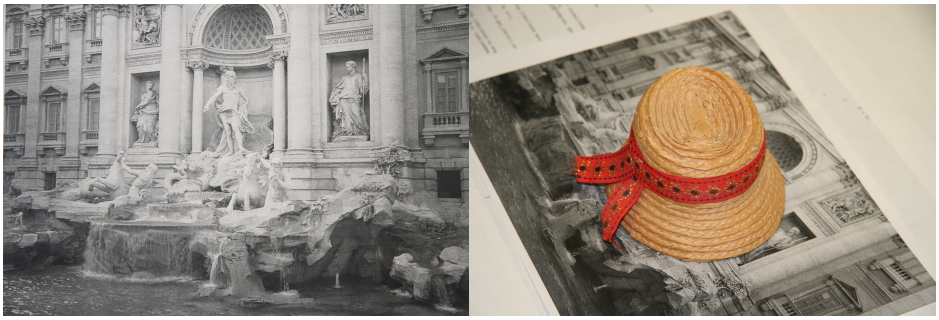
- Putative matches between the top-down and sequence image: $\mathbf{x}_i$ and $\mathbf{x}'_i$, $i = 1 \ldots n$

Algorithm

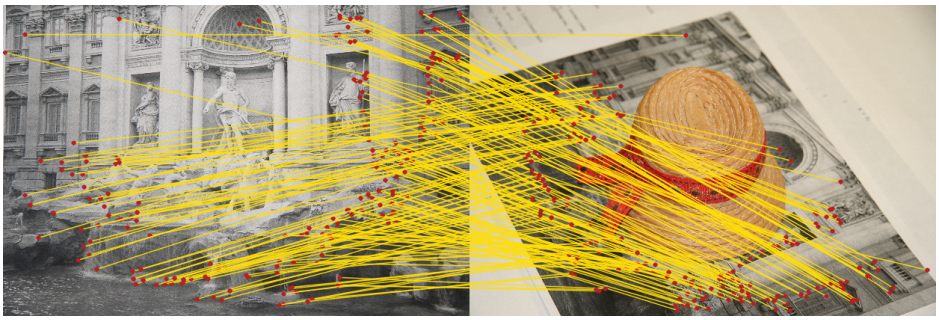
- **RANSAC Stage**
  - Repeat for N samples where N is updated as the algorithm proceeds
    1. Randomly select 4 of the putative correspondences and calculate the homography H_{est}
    2. Project all of the correspondences under the homography and calculate the distance d_i between the location of the putative match and the projected location
    3. Calculate the number of inliers, p, that are consistent with H_{est} as those correspondences with $d_i < t$ pixels where $t = \sqrt{5.99}\sigma$ is an accuracy threshold (σ = std. dev. error)
  - Return the homography $\bar{H} = \arg\max_p \left(H_{\text{est}}\right)$ with the greatest number of inliers
- **Refinement Stage**
  1. Project all feature points (not just SIFT matches) under the current $\bar{H}$ and determine the set of inliers for which $d_i < \sqrt{5.99}\sigma$
  2. Use all the inliers to determine the least squares estimate for $\hat{H}$
  3. Repeat stages (1.) $\rightarrow$ (2.) until the number of inliers converges
  4. If necessary, repeat stages (1.) $\rightarrow$ (3.) with decreasing values of σ to ensure sufficiently accurate results

Output

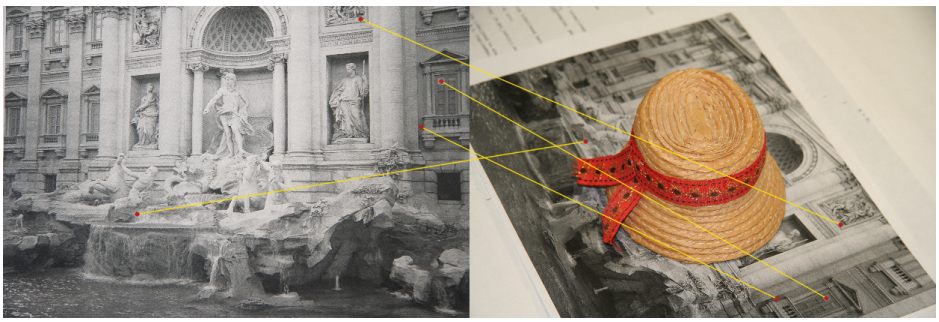
- Best linear homography estimate $\hat{H}$



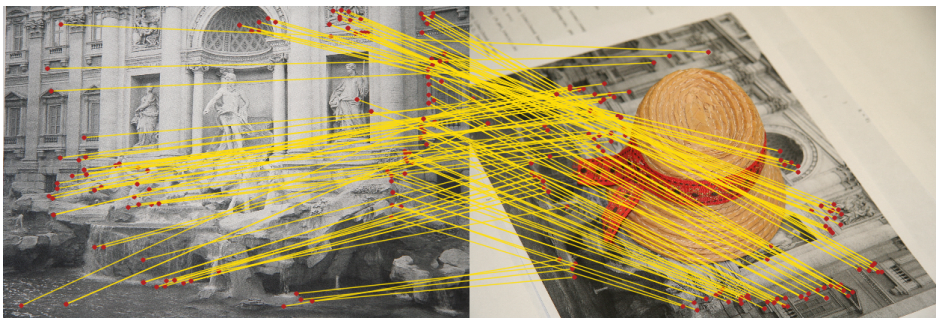
(a) Top-down and sequence image



(b) Results of SIFT matching (203 correspondences)



(c) Best matches selected by RANSAC stage (104 inliers)



(d) Refined matches used to estimate homography (129 correspondences)

Figure 3.2: Homography estimation using RANSAC. *SIFT features are found for the top-down and sequence images (a) and used to find putative correspondences (b) which contain outliers. The RANSAC stage selects the 4 correspondences with the best support (c) and then the final refinement stage provides a set of coherent correspondences (d) which are used to estimate the homography.*

matrix broken down into a rotation R and translation $\mathbf{t}$ such that we have $\alpha_u = \alpha_v = \alpha$ which we will term the focal length. Initially we fix the principle point $[u_0, v_0]^T$ to be at the centre of the image although this will be relaxed during the final bundle adjustment stage. Thus we have the intrinsic parameter calibration matrix as

$$K = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

and the full camera model given as

$$\tilde{\mathbf{u}} = \lambda K [R|\mathbf{t}] \tilde{\mathbf{X}} \tag{3.3}$$

or

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{3.4}$$

Zhang's method assumes a planar calibration pattern. Without loss of generality we may encode this plane as the plane $Z = 0$ therefore we have

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \tag{3.5}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots \\ \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{3.6}$$

which may be written as

$$\tilde{\mathbf{u}} = \lambda H [X, Y, 1]^T \tag{3.7}$$

where H is a 3×3 homography matrix. Now if we let

$$H = \begin{bmatrix} \vdots & \vdots & \vdots \\ \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{3.8}$$

we may write

$$\begin{aligned} H &= \lambda K \left[\mathbf{r}_1 \, \mathbf{r}_2 \, \mathbf{t} \right] \\ \Rightarrow \mathbf{h}_1 &= \lambda K \mathbf{r}_1 \\ \Rightarrow \lambda \mathbf{r}_1 &= K^{-1} \mathbf{h}_1 \end{aligned} \tag{3.9}$$

and similarly $\lambda \mathbf{r}_2 = K^{-1} \mathbf{h}_2$. Now we know that $\mathbf{r}_1$ and $\mathbf{r}_2$ are orthonormal and hence $\mathbf{r}_1 \cdot \mathbf{r}_2 = 0$. This leads to

$$\begin{aligned} \lambda^2 {\mathbf{r}_1}^T \mathbf{r}_2 &= \left(K^{-1} \mathbf{h}_1 \right)^T \left(K^{-1} \mathbf{h}_2 \right) \\ \Rightarrow 0 &= {\mathbf{h}_1}^T \left(K^{-T} K^{-1} \right) \mathbf{h}_2 \end{aligned} \tag{3.10}$$

and we also have

$${\mathbf{h}_1}^T \left(K^{-T} K^{-1} \right) \mathbf{h}_1 = {\mathbf{h}_2}^T \left(K^{-T} K^{-1} \right) \mathbf{h}_2 \, . \tag{3.11}$$

Now we note that ω, the image of the absolute conic [Hartley and Zisserman 2004], is given as

$$\omega = \left(K^{-T} K^{-1} \right) \tag{3.12}$$

$$= \frac{1}{\alpha^2} \begin{bmatrix} 1 & 0 & -u_0 \\ 0 & 1 & -v_0 \\ -u_0 & -v_0 & \left({u_0}^2 + {v_0}^2 + \alpha^2 \right) \end{bmatrix} . \tag{3.13}$$

As mentioned previously, we fix the principle point $[u_0, v_0]^T$ to be at the centre of the image for this initial estimation, therefore only α is unknown at this point. We wish to encode the constraints of (3.10) and (3.11) into an orthogonal least-squares framework therefore we need to evaluate

$$\begin{aligned} {\mathbf{h}_i}^T \omega \mathbf{h}_j &= \frac{1}{\alpha^2} \begin{bmatrix} h_{i1} & h_{i2} & h_{i3} \end{bmatrix} \begin{bmatrix} 1 & 0 & -u_0 \\ 0 & 1 & -v_0 \\ -u_0 & -v_0 & \left({u_0}^2 + {v_0}^2 + \alpha^2 \right) \end{bmatrix} \begin{bmatrix} h_{j1} \\ h_{j2} \\ h_{j3} \end{bmatrix} \\ &= {\mathbf{b}_{i,j}}^T \begin{bmatrix} \alpha^2 \\ 1 \end{bmatrix} \end{aligned} \tag{3.14}$$

where

$$\mathbf{b}_{i,j} = \begin{bmatrix} h_{i3} h_{j3} \\ h_{i1}(h_{j1} - u_0 h_{j3}) + h_{i2}(h_{j2} - v_0 h_{j3}) + h_{i3} \left(-u_0 h_{j1} - v_0 h_{j2} + ({u_0}^2 + {v_0}^2) h_{j3} \right) \end{bmatrix}$$

and $i, j \in \{1, 2\}$. Thus for every image in the sequence we evaluate the homography between the view and the top-down image. Each image confers two constraints in the

form of (3.10) and (3.11) which are encoded as two rows of V in the orthogonal least-squares problem

$$V \begin{bmatrix} \alpha^2 \\ 1 \end{bmatrix} = \mathbf{0} \ . \tag{3.15}$$

Thus every image provides the two rows

$$V = \begin{bmatrix} \vdots \\ \left[\mathbf{b}_{1,2}\right]^T \\ \left[\mathbf{b}_{1,1} - \mathbf{b}_{2,2}\right]^T \\ \vdots \end{bmatrix} \tag{3.16}$$

and α may be retrieved from the SVD of V by finding the the unit singular vector corresponding to the smallest singular value of V. This initial estimate for α is then further refined by a subsequent optimisation.

3.4.1 Focal Length Refinement and Structure Estimation

The linear estimate of the focal length may have a reasonable degree of error since the previous estimation assumed that the top-down image was taken with the camera's image plane exactly parallel to the world plane which will not be true in a realistic situation. To compensate for this we perform an optimisation to refine the initial estimate. Appendix 1 of [Triggs 1998] describes an algorithm for determining the relative orientation of two calibrated cameras from an unknown planar scene. The algorithm also estimates the normal of the world plane. Here we use this algorithm to formulate a cost function which optimises the focal length in order to minimise the disparity in estimates for the plane normals across the image sequence. This is possible since we always take homographies with respect to the top-down image which should observe the same world plane normal. We also resolve scale issues by fixing the world plane to be a set distance from the top-down camera position.

Let us denote the current estimate for the focal length as α^t and thus the current camera matrix K^t. We may then obtain the calibrated homography H'_m from the uncalibrated estimate H_m as

$$H'_m = K^{t\,-1} H_m K^t \ . \tag{3.17}$$

The algorithm of Appendix 1 of [Triggs 1998] will decompose this H'_m into a scale parameter ζ, which we may normalise to one since we use the same top-down reference image, and two possible solutions for the structure and plane normal: $R_m^{\{1,2\}}$, $\mathbf{t}_m^{\{1,2\}}$ and $\mathbf{n}_m^{\{1,2\}}$. The normal estimate should be the same for all images $m = 1...M$ with the other randomly

distributed. We may thus determine the correct solution by taking the average normal $\bar{\mathbf{n}}$ as

$$\bar{\mathbf{n}} = \frac{1}{M} \sum_m \mathbf{n}_m^1 + \mathbf{n}_m^2 \tag{3.18}$$

and then selecting the correct normal $\mathbf{n}_m$ as the closest estimate to this average such that

$$\mathbf{n}_m = \begin{cases} \mathbf{n}_m^1 & \text{if } \cos^{-1}\left(\mathbf{n}_m^1 \cdot \bar{\mathbf{n}}\right) < \cos^{-1}\left(\mathbf{n}_m^2 \cdot \bar{\mathbf{n}}\right) \\ \mathbf{n}_m^2 & \text{otherwise} \end{cases} . \tag{3.19}$$

We then take the average of the correctly chosen normal

$$\hat{\mathbf{n}} = \sum_m \mathbf{n}_m \tag{3.20}$$

and set our cost function $f(\alpha^t)$ to minimise the variance of the normals with respect to this average

$$f(\alpha^t) = \frac{1}{M} \sum_m \cos^{-1}\left(\mathbf{n}_m \cdot \hat{\mathbf{n}}\right) \ . \tag{3.21}$$

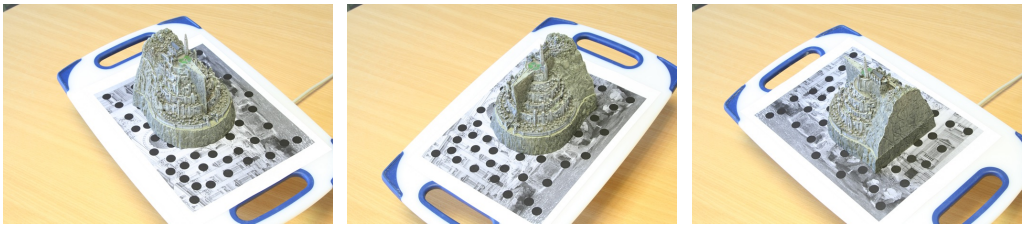
We may then optimise the value of α to minimise $f(\alpha^t)$ and then use this value to form our estimate of K and take the corresponding correct solutions of $R_m^{\{1,2\}}$ and $\mathbf{t}_m^{\{1,2\}}$ as the initial camera poses R_m and $\mathbf{t}_m$. These values provide a suitable initialisation for non-linear optimisation which will optimise all the existing parameters and include the principle point (u_0, v_0) as well as any radial distortion from the camera lens. Figure 3.3(b) shows the results of linear camera calibration on the sequence of Figure 3.3(a) with the general structure having been correctly recovered.

3.5 Non-Linear Optimisation for Calibration

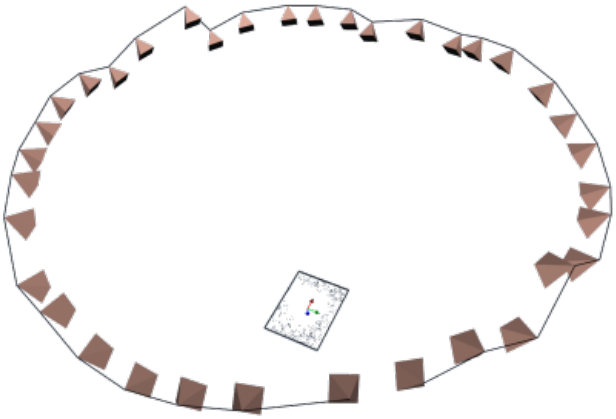
In order to obtain the most precise calibration it is necessary to perform non-linear optimisation of the camera parameters. Under the assumption of zero-mean Gaussian noise, the Maximum Likelihood Estimate (MLE) is obtained by performing bundle adjustment as detailed in § 2.2.1. Figure 3.3(c) shows the results of applying the non-linear optimisation. The bundle adjustment stage improves the RMS reprojection error by a factor of 100 over the results of linear calibration. These results show the system working even when a significant portion of the plane is occluded by an object.

3.6 Experiments

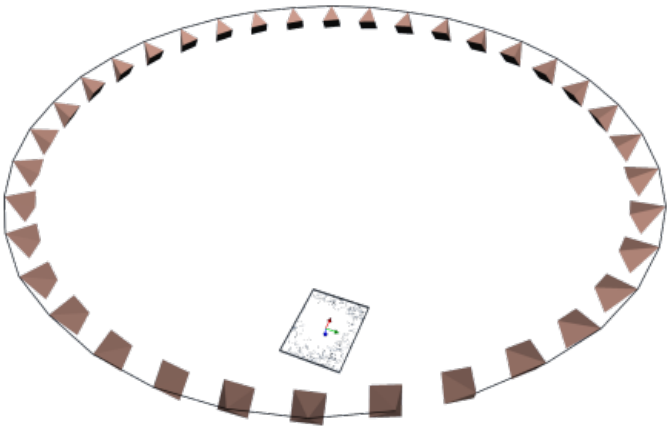
In order to allow for a quantitative comparison with the Bundler system of [Snavely et al. 2006] a series of experiments was performed using synthetic image sequences, such as



(a) Some of the input images



(b) The linear estimate of the camera calibration



(c) The camera calibration after bundle adjustment

Figure 3.3: Estimation of camera calibration for a turntable sequence. *A sequence of 36 images (a) at 8 MP resolution were taken on a turntable to verify the accuracy of the calibration procedure. The calibrated structure after linear estimation (b) has an RMS reprojection error of around 30 pixels. After bundle adjustment (c) the RMS reprojection error has been reduced to 0.3 pixels and the structure is clearly observable as circular motion.*

that of Figure 3.4(a), so that the ground-truth calibration was known. When provided with images obtained with a known and fixed camera intrinsic calibration matrix K we may observe that the planar calibration algorithm outperforms the Bundler system by comparing Figures 3.4(b) and 3.4(c). We note, in particular, that the greatest errors are in the position of the camera centre along the optical axis. This observation may be seen in the numerical results by introducing noise into the focal lengths used to produce the synthetic images. This simulates the varying auto-focus of the camera even when the coarse zoom is left unadjusted. We performed a series of runs varying both the extent of the noise in the focal length and the angle between the plane and the camera viewpoints. The planar algorithm was able to calibrate successfully under all conditions whereas the Bundler system had a failure rate of 30% and 60% at 30° and 45° angles to the plane respectively. Taking the averages over the runs, and neglecting the failed runs under Bundler, the relative error in camera position and rotation is given in Figures 3.5(a) and 3.5(b) respectively. In both cases the planar algorithm is found to outperform the Bundler system by over an order of magnitude.

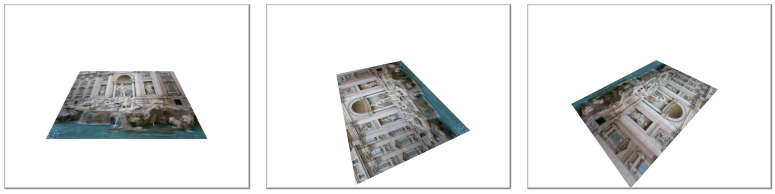
3.7 Discussion

The good performance of the algorithm with planar scenes is due not only to explicitly considering the plane as an integral part of the optimisation, but also due to the use of the top-down image during the matching process. By finding image correspondences between each viewpoint and the top-down image we protect against the effects of drift and effectively match across larger baselines. The problem with matching neighbouring images is that the parts of the scene that each image feature will only be matched to the closest neighbouring images since it becomes harder to match features as the baseline between the images increases since the effects of distortion (for example due to perspective projection) become more dominant. This leads to features being matched across only a few images at a time, not the entire sequence, and leads to 'drift' whereby the calibration results stray from the true calibration as more and more images are added since there is no fundamental reference point. For circular sequences this effect is often observed by the calibration failing to curl in on itself and thus failing to form a closed circle. By using the top-down image we are effectively matching features over a shorter baseline (the angle subtended between the top-down camera position and the image rather than all the way round the scene) and we avoid drift since the same image is used across the entire sequence. We may also note that, since the normal of the plane is also included in the optimisation, it is not necessary for the top-down image to be exactly parallel to the plane and so may still be

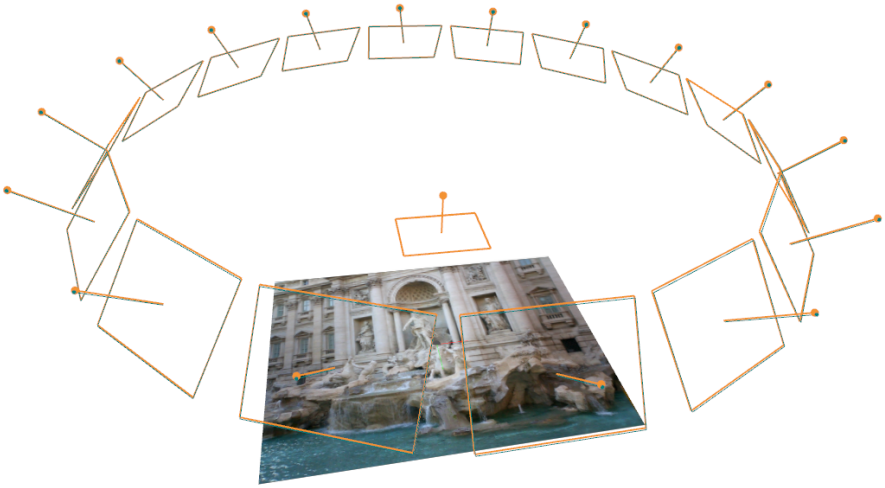
used if the reference plane is harder to obtain, for example the outdoor ground plane of a field.

The performance of the planar algorithm is confirmed to be susceptible to the angle between the plane and the camera viewpoint. This is expected since below a certain angle the SIFT matching process will fail since the features are only invariant to affine transformations whereas the images experience strong projective distortions as the viewpoint angle approaches the plane. If all the images are taken normally then we arrive at degenerate configurations of pure translation or pure rotation around the camera centre. Figure 3.5(b) shows that the smallest rotation errors occur when the angle between the camera and plane is 45°, between these two extremes.

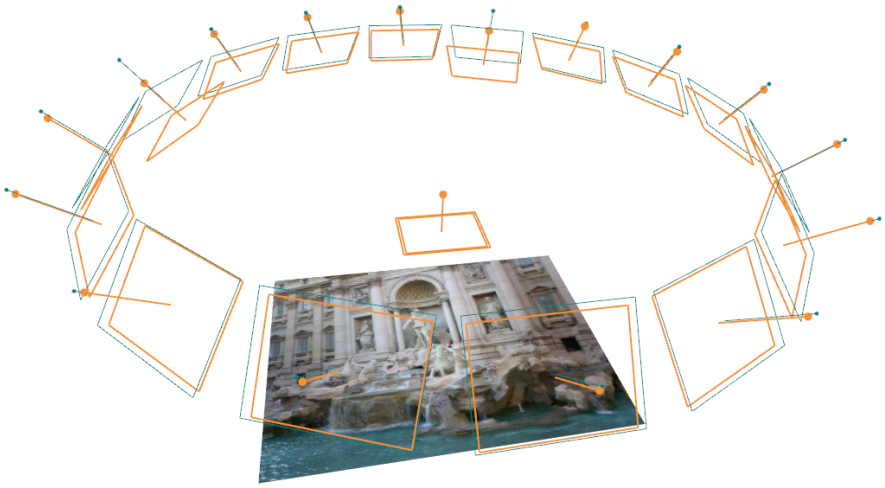
Figure 3.6 shows the calibration result from only 6 images of a chest. Here the Bundler system was unable to recover any calibration despite the fact that there are multiple, non-parallel planes in the scene. The fact that the plane comprising the front of the chest is dominant is sufficient to disrupt the calibration procedure.



(a) Synthetic images of a plane

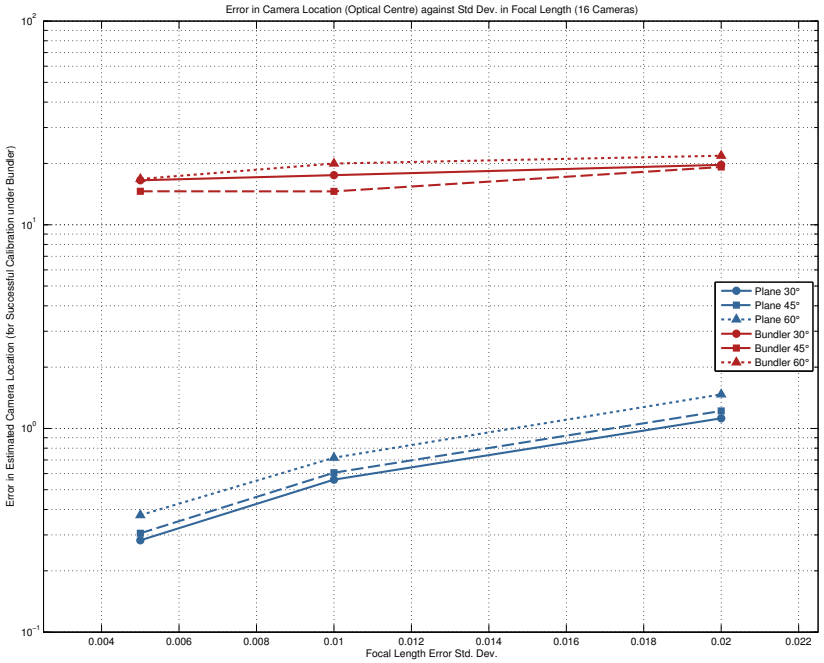


(b) Calibration result for planar algorithm

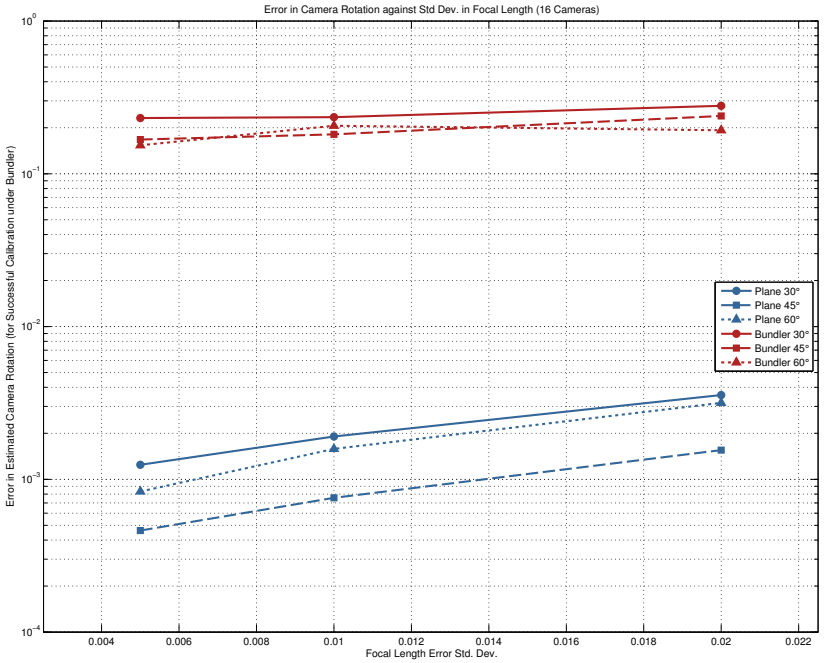


(c) Calibration result for Bundler

Figure 3.4: Comparison of calibration results with synthetic data without noise. *(a) Synthetic images of a plane (3 of 16 shown). (b) The estimated calibration under the planar algorithm in orange compared to the ground-truth calibration in green. (c) The estimated calibration under the Bundler algorithm in orange compared to the ground-truth calibration in green. We observe the superior performance of the planar calibration algorithm, particularly with respect to the camera centres.*



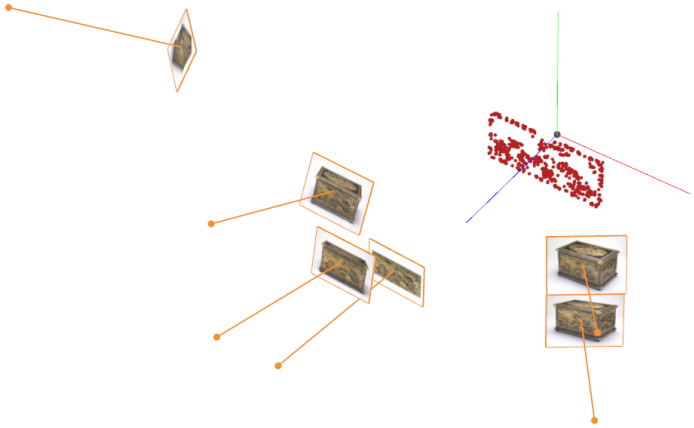
(a) Error in camera position



(b) Error in camera rotation

Figure 3.5: Comparison of calibration performance under noise. *The average relative error (log scale) in (a) the estimated position and (b) the estimated rotation of the camera for increasingly noisy focal lengths. Measurements are taken at three different angles above the plane. In all cases the planar algorithm outperforms Bundler by over an order of magnitude. The planar algorithm produced a calibration estimate in all cases, the averages for Bundler neglect runs for which no calibration was obtained.*

(a) Images of a chest containing a dominant plane



(b) Calibration result

Figure 3.6: The planar calibration algorithm for images of a chest. *(a) Images of a chest (3 of 6 shown) containing a dominant plane. (b) The calibration result obtained, the Bundler system of [Snavely et al. 2006] was unable to calibrate any of the images.*

CHAPTER 4

Random Fields for Segmentation and Reconstruction

As indicated in Figure 1.4, the third and fourth stages of the reconstruction pipeline are to perform a segmentation of the, now calibrated, input images and finally reconstruct the 3D model.

The goal of the segmentation is to extract the object of interest (i.e. the object to be reconstructed) out of the unwanted background of the remainder of the image; we are finding the *silhouette* of the object in each view. There are reconstruction algorithms that do not require object silhouettes. However, we may note that it is impossible to degrade the performance of any algorithm by providing the object silhouettes, since we may make the reasonable assumption that the image outside the silhouettes provides no further information to the object shape[1], and in the vast majority of cases an improvement is offered even if only in the form of reduced computation time. Furthermore, the algorithms that make use of silhouettes will often offer improvements in terms of accuracy and completeness of reconstructions since an objects silhouettes are well known to provide a strong constraint upon its shape, for example [Hernández and Schmitt 2004, Kolev and Cremers 2008, Baumgart 1974, Laurentini 1994, Cipolla and Giblin 1999]. In particular we may observe that the projection of the contour generators (outline of the silhouette [Cipolla and Giblin 1999]) and dense stereo matching provide complementary information [Hernández and Schmitt 2004, Kolev and Cremers 2008].

As suggested in Chapter 1, the recovery of shape from from images is one of the most established areas of research in the field of computer vision and has produced a wealth

[1]Here we refer to information used for the purpose of dense stereo matching. It is conceivable that this may not be the case if we were to produce an algorithm of greater intelligence and reasoning since the rest of the image will help with the inference of parameters such as scene lighting and so forth.

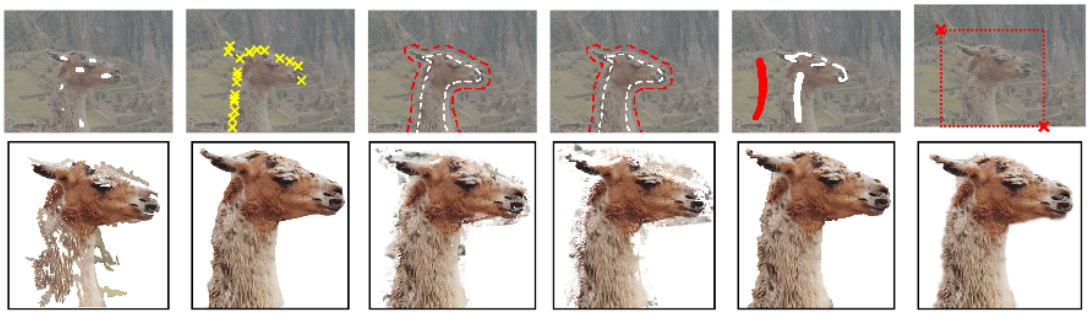


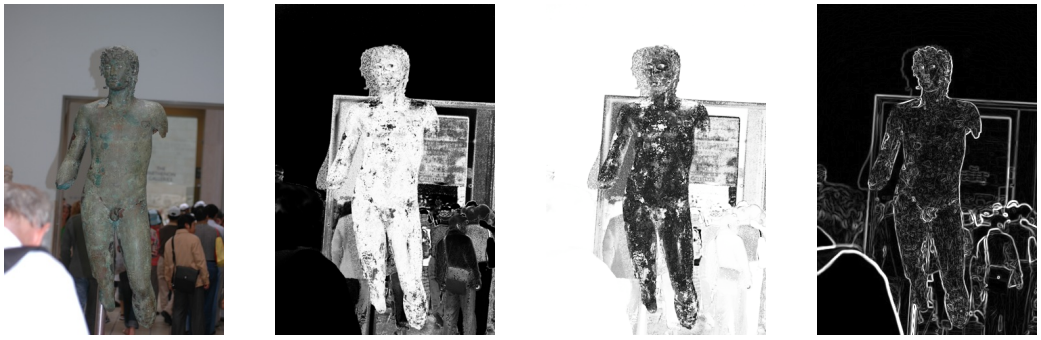
Figure 4.1: Differing user interaction requirements for some of the latest segmentation algorithms. *The top row shows the user interaction required to complete the segmentation or matting process: white brush/lasso (foreground), red brush/lasso (background), yellow crosses (boundary). Used with permission from [Rother et al. 2004].*

of literature. We present an overview to indicate the breath of the topic as well as providing the background of the specific techniques that our contribution is built upon. The interesting result is that the tasks of segmentation and reconstruction have much in common when both are thought of in the context of Random Fields and Markov models, more specifically the Markov Random Field (MRF) and the Conditional Random Field (CRF). Such formulations present an elegant approach and form the basis of our contributions in this area.

4.1 Segmentation

The task of foreground segmentation in computer vision is a challenging one, not least because in a given image the decision of what is foreground and what is background is in some sense an arbitrary one because the aims and requirements of individual users may differ greatly. Since we cannot expect an algorithm to be clairvoyant in this matter, the latest algorithms for foreground image segmentation adopt an interactive approach where a user must specify the object to be extracted. Figure 4.1, taken from [Rother et al. 2004], provides typical examples of the input required from the user. Whilst this interactive approach may be the best solution for foreground segmentation in arbitrary images, we demonstrate, in the algorithm presented in Chapter 5, that if we restrict our input to a calibrated image sequence focused on an object of interest we may perform segmentation of this object automatically across all the images without any user input.

In order to perform foreground/background segmentation we require an indication of what makes up the foreground. Once this is provided, for example by the user interactions demonstrated in Figure 4.1, we must then encode this knowledge in a model which allows us to classify a pixel or region of an image as part of the foreground or background.



(a) Input image **(b)** Foreground likelihood **(c)** Background likelihood **(d)** Edge likelihood

Figure 4.2: Example of foreground and background priors for image segmentation. *Taking a sample image (a), we produce a prior for pixels being part of the foreground (b) or the background (c). We may also derive a measure of the edge likelihood in the image (d). Here a mixture of Gaussians in colour space was used to model the foreground/background priors with the prior probabilities expressed from 0 (black) to 1 (white).*

This model encodes our prior beliefs about which regions are foreground and which are background in the image. The models used to express these priors include histograms of pixel intensities or texture, for example [Boykov and Jolly 2001], or probabilistic models, for example Gaussian mixtures models in colour space [Chuang et al. 2001, Blake et al. 2004].

Figure 4.2 provides an example of the prior model for the segmentation of an image of a statue. Once these models have been learnt we may note that the prior alone is not enough to separate the foreground from the background since the two are not separable in colour space. We must therefore incorporate some further assumptions in order to constrain the segmentation. Firstly, we believe that the silhouette edges should coincide with intensity edges in the original image. This gives rise to an edge probability term, such as Figure 4.2(d), that represents the probability of a transition between foreground and background at a particular location. Secondly, we may assume a spatial prior: we expect the silhouette to be continuous. This is obviously not always the case since we must pass from foreground to background at some point, rather it reflects a preference for solutions which offer low numbers of transitions. In fact we would also like to encourage these transitions to occur at the image edges we have previously identified. In summary, the majority of segmentation algorithms will make use of some or all of these three components:

1. Foreground and background likelihoods,
2. Edge likelihood,
3. Spatial prior.

A variety of algorithms for performing segmentation have been proposed in the literature, by way of illustration we may consider the following: level set methods [Yezzi and Soatto 2003], snakes [Cohen 1991], and normalised cuts [Shi and Malik 1997]. One of the most elegant frameworks for combining the three sources of information is the use of energy based methods operating on Random Fields (RF). To provide a background to these particular methods we proceed by providing an overview of the RF framework followed by a description of how it may be applied to the task of segmentation.

4.2 Shape Reconstruction

The discussion in § 1.4 highlights the challenges faced by any algorithm attempting to invert the imaging process and recover the lost information. Since we cannot perform an exact inversion we are faced with many ambiguities. In order to resolve these ambiguities, researchers have established a number of cues which help to constrain the problem, under a set of assumptions, and thus produce an estimate of the original shape in a tractable manner. These cues vary dependent upon the freedoms found in the parameters that make up the image, Figure 1.9. Neglecting occlusion for the moment, consider the following four parameters: shape, material, viewpoint and lighting. Our target is to produce an estimate of shape, which is initially unknown. We will make the assumption that we are unable to influence the materials in the scene and thus have to alter our algorithms to take this into account. This leaves us with two free parameters: lighting and viewpoint. We may therefore classify shape recovery methods in terms of the material of the object and the freedoms that we may exploit in terms of lighting and viewpoint.

If we start with viewpoint we may divide the field into methods that operate using images from a fixed position and those that exploit multiple views of the same scene. Single viewpoint techniques are unable to exploit the geometry of multiple viewpoints by matching locations and thus require stronger assumptions to account for this lack of data. If we are provided with only a single image then we are forced to use techniques such as shape from shading [Zhang et al. 1999]. Although the technique has the appeal of operating on a single image, the problem is known to be ill-posed and has led to poor results in general. If we have freedom to alter the lighting we may take multiple images from the same viewpoint and use photometric stereo techniques [Woodham 1980]. In general these fixed viewpoint techniques attempt to attain a measure of depth in the scene for each pixel in the image. Such a representation is termed a depth-map or 2.5D reconstruction. A complete 3D reconstruction remains elusive since we do not observe the full surface. Photometric stereo may be used to produce such a reconstruction if we allow the viewpoint to move in

addition to the lighting [Vogiatzis et al. 2006]. This technique is useful for reconstructing texture-less objects which are very challenging for matching based stereo techniques as we shall see later on. In the context of our research goals, § 1.2, these techniques are of limited interest since we would like to reconstruct objects outside of the laboratory where we are unable to control and alter light sources.

Given that we want our system to reconstruct objects in the real world we are left with the assumption of constant lighting but the freedom to choose multiple viewpoints. As we are at the final stage of the reconstruction pipeline, Figure 1.4, we are presented with a set of calibrated images and a set of silhouettes of the object of interest in each image. The discussion of § 4.1, and the associated references, has revealed the relationship between silhouettes and shape. The constraints offered by silhouettes are very robust since the silhouette of an object is invariant to lighting and surface material or reflectance. In addition to the individual constraints, the visual hull, formed by the combination of the silhouettes has many uses as discussed in § 1.3. Perhaps the most important is its definition as an outer bound on the surface to be reconstructed. Whilst the silhouettes of an object are useful, a reconstruction cannot be obtained from them alone since concave regions on an object will not affect the objects silhouette and consequently the generation of the visual hull and thus may not be recovered.

If we consider the images, rather than just the silhouettes, we have a much richer source of data for the inference of 3D shape. If we are prepared to make the assumption that the objects of interest have a textured surface we may attempt to find points in different images which correspond to the same point on the surface of the object. Given the calibration of the cameras, which we assume have different optical centres, we may then triangulate the position of the point on the surface. The task of general feature matching is studied in its own right. The review of [Mikolajczyk et al. 2005] details the current techniques including the SIFT matching process [Lowe 2004] used in Chapter 3 as part of the automatic calibration algorithm. These techniques are well suited to matching a sparse set of feature points as required by the calibration algorithms discussed in Chapters 2 and 3. If we wish to obtain a detailed reconstruction of the object we would like to obtain a dense set of correspondences. These techniques are often referred to as dense stereo algorithms and have been studied extensively, for example the review of [Scharstein and Szeliski 2002b].

As for photometric stereo, dense stereo algorithms produce a depth-map or 2.5D reconstruction. Our goal is to build complete 3D models and thus dense stereo algorithms are unsuitable if used in isolation. The principle of finding correspondences is based on the idea that the same portion of a surface should adopt a similar appearance when it is

visible in different images. This is referred to as photo-consistency. Thus if we have many views available to us, the task of reconstruction, using this cue, is to estimate a surface that is most photo-consistent with all of the images in the sequence. We must also keep in mind cases where our photo-consistency assumptions may be violated, for example reflections and specularities in the scene. This is the challenge faced by Multi-View Stereo (MVS) algorithms.

4.3 Multi-View Stereo

Multi-view stereo reconstruction has become a growing area of interest in recent years with many differing techniques achieving a high degree of accuracy [Seitz et al. 2006]. The techniques that have been applied differ across a range of design choices in terms of the scene representation used, the metric for establishing photo-consistency, the manner with which they deal with occlusion and the optimisation algorithms used to enforce regularisation constraints. The last factor is a consequence of the need to resolve the ambiguities introduced by the imaging process, discussed in § 1.4, by making assumptions about the smoothness of the observed surface, termed shape regularisation [Poggio et al. 1985]. This is particularly true for objects containing regions devoid of texture since we will be unable to ascertain any information from the photo-consistency metric. Over small regions we then fall back on the shape regularisation, however, if the texture-less regions are large or encompass the entire object, correspondence based stereo will fail and we must recover the shape using one of the other techniques mentioned in the previous section. For the remainder of this chapter and our contributions we assume we are dealing with textured scenes although we return to the discussion in Chapter 8.

4.3.1 Scene Representation

In order to extract a surface the algorithm must maintain a representation of the surface in 3D space. This representation will often have an impact on the type of shape regularisation and optimisation techniques available to us since we are trying to produce a tractable problem. We are also faced with objects of arbitrary topology observed from a range of viewpoints and thus we must be able to handle visibility constraints within the representation.

Depth-Maps

The first common representation is to maintain a set of depth-maps for each of the views. As discussed in the previous section, there are many binocular dense stereo algorithms,

[Scharstein and Szeliski 2002b], that may be applied, in a pairwise fashion, to the input images to build a series of depth-maps. Many of the most successful make use of Markov Random Field models for regularisation, a topic addressed by the last sections of this chapter, that allow for effective optimisation algorithms to be employed. Whilst this representation allows for very general topology and efficient optimisation the resulting surfaces are viewpoint dependent and it is very difficult to perform global visibility reasoning rather than treating each viewpoint independently. The work of [Kolmogorov and Zabih 2002] attempted to address the challenge of resolving the global interactions between the many views of the scene but requires a very costly optimisation algorithm, limited to a small number of depth layers, for even a fairly modest number of views.

Region Growing

The next set of algorithms consist of region growing approaches [Furukawa and Pons 2007, Habbecke and Kobbelt 2007, Goesele et al. 2007]. These methods are based on a plane-based photo-consistency measure. Starting from sparse correspondences they fit local patches and then try and grow these patches into surrounding regions by following the photo-consistency. They adopt an iterative approach to the occlusion problem, alternating between camera visibility and shape estimation. In addition they iterate the region growing with filtering stages to attempt to remove any incorrect regions. These methods provide some of the most accurate results due to the plane-based photo-consistency measure, [Scharstein and Szeliski 2002a], and the ability allow for evolving visibility or occlusion handling. However, this is traded off against a long computation time and the need to 'tune' a large number of parameters to get the best results.

Meshes

The use of polygonal meshes has passed from the graphics community to form the basis of several reconstruction algorithms based on stochastic optimisation techniques, for example [Vogiatzis et al. 2003], and continuous mesh evolution, for example [Hernández and Schmitt 2004, Fua and Leclerc 1995], achieving a high degree of accuracy [Scharstein and Szeliski 2002a]. Meshes allow a great degree of freedom in terms of shape regularisation but this comes at the expense of difficulties with unknown topologies, for example the method of [Hernández and Schmitt 2004] relies on the silhouettes of the object to contain the correct topology information since the mesh cannot be 'cut' during the optimisation.

Volumetric

The final type of scene representations aim to relax the dependence on viewpoint by adopting volumetric approaches. The first class consists of level-set based approaches [Faugeras and Keriven 1998, Jin et al. 2005, Pons et al. 2005, Kolev and Cremers 2008]. The surface is evolved using variational methods under a precomputed photo-consistency volume with the surface maintained as a level-set. These processes are numerically complex and often slow to converge, although this is changing due to the use of graphics cards for performing parallel computations [Kolev and Cremers 2008]. The methods have yet to produce very accurate results on the standard data sets [Scharstein and Szeliski 2002a].

The second class consist of 'voxel' representations comprising of a discrete subdivision of a volume, made up of a set of voxels, that may be labelled as inside or outside the object. The first approaches include voxel colouring or carving [Dyer and Seitz 1997, Kutulakos and Seitz 2000, Broadhurst et al. 2001]. These algorithms attempt to determine a set of photo-consistent voxels (that must lie on the object surface) and use the subsequent visibility constraints to remove voxels outside the surface. The resulting surfaces tend to be noisy since there is no straight forward method for performing shape regularisation.

More recent approaches have exploited the fact that the voxel representation is essentially a binary occupancy function to pose the reconstruction task as a segmentation problem within a 3D Markov Random Field model [Vogiatzis et al. 2005; 2007, Hornung and Kobbelt 2006a]. This approach allows for surface regularisation and arbitrary topologies as well as access to efficient optimisation techniques used within the segmentation community. This comes at the cost of making it difficult to build iterative occlusion reasoning, hence the use of occlusion robust photo-consistency metrics [Vogiatzis et al. 2007]. This framework also allows for strong shape priors to be used if the type of object being reconstructed is known [Sun et al. 2006]. The voxel segmentation approach lends itself to a two stage approach to object reconstruction that is studied further in the following section.

4.3.2 Two Stage Reconstruction

A successful strategy is to split the reconstruction process into two stages. The first is to estimate a series of depth-maps using local groups of the input images. The second stage then attempts to combine these into a global surface estimate, making use of registration and regularisation techniques. This two stage approach is an elegant formulation that allows different techniques to be chosen independently for the two stages. Most of these algorithms use some form of global optimisation strategy on a volumetric representation to extract a surface [Hernández and Schmitt 2004, Vogiatzis et al. 2005; 2007, Hornung

and Kobbelt 2006a, Goesele et al. 2006]. Some recent methods achieve a fast computation time by avoiding a global optimisation when merging depth-maps [Merrell et al. 2007, Bradley et al. 2008] with the work of [Strecha et al. 2006] between the two. The global solutions either pre-compute the visibility using a proxy for the shape, for example the visual hull [Vogiatzis et al. 2005], or use a robust photo-consistency measure, for example [Vogiatzis et al. 2007].

The estimation of local depth-maps is often performed using patch based methods [Scharstein and Szeliski 2002b]. The work of [Hernández and Schmitt 2004] makes use of the Normalised Cross-Correlation (NCC) metric as a matching cost between two patches. This method offers good performance for textured objects and has been the basis of several successful techniques [Goesele et al. 2006, Vogiatzis et al. 2007, Hornung and Kobbelt 2006b]. In the first stage of [Hernández and Schmitt 2004] a depth is estimated for each pixel independently. In the next stage the algorithm looks for consensus in depth estimates from multiple depth-maps. Since the individual depth-maps are known to contain outliers, this stage relies upon redundancy in the depth-maps to reject the them. In data-sets containing a large number of images (50-100) this approach performs quite well. In so called sparse data-sets (10-20 images) one expects very little redundancy in the reconstructed depth-maps, leading to a drop in reconstruction accuracy. This drop is actually observed in the performance of [Hernández and Schmitt 2004] in sparse data-sets with known ground truth data [Seitz et al. 2006].

4.4 Random Fields and The Markov Assumption

The Markov property is a simplifying assumption that is often used to simplify an inference problem for the sake of tractability and is used extensively in computer vision. Here we provide a brief introduction to the topic, particularly focusing on the Markov Random Field (MRF) which is the generalisation of the Markov property to 2D and 3D networks often used in vision problems. A much more in-depth treatment may be found, among others, in [Kindermann and Snell 1980].

If we consider a deterministic stochastic process consisting of random variables $\{Y_i\}$ then in general the probability of a given random variable Y_t taking a state y_t is dependent on all the previous states taken up to that point, that is we have

$$P\left(Y_t = y_t \,|\, \{Y_j\}, j = 0, ..., (t-1)\right) \quad . \tag{4.1}$$

We can see that if this is truly the case, as time continues the computational cost of performing inference on probabilities of future states can become very expensive. Intuitively we may assume that for a variety of processes, the states taken a long time ago must have

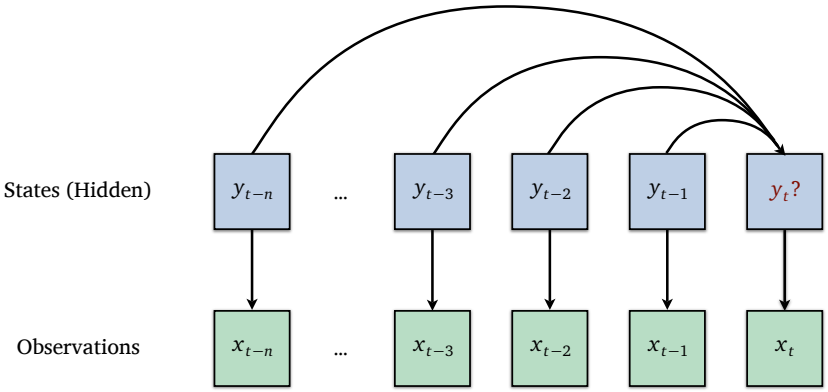
far less effect on the probabilities of the states the process is about to take than recent states. We may formalise this intuition by assuming the process is a Markov process. If we take things to the extreme we may state that we assume that the probability for the next state depends only on the previous state: this is the assumption of a first order Markov process. This allows us to write

$$P\left(Y_t = y_t \,|\, \{Y_j\}, j = 0, ..., (t-1)\right) = P\left(Y_t = y_t \,|\, Y_{t-1}, ..., Y_{t-n}\right) \tag{4.2}$$

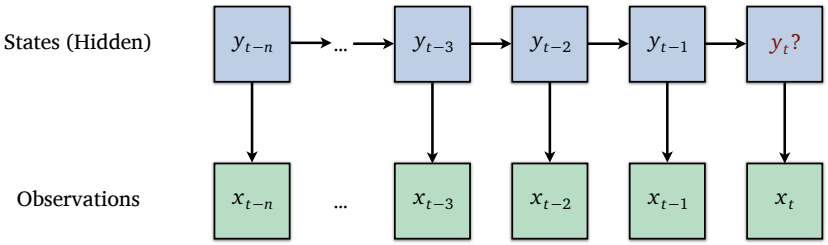
for an n^{th} order Markov process.

Figure 4.3 provides an illustration of this property. In Figure 4.3(a) we have a hidden process where we do not observe the states $\{y_i\}$ directly but rather a set of observations $\{x_i\}$ which we assume depend only on the corresponding hidden state. Figure 4.3(b) shows the same process with the first order Markov assumption. This is known as a Hidden Markov Model (HMM) and is often used in speech recognition, for example [Woodland and Young 1993]. The Markov model naturally extends to images where, rather than a series of temporal states, we have a random quantity defined over a grid of pixels. The probability of a given pixel will, in general, dependent on every other pixel however under a first order Markov assumption we end up with the 2D MRF of Figure 4.3(c) which greatly simplifies inference.

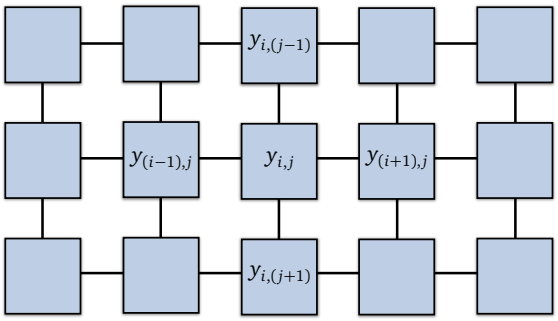
The term Markov Random Field is usually used to indicate a hidden Markov model where the hidden states obey the Markovian assumption and each observation is derived independently from only its corresponding state (the generalised 2D case of Figure 4.3(c)). For example, in the task of binary image segmentation we might assume that each observed pixel depends only upon whether its hidden state is foreground or background. The classification of Conditional Random Field is a further generalisation where each observation may be derived from any number of the hidden states or other observed data. In the case of the image segmentation example, we could argue that if we use image patches, texture information (applying filters to the image) or even learning colour models across the image then the observed pixels are no longer solely dependent on their corresponding hidden state and should be considered a CRF. Performing inference, or energy minimisation, using these CRF models should fully model all of these dependencies, however this often leads to intractable problems and removes the advantages of the Markovian assumption. To alleviate this problem it is common to derive the parameters of the model using the full dependencies but then assume that the observations are independent during the optimisation such that the advantages of the Markovian assumption are restored.



(a) Stochastic process with hidden states



(b) Process with Markov assumption



(c) Basic 2D Markov Random Field

Figure 4.3: Illustration of the Markov assumption. *If we take a general 1D stochastic process with hidden states* $\{y_i\}$ *and corresponding observations* $\{x_i\}$ *(a) the state transition probability for state* y_t *may depend on any previous state* $\{y_j\}$ *for* $j = 0, ..., (t-1)$*. Under the (first order) Markov assumption the state transition probability will only depend on the previous state (b). This framework may be extended to an undirected graph in N dimensions to give a Markov Random Field, for example a 2D MRF (c).*

4.5 Energy Minimisation for Random Fields

The RF framework is particularly useful for solving energy based models, in particular those with smoothness based priors. Many of the latest techniques are discussed in the recent review of [Szeliski et al. 2008]. It is common practice in computer vision problems to define an energy model over a grid of pixels and then calculate the desired solution as processes of minimising this energy. If we take our grid of pixels as an MRF we greatly simplify the energy model. We may derive this model as follows (for ease of comparison we will use the same notation as [Szeliski et al. 2008]). Firstly we assume we have a set of pixels p defined over an image $\mathcal{I}$ such that $p \in \mathcal{I}$. For each pixel p we wish to assign a label l_p from a discrete set of possible labels $\mathcal{L}$ such that $l_p \in \mathcal{L}$. Our energy term may then be composed as

$$E\left(\{l_p\}\right) = E_{\mathrm{d}}\left(\{l_p\}\right) + \lambda E_{\mathrm{s}}\left(\{l_p\}\right) \tag{4.3}$$

where $\{l_p\}$ is the set containing each pixel's allocated label and $E_{\mathrm{d}}(\cdot)$ and $E_{\mathrm{s}}(\cdot)$ are the data energy term and the smoothness energy term respectively. This data cost is the sum of data costs allocated on a per pixel basis $d_p(l_p)$ such that

$$E_{\mathrm{d}}\left(\{l_p\}\right) = \sum_p d_p(l_p) \ . \tag{4.4}$$

This is also termed the unary cost, since it is defined for a single pixel and label, and is equivalent to defining a cost to each label to each of the blue nodes (pixels) in the MRF of Figure 4.3(c). Following the grid notation of Figure 4.3(c), we may express each pixel using its position such that $p = (i, j)$. This allows us to more easily study the second energy term, the smoothness energy, which is defined for neighbouring pixels as

$$E_{\mathrm{s}}\left(\{l_p\}\right) = \sum_{\{p,q\}\in\mathcal{N}} V_{pq}(l_p, l_q) \tag{4.5}$$

where $\mathcal{N}$ is the set of all neighbouring pixels. This is also known as the pairwise term since it defined for pairs of neighbouring pixels and their associated labels and is thus equivalent to defining a cost function $V_{pq}(l_p, l_q)$ for each of the connecting edges in the MRF of Figure 4.3(c). Thus the cost function of (4.3) maps directly onto the MRF of Figure 4.3(c) and is thus the general form of any energy function defined on such an MRF. Our goal is to find an optimal labelling $\{\hat{l}_p\}$ which minimises the energy, that is

$$E\left(\{\hat{l}_p\}\right) = \arg\min_{\{l_p\}} E_{\mathrm{d}}\left(\{l_p\}\right) + \lambda E_{\mathrm{s}}\left(\{l_p\}\right) \ . \tag{4.6}$$

This means that a particular problem, under this common framework, is classified by the cardinality of the label set $|\mathcal{L}|$ and the form of the unary and pairwise functions $d_p(l_p)$ and

$V_{pq}(l_p, l_q)$. For example if we consider a binary label set $\mathcal{L} = \{0, 1\}$ and the pairwise term

$$V_{pq}(l_p, l_q) = \begin{cases} 0 & \text{if } l_p = l_q \\ 1 & \text{if } l_p \neq l_q \end{cases} \tag{4.7}$$

we get the Ising model, arguably the simplest energy function. This cost function may be solved exactly, i.e. the global minimum found, in polynomial time using the graph-cuts algorithm [Hammer 1965, Greig et al. 1989]. These leads us on to algorithms used to solve these models and a discussion of tractability.

Let us now expand our energy model from the binary set to a discrete set of labels $\mathcal{L}$ and allow less restrictive pairwise functions. Given that we have returned to our general formulation for the energy models, and that this has already been simplified to a Markov model from a fully connected graph, we note that obtaining the exact solution to an arbitrary instance of the problem is NP-hard and thus intractable, even for small images. Thankfully there are special cases where the exact solution is obtainable with an appropriate algorithm, for example graph-cuts with the Ising model, or we have an algorithm that is unable to provide a global optimum but may provide a strong local minima of the energy which corresponds to a reasonable solution. In the following discussion we provide an review of some of the top performing optimisation algorithms and when they may be applied. Of particular note are the conclusions of [Szeliski et al. 2008] surrounding the algorithms of choice. The discussion recommends both the graph-cuts algorithm for binary problems and the associated alpha-expansion algorithm of [Boykov et al. 2001] for multi-label problems as well as the Tree-Reweighted (TRW-S) message passing algorithm [Kolmogorov 2006]. These two algorithms are found to perform consistently well across a variety of vision problems and we make use of them in our segmentation and depth-map filtering algorithms.

4.5.1 Dynamic Programming

If we restrict our MRF model to a tree, thus ensuring that there are no loops in the graph, we may obtain an exact solution using a Dynamic Programming (DP) technique, for example the Viterbi algorithm.This solution will be exact for multi-label problems as well as binary ones. Whilst we might imagine the removal of loops in the graph to be very restrictive, it doesn't prevent the technique being useful for real problems, especially those that require fast solution times. Dynamic programming techniques have been used to solve binocular stereo matching problems where each scan line is treated independently [Ohta and Kanade 1985, Birchfield and Tomasi 1998]. The independence between the scan-lines does generate artifacts in the reconstructed depth-map, thus we lose the smoothness that

would be introduced by the fully connected MRF, but this can be mitigated, to a certain extent, to produce a real-time algorithm that also makes allowance for an occlusion model [Criminisi et al. 2007]. If we wish to use a full objective function then we are forced to look elsewhere and, if we wish to use a discrete label set, forego a globally optimal solution for a local minimum.

4.5.2 Belief Propagation

The Belief Propagation (BP) algorithm has two main variants, if we wish to identify a MAP estimate, or the most likely solution, we can use the max-product algorithm that we consider here. The sum-product algorithm, rather than identifying the minimum energy solution, produces a marginal distribution over all the label states. Belief propagation algorithms usually run on belief networks, a probabilistic graphical model that we map an MRF problem onto, and, as for DP, result in an exact solution when we have no cycles [Pearl 1988]. The message passing algorithm, however, is not restricted to networks without cycles and may be applied to cyclic networks [Frey and MacKay 1997]. In this loopy BP we no longer have the same guarantee of convergence to a global optimum, or indeed convergence, however it has been shown that the fixed points, if obtained, correspond to the stationary points of an approximation to the free energy of the MRF [Yedidia et al. 2005]. Loopy BP has been very successful in solving RF problems and work has been done to improve the computation time for solving vision problems [Felzenszwalb and Huttenlocher 2004].

4.5.3 Graph-Cuts

We have already discussed the use of the Graph-Cuts algorithm to obtain a globally optimal solution to the binary label Ising problem. The model can be formulated as a flow problem on a weighted graph and a solution obtained in polynomial time using the max-flow/min-cut algorithm. The graph of the MRF is complemented with two nodes, identified as the source and the sink. The set of weighted edges now includes edges from the original nodes to the source and sink. The algorithm partitions this new graph into two disjoint sets, one containing the source and one the sink. This corresponds to a binary labelling. In general, an optimal solution may be obtained to any problem that obeys the constraint of submodularity [Kolmogorov and Zabih 2004].

Work has been done to extend the use of this algorithm to multi-label cases, principally in the form of the α expansion or α-β swap algorithms [Boykov et al. 2001]. This is an iterative algorithm that performs a binary move at each sub-iteration that either swaps a pair of labels (α and β) or moves a set of labels to a new label (expansion move) such that

the energy is reduced. These algorithms are unable to produce a globally optimal solution but instead compute a strong local minimum to the problem, within a single α expansion or α-β swap 'move' of the true solution.

With each of the sub-iterations being performed using the Graph-Cuts algorithm, the swap or expansion moves must obey the submodularity constraint (requiring the pairwise interactions to be semi-metric or metric). Recent work has also produced the Quadratic Pseudo-Boolean Optimisation (QBPO) algorithm which extends the graph-cuts algorithm to non-submodular cost functions although it is not guaranteed to provide a complete labelling [Kolmogorov and Rother 2007].

4.5.4 Tree-Reweighted Message Passing

The Tree-Reweighted (TRW) message passing algorithm [Wainwright et al. 2005] is related to the loopy BP algorithm in its max-product form. The MRF graph is decomposed into a set of trees, under the guarantee that each edge is present in at least one of the trees, and a probability distribution is placed over the set. The update rule of loopy BP for each edge is then weighted by the probability that it is contained in a tree selected at random under the probability over the set of trees. At each iteration the algorithm maintains a lower bound on the energy that may be used to assess the quality of the solution, however, the original algorithm does not guarantee to increase this bound or converge. An improved algorithm is the sequential version (TRW-S) of [Kolmogorov 2006] that guarantees not to decrease the lower bound at each iteration.

4.6 Segmentation Using a Random Field Framework

Returning to the task of segmentation we can see that the energy model of (4.3) is well suited to encoding the three components of § 4.1. We may pose the problem as a binary label MRF or CRF with the labels as foreground and background. This allows the unary (or data) terms of (4.4) to encode the foreground/background likelihood and the pairwise (or smoothness) terms of (4.5) to encode both the spatial prior and the edge likelihood. This pairwise term may fulfil this role by penalising neighbouring pixels which swap from foreground to background unless they lie either side of an edge in the image. This elegant framework was introduced by [Boykov and Jolly 2001] combined with a method for obtaining the foreground/background likelihood. The authors demonstrated how a simple histogram learning process could determine this likelihood for each pixel and thus be used to allow a user to segment interactively a greyscale image. The paper also details how the method may be extended to N-Dimensional graphs.

This energy model approach to segmentation has proved to be very successful and confers state-of-the-art performance. The work of [Rother et al. 2004, Blake et al. 2004] showed that the framework could be extended to use more advanced prior models (a Gaussian mixture mode in colour-space in this instance) to offer an improved interactive segmentation tool. Further work on the optimisation tools has improved computation times so that interactive segmentation may be performed in real time on high resolution images by increasing efficiency [Lempitsky et al. 2007] and through the use of geodesics [Criminisi et al. 2008]. Work has also been performed on improving the performance or successive graph-cut optimisations [Kohli and Torr 2005] which are often performed by iterative algorithms.

4.7 Reconstruction Using a Random Field Framework

We now look at our second task, that of reconstruction, and note that it may be posed as a segmentation task where we are segmenting a 3D volume into 'inside an object' or 'empty space' rather than segmenting a 2D region into 'object' and 'background'. Again, we may construct a binary label problem, this time with the MRF or CRF defined on a set of voxels (discrete elements of volume). An example of such a graph may be found in Chapter 5, Figure 5.6. Using this framework for reconstruction leads to an elegant formulation based on a volumetric scene representation and has been demonstrated to be effective as a MVS method [Vogiatzis et al. 2005; 2007, Hornung and Kobbelt 2006a].

The use of a volumetric representation has many advantages. The resulting surface is defined by the volume, and therefore the representation is independent of the positions of the cameras which is an important property for the general case of MVS where there may be large numbers of images taken from arbitrary viewpoints. The volumetric definition also allows for the general case of arbitrary topology; any physically plausible shape may be easily represented by occupancy of a discrete volumetric grid and it is straight forward for optimisation schemes to move through different topologies in search of an optimal surface. Additionally, it is possible to use the Markov approximation to place continuity constraints on the volume which equate to smoothness constraints on the surface, often a necessary prior for many MVS systems, although with a reduction in flexibility when compared with mesh representations.

The main limitation of the volumetric method is the ability to adapt to surface visibility and occlusion. Whilst iterative methods that maintain direct estimates of the surface may update the surface visibility of each camera as the optimisation progresses this is very difficult to perform tractably when using Random Field solvers of the types discussed in

§ 4.5. A solution to this has been to use a two stage reconstruction approach (§ 4.3.2) that begins with the estimation of depth-maps using occlusion robust metrics followed by a global volumetric optimisation, to label the volume and hence recover the surface, that treats any occlusion errors as an additional source of noise [Vogiatzis et al. 2007].

In order to deal with the occlusion errors we require a proficient optimisation scheme, which are available for RF frameworks, and a method of increasing the signal-to-noise ratio. In the case of MVS, we can increase the signal-to-noise ratio by exploiting the redundancy in the image sequence: the same surface is observed from many different images. This returns us to the argument at the end of § 4.3.2 whereby sequences with a large number of images have sufficient redundancy to overcome the occlusion errors and thus benefit from the advantages of volumetric approaches to achieve good performance. However, when fewer images are available the additional noise introduced by the occlusion errors is more pronounced, reducing accuracy. The work presented in Chapter 7 addresses this problem by reducing the errors found in the depth-maps (in the first stage of the reconstruction process) so that fewer images (less redundancy) are required to achieve good reconstruction results.

CHAPTER 5

Automatic Object Segmentation

5.1 Introduction

The most recent advances in image segmentation adopt an interactive approach [Rother et al. 2004, Blake et al. 2004, Boykov and Jolly 2001] where the user is required to guide the segmentation by manually segmenting image regions. The information supplied by the user is used to learn the object's colour distribution and the globally optimal segmentation is achieved by the maximum flow algorithm [Kolmogorov and Zabih 2004]. This approach performs well on simple images, but the extent of user interaction required increases significantly if the object contains multiple distinct colours and if it is found in a cluttered or camouflaged environment.

If we now consider the case of a large number of images, required for high accuracy reconstructions, we can see that even a modest amount of user interaction on an individual image basis will represent a significant task using these interactive tools. In order to reduce the demands placed on the user we aim to exploit the constraints that exist within the image sequence. Note that the sequence contains a series of multiple views of the same rigid 3D object, therefore the segmentations must satisfy a *silhouette coherency* constraint [Hernández et al. 2007a]. This constraint follows from the knowledge that the images are formed from projections of the same rigid 3D object with the correct segmentations being the corresponding silhouettes. This suggests that by attempting to perform segmentation on isolated 2D images we are rejecting a great deal of information contained within the sequence. We further observe that, since the intended purpose of the image sequence is to reconstruct the object, we have a *fixation* constraint whereby the object of interest is always being focused upon by the camera and thus likely to be central to the viewpoint.

In order to exploit the coherency constraint we perform the segmentation across all images simultaneously. This is achieved by performing a binary segmentation of 3D space

where each 3D location (voxel) is labelled as 'object' or 'background'. The 3D segmentation is influenced by: (a) a colour model initially learnt from the fixation points and then refined in subsequent iterations and (b) a 3D shape prior based on minimal surface area.

The advantages of our approach over independent 2D interactive segmentations are twofold. Firstly, we replace the user supplied image labelling with the fixation constraint, which is usually satisfied by multi-view stereo sequences. Additionally we employ constraints that link multiple segmentations of the same rigid object to improve segmentation quality.

5.2 Prior Work

Object silhouettes have been used in a multi-view environment to provide calibration whether via epipolar tangency constraints [Giblin et al. 1994, Cipolla et al. 1995, Mendonça et al. 2001, Wong et al. 1999]or though agreement of the projected silhouettes or visual hull [Hernández et al. 2007a]. Our algorithm aims to exploit the second constraint (consistency of silhouettes) in reverse to take a calibrated image sequence and infer the silhouettes.

The work of Boykov [Boykov and Jolly 2001] introduced a framework for adopting an energy minimisation approach to segmentation where the resulting optimisation process is both tractable and unique thanks to the max-flow/min-cut algorithm. This offers a very elegant framework for segmentation since it is very simple to construct a graph and neighbourhood cliques from the uniform structure of a 2D image. This method also lends itself to a variety of related problems that may be posed as a Markov Random Field (MRF) formulation [Kolmogorov and Zabih 2004]. The energy function used for segmentation is composed of two terms, a boundary term which favours cuts along discontinuities in image intensities and a probabilistic term which encodes the predisposition of a pixel to be part of the foreground or the background. They demonstrated how a simple histogram learning process could be used to allow a user to segment interactively a greyscale image. The histograms were used to determine a likelihood for the pixel to be part of the object required by the user which was then encoded as the second term in the energy function. Although the paper details how the method may be extended to N-Dimensional graphs, the predominant focus is on the segmentation of 2D images.

The approach of adopting a user guided learning process was formalised in [Rother et al. 2004] where an interactive system for the segmentation of 2D colour images was proposed. As with our method, they adopt an iterative approach to the segmentation task by learning colour models of the object and the background. Figure 5.1 provides a typi-

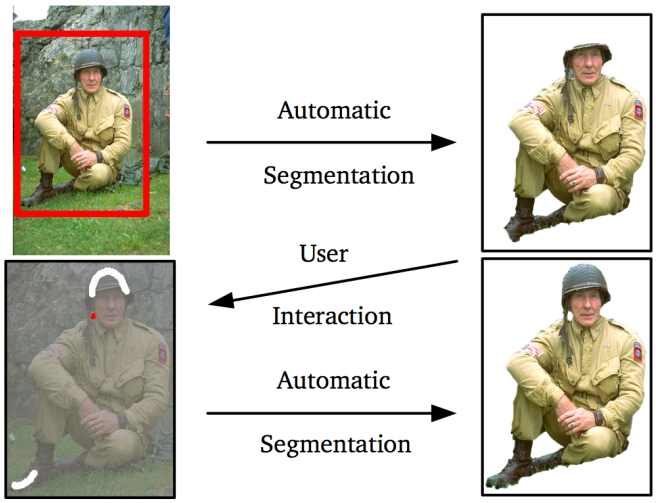


Figure 5.1: An example of the user interactive segmentation approach of [Rother et al. 2004]. *The user begins by drawing a box around the object of interest which produces an initial segmentation. The user then proceeds to make interactive corrections with brush strokes specifying additional areas of foreground (white) and background (red) until the desired result is achieved. Used with permission from [Rother et al. 2004].*

cal example of the segmentation process from a user's perspective. Although considered state-of-the-art for 2D images, the interactive demands placed on the user make the segmentation of a large sequence of images a sizeable task which we would like to avoid by adopting an automatic approach.

An extension of this idea is presented in [Wang et al. 2005], where a deforming object is segmented in a video sequence by a graph-cut on a 3D 'space-time' volume. Although related, the approach of [Wang et al. 2005] is not directly applicable to our problem since our image sequence is not obtained from video and as a result is not dense enough to form a continuous space-time volume over which we could hope to segment the object in the same manner.

In [Snow et al. 2000], a graph-cut based approach is used to estimate the voxel occupancy of a calibrated volume in space. Their approach is directly aimed at using an energy minimisation framework to regularise the process of combining a series of imperfect silhouettes. The main difference to our work is that they obtain these silhouettes as the result of a background subtraction process from a fixed, calibrated camera rig whereas we adopt an iterative learning approach requiring no prior knowledge of the object or environment.

The task of segmenting objects in multi-views has also been studied in [Yezzi and Soatto 2003]. Whilst also approaching the task in the 3D domain, here the authors use a level set method to evaluate the segmentation based on Lambertian scenes with smooth of constant albedo. Level set methods are known to be susceptible to local minima so

[Yezzi and Soatto 2003] relies on smooth albedo variation and a multi-resolution scheme to achieve convergence. In contrast, our method tolerates albedo discontinuities and, due to the graph-cut optimisation scheme, is guaranteed to converge to the globally optimal segmentation.

The concurrent work of [W. Lee and Boyer 2007] also addresses multiple view segmentation. Similarly to our algorithm they make use of the rigid object constraints to propagate information across views. Whilst they use graph-cuts to perform the segmentation, the images are segmented individually (a 2D graph-cut) rather than the volumetric graph-cut we use to segment over all images at once. This means that silhouette coherence is not guaranteed at each iteration of the algorithm although it is the target for the convergence of the algorithm.

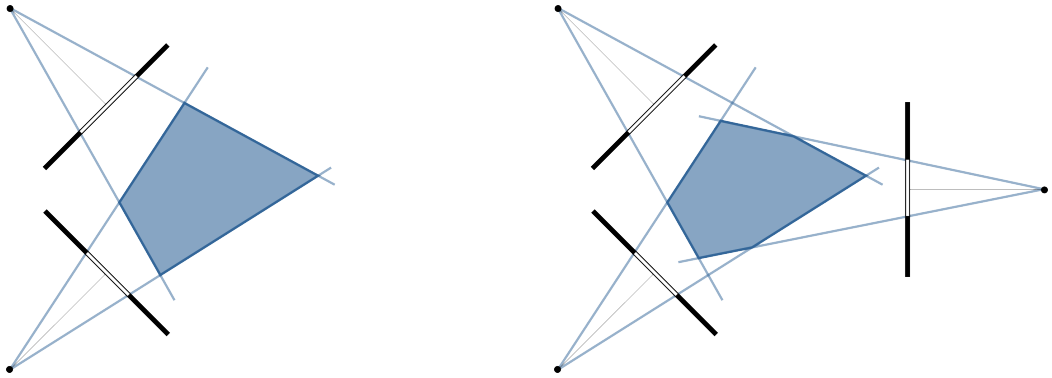
Camera fixation has been used to constrain vision problems, for example to aid recovery of camera motion and shape [Taalebinezhaad 1992], however we have not seen it used previously to assist segmentation across multiple views.

5.3 Multiple View Segmentation Constraints

We have shown that there is a demand for automatic segmentation of an object in multiple views in order to reduce the burden on the user. In order to achieve this we exploit two constraints on the image sequence. Since we are taking pictures of a rigid object we know that the silhouettes in each view must be coherent since they are generated by the same object, we term this constraint *silhouette coherency*. We also note that we are taking the photos of a specific object with reconstruction in mind, therefore we also exploit a *fixation condition* since the cameras will always be focusing on the object of interest.

5.3.1 Silhouette Coherency

Silhouette coherency has previously been used for camera calibration [Hernández et al. 2007a] under the further constraint of circular motion. Here we use the inverse approach and assume that given accurate calibration we can propagate knowledge across multiple views as segmentation proceeds. The intersection of silhouettes from multiple views forms the *visual hull* which must contain the object which cast the silhouettes (the maximal surface which could generate the silhouettes). Figure 5.2 demonstrates the construction of the visual hull. Figure 5.3 provides a demonstration of the reasoning behind the concept of the silhouette coherency constraint.



(a) Visual hull from two views

(b) Visual hull from three views

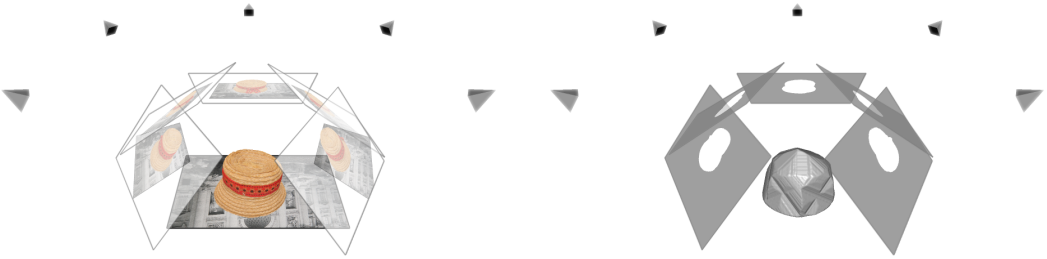
Figure 5.2: Construction of a visual hull in two dimensions. *The figures show a 2D slice through the camera centres and image planes. The visual hull is the intersection of the projection of the silhouettes in the image plane. The blue intersection of two silhouettes is shown in (a) and the change in the visual hull when a third image is added is observed in (b).*

5.3.2 Fixation Condition

In order to automate the segmentation process we need to provide a method of initialisation to replace the initial actions of the user in the standard approach, for example the initial box drawn by the user in Figure 5.1. In order to perform this initialisation we use a fixation constraint which is shown in Figure 5.4. By finding locations which have a large probability of being within the object of interest in the image we can gather sufficient data about the appearance of the object to develop an initial colour model for the object to start an iterative segmentation algorithm.

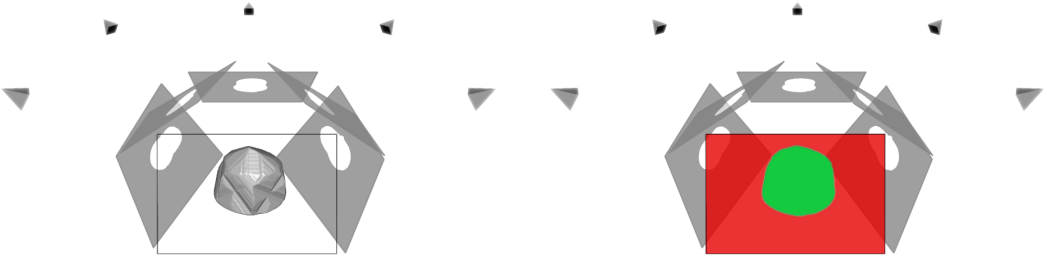
5.3.3 Problem Definition

We accept as an input a sequence of M images, $I_1 ... I_M$, of the object with each image made up of a set of pixels $\mathcal{P}_m$. We assume the images are calibrated which allows any location in the voxel volume, $\mathbf{X} \in \mathcal{R}^3$, to be mapped to its corresponding location in image I_m, such that $\mathbf{u}_m = \text{Proj}[\mathbf{X}]$. We also assume our fixation condition where the object is taken to be fully contained by and centred in each view. We form an array of N voxels, $\mathcal{V}$, from the intersection of the volumes projected by each of the images, thus the fixation constraint mandates that this volume contain the visual hull of the object. Each voxel has dimensions $(\delta x, \delta y, \delta z)$ and may be indexed as $v_n \in \mathcal{V}$. We intend to label each voxel as inside the object's visual hull ($\mathcal{O} \subset \mathcal{V}$) or outside ($\mathcal{B} \subset \mathcal{V}$), and thus part of the background in the image segmentation, such that $v_n \in \mathcal{O} \cup \mathcal{B}$. We also define $\mathbf{c}_{m,n} \in \mathcal{P}_m$



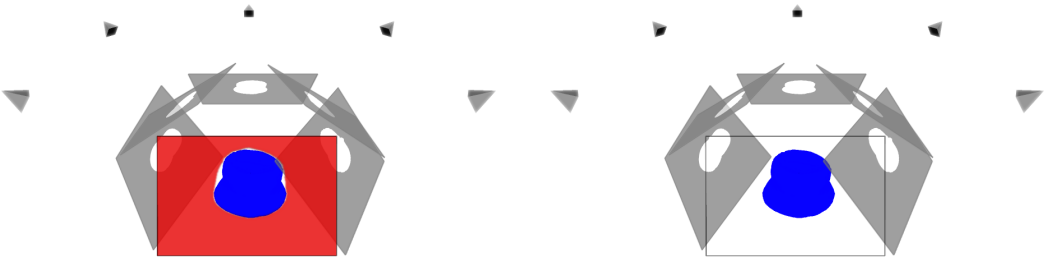
(a) An initial set of 5 views

(b) Given perfect segmentations, the visual hull is formed from the silhouettes



(c) A new image is taken from a different viewpoint

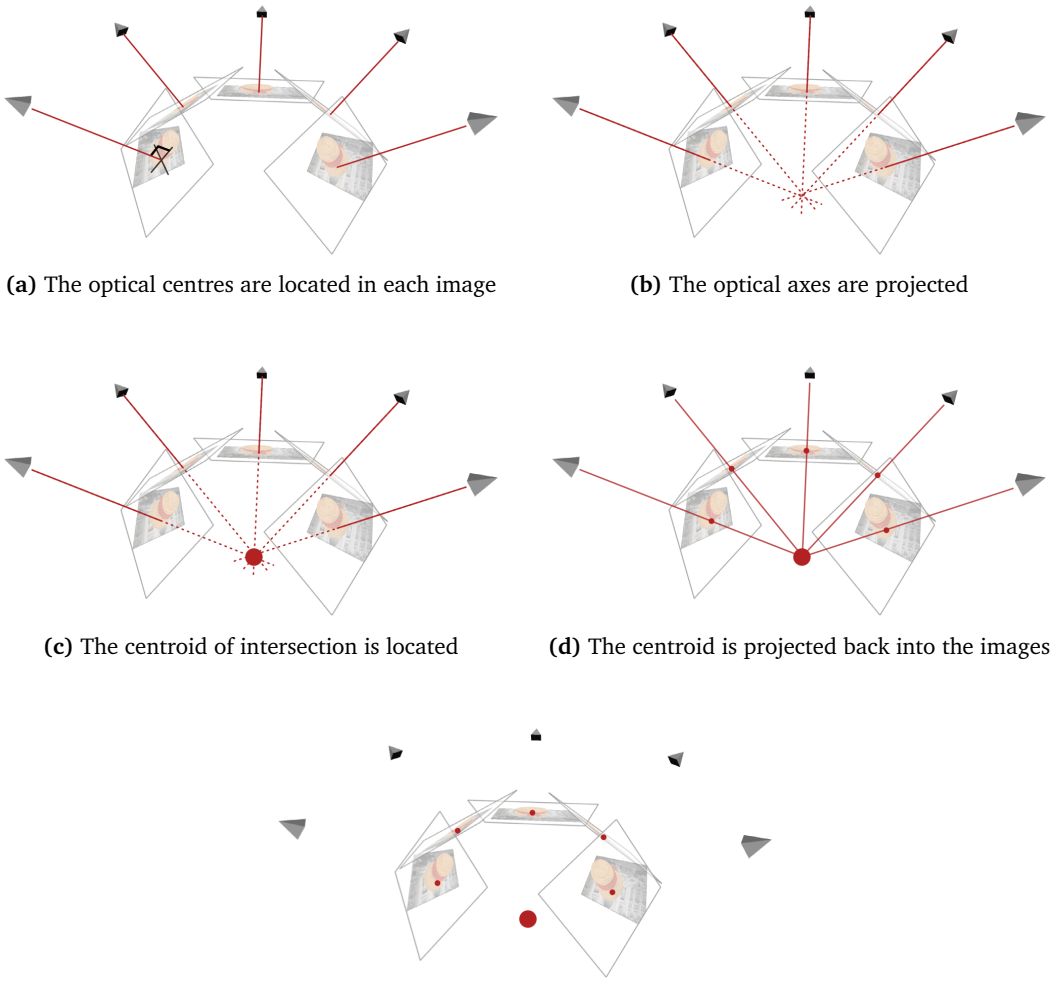
(d) The silhouette is constrained to lie inside the existing visual hull



(e) Thus the other views limit the range of possible silhouettes

(f) The desired silhouette

Figure 5.3: Illustration of silhouette coherency. *If we take a set of views of an object (a) with perfect segmentations then we may form the visual hull of the silhouettes (b). If we then wish to segment a new view (c) the desired silhouette is already constrained to be within the projection of the existing visual hull (d). Thus the range of possible silhouettes is substantially constrained (e) using the knowledge of the other silhouettes.*



(a) The optical centres are located in each image

(b) The optical axes are projected

(c) The centroid of intersection is located

(d) The centroid is projected back into the images

(e) Initialisation locations are found in each image

Figure 5.4: Using the fixation constraint for initialisation. *The optical centres are found in each image (a) and the optical axes projected into the volume (b). The centroid of intersection of these axes is found using a least square method (c) and the centroid back projected into each of the images (d) to find initialisation locations in each image (e).*

where $\mathbf{c}_{m,n} = I_m(\text{Proj}\left[v_n\right])$ is the RGB colour of the pixel which v_n projects to in image I_m. When formulating the voxel array as a graph we define a set of edges $\mathcal{E}$ containing neighbouring voxels in a six-connected sense.

5.4 Automatic Segmentation Algorithm

An overview of the automatic segmentation algorithm is given in Algorithm 3. The process starts by using the fixation condition to initialise colour models for the object and background. This allows us to enter an iterative loop which alternates between performing a volumetric graph-cut to produce a direct estimate of the visual hull and then, using the silhouettes of this current visual hull, updating the colour models for the object and background. This process continues until we have learnt accurate models for the foreground and background and converged to the visual hull of the object which produces the appropriate silhouettes. By performing the segmentation in the volume domain we segment across all images simultaneously and guarantee that silhouette coherence is always enforced. We also introduce a final stage 2D graph-cut around the converged silhouette boundaries for high resolution images where the resolution of the voxel array is too low to provide a pixel accurate result. The following two sections discuss the main components of the iterative stage: building the colour models and the volumetric graph-cut.

5.5 Building Colour Models

During the algorithm we develop colour models to provide probabilistic likelihoods for image pixels to be part of the object or background. Comparing the appearance of world surfaces between images is not ideal since we are forced to make quite strong assumptions about Lambertian reflectance, constant illumination of the scene and constant gain in the camera. Having said this, scenes of this nature are often a source of the best MVS results and these assumptions are often used by MVS algorithms whilst recovering the surface. Since we are targeting MVS we do not believe scenes approximating this nature represent too strict an assumption. In common with general practice in this area [Rother et al. 2004] we use a K component Gaussian Mixture Model (GMM), [Bishop 2006], in 3D colour space (red, green, blue or RGB) to model the likelihood. These models take the form of

$$p(\mathbf{c}\,|\,\pi_k,\mu_k,\Sigma_k) = \sum_{k=1}^{K} p(k)\,p(\mathbf{c}\,|\,\mu_k,\Sigma_k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{c}\,|\,\mu_k,\Sigma_k) \tag{5.1}$$

where $\mathbf{c}$ is a vector in colour space. Since we are using full covariance matrices in the mixture model, we notice negligible differences in the likelihood terms when using colour

Algorithm 3: **The iterative segmentation algorithm.**

Objective

- Generate the corresponding silhouettes, for each image, of the object fixated upon by the camera

Input

- A calibrated sequence of M images, $I_1 ... I_M$

Algorithm

- **Initialisation**
  1. Generate array of N voxels $\mathcal{V}$ from bounding volume of the sequence
  2. Extract fixation points for each image to initialise the colour models
- **Iterate until convergence of the visual hull**
  1. Construct cost function graph using the volume term (5.8) and the boundary term (5.11)
  2. Perform volumetric graph-cut to produce optimal segmentation (5.2)
  3. Extract the silhouettes from the current visual hull estimate
  4. Update the object colour model (5.5) using the new silhouettes over all the images
  5. Update the background colour models (5.6) using the new silhouettes for each image individually

Output

- The converged visual hull
- Extracted object silhouettes (guaranteed to be coherent from visual hull)

spaces other than RGB, for example the CIELAB colour space, as the requirement for Euclidean distances in the colour space to correspond to perceptual differences is removed. The probability distribution of the colour of the k^{th} component of the mixture is given by a normal distribution with mean μ_k and covariance Σ_k. Each of the individual components is weighted by the marginal probability of the component $p(k)$, termed the mixing coefficient and denoted π_k. The number of mixture components, K, provides a trade-off between computation time and model over-fitting against the discriminative power of the colour model. For our experiments we used $K = 5$.

At each iteration, GMMs are learnt for both the object and the background. The learning process consists of sampling the pixels as a sequence of colour vectors and using the Expectation-Maximisation (EM) algorithm to 'fit' the model parameters of the GMM to the sampled data [Bishop 2006]. We intend to exploit the fact that the object is seen in multiple views, therefore we build a full colour model for the object by sampling pixels from all the views using the silhouettes as a mask. This approach allows us to increase our knowledge about all the colours present in the object even if the initialisation fails to capture all the colours of the object. In this situation an automatic 2D segmentation would most likely fail. However, under our method the visual hull estimation forces a spatial coherency, thus when the graph-cut is performed the segmentation will generate a visual hull whose silhouettes should include a portion of the colour in at least one of the views. Since the colour model is built over all views, we only require one view to register the second colour as object in order to allow the subsequent iterations to add the colour to the model and extend the volume into the other region.

This is demonstrated in the hat sequence of Figure 5.5. The initialisation of the colour model, Figure 5.5(a), contained only the straw colour in each view, therefore the first iteration object colour model, Figure 5.5(b), fails to classify the red ribbon as object. The result of the first graph-cut segmentation, given in Figure 5.5(c), attempts to combine the separated straw coloured regions and in the process includes some of the ribbon in the segmentation. The second iteration learns an object model which propagates this knowledge of the ribbon over all the image sequences so that the object colour model includes the ribbon as in Figure 5.5(d). This is refined in the subsequent iteration to produce the correct segmentation of Figure 5.5(e), with the object colour model Figure 5.5(f), after the second iteration graph-cut.

In contrast to the object model, we expect the background model to vary over each image, therefore a separate GMM is learnt for the background of each image. Since we are unsure of the final visual hull at the start of the iteration, we conservatively estimate the background by sampling image pixels which aren't currently segmented as the object.

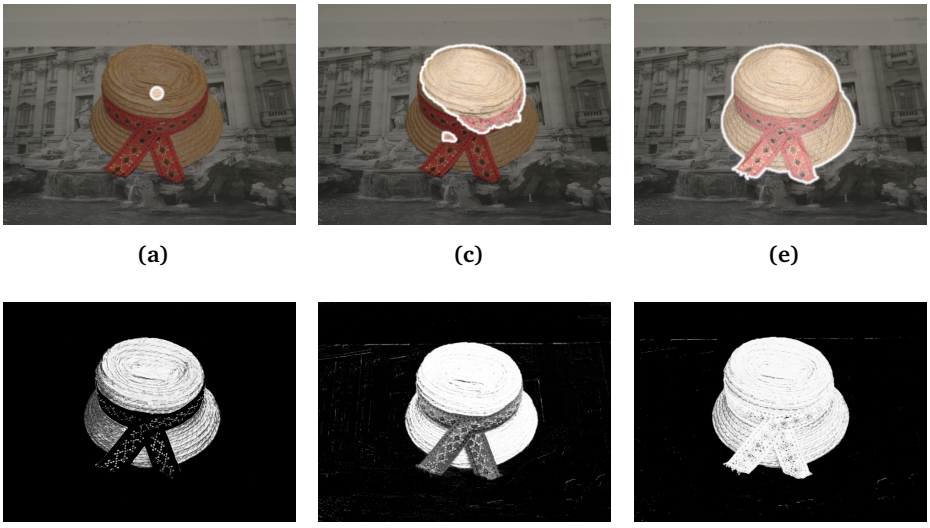


Figure 5.5: Iterative learning of the object colour model. *The fixation condition is used to provide a seed (a) for the first object colour model (b). In all images the seed areas contain only the straw colour. The result of the first graph-cut (c) contains areas of the red ribbon which is incorporated into the colour model (d). The second iteration refines this to produce the correct segmentation (e) with corresponding colour model (f). (b),(d),(f) Show the likelihood of being object $\hat{L}_{\mathcal{O}}$ under the colour model at each iteration (5.7).*

In the early stages, particularly during the first iteration, the sampled data will contain many pixels from the object itself. However, the localised object model will necessarily be a better fit to the data in and around the seed location, therefore the more generalised background model will not prevent the initial region from growing.

Since the algorithm is EM based it is possible that convergence will be to a local rather than global optimum. However, since we reject background pixels using spatial consistency, the algorithm should converge to a spatially coherent object.

5.6 Volumetric Graph Cuts

The task of segmentation is performed within the voxel array and the resulting silhouettes from the computed visual hull propagated back to the individual images. This ensures the set of image silhouettes are consistent with one another at every iteration. The segmentation operation is one of energy minimisation as

$$\hat{\mathcal{O}} \quad = \quad \arg\min_{\mathcal{O}\subset\mathcal{V}} \mathrm{E}(\mathcal{O},\mathcal{B},\{I_m\},\Theta) \ . \tag{5.2}$$

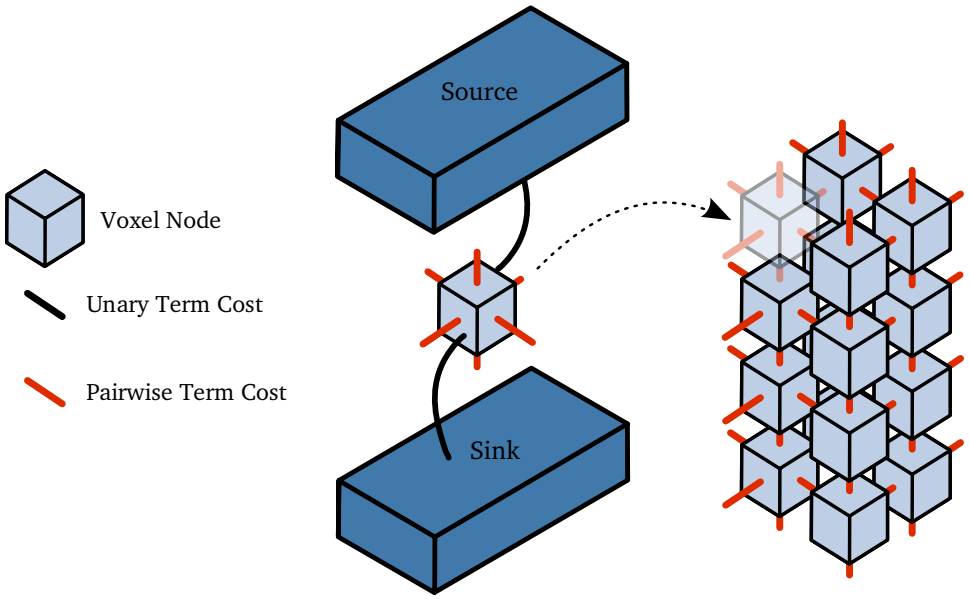


Figure 5.6: The voxel graph structure. *Each voxel forms a node in the graph and is connected to its six neighbours by a weighted edge. These edges encode the boundary term. Additionally, each node has two other connections: edges to the source and sink nodes. These edges encode the volume term.*

The energy to be minimised

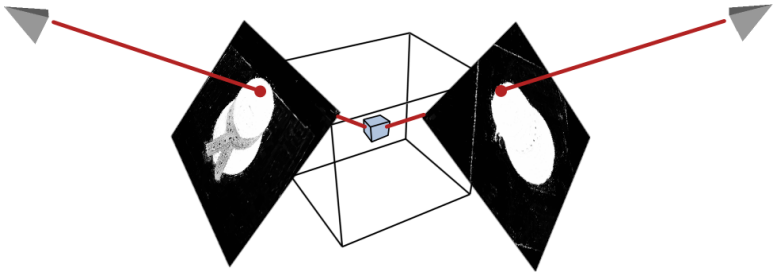
$$\mathrm{E}(\mathcal{O},\mathcal{B},\{I_m\},\Theta) \quad = \quad \lambda\,\mathrm{E}_{\mathrm{vol}}(\mathcal{O},\mathcal{B},\{I_m\},\Theta)+(1-\lambda)\,\mathrm{E}_{\mathrm{surf}}(\mathcal{O},\mathcal{B},\{I_m\}) \tag{5.3}$$

is comprised of two terms: a volumetric term and a boundary term. The parameter Θ denotes the collection of colour model parameters

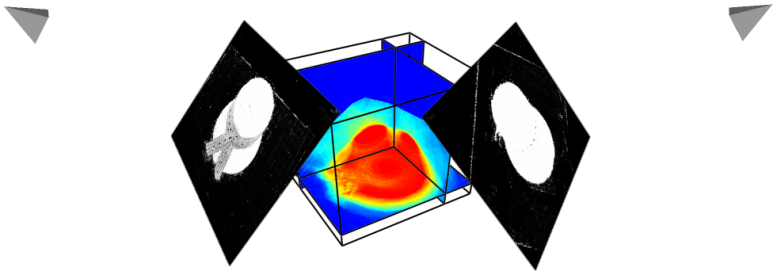
$$\Theta \quad = \quad \left[\,\{\pi_k^{\mathcal{O}}\},\{\mu_k^{\mathcal{O}}\},\{\Sigma_k^{\mathcal{O}}\},\{\pi_{k,m}^{\mathcal{B}}\},\{\mu_{k,m}^{\mathcal{B}}\},\{\Sigma_{k,m}^{\mathcal{B}}\}\,\right] \tag{5.4}$$

which are updated at each iteration using the results of the previous segmentation.

Figure 5.6 displays the voxel graph structure used to perform the volumetric graph-cut. Every voxel in the 3D volume becomes a node in a graph. The nodes are connected to their neighbours by edges (red in Figure 5.6) with weights dictated by the boundary cost term, that is the cost of cutting the edge such that one of the voxels is inside the visual hull and the other outside. In addition, every node has an additional edge (black in Figure 5.6) to the 'source' and to the 'sink' which correspond to the object (part of the visual hull) and the background (empty voxel) respectively. The weights of these edges are determined by the volume term and are construed as the probability of the voxel being object given the colour models. Since we must partition the graph so that each node is either connected to the source or sink we observe that the object and background links are complimentary which ties in with the probability (each voxel must be either



(a) Each voxel (node) is projected into each image



(b) A section through the likelihood cost volume

Figure 5.7: The volume term cost. *Every voxel (node) in the array is projected into all of the images of which two are shown in (a). The images shown are the likelihoods of being object given in (5.5). Cross-sections through the combined likelihood cost of (5.7) are shown in (b).*

object or background). The graph encodes a Conditional Random Field (with a first order Markov Assumption of independence used during the optimisation) and it can be solved in polynomial time using the max-flow/min-cut algorithm [Kolmogorov and Zabih 2004] to produce the globally optimal partition which minimises the energy cost of (5.2).

5.6.1 Volume Term

The volume term encodes the preference for a voxel to be classified as inside or outside the object. We therefore construct this term from the colour models of the individual views. For the m^{th} image in the sequence we can evaluate a likelihood term for the projected pixel colour $\mathbf{c}_{m,n}$ of voxel v_n to be part of the object

$$\mathrm{L}_{\mathcal{O}}(v_n, \Theta, m) \quad = \quad p(\mathbf{c}_{m,n} \,|\, \pi_k^{\mathcal{O}}, \mu_k^{\mathcal{O}}, \Sigma_k^{\mathcal{O}}) \tag{5.5}$$

or the background

$$\mathrm{L}_{\mathcal{B}}(v_n, \Theta, m) \quad = \quad p(\mathbf{c}_{m,n} \,|\, \pi_{k,m}^{\mathcal{B}}, \mu_{k,m}^{\mathcal{B}}, \Sigma_{k,m}^{\mathcal{B}}) \tag{5.6}$$

given the current GMM model parameters in the same form as in (5.1).

In the absence of any prior knowledge about the pixels class, we may form a classification probability of being object by normalising the likelihoods. We then sum over all

images for a single voxel, normalising by the number of images

$$\hat{\mathrm{L}}_{\mathcal{O}}(v_n,\Theta) \;=\; \frac{1}{M}\sum_{m=1}^{M}\frac{\mathrm{L}_{\mathcal{O}}(v_n,\Theta,m)}{\mathrm{L}_{\mathcal{O}}(v_n,\Theta,m)+\mathrm{L}_{\mathcal{B}}(v_n,\Theta,m)}\;. \tag{5.7}$$

Figure 5.7 details the construction of the volume term cost and an example of the final cost is given in Figure 5.9(a).

The final volume cost is given by

$$\mathrm{E}_{\mathrm{vol}}(\mathcal{O},\mathcal{B},\{I_m\},\Theta) \;=\; \sum_{v_n\in\mathcal{V}}\begin{cases}\left(1-\left[\hat{\mathrm{L}}_{\mathcal{O}}(v_n,\Theta)-\phi\right]\right) & v_n\in\mathcal{O}\\ \left(1+\left[\hat{\mathrm{L}}_{\mathcal{O}}(v_n,\Theta)-\phi\right]\right) & v_n\in\mathcal{B}\end{cases}\;. \tag{5.8}$$

Rather than use a per-image binary voting cost [Snow et al. 2000], we combine the probabilities from the individual images using an offset parameter $\phi\in[0,1]$. This parameter encodes the threshold level of the algorithm; were we simply to combine binary silhouettes it would represent the number of images which have to agree in order to start considering a particular voxel as part of the visual hull as seen in the case of ideal silhouettes where we have $\phi\rightarrow 1$ since $\hat{\mathrm{L}}_{\mathcal{O}}(v_n,\Theta)\rightarrow 1$ in the event of perfect object classification in all images. Thus ϕ controls the rate of convergence to the true solution against robustness to noise in the event of imperfect colour model classification in each image. Usually a conservative value of around 0.8 - 0.9 results in the best trade-off.

5.6.2 Boundary Term

The boundary term in (5.3) encodes the energy associated with the surface area of the object. In a similar manner to the 2D examples of [Rother et al. 2004, Boykov and Jolly 2001], we use the colour discontinuities within the images to project cutting planes through the voxel array which should form the boundaries of the visual hull. We identify the colour difference as the vector norm of the projected pixels of neighbouring voxels in each image as

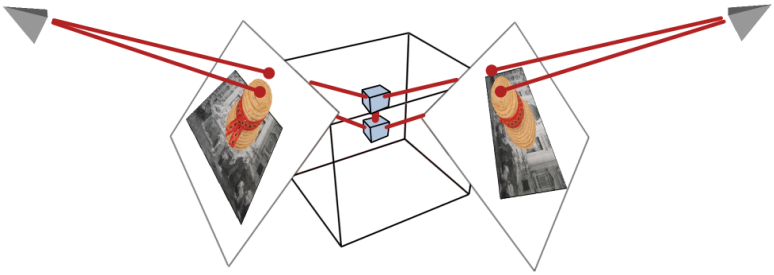
$$Z_m(v_i,v_j) \;=\; \left\|\,\mathbf{c}_{m,i}-\mathbf{c}_{m,j}\,\right\|^2\;. \tag{5.9}$$

Since we are estimating the boundary of the visual hull, the maximum colour difference across the image sequence is taken as

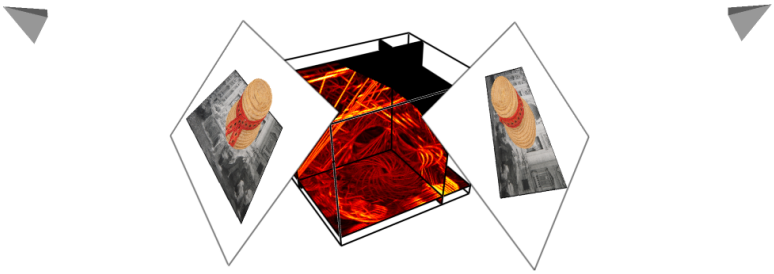
$$\hat{Z}(v_i,v_j) \;=\; \max_m\, Z_m(v_i,v_j) \tag{5.10}$$

and an exponential cost function, with β estimated from the images [Rother et al. 2004, Boykov and Jolly 2001], used to provide the standard Gibb's model of

$$\mathrm{E}_{\mathrm{surf}}(\mathcal{O},\mathcal{B},\{I_m\}) \;=\; \sum_{(v_i,v_j)\in\mathcal{E},\ \substack{v_i\in\mathcal{O}\\ v_j\in\mathcal{B}}} e^{-\beta\,\hat{Z}(v_i,v_j)}\;. \tag{5.11}$$



(a) Each pair of neighbouring voxels (nodes) are projected into each image



(b) A section through the maximum colour difference volume

Figure 5.8: The boundary term cost. *Every pair of neighbouring voxels (nodes) are projected into all of the images of which two are shown in (a). Cross-sections through the maximum colour difference volume of (5.10) are shown in (b).*

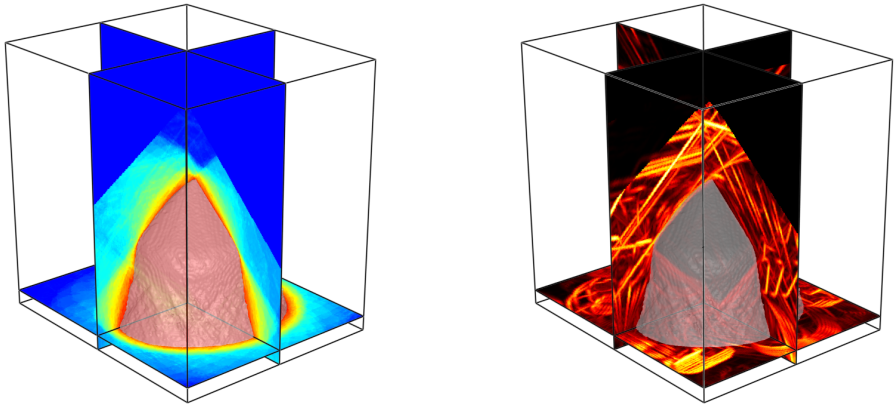
Figure 5.8 details the construction of the boundary term cost and an example of the final cost is given in Figure 5.9(b).

5.7 Experiments

Results of the automatic segmentation algorithm are provided in Chapter 8, specifically the hat sequence of Figure 8.2, the hand sequence of Figure 8.3 and the house sequence of Figure 8.5.

In addition to the full system testing of § 8.1, the segmentation algorithm was tested on the challenging statue sequence of Figure 5.10. For this sequence planar calibration was not possible and therefore the camera intrinsic parameters were calibrated separately followed by the application of structure from motion techniques [Hartley and Zisserman 2004] to track correspondences on the object itself and thus provide an initialisation for bundle adjustment.

The statue sequence consists of 69 images of a statue in the British Museum. The images were captured by hand at a resolution of 5 MP. As can be seen from the example images of Figure 5.10(a), the background continually changes including both the surrounding building and numerous different people and hence a very wide range of colours.



(a) Final volume term **(b)** Final boundary term

Figure 5.9: Final energy cost terms. *The final energy terms are shown with the converged visual hull. The volume term is shown in (a) with blue to red as low to high* $p(\text{object})$*. The boundary term is shown in (b) with red to orange as low to high* $p(\text{edge})$*.*

The lighting conditions also change dramatically as can be seen by comparing the 3^{rd} and 4^{th} images of Figure 5.10(a).

We can compare our results with an implementation of [Rother et al. 2004] applied independently to each image. Instead of user interaction, we generously supply the 2D method with the converged colour models of the 3D segmentation algorithm rather than just the fixation points. Figure 5.10(b) provides the silhouettes and Figure 5.10(c) shows the visual hull resulting from the combination of the 2D segmentation results.

The segmentation results after 9 iterations of our method are given in Figure 5.10(d). The majority of the silhouettes are recovered accurately offering significant improvements over the independent 2D segmentations. We observe that the left leg of the statue is not completely recovered. Since the leg structure is hollow, roughly half of the views see the inside of the sculpture which is different in colour and has failed to be modelled by the priors. Whilst the intersection of 2D binary silhouettes results in removing the whole leg, Figure 5.10(c), our 3D segmentation algorithm tries to resolve the discrepancy between the silhouette coherency which results in capturing half the leg as object.

5.8 Discussion

Figure 5.11 shows the converged object likelihoods for the statue sequence. We observe that there are many views where the object is not completely separable from the background in colour space. This is a limitation of the colour models used by the 2D and 3D algorithms and explains the poor 2D result when no further information, other than the

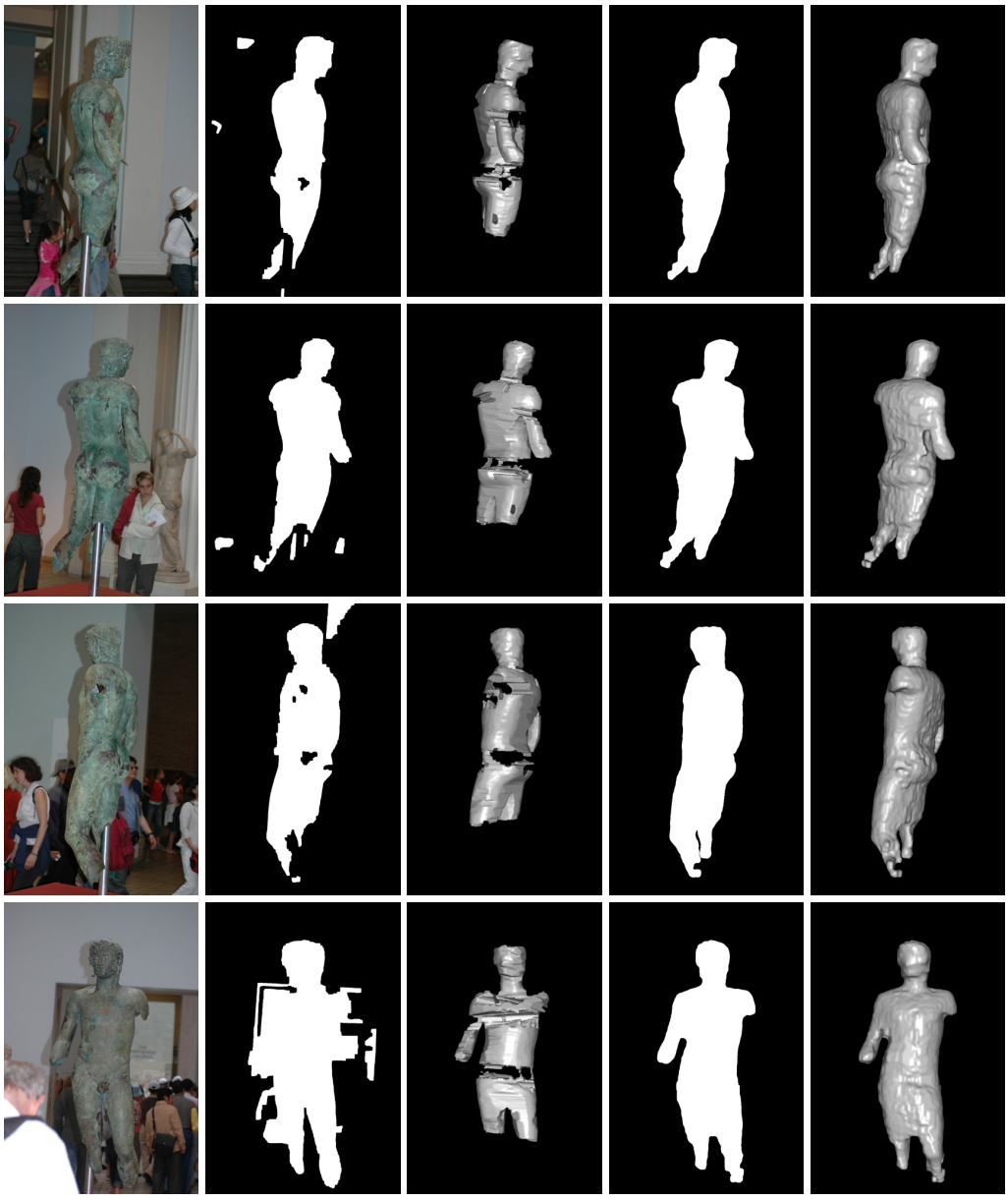


Figure 5.10: A single 3D segmentation improves multiple independent 2D segmentations. *The statue sequence contains 69 images of a statue observed in a continually changing background environment (a). Independent 2D segmentation results in incoherent silhouettes (b) and a poor visual hull (c). After convergence, our method produces silhouettes (d) and a visual hull (e) with the correct topology.*

Figure 5.11: Converged object likelihoods from the statue sequence. *The images show the final likelihood of object of (5.7) for the statue sequence of Figure 5.10.*

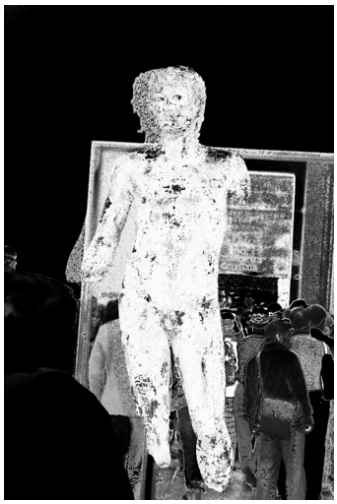
object priors, can be used to perform the segmentation which results in the incoherent silhouettes.

Figure 5.12 studies the limitations of the 3D algorithm, more specifically the effect of regularisation, in more detail. The algorithm over estimates the silhouette in areas where the colour model provides conflicting information (i.e. the object likelihoods are inconsistent across different images) due to the dominance of the spatial regularisation in these regions. At the extremities we observe an under estimate of the silhouette due to the fact that the statue is hollow, with a different appearance inside, which adds confusion to the graph-cut which is trying to estimate a solid body. In fact, if we enlarge the boundary size used for the final 2D graph-cut we can recover the correct segmentation, in this case, since we have strong edges in the original image.

We perform a more in-depth study of the limitations of automatic segmentation at the start of Chapter 6 (§ 6.3) and aim to overcome them, at the expense of sacrificing autonomy, with a small amount of input from the user as part of an interactive process.

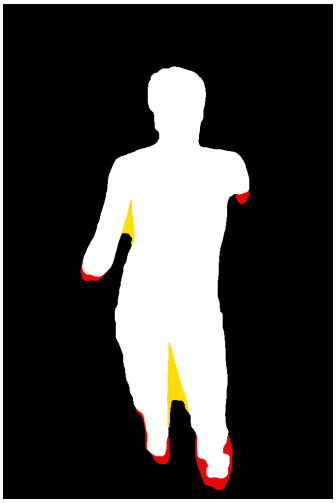
The voxel grid has a limited resolution based on the memory requirements for the graph-cut algorithm. For our results, voxel grids containing $N \sim 300^3$ were used and able to fit inside 4 GB of memory. Unfortunately the edge term is often densely populated, as observed in Figure 5.9(b), and thus the use of sparse data structures, for example an octree, will only confer a small reduction in memory requirements. This is not usually a problem if a final 2D boundary graph-cut is being performed since it will account for any mismatch in resolution between the image and the voxel grid.

(a) Input image

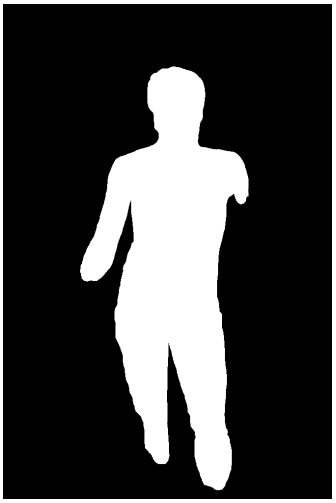


(b) Converged object likelihood

(c) 2D segmentation result



(d) 3D segmentation errors



(e) Actual silhouette recovered

Figure 5.12: Limitations of the segmentation algorithm. *An example of one of the statue sequence images (a) where the statue is not separable in colour space (b). The regularisation of the 3D algorithm improves upon the results of the 2D algorithm (c) but also results in over estimating the silhouette, shown in yellow in (d). The segmentation result also under estimates regions, shown in red in (d), but the full silhouette of (e) may be recovered using a final boundary 2D graph-cut around the silhouette from the converged visual hull.*

CHAPTER 6

A Clustering Approach to Object Segmentation

6.1 Introduction

We have already discussed the need to segment 3D objects, to aid the reconstruction process, in Chapters 4 and 5. We also note that there are limitations to the automatic segmentation algorithm of Chapter 5 both in terms of generality with respect to image sequences and in terms of the premise of the fixation condition and, consequently, automatic initialisation. We now approach the task of overcoming these limitations, specifically considering sequences of a more complex scene that violate the fixation condition.

One approach might be that of the co-segmentation algorithm [Rother et al. 2006]. Here a specific optimisation is used to co-segment (simultaneously) the entire sequence in 2D using the similarity in appearance of the object across multiple viewpoints, neglecting the information from the pose of the camera for each image. Unfortunately this approach breaks down when the appearance of the object changes significantly between viewpoints. On the other hand, we have already seen that recent advances in semi-interactive segmentation [Blake et al. 2004, Boykov and Jolly 2001] enable the segmentation of an object using small amounts of user input (e.g. a bounding box around the object of interest). However even a modest amount of manual labour becomes impractical for longer sequences of images if it must be performed on a per image basis.

This chapter details a practical solution that allows user interaction but with the aim of minimising the input required. To reduce the demand for user input we propose the use of inter-frame epipolar constraints arising from the rigidity of the scene and the known camera motion. We advocate a semi-automatic approach where the user is able to influence the final segmentation by labelling small amounts of pixels in the sequence that are

propagated according to the geometric constraints.

The key to our approach is the simplification of the segmentation problem by pre-clustering the scene pixels in two different levels. The first level is to perform a simple over-segmentation of the image pixels into *superpixel* regions. In the second level these superpixels themselves form clusters across the scene, according to appearance and consistency with the epipolar constraints. These larger clusters thus correspond to physical, spatially consistent scene objects rather than simply regions of similar appearance. Having computed this pre-clustering, the segmentation task essentially becomes one of selecting which cluster(s) make up the final object. To do this, the user must hand-label a small set of pixels as object and background. Since our pixel pre-clustering might be inconsistent with the user's labels, we use a graph-cut based relabelling process to restore consistency. The contributions are the following:

- The graph-clustering formulation of the calibrated sequence segmentation problem.
- A graph-cut based algorithm for establishing consistency between a user's manual pixel labels and our pixel pre-clustering.
- An easy to use, interactive segmentation for large image sequences.

6.2 Prior Work

There is a vast body of work related to interactive image segmentation (see [Blake et al. 2004] and references contained therein). However most of this work is concerned with segmenting a single image. Performing the interactive segmentation task for each image in a sequence individually, quickly becomes prohibitive as the length of the sequence increases. A simple extension of interactive 2D segmentation on video appeared in [Wang et al. 2005] where the user labels regions in a 3D space-time volume and the system segments a space-time region corresponding to a potentially deforming object. That method relies too heavily on the continuity of video, and hence cannot be applied to a typical, wide-baseline MVS sequence.

Apart from the contribution of Chapter 5, the most related work, also addressing the problem of segmenting a calibrated image sequence, can be found in [W. Lee and Boyer 2007]. In a similar manner to our automatic segmentation algorithm, in [W. Lee and Boyer 2007] viewing volumes of all the images are intersected and from this initial volume, a background colour model is learnt. Because they are highly related, these algorithms are compared to this new work in more detail in § 6.3.2.

Our work is related to [Sormann et al. 2006], where sequential multi-view segmentation is achieved by pre-clustering each image using Mean-Shift and then interactively segmenting the clusters using a graph-cut optimisation. The optimisation exploits multi-view constraints by sequentially segmenting a sequence of images, and using the previous result as a shape prior on the new image segmentation. In comparison, our work proposes a formulation of the multi-view constraints that simultaneously segment all the images.

The idea of over-segmenting an image sequence in the context of multi-view stereo has appeared in [Jancosek and Pajdla 2009]. The key difference of that paper to the present work is that its aim is a Multi-view stereo algorithm using superpixel over-segmentation to reduce computational load. Here we focus entirely on the segmentation task, using multi-view stereo constraints to propagate pixel labelling.

In [Snow et al. 2000], a graph-cut based approach is used to estimate the voxel occupancy of a calibrated volume in space. Their approach is directly aimed at using an energy minimisation framework to regularise the process of combining a series of imperfect silhouettes. The main difference is that they obtain these silhouettes as the result of a background subtraction process from a fixed, calibrated camera rig whereas our method requires no prior knowledge of the object or environment.

The task of segmenting objects in multi-views has also been studied in [Yezzi and Soatto 2003]. The authors use a level set method to evaluate the segmentation based on Lambertian scenes with smooth or constant albedo. Level set methods are known to be susceptible to local minima so [Yezzi and Soatto 2003] relies on smooth albedo variation and a multi-resolution scheme to achieve convergence. In contrast, our method tolerates albedo discontinuities.

Our work is also related to uncalibrated image co-segmentation [Rother et al. 2006] which aims at simultaneously segmenting an object out of a sequence of images, without using any geometric rigidity constraints. The advantage of that approach is that the method can segment non-rigid objects (i.e. a walking person). On the other hand, the method will not segment objects whose appearance varies dramatically with the viewpoint.

Finally, [Brostow et al. 2008] is an example of using multi-view constraints (in the form of Structure-from-motion results) to aid the task of per-pixel scene labelling where the labels have been predefined. Our work also uses multi-view rigidity constraints in a completely unsupervised clustering approach.

6.3 Problem Analysis

6.3.1 Single View Segmentation

As discussed in § 4.1, the task of segmentation is a challenging one, not least because in a given image the decision of what makes a good segmentation is in some sense arbitrary. Many of the latest algorithms adopt an interactive approach allowing feedback from the user to guide the segmentation process [Rother et al. 2004, Boykov and Jolly 2001]. In the case of segmentation in a single image two constraints are typically exploited. Firstly, we expect some form of colour or texture consistency within each segment and variation across segments. For example, consistency may be modelled by the use of intensity histograms [Boykov and Jolly 2001] or Gaussian mixture models in colour-space [Blake et al. 2004]. Secondly, we also make the assumption that segments are spatially continuous within the image. This prior may be enforced by modelling the segmentation task using, for example, an MRF [Boykov and Jolly 2001, Rother et al. 2004] or a level set method [Yezzi and Soatto 2003].

6.3.2 Limitations of Automatic Multiple View Segmentation

If we now consider the case of a calibrated sequence of images, we have seen in Chapter 5 that we may exploit scene rigidity to obtain a silhouette coherency constraint, between the images. This arises due to the fact that the corresponding segmentations in each image are projections from the same rigid 3D object. We were able to combine this constraint with the colour coherency and image spatial priors to perform automatic foreground/background segmentation of an object of interest across multiple images.

Whilst this approach works well for certain image sequences, it faces a number of limitations when addressing those that are more challenging. When the foreground and background are not readily separable in the feature space (colour in this instance) we are unable to separate the two just based on an appearance and image spatial consistency, rather we must rely on the silhouette consistency constraint. However, whilst the enforcement of silhouette consistency does compensate for the overlapping foreground/background distributions, the resulting segmentation will lose accuracy.

We can observe the loss of accuracy in Figure 6.1. Here we see the method of Chapter 5 failing to converge to the full object when applied to such a sequence. Figure 6.1(b) shows the algorithm to under estimate the silhouettes, in particular the head of the horse is missing. We see the difficulty faced by the generative model on this sequence by observing the image region shown in Figure 6.1(c) and the converged object likelihood in Figure 6.1(d) that fails to distinguish between the stone horse and background foliage.

This is a more extreme example of the limitations observed in Figure 5.12. Whilst there are many other images where the head of the horse may be more easily separated from the background there are a sufficient number of images that see the head as background to mean that the ϕ offset parameter, of (5.8) in § 5.6.1, would have to be so low (to allow the head to be recovered when it is more likely to be background in a number of images) that the silhouettes drastically over estimate the body of the horse.

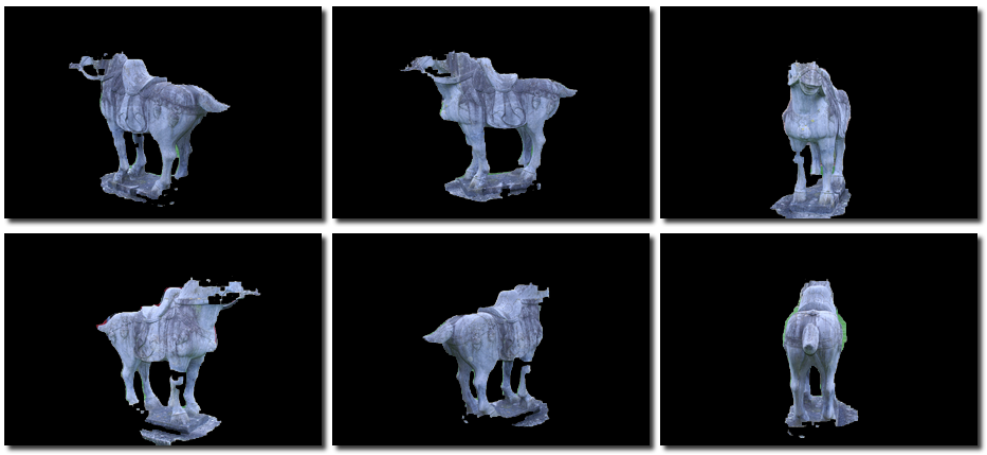
Previously we adopted an interactive approach that alternates between updating the generative colour models and enforcing spatial consistency across the views. Thus the issues with separating foreground and background under generative models will lead to two problems. Firstly, the algorithms will always run the risk of producing segmentations that are consistent but not the desired outcome. For example, the silhouettes in Figure 6.1(b) are spatially consistent with each other but don't correspond to the whole object. Secondly, the algorithms will require some form of automatic initialisation to begin their iterations. We also made use of a 'fixation condition', namely that the object of interest must be centred in all views, to initialise colour models whilst [W. Lee and Boyer 2007] begins from the intersection of the viewing volumes making the assumption of coherent colours in the background. Whilst both of these approaches are demonstrably successful with certain image sequences, it is not very difficult to find a more general sequence that will present difficulties. For example, sequences containing an off centre object or multiple objects with more elaborate backgrounds will present significant problems for initialisation as well many opportunities to become trapped in local minima. Figure 6.7(a) represents an example of such a sequence.

These problems present a fundamental limitation to the application and robustness of these algorithms. The single image segmentation algorithms make use of user feedback, in the form of interactive labelling, to overcome situations where the generative models fail to separate foreground and background and to provide suitable initialisation (for instance a tight bounding box [Lempitsky et al. 2009]). If we attempt to apply these techniques to multiple view sequences the user is faced with the sizeable task of providing hand labels to lots of images to which we must add the fact that the computation time required for both algorithms does not lend them to an interactive setting. If we want to perform segmentation on general multiple view sequences we need an algorithm that minimises the amount of input from the user and operates at interactive speeds.

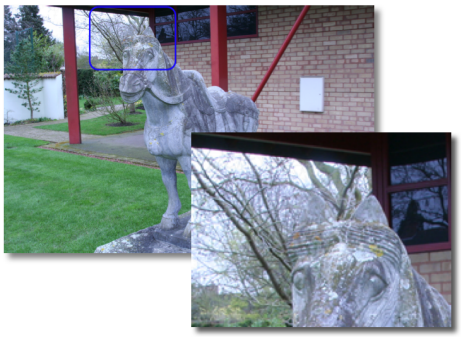
6.3.3 The Clustering Approach to Single View Segmentation

Although we assume that an algorithm is not capable of determining the user's segmentation criteria ahead of time, we may still perform some general processing on the image

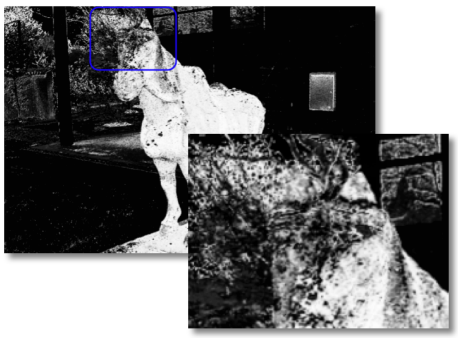
(a) Images of a horse sculpture



(b) Result using automatic segmentation algorithm



(c) Zoomed image



(d) Zoomed likelihood

Figure 6.1: Limitations of a generative colour model. *(a) 6 of the 36 images calibrated images of a horse sculpture. (b) The automatic segmentation results obtained for the sequence using the method of Chapter 5. We note that the silhouettes slightly over estimate the body of the horse whilst the head of the horse is not recovered. (c) A zoomed region of one of the images that displays the difficulty in separating foreground and background based on colour or texture. (d) The converged foreground likelihood of the same region reflects this difficulty by failing to distinguish clearly foreground and background.*

with the goal of making it easier for the user to select their appropriate segmentation. Thus we may consider an alternative approach to single image segmentation: we perform an initial processing stage to produce over-segmentation of the image. This over-segmentation may then be presented to the user, or perhaps a subsequent learning algorithm, in order for them to select the regions appropriate to their desired segmentation, obviously a much smaller task than asking the user to label pixels individually. This approach has been studied extensively for single images [Ren and Malik 2003, Levinshtein et al. 2009] including attempts to learn the properties of boundaries in natural images to automate the process as much as possible. Such techniques usually correspond to the combination of a clustering technique with a domain specific kernel.

Our approach is to extend this clustering philosophy to multiple view sequences such that we may precompute an over-segmentation (set of clusters) across all the images in the sequence such that these segments correspond to the same object observed in multiple views. This then allows the user to obtain their desired segmentation by selecting the appropriate clusters simultaneously across all the images without having to label each one individually.

6.3.4 A Clustering Approach to Multiple View Segmentation

We would propose to group regions of similar appearance (in our case colour) that correspond to consistent spatial locations across the image sequence with the task of interpretation left to the user as a process of labelling the regions appropriately. The strength of such an approach comes from a clustering framework that combines all of the segmentation cues, consistency of appearance (colour) as well as spatial consistency within the a single image and between the different views, in a single process. This removes the need to adopt an iterative technique that alternates between the different constraints.

We incorporate these cues by performing a graph based clustering with a kernel that promotes colour consistency and creating edges between images that obey the epipolar geometry induced by the location viewpoint of the images (obtained from the camera calibration). Since the clusters span across multiple views, we can reduce the labelling requirements placed on the user. Any information provided is automatically propagated across the whole sequence.

This approach presents several challenges. Firstly, multiple view image sequences will often contain many images ($50 - 100$) and it is also common to operate with high resolution images (> 6 MP). This results in a very sizeable data-set to process. Our solution is to perform an initial over-segmentation of each individual image in the sequence to represent each image with $\sim$4000 superpixels. The over-segmentation process is designed to

produce superpixels that respect boundaries within their original image and thus we may be hopeful that the final segmentation boundaries obtained will also respect those boundaries. If it is necessary to obtain highly accurate silhouettes, a final post-processing step may be performed using the superpixel segmentation as an input. The extraction of the segmentation alpha matte is another possible post-processing step.

The other main challenge, one faced by all clustering algorithms, is a method of determining the number of clusters K, a parameter required by the majority of clustering techniques. This would be a particularly difficult parameter to determine accurately since it is ultimately subjective. Furthermore a small K may result in a single cluster containing foreground and background regions whilst a large K increases the labelling demand on the user.

Instead of attempting to specify the correct number of clusters K a priori, we adaptively refine the clustering results, at interactive speeds according to user input. One possible method to achieve this would be to compute a tree hierarchy of clusters. With that approach if a user would like to split a cluster (in the case that the initial K was too small) we would have already precomputed a subdivision. However, this may still result in a large number of clusters since we have no guarantees that our precomputed split reflects the users desired split and thus we may have to compute and store multiple possible subdivisions. Instead we set K such that the resulting cluster sizes allow any cluster to be split tractably at interactive speeds. This allows us to use a graph-cut technique to split the cluster based on the labelling provided by the user and thus more likely to end up with the desired segmentation with the least number of clusters.

The following section details all the steps in the clustering algorithm.

6.4 Algorithm

6.4.1 Overview

We begin with a set of M images $I_1 .. I_M$ with known camera calibration. The calibration may be obtained automatically for a wide range of scenes. We pose the multiple view segmentation process as a graph clustering problem. For the sake of tractability, as discussed in § 6.3.4, we begin by over-segmenting each image I_m to obtain a set of superpixels $\{s_{j_m}\},\ j_m = 1 .. J_m$ with $J_m \sim 4000$. These superpixels then form the vertices (or nodes) $\mathcal{S} = \{s_{j_m}\}$ in an graph $\mathcal{G} = (\mathcal{S}, \mathcal{W})$. The edges $\mathcal{W}$ of the graph are represented by the edge adjacency matrix W where W_{j_m,j_n} represents the weight of the edge between s_{j_m} and s_{j_n} and a value of $W_{j_m,j_n} = 0$ indicates the absence of an edge. The construction of the W matrix is described in § 6.4.2. We then perform a spectral clustering of $\mathcal{G}$ to obtain K clusters using

the algorithm of [Meila and Shi 2001] as discussed in § 6.4.3. In § 6.3.4 we describe how the value of K is chosen such that the clusters produced contain a sufficiently low number of superpixels to allow any subsequent splitting to be performed at interactive speeds. This parameter is consequently dependent on computational resources. The final stage of the segmentation process is to present the resulting clusters to the user to allow for cluster labelling and refinement via a graph-cut method. This is outlined in § 6.4.4.

6.4.2 Generating the Weight Matrix W

The algorithm that generates the weight matrix W is provided in Algorithm 4 and demonstrated diagrammatically in Figure 6.2. The simplest algorithm would be to take each superpixel s_{j_m} and connect it to all the superpixels in all the other images that satisfy the epipolar constraint of (6.1) with a weight determined by the colour consistency (6.2). This approach suffers from two problems. The first is demonstrated by Figure 6.3. Unfortunately, constraining neighbouring superpixels to lie on epipolar lines is not sufficient to guarantee that superpixels are matched correctly since image regions of similar colour but belonging to different objects may also lie on the epipolar line, as shown in Figure 6.3(a).

Given that we are already computing colour consistencies (6.2) for all the superpixels found on the epipolar lines in neighbouring images it makes sense to combine the information from all the different views in a weak stereo algorithm to estimate the likely depth of the superpixel and thus identify matches which correspond to a physical object at this location in space. We do this by forming the histograms over depth bins in (6.4) and (6.5). This addition is not computationally intensive but results in marked improvements in obtaining correct edge matches, as shown in Figure 6.3(b). The product in (6.5) encourages consensus between the neighbouring views. However, due to occlusion, the correct depth may be discarded due to an occluded view erroneously registering low colour consistency. To make the product robust to this effect we include factors α, and $\bar{\alpha} = (1 - \alpha)$ that act as a mixture with an outlier distribution.

The second issue is one of tractability. Even using the superpixels from over-segmenting the image, we still have a very large problem size. The horse sequence of Figure 6.1(a), for example, contains $J = \sum_m J_m \sim 160,000$ superpixels. Thus the clustering stage has to solve a $J \times J$ eigenproblem. In order to ensure that this problem is tractable the W matrix must be sparse. The epipolar constraint already promotes a degree of sparsity in the matrix, however, we can reduce the computational demand if we can increase sparsity without loss of useful information. The histogram depth binning process encourages this since the incorrect matches will be given a very low weight and may thus be safely thresholded from W without affecting the resulting clusters. This is indicated by the reduction in matches

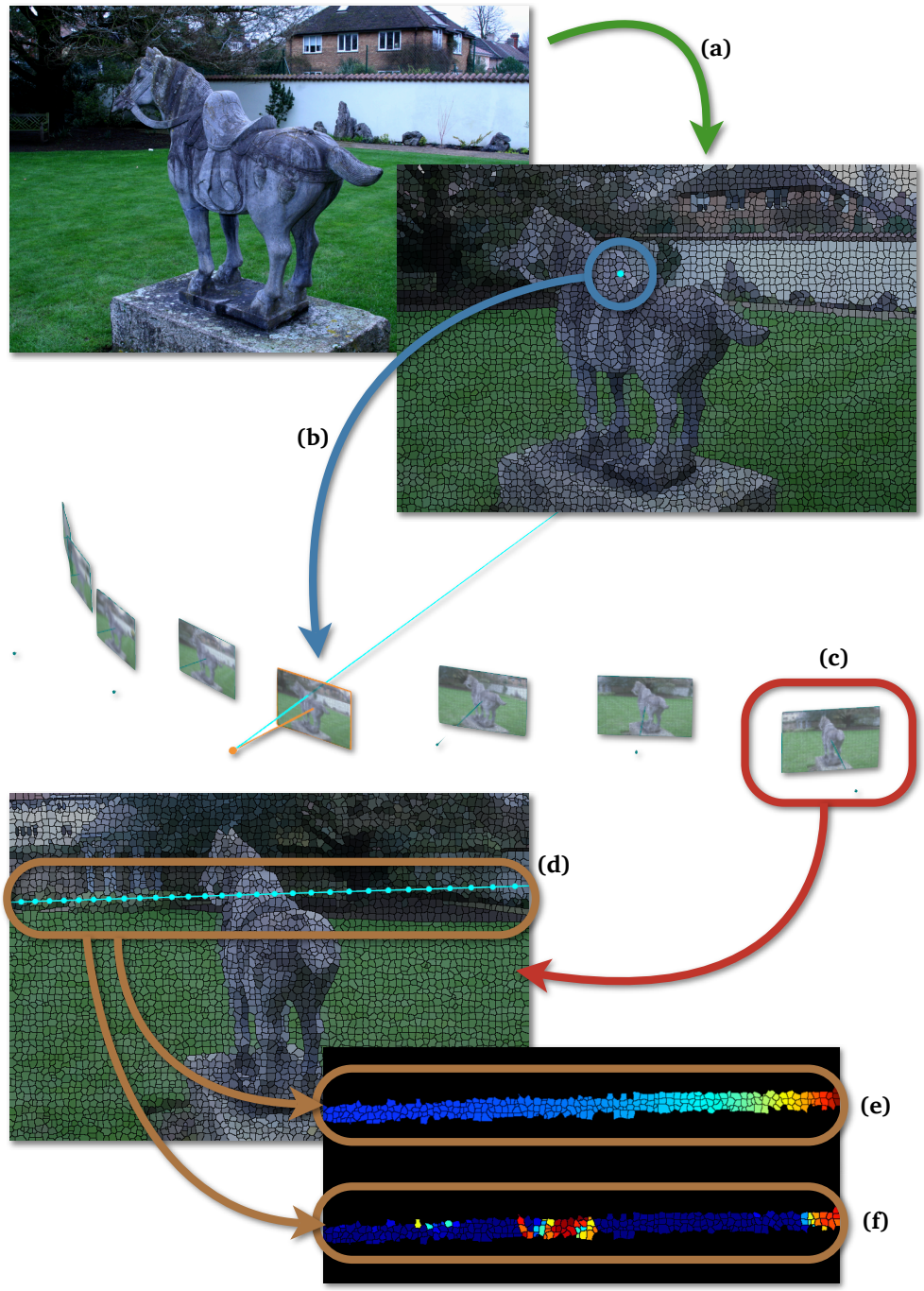


Figure 6.2: Illustration of the construction of the weight matrix W. *(a) Each of the initial images I_m is over-segmented to produce a superpixel representation $\{s_{j_m}\}$. (b) Every superpixel s_{j_m} is projected into a set of neighbouring images using epipolar geometry. (c) Each of the neighbouring images $I_n \in \mathcal{N}_m$ is selected in turn. (d) The set of superpixels $\{s_{j_n}\}$ that lie along the epipolar line is found. (e) The depth and (f) colour consistency is found for each s_{j_n} and used to construct the histogram of (6.5).*

Algorithm 4: **The weight matrix generation algorithm.**

Initialisation

foreach *image* $I_m,\ m = 1..M$ **do**
group the pixels into a set of superpixels $\{s_{j_m}\},\ j_m = 1..J_m$
determine each superpixel's mean colour $\mathbf{u}_{j_m} \in \mathcal{R}^3$ and position $\mathbf{x}_{j_m} \in \mathcal{P}^2$
end

Generate weight matrix W

foreach *image* $I_m,\ m = 1..M$ **do**
find set of neighbouring images $\mathcal{N}_m \subset \{I_n \mid n \neq m\}$ to I_m
foreach *superpixel* $s_{j_m},\ j_m = 1..J_m$ **do**
foreach *neighbouring image* $I_n \in \mathcal{N}_m$ **do**
find the epipolar line $\mathbf{l}_{(j_m,n)} = F_{(m,n)}\,\mathbf{x}_{j_m}$
find the set of superpixels that lie on the epipolar line

$$\left\{s_{j_n} \mid \mathrm{dist}\left(\mathbf{l}_{(j_m,n)}, \mathbf{x}_{j_n}\right) < \delta\right\} \tag{6.1}$$

foreach *superpixel* s_{j_n} **do**
find the colour consistency

$$\mathrm{c}\left(j_m, j_n\right) = \exp\left(-\lambda \left|\left|\mathbf{u}_{j_m} - \mathbf{u}_{j_n}\right|\right|_2^2\right) \tag{6.2}$$

find the depth

$$\mathrm{d}\left(j_m, j_n\right) = \mathrm{triangulate}\left(\mathbf{x}_{j_m}, \mathbf{x}_{j_n}\right) \tag{6.3}$$

end
construct a histogram $\mathrm{h}_{j_m,n}$ over depth bins $\hat{\mathrm{d}}_i$ such that

$$\mathrm{h}_{j_m,n}\left[\hat{\mathrm{d}}_i\right] = \max\left(\left\{\mathrm{c}\left(j_m, j_{n_i}\right) \;\middle|\; \mathrm{d}\left(j_m, j_{n_i}\right) \in \hat{\mathrm{d}}_i\right\}\right) \tag{6.4}$$

end
estimate the probability of the correct matching depth in bin $\hat{\mathrm{d}}_i$ as

$$\mathrm{p}_{j_m}\left[\hat{\mathrm{d}}_i\right] = \prod_{I_n \in \mathcal{N}_m} \frac{\alpha}{N_B} + \bar{\alpha}\left(\frac{\mathrm{h}_{j_m,n}\left[\hat{\mathrm{d}}_i\right]}{\sum_o \mathrm{h}_{j_m,n}\left[\hat{\mathrm{d}}_o\right]}\right) \tag{6.5}$$

where N_B is the number of bins and α an outlier ratio
foreach *neighbouring image* $I_n \in \mathcal{N}_m$ **do**
foreach *superpixel* $s_{j_n} \mid \left(\mathrm{dist}\left(\mathbf{l}_{(j_m,n)}, \mathbf{x}_{j_n}\right) < \delta\right)$ **do**

$$W_{j_m,j_n} = \mathrm{p}_{j_m}\left[\mathrm{d}\left(j_m, j_n\right)\right]\,\mathrm{c}\left(j_m, j_n\right) \tag{6.6}$$

end
end
end
end

found in Figure 6.3(b) vs. Figure 6.3(a) that were both thresholded at the same level.

The number of neighbours, $|\mathcal{N}_m|$, is also a question of the availability of computational resources since increasing the number of neighbours reduces the sparsity.

6.4.3 Performing Spectral Clustering

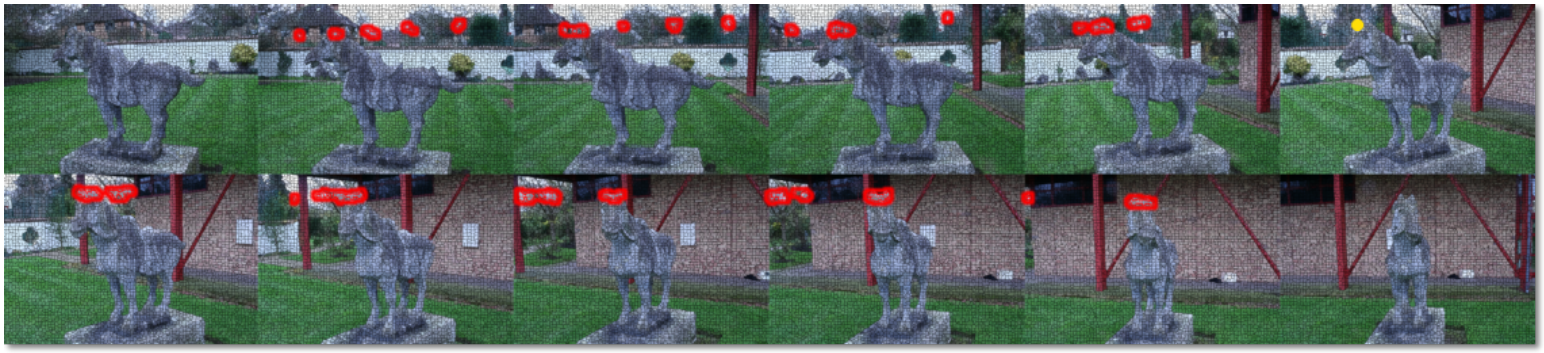
Once the weight matrix W has been generated, clustering is relatively straightforward although it is the most computationally expensive part of the process. We use the algorithm of [Meila and Shi 2001] which computes a set of K clusters in a single process, rather than forming a hierarchical clustering process. The clustering proceeds by obtaining the stochastic matrix $P = D^{-1}W$ where $D_{j_m,j_m} = \sum_{j_n} W_{j_m,j_n}$ is a diagonal 'normalising' matrix. Next we find the K largest eigenvectors in $P\mathbf{v} = \gamma\mathbf{v}$ which are concatenated to form a $J \times K$ matrix V. We then cluster the rows of V in $\mathcal{R}^K$ using K-Means to find K clusters used to label the superpixels.

6.4.4 User Interaction

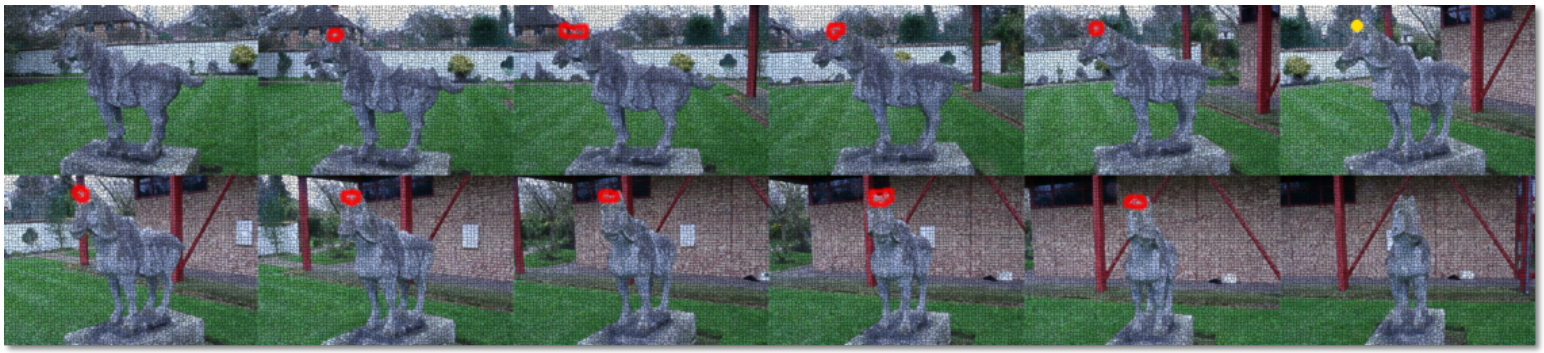
Once we have allocated each of the superpixels to a cluster, the user can now label and refine the clusters by drawing on one or more images with an appropriate 'foreground' or 'background' brush. Once any part of a cluster is labelled, the label will propagate across the entire cluster. If insufficient clusters were used during the initial segmentation, a cluster may contain foreground and background regions. This is identified by the user allocating different labels to superpixels in the same cluster. When this occurs we perform a binary graph-cut to separate the cluster. The graph-cut is performed on the sub-graph $\mathcal{G}_k \subset \mathcal{G}$ corresponding to the nodes in cluster k. We simply allocate the unary terms as the labellings provided by the user, with uninformative unaries on the unlabelled nodes, and the pairwise terms are obtained from the sub-matrix W_k obtained from the rows and columns of W corresponding to superpixels in cluster k. This refines the clustering by splitting the cluster into two, each with a different labelling. Because we have reduced the problem size down to the number of nodes in a cluster this may be performed in real-time in an interactive process that allows the user to label quickly the entire image sequence.

6.5 Experiments

Figure 6.4 provides an overview of the entire segmentation process. The input images are over-segmented and then run through the spectral clustering process to produce an initial set of clusters, shown in Figure 6.4(e). The user wishes to segment the fountain and we can see that the initial clusters that contain the fountain also overlap onto the wall.



(a) Edges without depth bin histogram



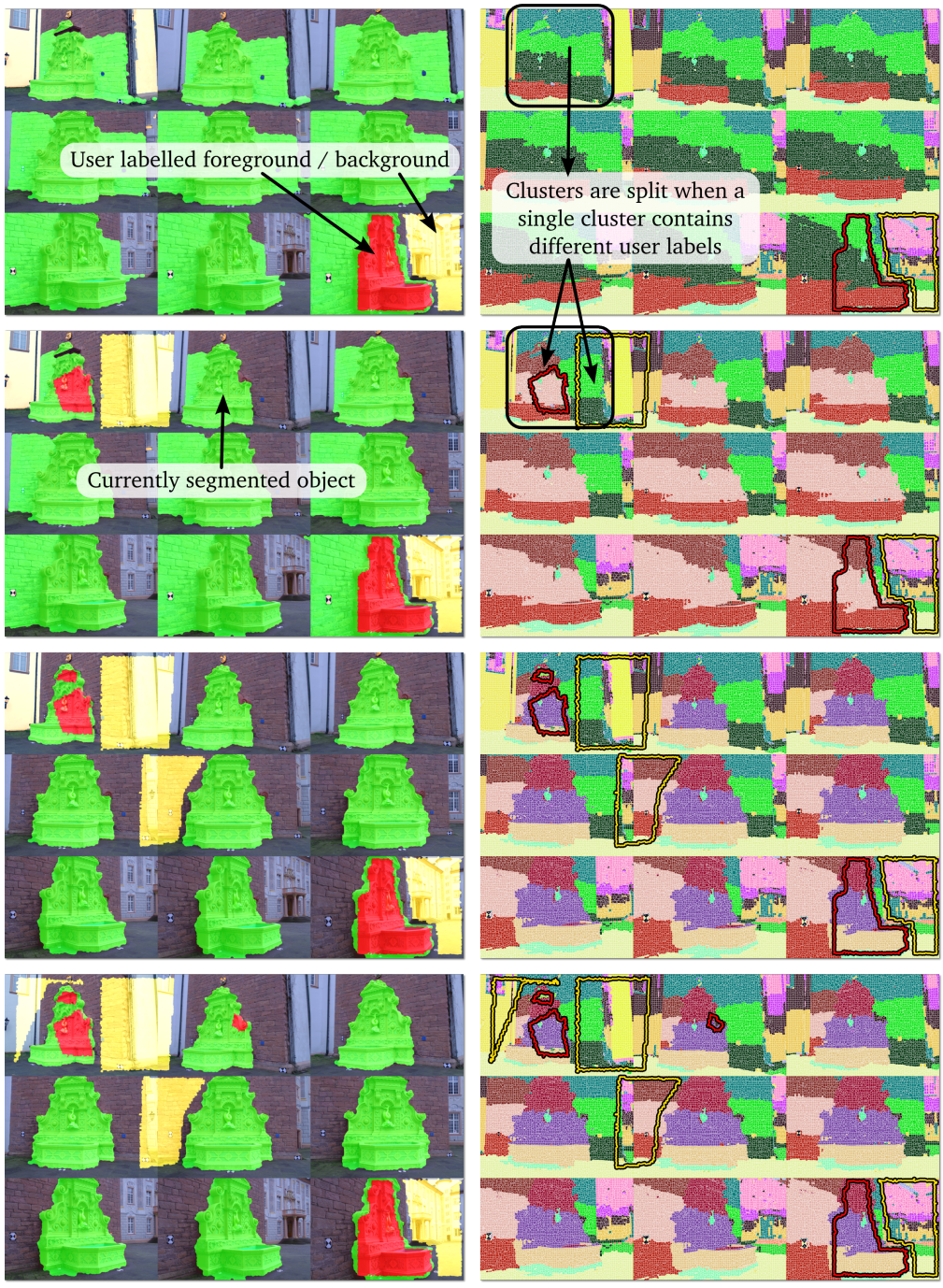
(b) Edges with depth bin histogram

Figure 6.3: The effect of the depth histogram. *The image superpixels connected (i.e. those with non-zero coefficients in the weight matrix W) to the yellow superpixel (top right image) are shown outlined in red. This is a graphical portrayal of the row of the weight matrix W corresponding to the yellow superpixel. (a) Shows the superpixels that form edges in the graph when W is calculated without using the histogram stage of Algorithm 4. Whilst the epipolar constraint is satisfied, we observe that a large number of superpixels are matched incorrectly since depth information is ignored and only colour similarity is used for matching. (b) Shows the edges in the graph found when using the histogram stage of Algorithm 4 to calculate W, thus including depth information as well as colour similarity for matching. The depth binning rejects almost all the incorrect matches by forming a consensus on the correct depth bin of the original superpixel. In both cases, the matches shown are superpixels connected to the reference (yellow) superpixel with an edge weight above a constant threshold.*

(a) Input images

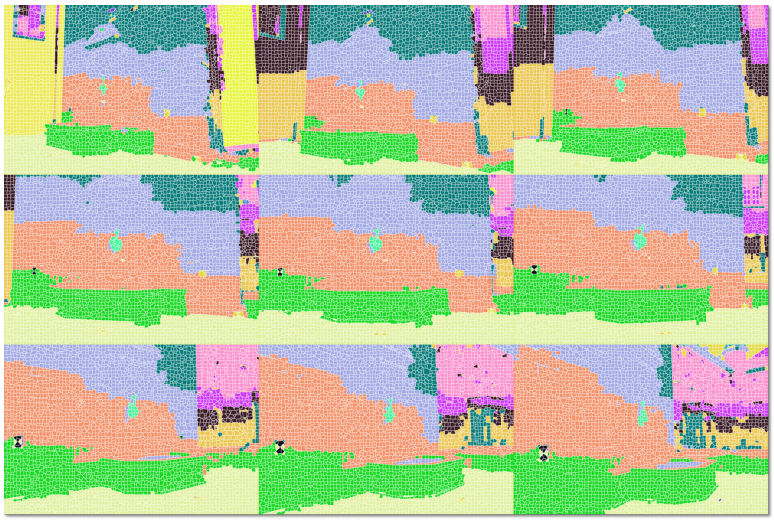
(b) Final segmentation result

Figure 6.4: Interactive segmentation results for the fountain sequence. *(a) Images (9 of 11) of the fountain sequence (data-set from [Strecha et al. 2008]). (b) The final segmentation result after performing the initial clustering and the interactive labelling process summarised in (c) and (d).*



(c) Updated segmentations over iterations **(d)** Updated clusters over iterations

Figure 6.4: Interactive segmentation results for the fountain sequence (Continued). *Interactive labelling of the clusters, resulting in clustering refinement when necessary, as an interactive process. (c) The iteratively updated segmentation with red (foreground) and yellow (background) colouring indicating user supplied labels and green the current segmentation result. (d) The corresponding clustering refined at each iteration.*



(e) Initial clustering

(f) Final clustering

Figure 6.4: Interactive segmentation results for the fountain sequence (Continued). *(e) The result of the initial clustering algorithm for $K = 15$ and (f) the final clustering result after refinement based on the interactive labelling.*

In an interactive process the user hand labels foreground (red) and background (yellow) regions in the image as shown by the colour overlays in Figure 6.4(c). Here the iterative process is demonstrated by four 'snapshots' of intermediate states from start (at the top) to finish (at the bottom). This process is guided by feedback in the form of the continually updated current segmentation (the green overlay) which is recomputed after each addition from the user. At any stage, if the user labels the same cluster with different labels, the affected clusters are automatically refined to separate the labels, shown in Figure 6.4(d). An example of this is the first two images of Figure 6.4(d) where the foreground and background (red and yellow) labelling indicate that the clusters that contain wall and fountain must be split. This leads to the final segmentation of Figure 6.4(b) and the refined clustering of Figure 6.4(f) with very little input required from the user, far less than would be required to segment each image individually.

Figure 6.6 shows the results of running the segmentation algorithm on the horse sequence of Figure 6.1(a). We observe that, at the expense of a few brush strokes by the user, the algorithm correctly segments the entire horse when compared with the automatic approach in Figure 6.1(b). Here we have made use of the camera intersection bounding box to label automatically parts of the image as background, Figure 6.6(b). Whilst this is useful, in terms of saving time, it is not a requirement of the algorithm which may have the user provide the only source of labelling. Whilst the superpixels offer a considerable advantage, making the clustering process tractable, they are also a limitation on the accuracy of the final segmentation since superpixel boundaries may fail to coincide with true image boundaries as seen in Figure 6.5.

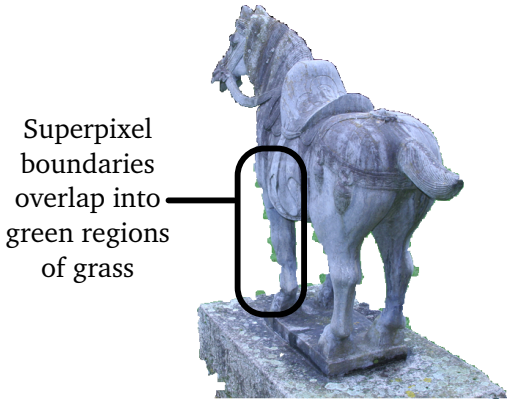


Figure 6.5: Close-up on superpixel boundaries. *An example of boundary errors in the segmentation due to failure of superpixel boundaries to lie on true image boundaries.*

Figure 6.6(e) shows the segmentation result directly after the user labelling process. Some errors are visible as 'orphaned' superpixels where the silhouette constraints have not been completely enforced. If these segmentations are intersected volumetrically (to

enforce silhouette coherency) we obtain the results of Figure 6.6(f). We can quantify these errors by comparing the two sets of segmentations with ground-truth segmentations. This comparison is shown in Figures 6.6(g) and 6.6(h). The pediment of the statue, shown in grey, is ignored for the purposes of the evaluation to allow for a fairer comparison (it is outside the bounding box in the automatic segmentation results).

	Segmentation Pixels:		
Algorithm	Correct	False Background	False Object
Automatic (Chapter 5)	82.4%	17.6%	2.1%
Clustering before Vol. Intersection	97.4%	2.6%	10.5%
Clustering after Vol. Intersection	95.7%	4.3%	1.3%

Table 6.1: Comparison of quantitative segmentation errors for the horse sequence. *The results of the automatic segmentation algorithm of Chapter 5 are compared with the clustering segmentation results before and after volumetric intersection.*

Table 6.1 provides the numerical segmentation errors shown as percentages of pixels (relative to the ground truth labelling) that are correctly labelled and those that are falsely labelled, as background (under estimate the silhouette) or foreground (over estimate the silhouette). We can see that with only a few strokes from the user the result before volumetric intersection has greatly increased the number of correctly labelled pixels and reduced the areas where the silhouette under estimates the object substantially. This is at the expense of increasing the number of pixels falsely labelled as belonging to the object. However, if we perform a volumetric intersection of the results we can reduce the number of these false object pixels for a reasonable drop-off in correct and false background labelling error. Clearly an interactive labelling approach should allow very accurate segmentation since, in the limit, the user could label each superpixel individually. Here we have tried to mitigate this effect by limiting the number of user strokes to a reasonable level, the whole labelling process is performed in less than a minute.

Figure 6.7 provides segmentation results for a table top scene that contains multiple objects and a wide range of colours. Such a scene is clearly not suitable for the fixation initialisation process of Chapter 5 and the presence of multiple colours in the foreground and background make it very difficult for the iterative algorithms of Chapter 5 or [W. Lee and Boyer 2007] to avoid local minima and converge to a meaningful solution. Again, a minimal demand is placed on the user who is free to segment readily any of the objects

individually.

For all experiments we use $\lambda = 1$ having normalised the colour data of an image sequence using linear PCA. The outlier ratio α was set to 0.25, we used the 8 nearest neighbouring images for each image and $K = 30$. The over-segmentation stage took an average of 30s per image for 4000 superpixels and a further 120s to generate the weight matrix. Solving the eigenproblem was the most computationally expensive stage, taking 15 minutes to run for the horse sequence.

6.6 Discussion

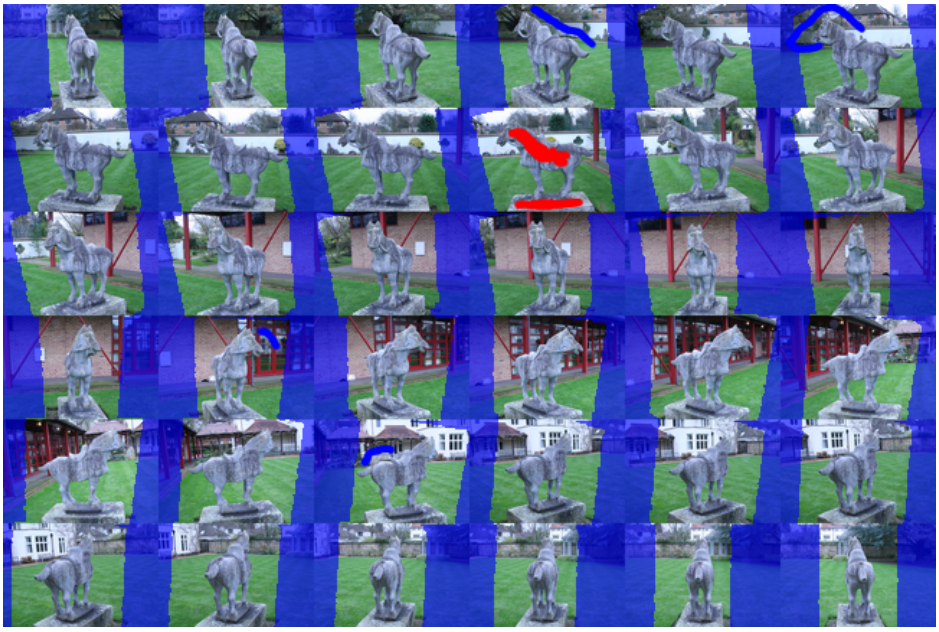
Our experiments have demonstrated that we may successfully segment an object from a calibrated image sequence using a two stage approach of an initial clustering followed by an iterative user-guided labelling. The elegant formulation as a graph-clustering problem allows a collection of spatially meaningful clusters to be obtained tractably across all the images. The clustering results allow the user to be presented with a labelling task that operates across the whole image sequence simultaneously to minimise the demands placed on the user's time and effort. We have also found that the initial clustering results sufficiently reduce the problem size such that individual clusters may be refined in real-time if the user labelling conflicts with the current clustering results.

We have shown that this process allows us to improve segmentation performance over the automatic segmentation algorithm of Chapter 5 on challenging sequences as well as dealing with segmentation tasks with constraints that are too complex for the simple fixation condition used previously. However, the method is not without its limitations which we will now discuss further.

6.6.1 Limitations

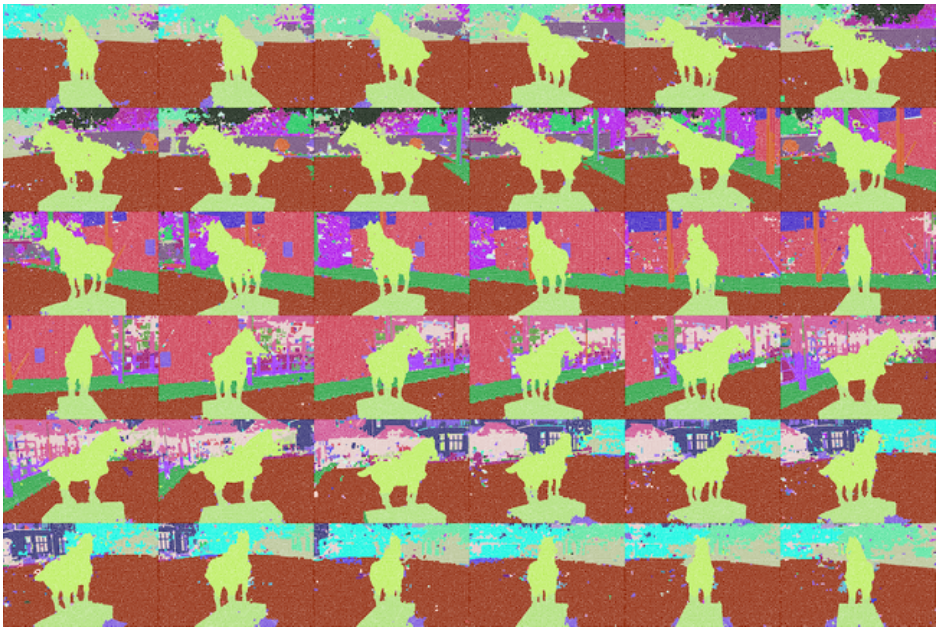
We have displayed, in Figure 6.5, the segmentation errors that occur when superpixel boundaries fail to coincide with the actual image boundary. A possible solution to this problem is to adopt a post-processing stage similar to the one proposed in Chapter 5. Here we may use the known size of the superpixels to set the border radius for a final 2D boundary graph-cut, performed on each image individually, to sharpen the final segmentation using the strong image edges nearest to the superpixel segmentation result. If the segmentation is being used for an approximating surface for reconstruction, in the form of a visual hull, it may be less critical to obtain an exact segmentation and slightly overestimating the silhouette may be more robust (to ensure the true surface is contained within the visual hull).

(a) The complete image sequence

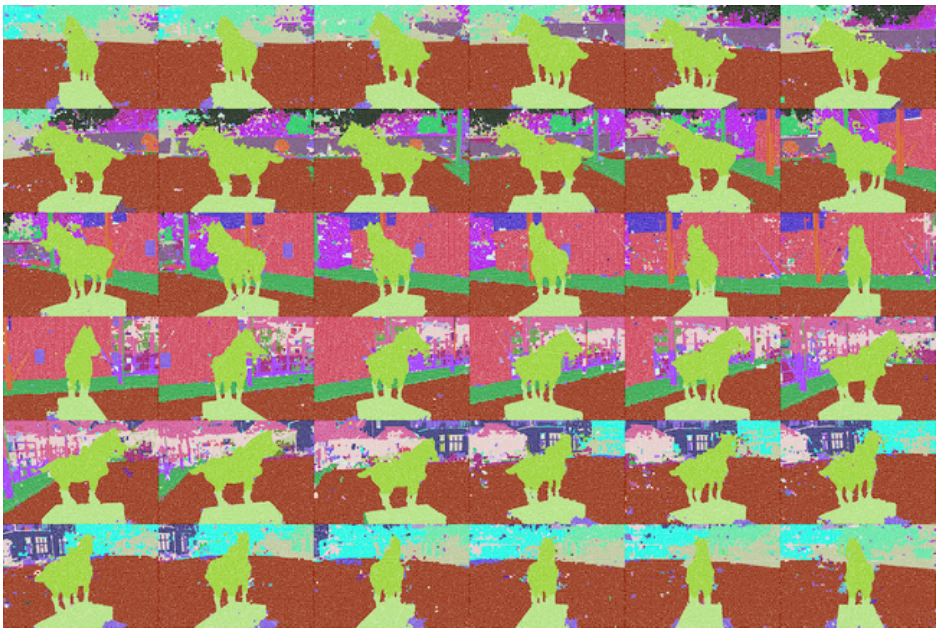


(b) User labelling

Figure 6.6: Segmentation results on the horse image sequence. *(a) All 36 images of the horse sequence. (b) The labelling strokes provided by the user (red for foreground, blue for background) with areas outside the bounding box as background (blue overlay).*



(c) Initial clusters



(d) Refined clusters

Figure 6.6: Segmentation results on the horse image sequence (Continued). *(c) The initial clusters produced by the spectral clustering of W. (d) The clusters after labelling and refinement.*

(e) Segmentation result before volumetric intersection

(f) Segmentation result after volumetric intersection

Figure 6.6: Segmentation results on the horse image sequence (Continued). *(e) The segmentation result after user labelling and (f) the final result after volumetric intersection.*

(g) Segmentation errors before volumetric intersection

(h) Segmentation errors after volumetric intersection

Figure 6.6: Segmentation results on the horse image sequence (Continued). *(g) The segmentation error before and (h) after volumetric intersection. Areas where the algorithm over estimates the silhouettes are shown in yellow with areas that it under estimates the silhouettes shown in red. Note that the pediment, shown in grey, is not included in the quantitative analysis. This is to provide a fair comparison since it is not fully observed in all images and will be mostly removed by the volumetric intersection. If it were considered it would dominate the pixel errors, providing a biased impression of performance since we actually care about the accuracy of the boundaries of the object observed fully in all views (the horse).*

We may also note the improvement offered by performing a volumetric intersection of the segmentation results (to guarantee silhouette coherency), quantified in Table 6.1. Whilst this might be a simple operation to insert into each iteration of the user labelling process, we have kept the two results (before and after intersection) separate in order to evaluate directly how well the clustering approach, more specifically the weight matrix W of (6.6), is capturing and enforcing 3D spatial consistency. The results obtained for the horse sequence indicate that the colour consistency along the epipolar lines, even with the inclusion of the weak stereo weighting of § 6.4.2, is not always sufficient to resolve the difference between object and background when the colours are similar, for example the errors in Figure 6.6(g).

When considering the use of colour information we would identify two areas for discussion. The first concerns the comparison of colour between images. In the sequences used for reconstruction we are assuming that the images were taken at the same time with the same camera under constant illumination. We hope that this ensures there are no gross errors in colour consistency between the images, for example those that would occur if some images were taken at night and the others during the day for an outdoor scene. Of course, even under these assumptions we are not guaranteed complete colour consistency however we can improve the situation by making use of the image point correspondences obtained during calibration. If we have a set of point correspondences across images we can perform a colour correction for the images by selecting a reference image and fitting a colour correction model to each of the remaining images based on the consistency of colours at corresponding image points. This is a technique used in the image mosaicing community, for example [Capel 2004]. We have applied this technique to the sequences presented in this chapter using an affine colour correction model.

The second area of discussion is to look at the model used to determine colour consistency. In order to gain further insight we may contrast the colour model used here against the one used in the automatic algorithm of Chapter 5. To illustrate the problem we provide a recap of the generative colour model used by the automatic algorithm.

In the automatic algorithm we maintain two Gaussian Mixture Models (GMMs) which encode probability distributions, one for the foreground and one for the background, over all possible colours. Thus we may take a single colour sample and determine a likelihood that it belongs to the foreground or background. The problem is, as previously discussed, that colours are often shared by the foreground and the background and so, depending on how we estimate the GMM from the data, we may end up in a situation where the likelihoods are either uninformative (equally likely to be object or background) or incorrect (if the colour is predominantly found in the other category).

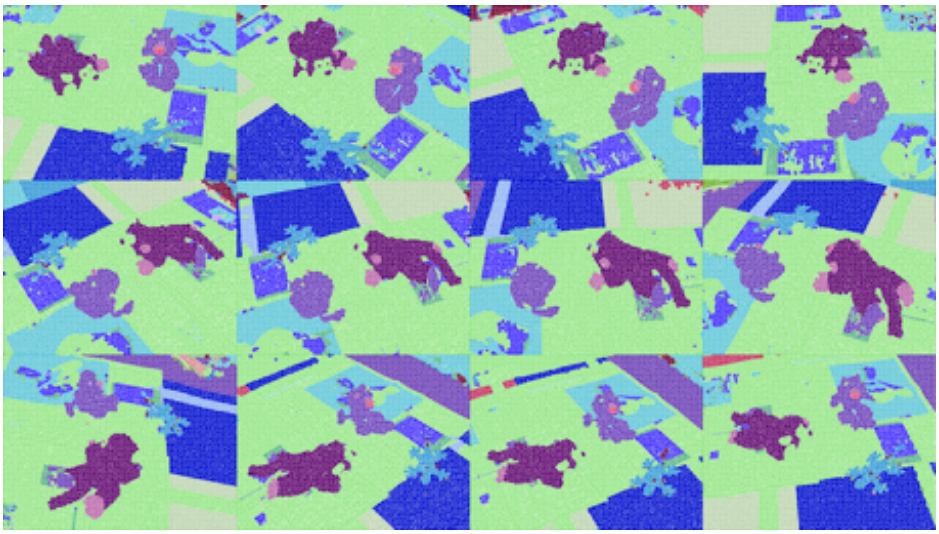
In contrast to this generative colour model, the clustering algorithm makes use of a kernel function to compare colours. Consequently, instead of taking a single colour sample and evaluating a likelihood, we take two colour samples and evaluate how similar they are. This has the advantage that we can use extra information, such as the epipolar distance or depth, to modulate the colour similarity and so the same two colour samples will have a different similarity score dependent on their location. Therefore we have found a solution to the problem of the overlapping foreground and background colour distributions.

The problem, however, is that this solution comes at the expense of having to set explicitly the kernel parameters for the comparison in colour space. At the moment the clustering kernel makes use of the Euclidean distance in colour space whereas the GMMs of the generative model effectively translate to a comparison metric using a data-dependent Mahalanobis distance. This distance more accurately reflects colour difference since it is dependent on the colour distribution observed in the object and background. In an attempt to mitigate the impact of the Euclidean distance, we use linear PCA to map the colour samples into a space where the distance is more meaningful, also known as data 'whitening' [Bishop 2006]. However, this does not eliminate the problem entirely since the distance is effectively a single Gaussian approximation of the GMMs used in Chapter 5. We would identify resolving this issue as an avenue for future research.

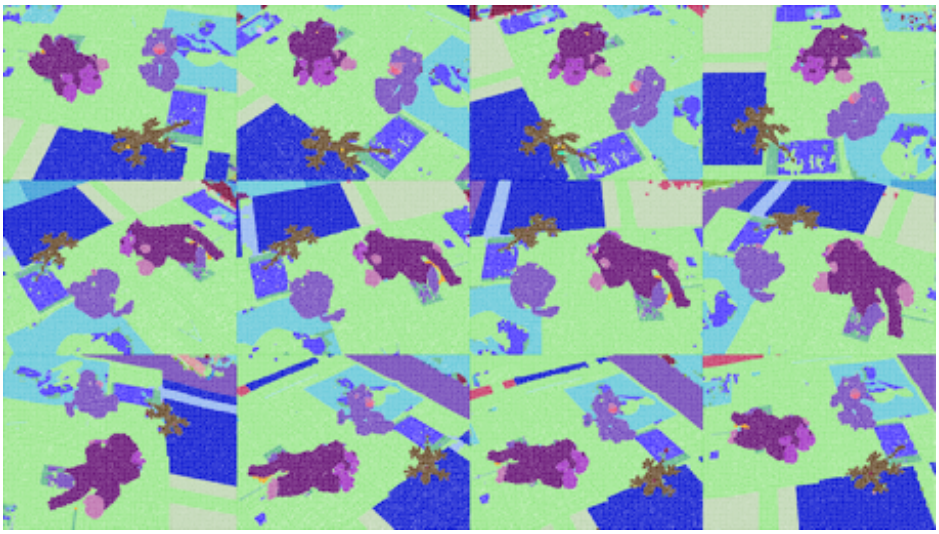
(a) Images of table top scene

(b) User labelling

Figure 6.7: Segmentation results on a table top scene. *This is a good example of a sequence where the fixation condition of Chapter 5 would be insufficient to initialise an automatic segmentation process. (a) 12 of 16 images of a table top scene containing multiple objects and colours with no clear object of interest. (b) The labelling strokes provided by the user (red for foreground, blue for background) with areas outside the bounding box as background (blue overlay).*



(a) Initial clusters



(b) Refined clusters

Figure 6.8: Segmentation results on a table top scene (Continued). *(a) The initial clusters produced by the spectral clustering of W. (b) The clusters after labelling and refinement.*

(a) Segmentation result

Figure 6.9: Segmentation results on a table top scene (Continued). *(a) The final segmentation result.*

CHAPTER 7

Depth-Map Estimation

7.1 Introduction

The topic of multi-view stereo was introduced in § 4.3 and observed that the current top performing algorithms, for example [Hernández and Schmitt 2004], suffer a significant drop in performance when moving from a large number of images (50-100) to sparse data-sets (10-20 images) where one expects far less redundancy across the sequence. In common with many other algorithms, [Hernández and Schmitt 2004] adopts an elegant two stage approach. Firstly a series of depth-maps are estimated using local groups of the input images. This is followed by the second stage which combines the depth-maps to form a global surface estimate. In this chapter we show that if individual depth-maps are filtered for outliers prior to the fusion stage, good performance can be maintained in sparse data-sets. Our strategy is to collect a list of good hypotheses for the depth of each pixel. We then chose the optimal depth for each pixel by enforcing consistency between neighbouring pixels in the depth-map. A crucial element of the filtering stage is the introduction of a possible *unknown* depth hypothesis for each pixel, which is selected by the algorithm when no consistent depth can be chosen. This pre-processing of the depth-maps allows the global fusion stage to operate on fewer outliers and consequently improve the performance under sparsity of data.

7.2 Prior Work

A taxonomy of the established methods for dense stereo may be found in [Scharstein and Szeliski 2002b]. Most of these methods use matching costs to assign each pixel to a set of disparity levels within the image. The earlier algorithms maintained relatively few separate levels and were more targeted towards depth based segmentation rather than detailed

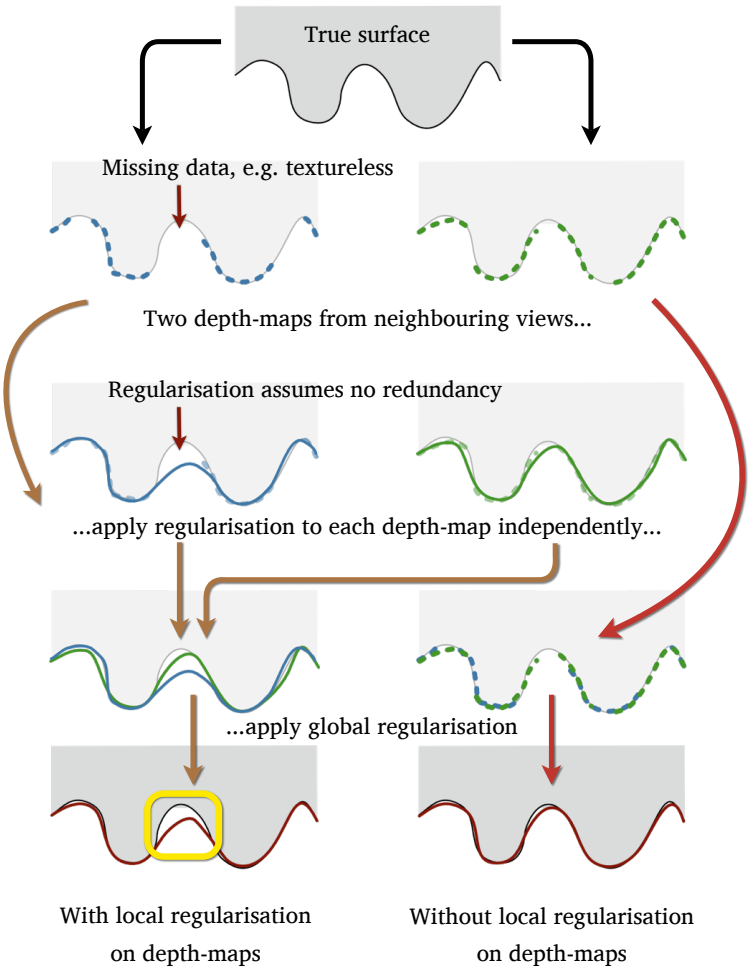


Figure 7.1: When to perform regularisation. *The use of an existing binocular dense stereo algorithm would lead to regularisation being performed on local image pairs since the algorithms are designed to be used when there is no further redundancy in the image sequence (only a pair of images). The use of spatial regularisation (e.g. in texture-less regions) will corrupt the surface produced by the final global regularisation.*

reconstruction. The latest algorithms [Scharstein and Szeliski 2002a] obtain depth-maps with greater accuracy. Since these algorithms only have pairs of images available, they can make no use of redundancy across multiple images in a data set and thus they use spatial regularisation and optimisation schemes which attempt to infer information about the depths. Whilst we also exploit a spatial regularisation constraint, we only allow the optimisation to choose from a set of discrete depths, well localised by the NCC peaks. This contrasts with methods which allow the depth of each pixel to vary continuously whilst minimising some cost function. Figure 7.1 illustrates why we shouldn't use an existing binocular dense stereo algorithm when performing MVS reconstructions.

Some of the best performing algorithms make use of an occluded state. This may be via an explicit estimation of a disparity map, for example [Sun et al. 2005] or internally as part of an optimisation routine [Criminisi et al. 2007]. We make use of the unknown state in a similar manner however we also use it recognise the other failure modes of NCC matching, discussed in § 7.3, since they are indistinguishable.

The work of [Hernández and Schmitt 2004] proposed the robust NCC matching technique which we extend in our algorithm. Outlier rejection is accomplished through redundancy in the image sequence. The works of [Goesele et al. 2006, Vogiatzis et al. 2007] have used derivatives of this technique with slight modifications, for example the inclusion of a Parzen window to filter the consensus matches in [Vogiatzis et al. 2007]. The work of [Hornung and Kobbelt 2006b] proposed a new, colour normalised super-sampling approach to correct for projective warping errors and also provided improved computation time with an efficient GPU implementation.

Recent work has demonstrated that depth-map estimation and integration paradigm may be used to produce accurate results with greatly reduced computation time [Bradley et al. 2008] or real-time [Merrell et al. 2007]. Again the reliance upon redundancy in the image sequence is paramount, for example the visibility computations of [Merrell et al. 2007].

Since our contribution affects only the depth-map estimation, the global stage may be considered separately. The works of [Curless and Levoy 1996, Zach et al. 2007] present complementary algorithms for range image integration. Here, the depth-maps produced by our algorithm would provide a suitable set of range images. The use of a volumetric graph-cut to extract the surface was proposed in [Vogiatzis et al. 2005] and extended in [Vogiatzis et al. 2007] to include the robust NCC photo-consistency. Other works have shown the graph-cut formulation to perform well as a global optimisation stage [Hornung and Kobbelt 2006a, Sinha et al. 2007].

The work of [Park and Kak 2004] uses multiple depth hypotheses as a result of reflec-

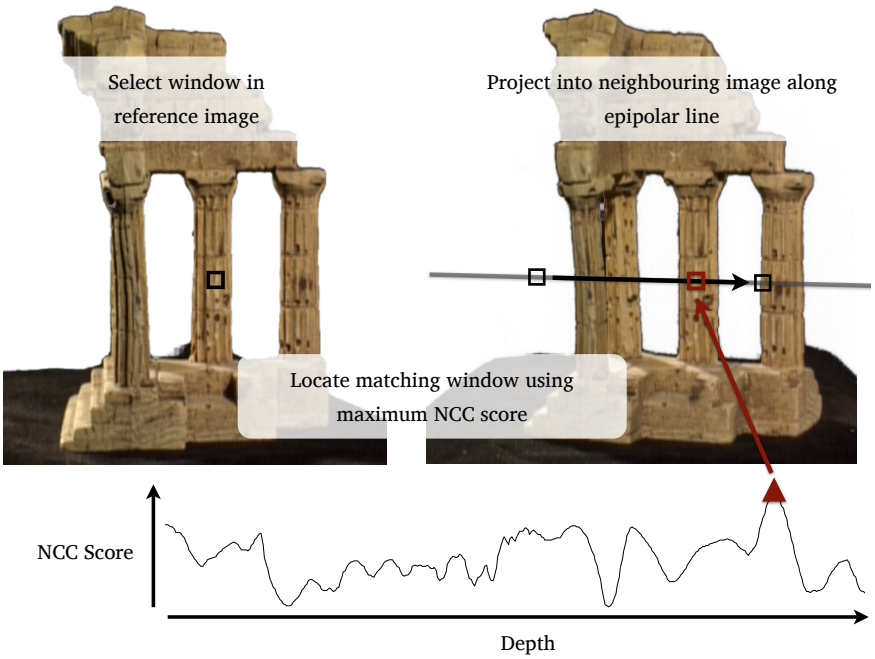


Figure 7.2: Normalised Cross-Correlation based window matching. *For each pixel in the reference image, a window, centred on the pixel, is taken. A series of windows are then taken from a neighbouring image, along the epipolar line, corresponding to different depth projections of the original pixel. The windows from the different images are then compared using the NCC metric to give a series of matching scores against depth.*

tions during the active 3D scanning of specular objects. Here a different framework, also based on spatial consistency, is used to reject false matches. The work of [Woodford et al. 2007] makes use of multiple hypotheses for the related problem of new-view synthesis. They also make use of a CRF optimisation, here using a truncated quadratic kernel, to solve their synthesis problem.

7.3 Normalised Cross-Correlation for Photo-Consistency

Normalised Cross Correlation (NCC) may be used to define an error metric for matching two windows in different images. Figure 7.2 provides an example of using NCC and epipolar geometry to perform window based matching. If we fix a pixel location in a reference image, for each possible depth away from that pixel we get a corresponding pixel in the second image. By computing the NCC between windows centred in those two pixels we can define a matching score as a function of depth for the reference pixel. We refer to this function as the *correlation curve* of the pixel. A typical correlation curve will exhibit a very sharp peak at the correct depth, and possibly a number of secondary peaks in other depths.

Figure 7.3 demonstrates the importance of the peaks of the correlation curve. Here

we use a simple implementation of the alpha-expansion algorithm of [Boykov et al. 2001] to estimate the disparity between the two images of Figure 7.3(a). The first test uses a dense cost volume with the NCC score computed at disparity intervals of a quarter of a pixel across the complete disparity range. This results in a large number of disparity levels and a time consuming optimisation. The second test uses the same cost volume but with the values set to the minimum score everywhere except at the peaks. The results of the two optimisations are nearly identical with the only discrepancies found in occluded regions. By comparing the NCC scores corresponding to the estimated disparity levels, in Figures 7.3(g) and 7.3(h), we can see that when the peak value is not on the surface (a dark blue region) the corresponding NCC value in the full cost volume has no correlation with the true surface and as such is uninformative.

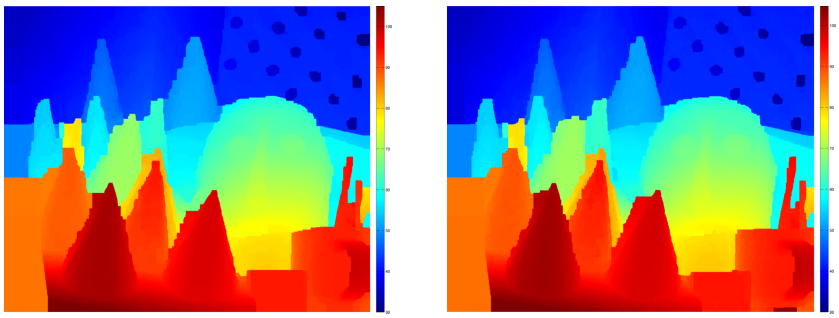
In [Hernández and Schmitt 2004] a depth-map is generated for each input image using this matching technique for neighbouring images. For each pixel a number of correlation curves are computed (using the neighbouring viewpoints) and the depth that gives rise to most peaks in those curves is selected as the depth for that pixel. See [Hernández and Schmitt 2004] or [Vogiatzis et al. 2007] for details. This process results in an independent depth estimate for each pixel. These depth estimates will unavoidably contain a significant percentage of outliers which must be dealt with in the subsequent step of [Hernández and Schmitt 2004] which is the volumetric fusion of multiple depth-maps. In data-sets with a large number of images this is is overcome by the redundancy in the depth estimates. The same surface point is expected to be covered by many different depth-maps, some of which will have the right depth estimate. In sparse data-sets however, each surface point may be seen by as few as two or three depth-maps. It is therefore crucial that outliers are minimised in the depth-map generation stage.

In this work we focus on the two most significant failure modes of NCC matching which are (1) the presence of repetitions in the texture and (2) complete matching failure due to occlusion, distortion and lack of texture. These are now described in more detail.

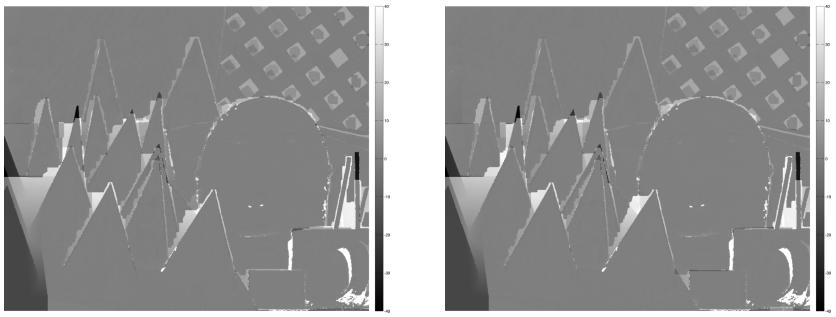
7.3.1 Repeated texture

In general, there is no guarantee that the appearance of a patch is unique across the surface of the object. This results in correlation curve peaks at incorrect depths due to repeated texture or 'false' matches (Figure 7.4). A larger window size is more likely to match uniquely to the true surface, reducing the number of false matches. However the associated peak will be broader and less well localised, reducing the accuracy of the depth estimate. The absolute value of the NCC score at a peak reflects how well the two windows match. Thus one might expect the peak with the maximum score to be the true peak. Unfortunately,

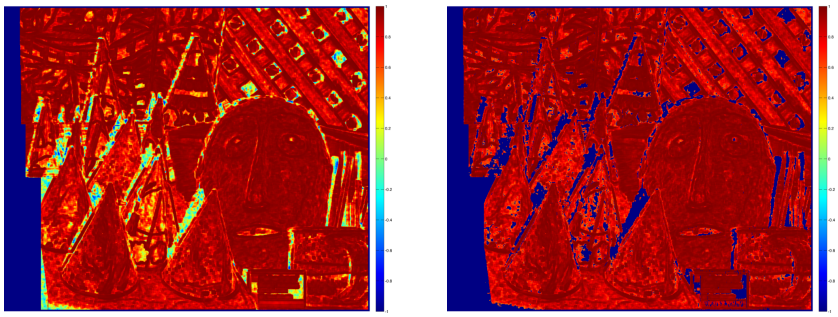
(a) Input image pair of the cones data-set **(b)** Ground-truth disparity



(c) Estimated disparity (full) **(d)** Estimated disparity (peaks)



(e) Disparity error (full) **(f)** Disparity error (peaks)



(g) NCC score (full) **(h)** NCC score (peaks)

Figure 7.3: Comparison of depth estimation using full NCC data and just the peaks. *(a) The two input images and (b) the ground truth disparity. Images on the left show results obtained using the full NCC cost volume whilst those on the right use just the peaks. A basic implementation of the alpha-expansion algorithm of [Boykov et al. 2001] was used. (c)-(d) The estimated disparity, (e)-(f) the error compared to the ground truth and (g)-(h) the NCC score for the chosen disparity.*

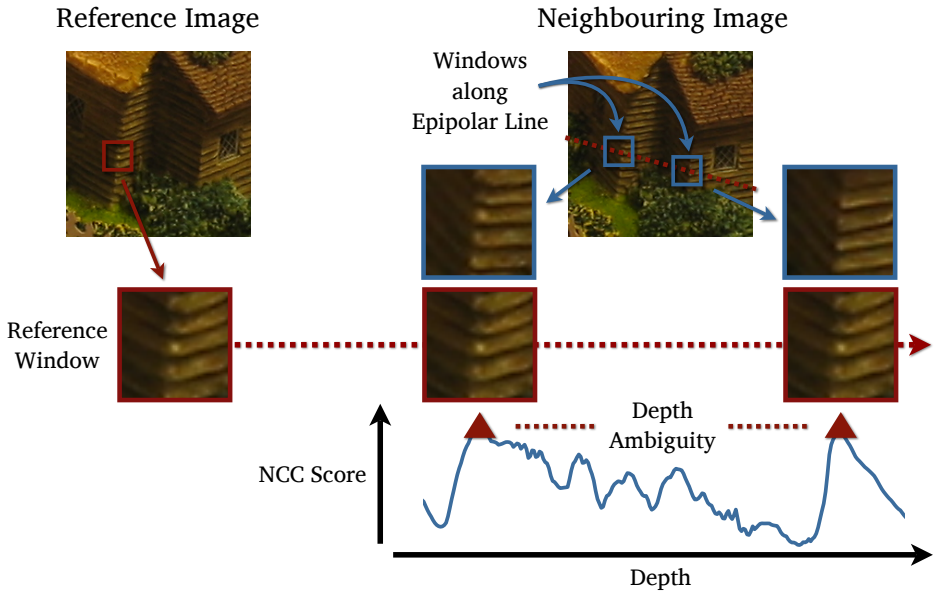


Figure 7.4: Example of ambiguity from repeated texture. *As we compare the reference window to the windows along the epipolar line there is no guarantee that only the true corresponding window will have a similar appearance. In this example we can see that the NCC score contains two large peaks, one at the true depth and the other on another edge of the house. There is also no guarantee that the true match will have a greater score (for example due to imaging noise, calibration errors or perspective distortion).*

the appearance of false matches due to repeated texture may result in false peaks having similar or even greater scores than the true surface peak (Figure 7.6(a)). To identify the correct peak, we propose to apply a spatial consistency constraint across neighbouring pixels in the depth-map. The underlying assumption is that if a peak corresponds to the true surface, the neighbouring pixels should have peaks at a similar depth. The exceptions to this are occlusion boundaries, which are however catered for under the next failure mode.

7.3.2 Matching failure

The second failure mode is comprised of occlusion errors, distorted image windows (due to slanted surfaces) and lack of texture. In all of these cases, the correlation curve will not exhibit a peak at the true depth of the surface, resulting in only false peaks. Furthermore no spatial consistency can be enforced between the pixel in question and its neighbours. In this situation we would like to acknowledge that the depth at this pixel is unknown and should therefore offer no vote for the surface location.

Figure 7.5 summaries the failure modes and proposes a consequential conjecture that the true depth is either found on spatially consistent peaks or not to be found on any peak.

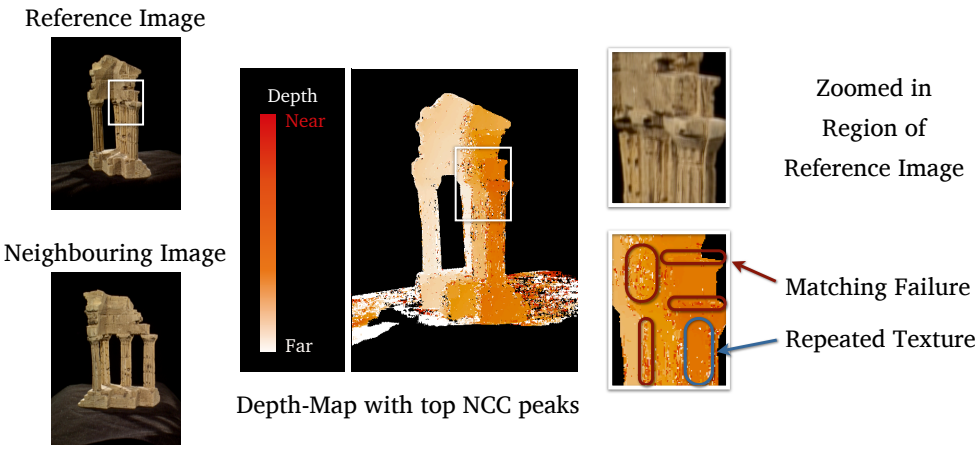


Figure 7.5: Example of the failure modes of NCC matching. *In the case of false matches due to repeated texture, for example on the columns of the temple, we will observe a peak at the true depth plus a number of false peaks. In this case the false peaks should be spatially inconsistent. In the event of a matching failure we find that there is no peak at the true depth, only false peaks that again should be inconsistent. Thus we may propose the conjecture that the true depth is either found on spatially consistent peaks or not to be found on any peak.*

This may be used to justify the following solution. In order to take account of these failure modes we propose an optimisation strategy which makes use of a discrete label Conditional Random Field (CRF). The CRF allows each pixel to choose a depth corresponding to one of the top NCC peaks which is spatially consistent with neighbouring pixels or select an *unknown* label to indicate that no such peak occurs and there is no correct depth estimate. This process means that the returned depth-map should only contain accurate depths, estimated with a high degree of certainty, and an *unknown* label for pixels which have no certain associated depth. Figure 7.6 illustrates the optimisation for a 1D example of neighbouring pixels across an occlusion boundary.

In the absence of texture, the image patch will become homogeneous and the variance will become small. Since it will be impossible to detect peaks in such a situation, and considering that the NCC computation normalises by the variance, we threshold the variance of patches such that low values (e.g. $\sigma^2 < 10^{-5}$) result in the patch being ignored and no peak can be found at this location.

7.4 Depth-Map Estimation

Our proposed algorithm estimates the depth for each pixel in the input images. It proceeds in two stages: Initially we extract a set of possible depth values for each pixel using NCC as a matching metric. We then solve a multi-label discrete CRF model which yields the depth

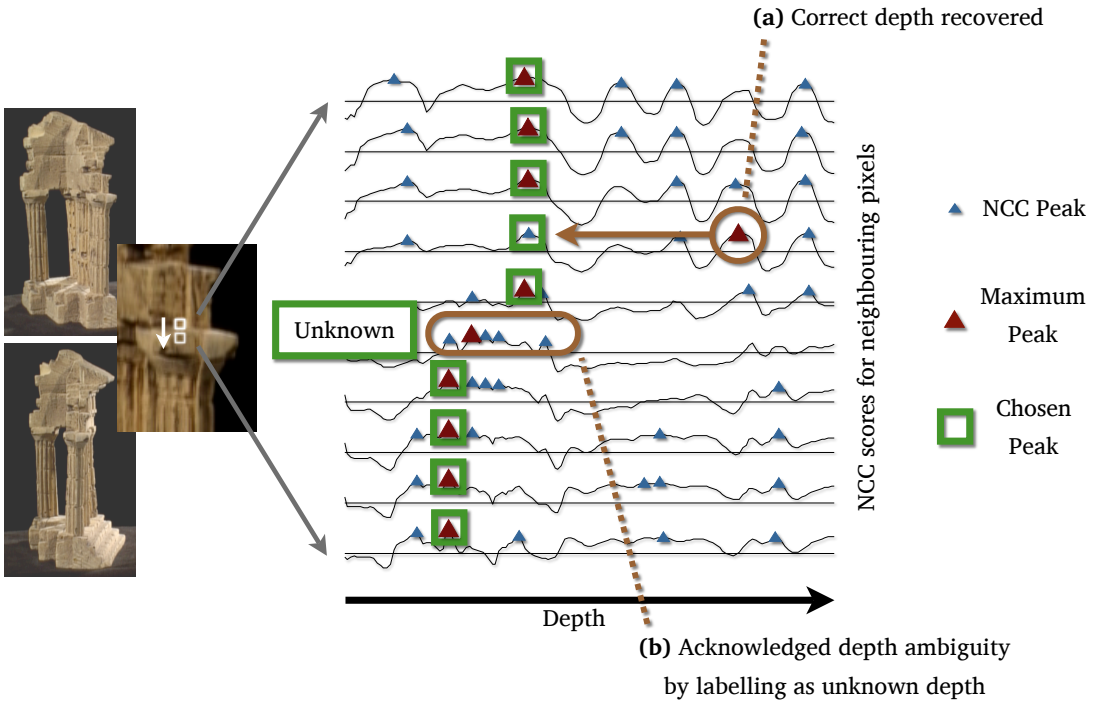


Figure 7.6: Illustration of the CRF optimisation applied to neighbouring pixels. *Existing method return the maximum peak which results in outliers in the depth estimate. The optimisation corrects an outlier to the true surface peak (a) and introduces an unknown label at the occlusion boundary (b)*

assignment for every pixel. One of the key features in this process is the inclusion of an *unknown* state in the CRF model. This state is selected when there is insufficient evidence for the correct depth to be found.

7.4.1 Candidate Depths

The input to our algorithm is a set of calibrated images $\mathcal{I}$ and the output is a set of corresponding depth-maps $\mathcal{D}$. In the following, we describe how to acquire a depth-map for a reference image $I_{\text{ref}} \in \mathcal{I}$. Let $N(I_{\text{ref}})$ denote a set of 'neighbouring' images to I_{ref}.

As proposed in § 7.3, we wish to obtain a hypothesis set of possible depths for each pixel $p_i \in I_{\text{ref}}$. Taking each pixel in turn, we project the epipolar ray into a second image $I_n \in N(I_{\text{ref}})$ and sample the NCC matching score over a depth range $\rho_i(z)$. We compute the score using a rectangular window centred at the projected image co-ordinates. One of the advantages of the multiple depth hypotheses is the ability to use a smaller matching window to provide a faster computation and improved localisation of the surface. Once we have obtained the sampled ray we store the top K peaks $\hat{\rho}_i(z_{i,k}), k \in [1, K]$ with the greatest NCC score for each pixel. Depending on the number of images available, and the width of the camera baseline, this process may be repeated for other neighbouring images.

We then continue to the optimisation stage with a set of the best K possible depths, and their corresponding NCC scores, over all neighbouring images of I_{ref}.

7.4.2 CRF Formulation

At this stage a set of candidate depths $\hat{\rho}_i(z_{i,k}), k \in [1,K]$, for each pixel p_i in the reference image I_{ref} has been assigned and we wish to determine the correct depth-map label for each pixel. As described in § 7.3, we also make use of an *unknown* state to account for the failure modes of NCC matching.

We model the problem as a discrete CRF where each each pixel has a set of up to $(K+1)$ labels. The first K labels, fewer if an insufficient number of peaks were found during the matching stage, correspond to the peaks in the NCC function and have associated depths $z_{i,k} \in \mathcal{Z}_i$ and scores $\hat{\rho}_i(z_{i,k})$. The final state is the *unknown* state $\mathcal{U}$. If the optimisation returns this state, the pixel is not assigned a depth in the final depth-map. For each pixel we therefore form an augmented label set $z'_{i,k} \in \{\mathcal{Z}_i, \mathcal{U}\}$ to include the unknown state.

The optimisation assigns a label $\bar{k}_i \in \{1 \ldots K, \mathcal{U}\}$ to each pixel p_i. The cost function to be minimised consists of unary potentials for each pixel and pairwise interactions over first order cliques. The cost of a labelling $\bar{\mathbf{k}} = \{\bar{k}_i\}$ is expressed as

$$E\left(\bar{\mathbf{k}}\right) = \sum_i \phi(\bar{k}_i) + \sum_{(i,j)} \psi(\bar{k}_i, \bar{k}_j) \tag{7.1}$$

where i denotes a pixel and (i,j) denote neighbouring pixels.

The following sections discuss the formulation of the unary potentials $\phi(\cdot)$ and pairwise interactions $\psi(\cdot,\cdot)$.

7.4.3 Unary Potentials

The unary labelling cost is derived from the NCC score of the peak. We wish to penalise peaks with a lower matching score since they are more likely to correspond to an incorrect match due to occlusion or noise. The NCC process will always return a score in the range $[-1,1]$. As in [Vogiatzis et al. 2007], we take an inverse exponential function to map this score to a positive cost in a fashion that correlates with the observation that only relatively high NCC matching scores (> 0.5) tend to correspond to correct and accurate estimates of the true surface. The more abrupt transition of the exponential function is preferable to a simple linear mapping and is found to be more robust in practice.

The unary cost for the *unknown* state is set to a constant value ϕ_{U}. This term serves two purposes. Firstly it acts as a cut-off threshold for peaks with poor NCC scores which have no pairwise support (neighbouring peaks of similar depth). This mostly accounts for

peaks which are weakly matched due to distortion or noise. Secondly it acts as a truncation on the depth disparity cost of the pairwise term. By assigning a low pairwise cost between peaks and the *unknown* state, the constant unary cost will effectively act as a threshold on the depth disparity to handle the case of an occlusion boundary. Thus the final unary term is given by

$$\phi\left(k_i = x\right) = \begin{cases} \lambda\, e^{-\beta\,\hat{\rho}_i(z_{i,x})} & x \in [1 \ldots K] \\ \phi_{\mathrm{U}} & x = \mathcal{U} \end{cases} . \tag{7.2}$$

To reiterate, the variance of the individual windows is calculated prior to comparison and patches with low variance (corresponding to homogeneous regions without texture) are ignored and no peak may be returned. In our experiments we threshold the variance at 10^{-5}.

7.4.4 Pairwise Interactions

The pairwise labelling cost is derived from the disparity in depths of neighbouring peaks. As has been previously mentioned, this term is not intended to provide a strong regularisation of the depth-map. Instead it is used to try and determine the correct peak, corresponding to the true surface location, out of the returned peaks. We observe that the correct peak may not have the maximum score. Therefore if there is strong agreement on depth between neighbouring peaks, we take this to be the true location of the surface.

When dealing with the depth disparity term we are really considering surface orientation; whether the surface normal is pointing towards or away from the camera. Under a perspective projection camera model it is therefore necessary to correct for the absolute depth of the peaks rather than simply taking the difference in depth. We perform this correction by dividing by the average depth of the two peaks. The resulting pairwise term is given by

$$\psi\left(k_i = x, k_j = y\right) = \begin{cases} 2\frac{\left|z_{i,x} - z_{j,y}\right|}{\left(z_{i,x} + z_{j,y}\right)} & x \in [1 \ldots K] \quad y \in [1 \ldots K] \\ \psi_{\mathrm{U}} & x = \mathcal{U} \quad y \in [1 \ldots K] \\ \psi_{\mathrm{U}} & x \in [1 \ldots K] \quad y = \mathcal{U} \\ 0 & x = \mathcal{U} \quad y = \mathcal{U} \end{cases} . \tag{7.3}$$

We set ψ_{U} to a small value to encourage regions with many pixels labelled as *unknown* to coalesce. This acts as a further stage of noise reduction since it prevents spurious peaks with high scores but no surrounding support from appearing in regions of occlusion.

The formulation of the pairwise term introduces a bias to 'fronto-parallel' surfaces, that is surfaces with a normal pointing directly at the reference camera (and thus neighbouring pixels share a similar depth). Whilst this may seem an unnatural prior, we must again consider the properties of the NCC window matching process. If we are looking for an estimate of the surface from a particular camera using NCC window matching we will obtain accurate estimates when the surface normal points towards the camera since the distortion of the surface in the corresponding matching windows will be minimal. As the normal points further away from the reference image not only will the distortion increase (making it harder to find a peak corresponding to the correct match) the error in the depth estimate for the window also increases. Thus whilst a bias to fronto-parallel surfaces may reject some true surface peaks, these are likely to be regions where the surface faces away from the camera and will thus produce poor depth estimates and it would be preferable to rely on other reference images to estimate the location of this portion of the surface.

7.4.5 Optimisation

To obtain the final depth-map we need to determine the optimal labelling $\hat{\mathbf{k}}$ such that

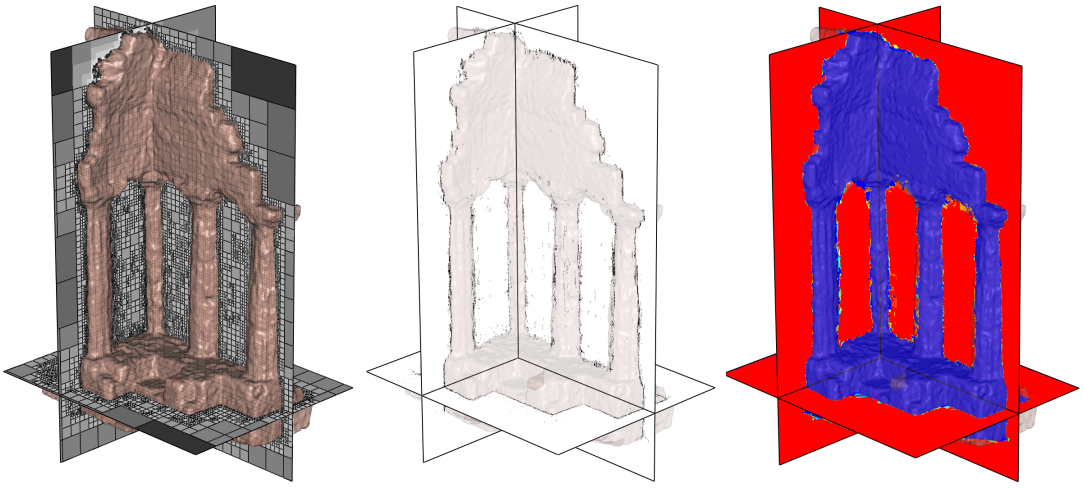
$$E(\hat{\mathbf{k}}) = \arg\min_{(\bar{\mathbf{k}})} \sum_i \phi(\bar{k}_i) + \sum_{(i,j)} \psi(\bar{k}_i, \bar{k}_j) \ . \tag{7.4}$$

Since in the general case this is an NP-hard problem we must use an approximate minimisation algorithm to achieve a solution. The most well-known techniques for solving problems of this nature are based on graph-cuts and belief propagation. Instead, we use the recently developed sequential tree-reweighted message passing algorithm, termed TRW-S, of [Kolmogorov 2006]. This has been shown to outperform belief propagation and graph-cuts in tests on stereo matching using a discrete number of disparity levels. The algorithm simultaneously minimises the energy to a local minimum whilst maximising a lower bound on the global minimum, thus if the two coincide a global minimum has been achieved. Although we are by no means guaranteed that the global minimum is attainable, the lower bound is useful for checking for convergence and evaluating the performance of the algorithm.

7.5 Extension to Multi-View Stereo Framework

As discussed in § 4.7, the detailed evaluation of [Seitz et al. 2006] demonstrates that volumetric methods display state-of-the-art performance both in terms of accuracy and completeness. Some of the most successful create a 3D cost field within a volume and the reconstruction task is then to extract the optimal surface from this volume. Algorithms developed for segmentation problems are commonly used to extract the surface.

In order to evaluate the improvement to multi-view stereo we combined our depth-map estimation with a modified version of the volumetric regularisation framework of [Vogiatzis et al. 2007]. This method uses a volumetric graph-cut to recover the surface from a volume subdivided into a set of discrete voxels as in Chapter 5, Figure 5.6. Each voxel becomes a node in a 3D binary Random Field where the voxel must be labelled as inside or outside the object with the surface recovered as the boundary between the two.



(a) Multi-resolution octree **(b)** Photo-consistency cost volume **(c)** Visibility cost volume

Figure 7.7: Example of the cost volumes used for surface recovery. *The cost volumes for the sparse temple sequence (16 images) [Seitz et al. 2006] are shown with the resulting surface rendered transparently. (a) The data-dependent multi-resolution grid used to reduce the storage requirements and the size of the graph. The octree structure is shown from dark grey at the root node to white at the leaves. (b) The photo-consistency cost volume obtained by combining all 16 depth-maps. White indicates regions of low photo-consistency and black indicates regions of high photo-consistency. The subdivision of the octree is dependent on this term so fine detail voxels are only present close to the surface. (c) The visibility cost volume obtained as the probabilistic log-evidence of the voxel being visible from at least one camera and therefore empty. Red indicates the evidence is in favour of visible (outside the surface) and blue the evidence is in favour of not visible (assumed to be inside the surface).*

The energy formulation allows for two terms in the cost function and Figure 7.7 provides an example of these for the temple dataset. The first is the unary inside/outside labelling cost, shown in Figure 7.7(c). This encodes the likelihood that a particular voxel is part of the object or empty space. The recent work of [Hernández et al. 2007b] shows how depth-maps may be used to evaluate a probabilistic visibility measure for each voxel in the volume. This term may be used to estimate whether or not the voxel in question resides in empty space and is therefore visible from the cameras. From this it is possible to derive an appropriate cost for the unary term related to the likelihood of visibility.

The second term is the pairwise discontinuity cost, shown in Figure 7.7(b). This term represents the likelihood that the surface boundary lies between two neighbouring voxels and therefore contains the photo-consistency information from all the depth-maps. This term may be derived directly from the individual depth-maps by projecting each of them into the volume and summing the information.

In [Boykov 2003] the authors show that the energy cost is a discrete approximation to the sum of a weighted surface area of the boundary (the pairwise terms) and a weighted volume of the object (the unary terms) as

$$E(S) \quad = \quad \int_S \rho\left(\mathbf{x}\right) \mathrm{d}A + \int_{\mathrm{vol}(S)} \sigma\left(\mathbf{x}\right) \mathrm{d}V \ . \tag{7.5}$$

where we have S as the boundary of the recovered surface containing a volume $\mathrm{vol}(S)$ with $\rho\left(\mathbf{x}\right)$ as the photo-consistency volume and $\sigma\left(\mathbf{x}\right)$ as the visibility volume. Note that values of $\rho\left(\mathbf{x}\right)$ are low in photo-consistent regions and values of $\sigma\left(\mathbf{x}\right)$ are low in regions that are not visible and therefore assumed to be inside the object. Thus the surface S with the minimum energy in (7.5) should correspond to a minimal surface that passes through photo-consistent regions and contains a volume that is not visible by any of the cameras.

This framework is ideal for use with our depth-maps since it provides global regularisation using all the available data, a key advantage of our approach. Rather than perform regularisation on individual depth-maps to recover uncertain regions, we only return depths with a high degree of confidence associated with them. Thus other depth-maps may be able to fill in the areas where a particular depth-map is uncertain. In the event that there are still regions of the surface which are not determined precisely by any of the depth-maps, the regularisation should be performed by a global method which takes into account the data from all the depth-maps rather than an amalgamation of estimates from individual depth-maps.

As suggested in [Vogiatzis et al. 2007], memory requirements and computation time for large voxel grids may be reduced by using a multi-resolution subdivision of the volume using an octree as shown in Figure 7.7(a). The subdivision of the volume is data-dependent and driven by the photo-consistency term such that fine scale voxels (at the leaves of the octree) are only found near to the surface and regions far from the surface are more coarsely divided. As an example of the saving in graph size, using 400^3 voxels for the temple surface of Figure 7.7(a) would have required approximately 64×10^6 nodes and 192×10^6 edges whereas a 10-level octree results in 3×10^6 nodes and 11×10^6 edges. Since the subdivision is data dependent it is impossible to anticipate precise savings but these results are typical for the image sequences we have used.

7.5.1 Depth-Map Acquisition

The first stage of the reconstruction process is to acquire the depths maps. Our method is to select an image and project rays into the nearest neighbouring images in a sequential process. We maintain a cumulative store of the K top scoring NCC peaks for each pixel. This provides an even greater degree of robustness against occlusion than the technique of [Vogiatzis et al. 2007] and is easier to implement in a parallel environment such as a GPU. Rather than requiring peaks from multiple images to fall in the same location, we only have to observe accurately a surface location in a single pair of images and rely on the surrounding support of peaks to identify the correct peaks. The speed of the depth-map computation maybe increased by using the object silhouettes to avoid performing NCC matching calculations in regions outside the possible surface locations. Extraction of silhouettes for multi-view stereo may be performed automatically or interactively using Chapter 5 or Chapter 6.

7.5.2 Surface Recovery

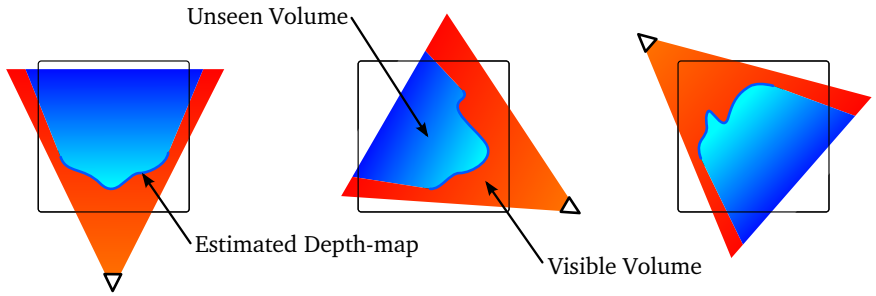
Integrating our depth-maps with the framework of [Vogiatzis et al. 2007] and [Hernández et al. 2007b] is a simple and elegant process and Figure 7.8 provides an illustration. For the visibility volume we may project the same probability of visibility along each ray as [Hernández et al. 2007b] when we have a known depth. For pixels labelled as *unknown* we simply project a likelihood of 0.5 to indicate that this pixel provides no information about visibility. For the photo-consistency cost we adopt a 'binning' approach. For each voxel in the volume we take the sum of the projected depths of all the pixels in all the depth-maps which fall inside the voxel, weighted by their NCC scores. If a pixel is labelled as *unknown* then it plays no part in the photo-consistency cost.

The final optimisation follows in the same manner as [Vogiatzis et al. 2007] with the graph-cut used to segment the volume. The iso-surface is extracted and smoothed using a snake [Hernández and Schmitt 2004] to perform 'intelligent' smoothing making use of the photo-consistency volume.

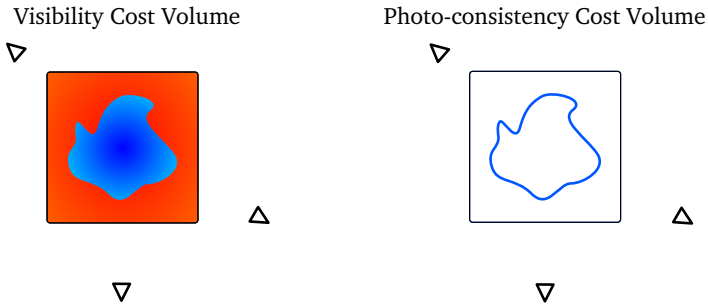
7.6 Experiments

7.6.1 Implementation

To improve the computation time for our depth-maps we perform the NCC matching by taking advantage of the parallel processing and texture facilities of the GPU of modern graphics cards. The GPU code improves performance by up to an order of magnitude



(a) Individual depth-maps



(b) Resulting cost volumes

Figure 7.8: Illustration of combining depth-maps to form the final cost volumes. *(a) The individual depth-maps contain both the estimate of the surface location and information about visible regions of the volume (which should contain empty space). (b) The information from all the depth-maps is combined to produce the visibility cost volume, as the probabilistic evidence that a voxel is visible from at least one camera, and the photo-consistency volume which is the sum of all the NCC peaks that are found in each voxel, weighted by the NCC score.*

depending on the window size. One of the advantages of our method is the ability to use small windows which result in greater precision of the surface location but which introduce a significant amount of noise which will adversely affect many of the existing techniques. The use of the smaller window also results in a greater saving in computational efficiency since the GPU offers improved performance with small kernels.

For the optimisation of the discrete CRF for the depth-map we use the TRW-S implementation of Kolmogorov [Kolmogorov 2006]. We also use Kolmogorov's implementation of the graph-cut algorithm [Boykov and Kolmogorov 2004]. Our implementation, running on a 3.0 GHz machine with an nVidia Quadro graphics card, can evaluate 900 NCC depth slices in 20 seconds for the temple sequence (image resolution 640×480). The TRW-S optimisation has a typical run time of 20 seconds for the same images. The final volumetric graph-cut typically runs in under 5 minutes for a 350^3 voxel array.

For all the experiments we used the following parameter values: $\beta = 1$, $\lambda = 1$, $\phi_U =$

0.04 and $\psi_U = 0.002$. We used an NCC window size of 5×5.

7.6.2 Depth-Maps

Figure 7.9 illustrates the improvement of our method over the voting schemes of [Hernández and Schmitt 2004, Vogiatzis et al. 2007]. Figure 7.9(b) shows the depth that would be determined by simply taking the NCC peak with the greatest score. Our method, implemented here with $K = 9$ peaks, is able to select the peak corresponding to true surface peak from the ranked candidate peaks and Figure 7.9(d) illustrates that a significant proportion of the true surface peaks are not the absolute maximum. We also observe that pixels are correctly labelled with the *unknown* state along occlusion boundaries and along areas such as the back wall of the temple and edges of the pillars where the surface normal is oriented away from the camera. Looking at the rendering of this depth-map and its neighbour, Figure 7.9(e-g), we can observe that very few erroneous depths are recovered and we observe that the combination of the two depths maps align and complement each other rather than attempting to fill in the holes on the individual depth-maps which would impact the subsequent multi-view stereo global optimisation.

Figure 7.10 shows the results on the 'cones' data-set which forms part of the standard dense stereo evaluations images and consists of a single stereo pair with the left image shown. Our depth-map again shows a high degree of detail on textured surfaces and we correctly identify occlusion boundaries with the *unknown* state. Further more the algorithm also correctly textures the failure modes of NCC by returning the *unknown* state in texture-less regions where the matching fails to localise accurately the surface.

7.6.3 Multi-View Stereo Evaluation

In order to evaluate the improvement of our depth-maps to multi-view stereo we ran our algorithm on the standard evaluation 'temple' data-set. Table 7.1 provides the accuracy and completeness measures of [Seitz et al. 2006] against the ground-truth data for the object. In terms of both accuracy and completeness our results provide a significant improvement in both the sparse ring and ring data-sets. In particular we observe that the results for the sparse ring offer greater accuracy than the other algorithms [Scharstein and Szeliski 2002a] running on the ring sequence (3 times as many images) with the exception of [Hernández and Schmitt 2004]. In the case of [Bradley et al. 2008] the accuracy is improved by returning an incomplete mesh, only regions that are reliably observed in multiple views are returned. This increases the accuracy, since no global regularisation is performed, at the expense of completeness. It should be noted that this method is not robust since increasing the number of images, the ring data-set, results in a decrease in

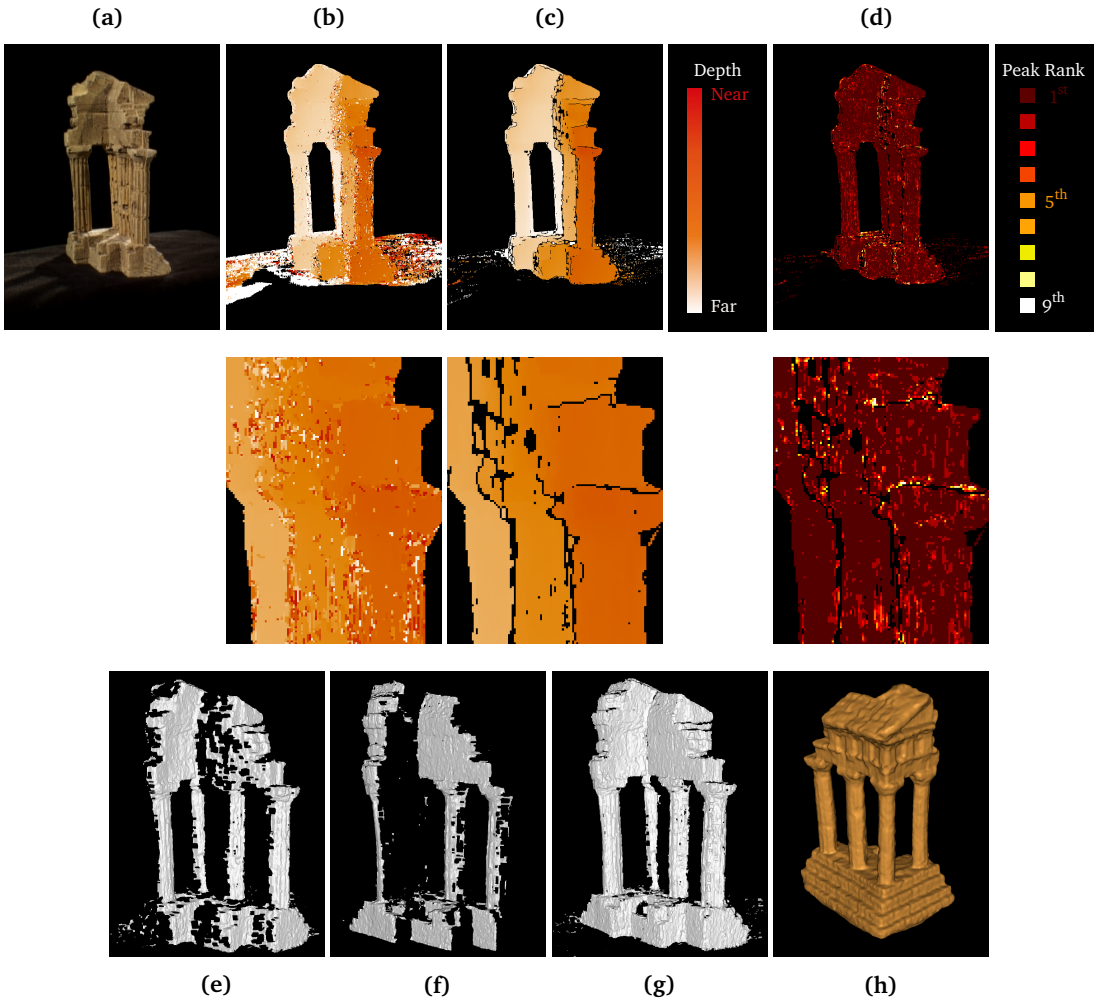
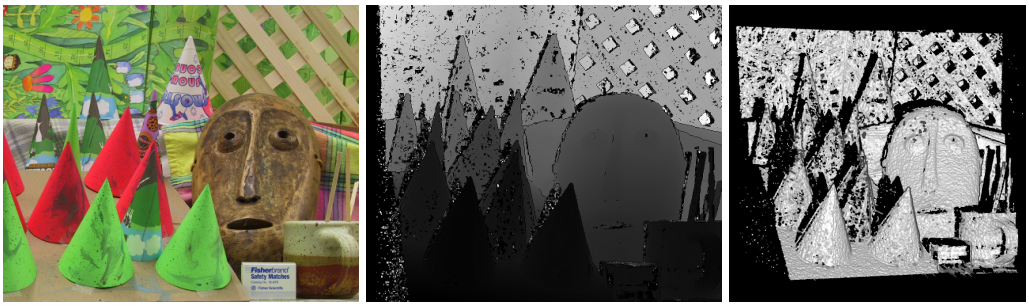


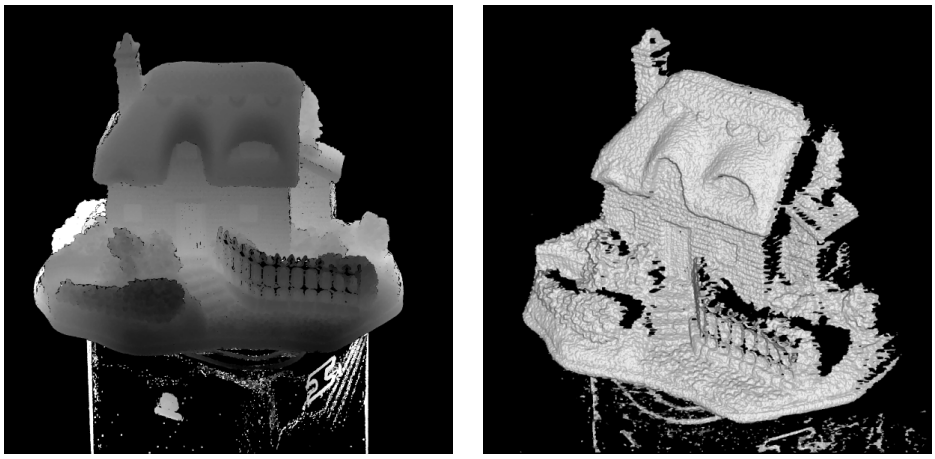
Figure 7.9: Results of the depth-map estimation algorithm. *Two neighbouring images are combined with the reference image (a). If we simply took the NCC peak with the maximum score, as in [Hernández and Schmitt 2004], we would obtain (b). The result of our algorithm (c) shows a significant reduction in noise. We have corrected noisy estimates of the surface and the* unknown *state has also been used to denote clearly occlusion boundaries and remove poorly matched regions. The number of the correct surface peak returned, ranked by NCC score, is displayed in (d) where dark red indicates the peak with the greatest score. The rendered depth-map is shown in (e) along with the neighbouring depth-map (f) with (g) showing the two superimposed. The final reconstruction (h) for the sparse temple sequence (16 images) of [Seitz et al. 2006]*



(a) Left image **(b)** Recovered depth-map **(c)** Rendered depth-map

Figure 7.10: Single view stereo results for the 'Cones' data set. *The left image of the stereo pair is shown in (a) with the recovered depth-map in (b), rendered in (c)*

(a) Input images



(b) Recovered depth-map **(c)** Rendered depth-map

Figure 7.11: Depth-map obtained from only three images of a model house. *(a) The reference image and its neighbouring pair of images. (b) The recovered depth-map, rendered in (c). As well as achieving a high degree of accuracy on surface detail our algorithm has correctly recovered the occlusion boundaries and removed outlying depth estimates.*

accuracy. The most interesting comparison is with [Vogiatzis et al. 2007] since this is the basis for the MVS algorithm that we improve with our depth-map estimation. As expected, we see the greatest improvement with the sparse data set. It is also interesting that we still provide an improvement with the full data set. This shows that even with a large number of redundant depth-maps, and consequently the improved signal-to-noise ratio expected, removing depth outliers still improves the final estimate of the surface

	Accuracy / Completeness		
Method	Full (312 images)	Ring (47 images)	SparseRing (16 images)
[Hernández and Schmitt 2004]	0.36mm / 99.7%	0.52mm / 99.5%	0.75mm / 95.3%
[Goesele et al. 2006]	0.42mm / 98.0%	0.61mm / 86.2%	0.87mm / 56.6%
[Hornung and Kobbelt 2006a]	0.58mm / 98.7%	–	–
[Pons et al. 2007]	–	0.60mm / 99.5%	0.90mm / 95.4%
[Furukawa and Pons 2007]	0.54mm / 99.3%	0.55mm / 99.1%	0.62mm / 99.2%
[Bradley et al. 2008]	–	0.57mm / 98.1%	0.48mm / 93.7%
[Vogiatzis et al. 2005]	1.07mm / 90.7%	0.76mm / 96.2%	2.77mm / 79.4%
[Vogiatzis et al. 2007]	0.50mm / 98.4%	0.64mm / 99.2%	0.69mm / 96.9%
Our Results	**0.41mm / 99.9%**	**0.48mm / 99.4%**	**0.53mm / 98.6%**

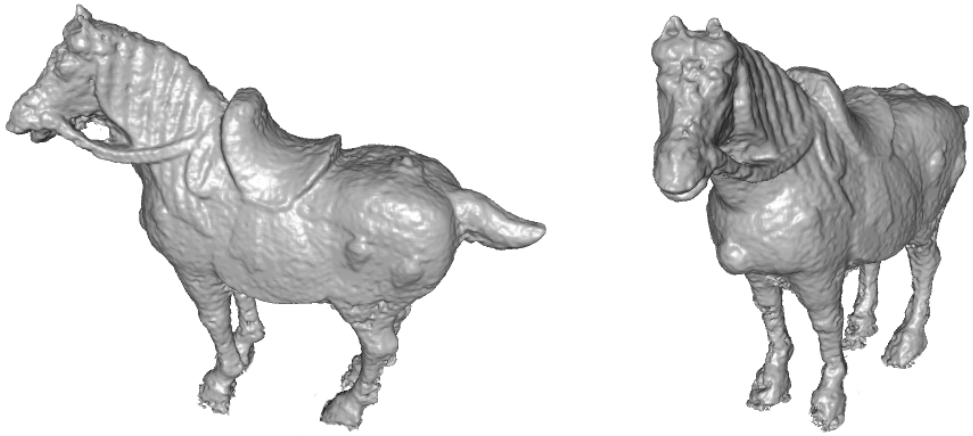
Table 7.1: Comparison with the ground truth evaluation used in [Seitz et al. 2006]. *The table features the latest reconstruction algorithms and identifies the state-of-the-art performance of our algorithm, both in terms of accuracy and completeness.*

Figure 7.12 shows a result obtained using the full MVS system on images of a horse sculpture in Wolfson College. The algorithm makes use of the silhouettes of Figure 6.6, obtained using the clustering segmentation algorithm. The texture map is obtained using the method described in [Hernández 2004].

7.7 Discussion

The experimental results indicate that we can improve the results of a state-of-the-art reconstruction algorithm, [Vogiatzis et al. 2007], when targeting sparse data sets. This increase in accuracy is due to the rejection of outlying depth estimates by accounting for the failure modes of NCC window matching. Using the multiple hypotheses followed by

(a) 3 of 36 input images



(b) 3D model from novel viewpoints

(c) 3D model with texture

Figure 7.12: Reconstruction of a horse. *(a) 36 images were used to recover (b) a 3D model, also shown with texture (c), using the full MVS system of [Campbell et al. 2008].*

filtering structure allows us to use smaller NCC matching windows, which would result in reduced performance when taking, say, the top NCC score, to more accurately localise the surface.

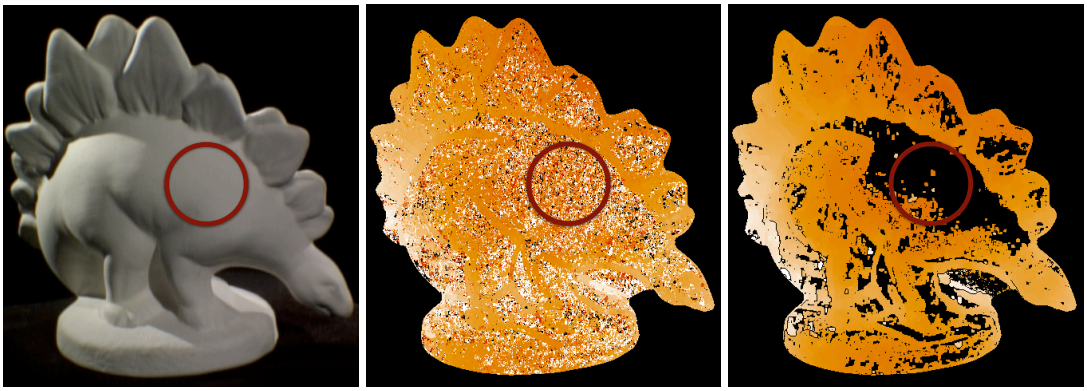
When looking at the number of peaks, K, we can see that we would wish to choose as many as possible. Unfortunately memory limitations for the optimisation algorithm will limit the actual number we are able to take. Whilst the memory requirements scale linearly with K they also scale linearly with the number of pixels (an approximation assuming a constant aspect ratio for the images). This means that moving to high resolution images will have a limiting effect on the number of peaks, in practice we are limited to around $K = 9$ for 6 MP images to fit in 4 GB of RAM. However, we observe, for example in Figure 7.9(d), that the number of peaks returned decreases with the ranking of the peaks. Thus we are not losing many true surface estimates by artificially reducing K. This leads to an observed sub-linear increase in computational complexity with K, since the lower order peaks are rarely required, however this is not guaranteed and very few bounds on convergence can be made when using TRW-S with a discrete random field with a large set of possible labels.

In ideal circumstances we would like to keep all K peaks when performing the global optimisation since by selecting a single peak for each depth-map, making an early decision, we are throwing away data. The problem with keeping all K peaks, for example computing a true marginal rather than taking a MAP estimate, is that it leaves the global optimiser with a very difficult optimisation problem with a large number of local minima. In the case of sparse image sequences, we might also be in a situation where there is insufficient redundancy, in terms of the number of views, to resolve this ambiguity. As improvements are made in non-linear optimisation it may be possible to keep all K peaks into the global optimisation stage and we identify this as an avenue for future work.

7.7.1 Limitations

The main limitations of the method are failure in the absence of texture, which is common to all NCC based methods, and also the algorithm can only reject repeated texture to a certain extent.

Figure 7.13 illustrates the failure due to lack of texture. We should note, however, that the algorithm has returned the unknown state rather than the large number of outliers produced by taking the maximum peak so we have certainly improved the result. From this we can infer that NCC window matching will always fail on textureless regions however it is possible to detect and remove the majority of these failed pixels, which will be returned with the unknown label, using the algorithm presented.



(a) Reference image **(b)** Depth of top NCC score **(c)** Result of our algorithm

Figure 7.13: Failure to recover the surface in the absence of texture. *The lack to texture in the shadow region of the input image (a) leads to failure in the NCC matching as shown by the large number of outliers obtained by taking the maximum peak (b). The algorithm fails to recover the surface in the shadow but succeeds in removing almost all the outliers (c).*

(a) Depth of top NCC score **(b)** Result with smaller ψ_U **(c)** Result with larger ψ_U

Figure 7.14: Result of increasing the coalescing parameter. *(a) The estimated depth taking the top NCC score. As we increase the coalescing parameter ψ_U from (b) to (c) we require the depth to be consistent over a greater spatial area and reduce errors due to consistent repeated texture at the possible expense of labelling some correct depths as unknown.*

If we have a large area of repeated texture then the incorrect peaks may also be spatially consistent and thus the algorithm will be unable to guarantee that the correct peaks are returned. To a certain extent, we can combat this effect by adjusting the coalescing parameter ψ_{U}. Figure 7.14 demonstrates this effect on the fountain sequence shown in Figure 1.8. Whilst we have presented results with a default set of parameters it may the case that performance can be improved for specific sequences by adjusting the parameter ψ_{U}; this is a topic for further study.

CHAPTER 8

Conclusion and Future Work

8.1 Automatic Reconstruction Results

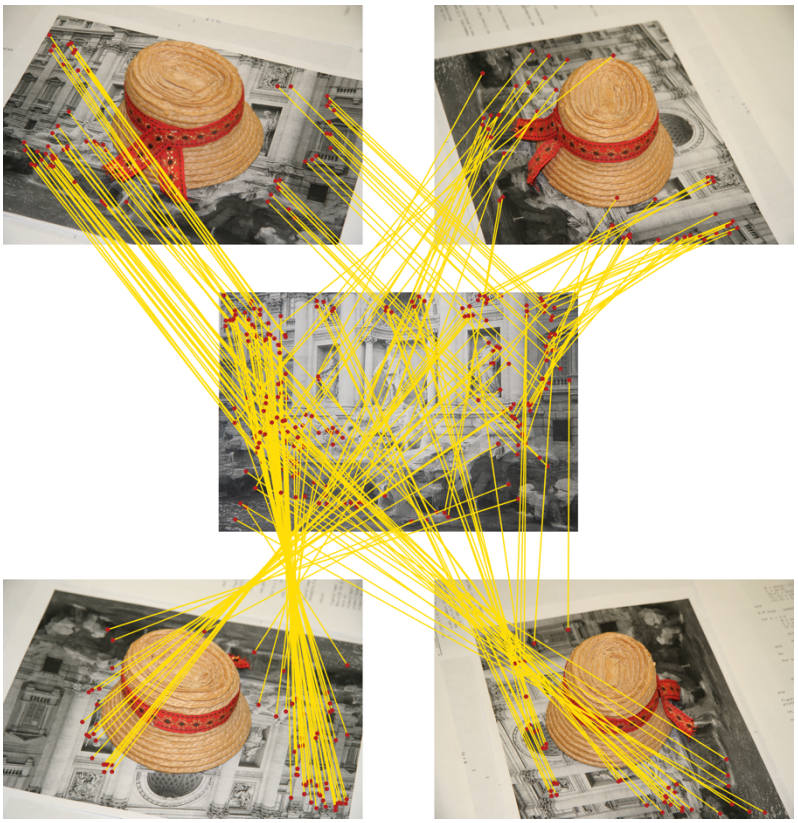
We begin our conclusion by providing some examples of our practical reconstruction system to illustrate our combined contributions. The hat sequence consists of 30 images of a hat, some examples are given in Figure 8.2(a), taken by hand at a resolution of 4 MP . The hat is located upon a photo of the Trevi fountain that provides a suitable textured plane for the automatic calibration to proceed, shown in Figure 8.1, as described in Chapter 3.

Figure 8.2 shows the segmentation results for the sequence obtained using the automatic algorithm of Chapter 5. The silhouettes and visual hulls shown were obtained after 3 iterations. In addition to the segmentation process being performed without user input, our planar calibration procedure is also fully automatic and thus no user interaction, other than capturing the images themselves, was required to complete this segmentation. Figure 8.2(e) shows a 3D model reconstruction of the hat using an implementation of [Hernández and Schmitt 2004]. The object texture is also automatically recovered and may be used to perform the texture mapping shown in Figure 8.2(f).

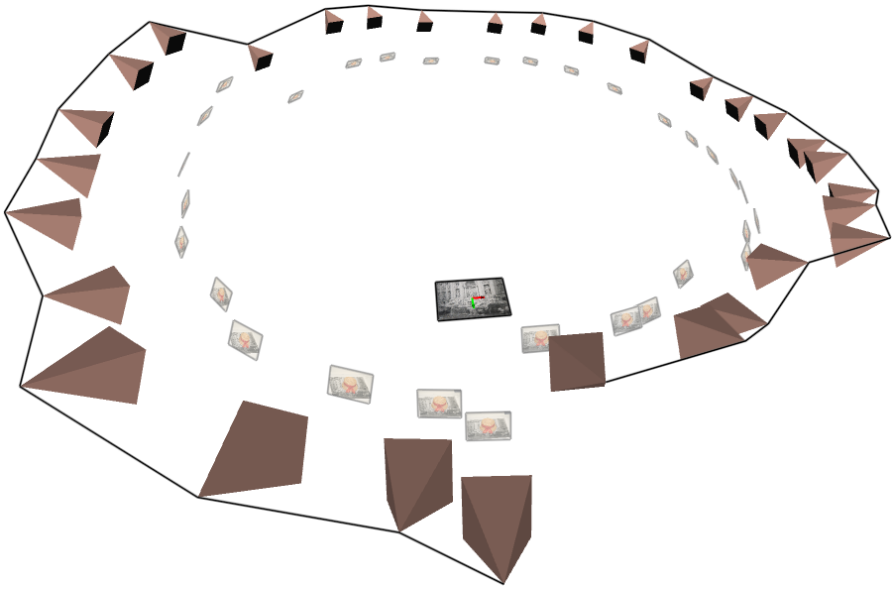
The object colour model probabilities, Figure 8.2(b), are observed to have correctly captured all the object colours resulting in the accurate visual hull of Figure 8.2(d) with corresponding silhouettes Figure 8.2(c).

The system used to provide the results was based on an Intel Xeon Processor with 4 GB of RAM running at 2.6 GHz. The implementation required 5 minutes per iteration for the hat sequence with a voxel array size of 230^3. The complexity scales linearly with the number of pixels and the majority of the time is spent fitting and evaluating the colour mixture models.

Figure 8.3 shows the segmentation and reconstruction process for a sequence of images of a hand. Again, automatic calibration was used, this time a newspaper provided a suit-



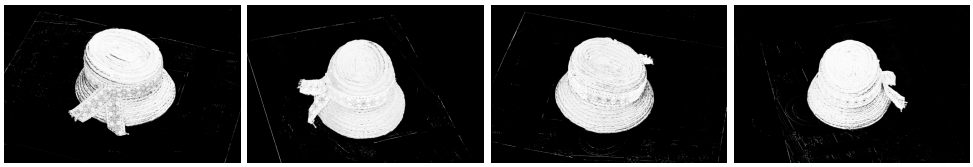
(a) Correspondences used for bundle adjustment



(b) The final camera calibration

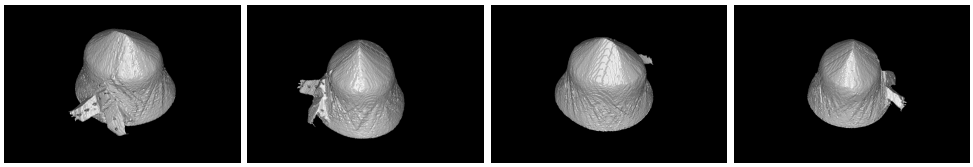
Figure 8.1: Automatic calibration of the hat sequence. *(a) The automatic homography estimation yields a set of correspondences that are used to perform bundle adjustment. (b) The final camera calibration obtained after bundle adjustment.*

(a) Input images

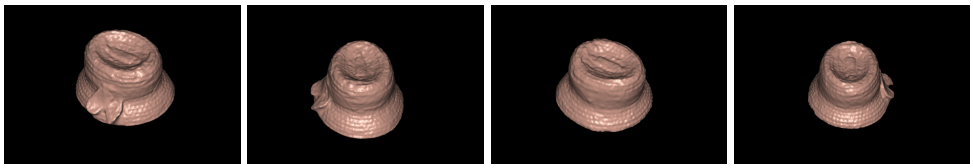


(b) Converged object likelihoods

(c) Converged silhouettes



(d) Converged visual hull

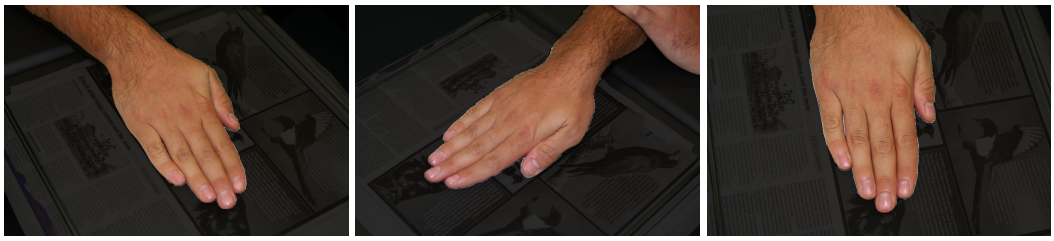


(e) Automatic reconstruction

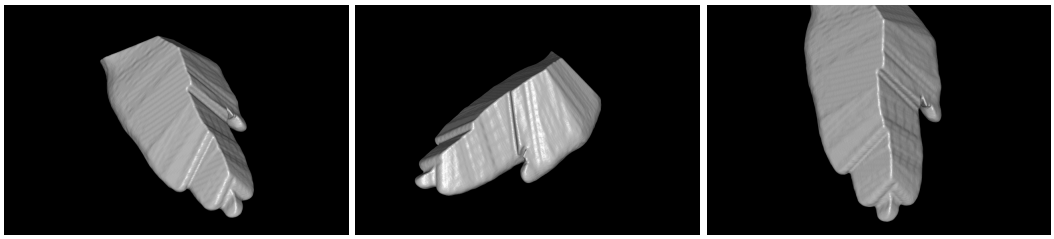
(f) Reconstruction with texture mapping

Figure 8.2: Automatic calibration, segmentation and reconstruction of the hat sequence. *(a) The input images (4 of 30 shown) and the automatically acquired calibration allow the colour models to converge to capture all the object colours (b) and the graph-cut at convergence produces the correct silhouettes (c) and visual hull (d) which may be used to produce a reconstruction of the object (e) and also perform texture mapping (f).*

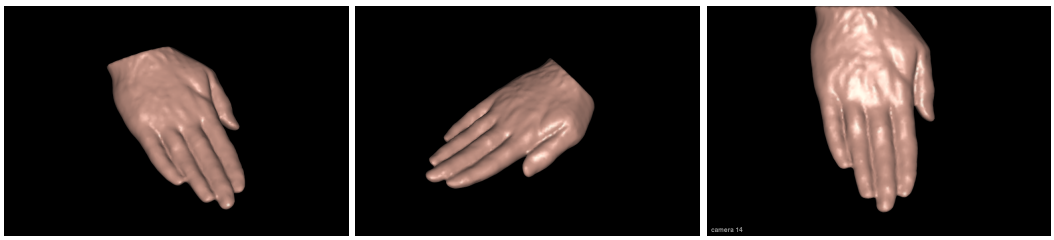
(a) Input images



(b) Silhouettes



(c) Visual hull



(d) 3D reconstruction

Figure 8.3: Automatic calibration, segmentation and reconstruction of the hand sequence. *(a) Input images (3 of 18 shown) taken with a handheld camera and automatically calibrated using the method of Chapter 3. (b) Silhouettes obtained automatically using the segmentation algorithm of Chapter 5 and the corresponding visual hull (c). (d) The final model was obtained automatically using the algorithm of Chapter 7.*

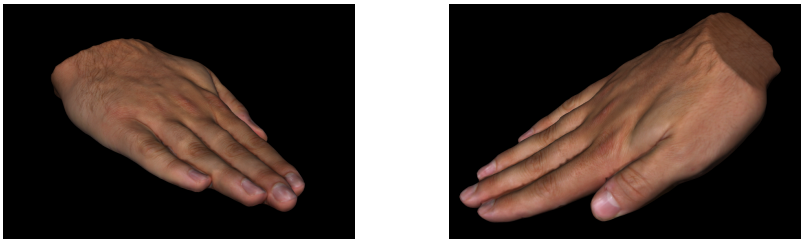


Figure 8.4: Texture mapped reconstruction of the hand. *Recovery of the texture map is also an automatic process and allows the hand to be rendered from novel viewpoints not present in the input image sequence.*

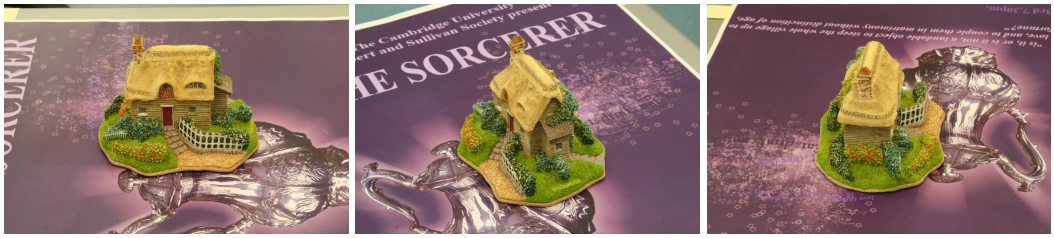
able textured plane. Slight modifications were made to the algorithm to cut the voxel array since the arm extends out of the image volume and the segmentation algorithm is designed for objects to be completely contained by the visible volume of the cameras. Automatic reconstruction was performed using the algorithm of Chapter 7 to produce the final surface. Figure 8.4 shows some novel views of the texture mapped hand reconstruction.

Figure 8.5 shows the results of segmentation and reconstruction for a sequence of images of a toy house with a theatre poster providing the textured plane for calibration. Again, the system made use of Chapters 3, 5 and 7. All the parameters were set to the values described in the experiments section of each of the corresponding chapters.

For sequences with more challenging background clutter, in particular where the object and background are not separable in colour space, the interactive segmentation algorithm of Chapter 6 may be used instead of the automatic method of Chapter 5. Figure 8.6 shows a reconstruction result for a horse statue in Wolfson College. The entire process may be performed in under 40 minutes on a quad-core Intel Xeon Processor with 8 GB of RAM running at 2.6 GHz and requires less than a minute of time from the user for the interactive segmentation.

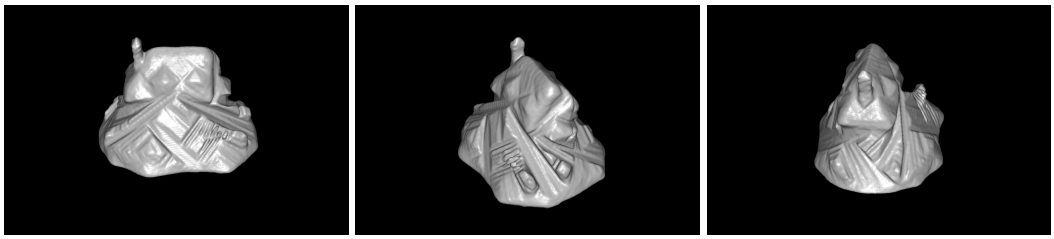
8.2 Conclusion

This work has demonstrated that computer vision algorithms can be developed to not only recover 3D shape from visual information but do so robustly in an unguided and autonomous fashion. These advances make it possible to extend the use of these technologies from the laboratory to the world at large. This opens the way to make use of these algorithms in many of the applications outlined in § 1.1. Furthermore, it allows the people who will benefit from the technology to use the systems directly without having to call on computer vision experts. The only equipment required is a digital camera and thus the technology is inexpensive considering the mass production, and consequential prevalence,

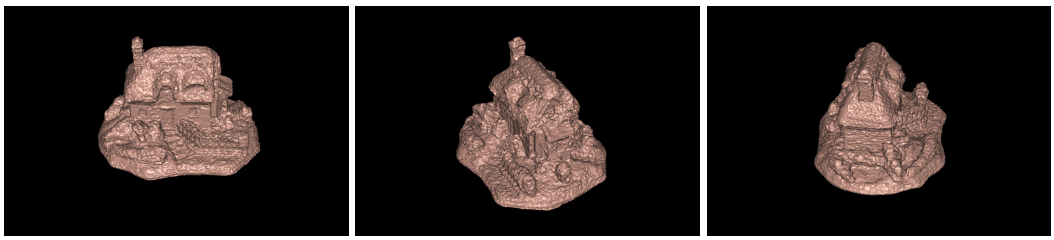


(a) Input images

(b) Silhouettes



(c) Visual hull

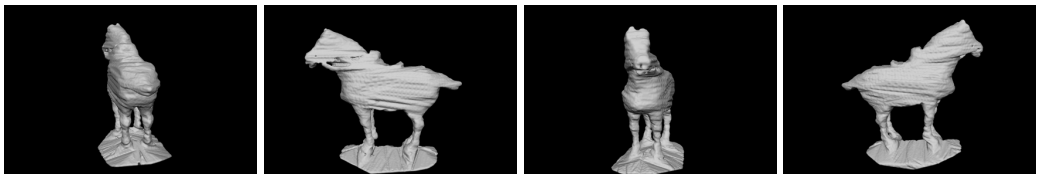


(d) 3D reconstruction

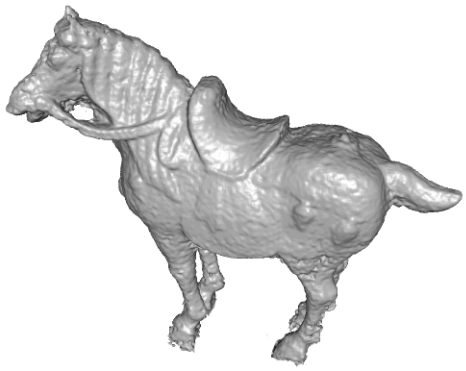
Figure 8.5: Automatic calibration, segmentation and reconstruction of the house sequence. *(a) Input images (3 of 38 shown) taken with a handheld camera and automatically calibrated using the method of Chapter 3. (b) Silhouettes obtained automatically using the segmentation algorithm of Chapter 5 and the corresponding visual hull (c). (d) The final model was obtained automatically using the algorithm of Chapter 7.*

(a) Input images

(b) Silhouettes



(c) Visual Hull



(d) 3D model

(e) 3D model with texture

Figure 8.6: Automatic calibration, interactive segmentation and automatic reconstruction of the horse sequence. *(a) Input images (4 of 36 shown) taken with a handheld camera and automatically calibrated using the method of [Snavely et al. 2006]. (b) Silhouettes obtained using the interactive segmentation algorithm of Chapter 6, requiring less than a minute of time from the user, and the corresponding visual hull (c). (d) The final model was obtained automatically using the algorithm of Chapter 7 and textured mapped (e).*

of cameras in modern society. This is an important milestone in the adoption of computer vision technologies.

Automatic Segmentation Algorithm

The results of our experiments indicate that our method confers considerable advantages for automatic object segmentation over the best 2D algorithms. The volumetric approach makes use of the silhouette coherency constraint to perform the segmentation in 3D, allowing the object to be segmented in all images simultaneously. This allows us to combine the learnt colour model, containing information from all views of the object, with a 3D shape prior to produce a more accurate result. We have also shown that it is possible to exploit a fixation constraint in order to initialise an iterative estimation algorithm to converge to the visual hull of an object observed in multiple views and thus avoid the need for any interaction from the user, thus making the whole process automatic.

Interactive Segmentation Algorithm

We have shown that the task of segmenting a calibrated image sequence may be elegantly posed as a graph-clustering problem and our results have demonstrated that this may be solved tractably to produce spatially meaningful clusters across the images. This allows us to present a user with an easy to use, interactive segmentation process that minimises the user's input whilst performing segmentations across many images. In the event that our initial clustering is not in keeping with the user's requirements, we have proposed a technique to refine our initial clusters, in keeping with the information provided by the user, at interactive speeds.

Depth-Map Estimation Algorithm

The results of our experiments confirm that our method offers a significant improvement in performance over the current state-of-the-art reconstruction algorithms when running on sparse data sets. By explicitly accounting for the failure modes of the NCC matching technique we are able to produce depth-maps which accurately locate the true surface in noise, allowing the use of small matching windows. We are also able to identify where the surface estimate is inconsistent, due to lack of texture or occlusion, and label pixels as having unknown depths. Returning this unknown state, rather than providing a form of local regularisation, allows the subsequent global regularisation to be performed over all the depth-maps using the best possible data. If there are unknown surface regions that are not recovered by the depth-map a global regularisation scheme is in a much bet-

ter position to estimate the surface since it has access to all of the depth-maps. This is particularly important in the case of the sparse ring temple data-set and we believe it is primarily responsible for the improved performance over other methods. We also note that our depth-map estimation algorithm may be integrated with a variety of multi-view stereo algorithms [Hernández and Schmitt 2004, Vogiatzis et al. 2007, Goesele et al. 2006, Hornung and Kobbelt 2006a, Merrell et al. 2007, Bradley et al. 2008, Sinha et al. 2007] where is should confer similar improvements in performance.

8.3 Future Work

The work presented also has its limitations and these lead to a number of avenues for future research. The reconstruction approach we have presented manages to retrieve shape automatically when the object fulfils certain criteria. The matching techniques used by many multi-view stereo systems, including [Campbell et al. 2008], rely on the object possessing a textured and diffuse surface. Other objects will not be well modelled:

- **Texture-less objects.** Objects that lack texture, for example a porcelain model, can be reconstructed by other techniques, such as photometric stereo [Vogiatzis et al. 2006]. Unfortunately these techniques rely on assumptions about the lighting in the scene, in fact the majority require the ability to control the position of the lighting. Outside the laboratory we can exert little influence on the lighting of scenes and thus we are will need new methods to perform robust and automatic recovery of the shape of these objects.

- **Specular and translucent objects.** Algorithms that can reliably recover the shape of objects that are specular, or shiny, have yet to be firmly established. We can identify the challenge by considering the reflections seen in the body of a car which will vary with viewpoint and thus promote the hallucination of artificial surfaces. Taken to a greater extreme, consider translucent or transparent objects. For example we may only infer the shape of a glass vase through its distortion of light rays from the surrounding scene.

In both these cases we find that there is insufficient information to reconstruct the object when it is considered in isolation. Taking into account the scene surrounding the object will allow us to overcome the shortcomings of existing techniques with respect to object material and texture without the need to control other scene parameters, such as lighting.

Alongside the demand for reconstruction techniques for wider ranges of objects there exists the challenge of reducing the number of images required. Our work on reconstruction under sparse data sets has demonstrated that we can reduce the number of images

down to the range of 10-20 images whilst maintaining accuracy. However, we may be mindful of the fact that a human can be given a single postcard depicting a location they have never seen before and have little difficulty inferring geometry and shape in the scene. As outlined in § 1.4, the human visual system might be able to exploit a wealth of prior knowledge to constrain the image inversion problem. We would propose that learning and exploiting these priors represents the most important challenge in visual reconstruction.

Our latest research has involved looking at methods of enforcing priors using the concept of sparsity. The root of the idea is to enforce a prior by finding a transformation which takes the visual input data to a domain with a sparse representation. If we choose the sparse domain to relate to the prior we can see that the transformation represents the extraction of all the pertinent information under that prior.

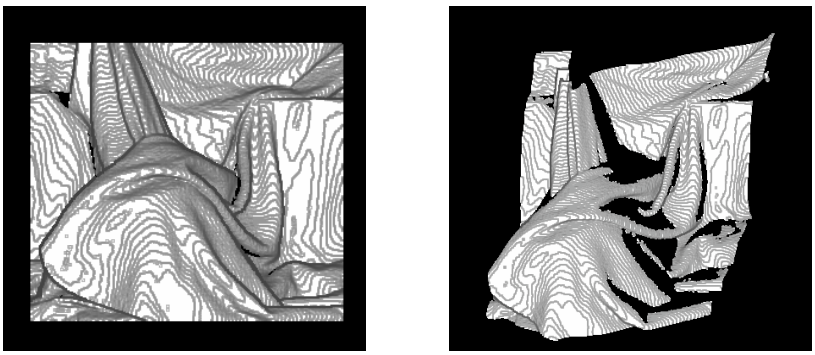
As an illustration we may consider the estimation of depth from a stereo pair. Many stereo algorithms exploit a constraint that surfaces in the world are generally smooth and continuous. If we can identify a transformation which encodes a smooth and continuous depth-map as a sparse signal we can conclude that our smoothness prior will be enforced if we are able to constrain our solution to be sparse under the transformation. Recent work within the signal processing community on 'compressed sensing', for example [Donoho 2006], has looked at the theory of sparse signals and algorithms to enforce sparsity. By selecting a transformation that encodes smooth surfaces as sparse signals, for example the wavelet transform of [Kingsbury 2001], and enforcing the sparsity using a compressed sensing framework we obtain the result in Figure 8.7. The result in Figure 8.7(c) demonstrates that the smoothness prior has been enforced and the result is arguably superior than the 'ground truth' data, Figure 8.7(b), obtained using a structured light scanner [Scharstein and Szeliski 2002b].

The result of Figure 8.7(c) may be obtained using the FISTA algorithm [Beck and Teboulle 2009] to minimise a convex energy function of the form

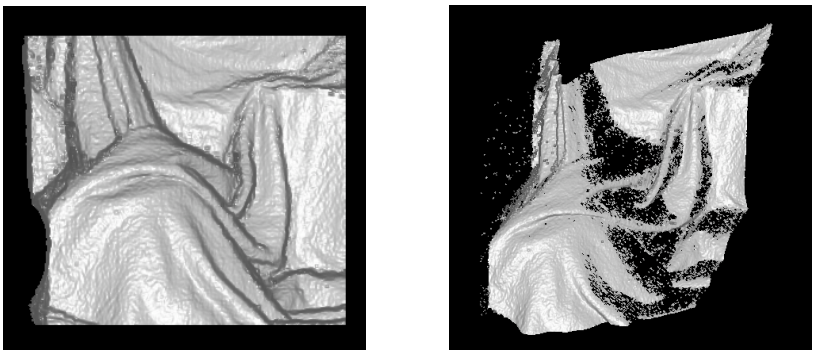
$$E(\mathbf{z}) \quad = \quad \left\| A\mathbf{z} - \mathbf{d} \right\|^2 + \lambda \left\| \Psi\, \mathbf{z} \right\|_1 \tag{8.1}$$

with $\mathbf{z}$ as the pixel depths, $\mathbf{d}$ as the depth of the chosen NCC peak (from Chapter 7) and Ψ as the Dual-Tree Complex Wavelet Transform (DT-CWT) [Kingsbury 2001]. The $L2$-norm formulation of the data term $\|A\mathbf{z} - \mathbf{d}\|^2$ derives from the assumption of a Gaussian distribution between the chosen NCC peak and the true surface. This term may not be complete (it may contain missing data) since pixels labelled as unknown will not be included, hence the matrix A serves to ignore these unknown pixels in the data term. The regularising term (multiplied by a regularising constant λ) is formulated as an $L1$-norm error term of the magnitude of the DT-CWT transform of the depth. The regularisation term exploits our prior that the depth-map is piecewise smooth, or more precisely, may be composed

(a) The pair of input images



(b) Ground truth



(c) Initial result enforcing sparsity prior

Figure 8.7: Initial stereo results using sparsity prior. *The compressed sensing algorithm was run on a pair of images (a) from a standard stereo evaluation data-set [Scharstein and Szeliski 2002a]. The ground-truth data (b), obtained using a structured light scanner [Scharstein and Szeliski 2002b], displays quantisation artifacts whilst the wavelet sparsity prior produces a result that is arguably more accurate.*

of piece-wise regions that may be expressed as low order polynomials. Functions of this sort are known to be sparse in the wavelet domain and thus we wish the regularising term to favour solutions for which $\Psi\,\mathbf{z}$ is sparse. The definition of sparsity (few non-zero coefficients) suggest that we should minimise the $L0$-norm $\|\Psi\,\mathbf{z}\|_0$, however such a cost function is no longer convex and very difficult to optimise. The choice of the $L1$-norm $\|\Psi\,\mathbf{z}\|_1$ exploits the main result of compressed sensing [Donoho 2006] that demonstrates that, under certain circumstances, a sparse signal may be recovered with fewer samples than the Nyquist limit dictates through minimisation of the $L1$-norm instead of the $L0$-norm.

This initial investigation suggests that the use of sparsity has great potential for representing priors and presenting algorithms to perform inference using them. The next target will be to produce a system that has the ability to use priors for general scenes rather than the somewhat contrived circumstances of Figure 8.7. Under this framework this corresponds to learning the transformation into the sparse domain.

The task of identifying our own priors and encoding them in algorithms is too sizeable a challenge for reconstruction of arbitrary scenes of the world. Instead we need to produce algorithms that learn these priors from training data and images. This is a very difficult challenge and will require the development of completely new algorithms and representations. It is also a challenge we might consider to be particularly exciting because we believe that the new theory and understanding that must go hand-in-hand with research of this nature will hopefully shed new light on how we might interpret the human visual system and the techniques and representations we use to model the visual world.

Bibliography

Y.I. Abdel-Aziz and H.M. Karara. Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In *Proc. ASP/UI Symp. on Close-Range Photogrammetry*, pages 1–18, 1971.

M. Armstrong, A. Zisserman, and P.A. Beardsley. Euclidean structure from uncalibrated images. In *Proc. 5^{th} British Machine Vision Conference*, pages 509–518, 1994.

B.G. Baumgart. *Geometric modelling for computer vision*. PhD thesis, Standford University, 1974.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, January 2009.

S. Birchfield and C. Tomasi. Depth discontinuities by pixel-to-pixel stereo. In *Proc. 6^{th} Intl. Conf. on Computer Vision*, pages 1073–1080, 1998.

C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

A. Blake, C. Rother, M. Brown, P. Perez, and P.H.S. Torr. Interactive image segmentation using an adaptive GMMRF model. In *Proc. 8^{th} Europ. Conf. on Computer Vision*, pages 428–441, 2004.

Y. Boykov. Computing geodesics and minimal surfaces via graph cuts. In *Proc. 9^{th} Intl. Conf. on Computer Vision*, pages 26–33, 2003.

Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *Proc. 8^{th} Intl. Conf. on Computer Vision*, pages 105–112, 2001.

Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9): 1124–1137, September 2004.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, November 2001.

D. Bradley, T. Boubekeur, and W. Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1063–6919, 2008.

A. Broadhurst, T. Drummond, and R. Cipolla. A probabilistic framework for space carving. In *Proc. 8^{th} Intl. Conf. on Computer Vision*, pages 338–393, 2001.

G.J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *Proc. 10^{th} Europ. Conf. on Computer Vision*, pages 44–57, 2008.

D.C. Brown. The bundle adjustment — progress and prospects. *Int. Archives Photogrammetry*, 21(3), 1976. Paper 3−03 (33 pages).

M. Brown and D. Lowe. Unsupervised 3D object recognition and reconstruction in unordered datasets. In *Intl. Conf. on 3D Imaging and Modelling*, pages 56–63, 2005.

N.D.F. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Automatic 3D object segmentation in multiple views using volumetric graph-cuts. In *Proc. 18^{th} British Machine Vision Conference*, pages 530–539, 2007.

N.D.F. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *Proc. 10^{th} Europ. Conf. on Computer Vision*, pages 766–779, 2008.

N.D.F. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Automatic 3D object segmentation in multiple views using volumetric graph-cuts. *Image and Vision Computing*, 28 (1):14–25, January 2010.

D. Capel. *Image Mosaicing and Super-Resolution*. Springer-Verlag, 2004.

Y.Y. Chuang, B. Curless, D. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 264–271, 2001.

R. Cipolla and P. Giblin. *Visual Motion of Curves and Surfaces*. Cambridge University Press, 1999.

R. Cipolla, K. Åström, and P. Giblin. Motion from the frontier of curved surfaces. In *Proc. 5^{th} Intl. Conf. on Computer Vision*, pages 269–275, 1995.

L.D. Cohen. On active contour models and balloons. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 53(2):211–218, March 1991.

A. Criminisi, J. Shotton, A. Blake, C. Rother, and P.H.S. Torr. Efficient dense stereo with occlusions for new view-synthesis by four-state dynamic programming. *Intl. Journal of Computer Vision*, 71(1):89–110, January 2007.

A. Criminisi, T. Sharp, and A. Blake. Geos: Geodesic image segmentation. In *Proc. 10^{th} Europ. Conf. on Computer Vision*, pages 99–112, 2008.

B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. of the ACM SIGGRAPH*, pages 303–312, 1996.

G. Das. A mathematical approach to problems in photogrammetry. *Empire Survey Review*, 10(73):131–137, 1949.

D. Donoho. Compressed sensing. *IEEE Trans. Information Theory*, 52(4):1289–1306, 2006.

C. Dyer and S. Seitz. Photorealistic scene reconstruction by voxel coloring. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1067–1073, 1997.

O. Faugeras. Stratification of 3D vision: projective, affine and metric representations. *J. Optical Society of America*, 12(3):465–484, March 1995.

O. Faugeras and R. Keriven. Variational principles, surface evolution, PDEs, level set methods and the stereo problem. *IEEE Trans. on Image Processing*, 7(3):335–344, 1998.

O. Faugeras and Q. Luong. *The Geometry of Multiple Images*. MIT Press, 2001.

O. Faugeras and G. Toscani. The calibration problem for stereo. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 15–20, 1986.

O. Faugeras, Q. Luong, and S.J. Maybank. Camera self-calibration: theory and experiments. In *Proc. 2^{nd} Europ. Conf. on Computer Vision*, pages 321–334, 1992.

P. Felzenszwalb and D. Huttenlocher. Efficient belief propagation for early vision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 261–268, 2004.

M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6): 381–395, 1981.

B. Frey and D. MacKay. A revolution: belief propagation in graphs with cycles. In *Advances in Neural Information Processing Systems*, pages 479–486, 1997.

P. Fua and Y. Leclerc. Object centred surface reconstruction: combining multi-image stereo and shading. *Intl. Journal of Computer Vision*, 16(1):35–56, 1995.

Y. Furukawa and J.-P. Pons. Accurate, dense, and robust multi-view stereopsis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

P. Giblin, F. Pollick, and J. Rycroft. Recovery of an unknown axis of rotation from the profiles of a rotating surface. *J. Optical Society America*, 11(7):1976–1984, 1994.

M. Goesele, B. Curless, and S. Seitz. Multi-view stereo revisited. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2402–2409, 2006.

M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. Seitz. Multi-view stereo for community photo collections. In *Proc. 11^{th} Intl. Conf. on Computer Vision*, 2007.

D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *J. Royal Statistical Society*, 51(2):271–279, 1989.

M. Habbecke and L. Kobbelt. A surface-growing approach to multi-view stereo reconstruction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

P. Hammer. Some network flow problems solved with pseudo- boolean programming. *Operations Research*, pages 388–389, 1965.

R.M. Haralick, C.N. Lee, K. Ottenberg, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 592–598, 1991.

R. Hartley. Euclidean reconstruction from uncalibrated views. In *Applications of Invariance in Computer Vision*, pages 237–256, 1993.

R. Hartley. In defense of the eight-point algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(6):580–593, June 1997.

R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

C. Hernández. *Stereo and Silhouette Fusion for 3D Object Modeling from Uncalibrated Images Under Circular Motion*. PhD thesis, École Nationale Supérieure des Télécommunications, 2004.

C. Hernández and F. Schmitt. Silhouette and stereo fusion for 3D object modelling. *Computer Vision and Image Understanding*, 96(3):367–392, December 2004.

C. Hernández, F. Schmitt, and R. Cipolla. Silhouette coherence for camera calibration under circular motion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(2):343–349, February 2007a.

C. Hernández, G. Vogiatzis, and R. Cipolla. Probabilistic visibility for multi-view stereo. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007b.

A. Heyden and K. Astrom. Euclidean reconstruction from constant intrinsic parameters. In *Intl. Conf. Pattern Recognition*, pages 339–343, 1996.

A. Hornung and L. Kobbelt. Hierarchical volumetric multi-view stereo reconstruction of manifold surfaces based on dual graph embedding. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 503–510, 2006a.

A. Hornung and L. Kobbelt. Robust and efficient photoconsistency estimation for volumetric 3D reconstruction. In *Proc. 9^{th} Europ. Conf. on Computer Vision*, pages 179–190, 2006b.

M. Jancosek and T. Pajdla. Segmentation based multi-view stereo. In *Computer Vision Winter Workshop*, 2009. Paper 9.

E.T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

H. Jin, S. Soatto, and A. Yezzi. Multi-view stereo reconstruction of dense shape and complex appearance. *Intl. Journal of Computer Vision*, 63(3):175–189, 2005.

R. Kindermann and J. L. Snell. *Markov Random Fields and Their Applications*. American Mathematical Society, 1980.

N.G. Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Journal of Applied and Computational Harmonic Analysis*, 10(3):234–253, 2001.

P. Kohli and P.H.S. Torr. Effciently solving dynamic markov random fields using graph-cuts. In *Proc. 10^{th} Intl. Conf. on Computer Vision*, pages 922–929, 2005.

K. Kolev and D. Cremers. Integration of multiview stereo and silhouettes via convex functionals on convex domains. In *Proc. 10^{th} Europ. Conf. on Computer Vision*, pages 752–765, 2008.

V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, October 2006.

V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts—a review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(7):1274–1279, July 2007.

V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph-cuts. In *Proc. 7^{th} Europ. Conf. on Computer Vision*, pages 82–96, 2002.

V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):147–159, November 2004.

E. Kruppa. Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *Math, Naturw. Abt. IIa*, 122:1939–1948, 1913.

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

A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, February 1994.

V. Lempitsky, C. Rother, and A. Blake. Logcut — efficient graph cut optimization for markov random fields. In *Proc. 11^{th} Intl. Conf. on Computer Vision*, 2007.

V. Lempitsky, P. Kohli, C. Rother, and T. Sharp. Image segmentation with a bounding box prior. In *Proc. 12^{th} Intl. Conf. on Computer Vision*, pages 277–284, 2009.

A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(12):2290–2297, 2009.

D.G. Lowe. Distinctive image features from scale-invariant keypoints. *Intl. Journal of Computer Vision*, 60(2):91–110, 2004.

D. Marr. *Vision*. W.H.Freeman & Co., 1982.

S. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *Intl. Journal of Computer Vision*, 8(2):123–151, August 1992.

M. Meila and J. Shi. A random walks view of spectral segmentation. In *10th Int. Workshop on Artificial Intelligence and Statistics*, 2001.

P.R.S. Mendonça, K.-Y.K. Wong, and R. Cipolla. Epipolar geometry from profiles under circular motion. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(6):604–616, 2001.

P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nistér, and M. Pollefeys. Real-time visibility-based fusion of depth maps. In *Proc. 11^{th} Intl. Conf. on Computer Vision*, 2007.

K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *Intl. Journal of Computer Vision*, 65(1):43–72, 2005.

D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(12):756–770, June 2004.

Y. Ohta and T. Kanade. Stereo by two-level dynamic programming. *Intl. Joint Conf. on Artificial Intelligence*, pages 1120–1126, 1985.

J. Park and A.C. Kak. Multi-peak range imaging for accurate 3D reconstruction of specular objects. In *Proc. 6^{th} Asian Conf. on Computer Vision*, 2004.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman, 1988.

T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317(6035):314–319, 1985.

M. Pollefeys and L. Van Gool. A stratified approach to metric self-calibration. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 407–412, 1997.

M. Pollefeys, L. Van Gool, and A. Oosterlinck. The modulus constraint: A new constraint for self-calibration. In *Proc. 13^{th} Intl. Conf. on Pattern Recognition*, pages 349–353, 1996.

J.-P. Pons, R. Keriven, and O. Faugeras. Modelling dynamic scenes by registering multi-view image sequences. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 822–827, 2005.

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

L. Quan and Z.D. Lan. Linear n $\geq$ 4-point pose determination. In *Proc. 6^{th} Intl. Conf. on Computer Vision*, pages 778–783, 1998.

X. Ren and J. Malik. Learning a classification model for segmentation. In *Proc. 9^{th} Intl. Conf. on Computer Vision*, pages 10–17, 2003.

C. Rother, V. Kolmogorov, and A. Blake. "grabcut": interactive foreground extraction using iterated graph cuts. In *Proc. of the ACM SIGGRAPH*, pages 309–314, 2004.

C. Rother, T. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching — incorporating a global constraint into MRFs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 993–1000, 2006.

D. Scharstein and R. Szeliski. Middlebury evaluation website, 2002a. URL `http://vision.middlebury.edu/`.

D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Intl. Journal of Computer Vision*, 47(1–3):7–42, 2002b.

S. Segvic, G. Schweighofer, and A. Pinz. Performance evaluation of the five-point relative pose with emphasis on planar scenes. In *Proc. of AAPR/OAGM*, pages 33–40, 2007.

S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 519–528, 2006.

J. Shi and J. Malik. Normalized cuts and image segmentation. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 731–737, 1997.

S.N. Sinha, P. Mordohai, and M. Pollefeys. Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh. In *Proc. 11^{th} Intl. Conf. on Computer Vision*, 2007.

N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: Exploring image collections in 3D. In *Proc. of the ACM SIGGRAPH*, pages 835–846, 2006.

D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 345–353, 2000.

M. Sormann, C. Zach, and K. Karner. Graph cut based multiple view segmentation for 3d reconstruction. In *Intl. Symposium on 3D Data Processing Visualization and Transmission*, pages 1085–1092, 2006.

C. Strecha, R. Fransens, and L. Van Gool. Combined depth and outlier estimation in multi-view stereo. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 2394–2401, 2006.

C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

P. Sturm. A case against kruppa's equations for camea self-calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(10):1199–1204, 2000.

J. Sun, Y. Li, S.B. Kang, and H.-Y. Shum. Symmetric stereo matching for occlusion handling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 399–406, 2005.

Y. Sun, P. Kohli, M. Bray, and P.H.S. Torr. Using strong priors for stereo. In *Comp. Vision, Graphics and Image Proc.*, pages 882–893. Springer-Verlag, 2006. LNCS 4338.

I.E. Sutherland. Sketchpad: A man-machine graphical communication system. Technical Report Technical Report 296, MIT Lincoln Laboratories, 1963.

R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(6):1068–1080, June 2008.

M.A. Taalebinezhaad. Direct recovery of motion and shape in the general case by fixation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(8):847–853, August 1992.

P.H.S. Torr and D.W. Murray. Outlier detection and motion segmentation. In P. S. Schenker, editor, *Sensor Fusion VI*, pages 432–443. SPIE volume 2059, 1993.

P.H.S. Torr and A. Zisserman. Robust computation and parametrization of multiple view relations. In *Proc. 6^{th} Intl. Conf. on Computer Vision*, pages 727–732, 1998.

B. Triggs. Autocalibration and the absolute quadric. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1997.

B. Triggs. Autocalibration from planar scenes. In *Proc. 5^{th} Europ. Conf. on Computer Vision*, pages 89–105, 1998.

B. Triggs. Camera pose and calibration from 4 or 5 known points. In *Proc. 7^{th} Intl. Conf. on Computer Vision*, pages 278–284, 1999.

B. Triggs, P. McLauchlan, R.I. Hartley, and A.W. Fitzgibbon. Bundle adjustment: A modern synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–372, 1999.

R. Tsai. A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE J. Robotics and Automation*, 3 (4):323–344, 1987.

G. Vogiatzis, P.H.S. Torr, and R. Cipolla. Bayesian stochastic mesh optimisation for 3D reconstruction. In *Proc. 14^{th} British Machine Vision Conference*, pages 711–718, 2003.

G. Vogiatzis, P.H.S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 391–398, 2005.

G. Vogiatzis, C. Hernández, and R. Cipolla. Reconstruction in the round using photometric normals and silhouettes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1847–1854, 2006.

G. Vogiatzis, C. Hernández, P.H.S. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(12):2241–2246, December 2007.

W. Woo W. Lee and E. Boyer. Identifying foreground from multiple images. In *Proc. 8^{th} Asian Conf. on Computer Vision*, pages 580–589, 2007.

M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on trees: message passing and linear programming. *IEEE Trans. Information Theory*, 51(11):3697–3717, 2005.

J. Wang, P. Bhat, R.A. Colburn, M. Agrawala, and M.F. Cohen. Interactive video cutout. In *Proc. of the ACM SIGGRAPH*, pages 585–594, 2005.

K-Y.K. Wong, P.R.S. Mendonça, and R. Cipolla. Reconstruction and motion estimation from apparent contours under circular motion. In *Proc. 10^{th} British Machine Vision Conference*, pages 83–92, 1999.

O.J. Woodford, I.D. Reid, and A.W. Fitzgibbon. Efficient new view syntesis using pairwise dictionary priors. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

R.J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19(1):139–144, 1980.

P.C. Woodland and S.J. Young. The htk tied state continuous speech recognition. In *Proc. EuroSpeech*, pages 2207–2210, 1993.

J. Yedidia, W. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Information Theory*, 51(7):2282–2312, July 2005.

A. Yezzi and S. Soatto. Stereoscopic segmentation. *Intl. Journal of Computer Vision*, 53(1): 31–43, January 2003.

C. Zach, T. Pock, and H. Bischof. A globally optimal algorithm for robust TV-L1 range image integration. In *Proc. 11th Intl. Conf. on Computer Vision*, 2007.

R. Zhang, P.S. Tsai, J. Cryer, and M. Shah. Shape from shading: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(8):690–706, August 1999.

Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, November 2000.