3-D Reconstruction of Multi-object Scenes from Multiple Images

Matthew Grum

This thesis is submitted in partial fulfilment of the
requirements for the degree of
Doctor of Philosophy.

University of York
York
YO10 5DD
UK

Computer Science

April 2010

Abstract

Photorealistic 3-D models are used in a wide variety of applications from entertainment and games, through to simulation, training . Algorithms to automatically create such models from ordinary photographs can vastly reduce the workload and expense associated with acquiring such models. The vast majority of research into reconstructing 3-D models from images has concentrated on the case of single objects.

This thesis presents a method to model complex multi-object scenes in a series of steps starting with a set of images which surround a scene and finally producing a complete photorealistic representation of the objects. The probabilistic space carving algorithm is used to provide an initial estimate of shape as it makes no assumptions about the shape of the scene aside from the bounding cuboid. This representation is smoothed by fitting a Radial Basis Function implicit surface, which smoothes noise and interpolates any missing data. Errors which persist are addressed by a matching surface points between images and estimating the perspective transformation between them which is used to calculate the correct position for the point, which is consistent with the input images. The model may be corrected by constraining the surface to pass through these points. The smoothing properties of RBFs can cause problems by interpolating across objects which are close together, causing them to be joined in the representation. A method is presented to correct this by enforcing consistency between edges in 2-D and 3-D.

Experiments are conducted using real image sequences of complex multi-object scenes. Both qualitative and quantitative evaluations are performed demonstrating the effectiveness of the methods presented. In addition to modelling all of the objects present, colour surfaces are produced from which even fine text is legible. A detailed study is undertaken into the factors which influence the effectiveness of techniques to recover partially or fully fused objects and conclusions are drawn which hint at the ultimate limit of accuracy in the case of multiple objects.

Declaration

This thesis has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree other than Doctor of Philosophy of the University of York. This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by explicit references.

I hereby give consent for my thesis, if accepted, to be made available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

Publications

Parts of the work in this thesis have been published in the following conferences:

M. Grum, A.G. Bors, “Enforcing image consistency in multiple 3-D object modelling,” *Proc. Int. Conference on Pattern Recognition*, Tampa, FL, USA, 12-15 Oct. 2008, pp. 132-135.

M. Grum, A.G. Bors “Multiple image disparity correction for 3-D scene representation,” *Proc. IEEE Inter. Conf. on Image Processing*, San Diego, CA, USA, 12-15 Oct. 2008, pp. 209-212.

M. Grum, A.G. Bors “Refining implicit function representations of 3-D scenes,” *Proc. British Machine Vision Conference* Warwick, UK, 10-13 Sep. 2007, vol. II, pp. 710-719.

M. Grum, A.G. Bors, “3-D scene modelling from multiple images using radial basis function networks,” *Proc. IEEE Workshop on Machine Learning for Signal Processing*, 27-29 Aug. 2007, Thessaloniki, Greece, pp. 105-110.

Contents

List of Tables

List of Figures

Notation

In general sets are indicated by caligraphic capital letters, $\mathcal{S}$, matrices and tensors by capital italics, $\mathbf{M}$, column vectors by lowercase bold, $\mathbf{v}$, and scalars by lowercase italics, s. In addition to this the following notations are commonly used:

$\mathcal{I} = \{\mathbf{I}_i \| i \in 1..n\}$	the set of n input images
$\mathbf{u} = \mathbf{I}_i^{u,v}$	a [colour] pixel at coordinates u, v in image i
$\mathcal{P} = \{\mathbf{P}_i \| i \in 1..n\}$	the set of corresponding projection matrices
$\mathcal{C}_i$	a contour corresponding to image i
$\mathbf{u} \in \mathcal{C}_i$	a point in 2-D which lies within the contour
$\mathcal{V}$	the visual hull of an object or group of objects
$\mathbf{x} = (x, y, z)^\mathsf{T}$	a point in 3-D space, made up of three coordinates
$\phi(x)$	a radially symmetric basis function
$\mathbf{c}$	the centre of a basis function (a point in space)
$\overrightarrow{\mathbf{n}}$	normal vector (of unit length)

Please note the difference between the projection matrix $\mathbf{P}$ and the probability of an event $P(e)$. The following terms have a specific meaning with this thesis. Colour - a triplet of values that determine the appearance of a point on a surface or image, capturing lightness as well as the hue. Thus dark blue, light blue, black, white etc. are different considered to be different colours. This is slightly different to the definition of colour as a particular wavelength of light. Texture - the pattern of colours across a surface. This does not refer to the tactile quality of surfaces in any way.

Chapter 1

Introduction

We live in a three-dimensional world. For computers to be able to interact with this world they need sensory information about their 3-D environment. The field of computer vision lies at the confluence of many areas of research, such as artificial intelligence, robotics, optimisation, pattern recognition and signal processing. The ultimate goal is to allow computers to understand and reason about the outside world visually, in a similar way to human beings. 3-D computer vision concerns the process of inferring spatial relationships from data obtained using cameras or other electromagnetic sensory equipment. Applications of 3-D vision include automated navigation, tracking and interpretation of motion, face recognition and 3-D modelling.

The earliest origins of 3-D computer vision can be traced to the study of photogrammetry - making measurements using photographs. Photogrammetry dates back to the mid-nineteenth century and is almost as old as photography itself. It was used principally by architects and cartographers for the purpose of creating accurate diagrams and is still used today in areas where direct measurements cannot be made. In photogrammetry the emphasis is placed firmly on accuracy. This requires high-quality carefully constructed camera equipment (to minimise the distortion introduced by the lens elements and any other imperfections) and full knowledge of the cameras internal geometry. Special markers are usually attached to objects which are then photographed from different angles. From these photographs the absolute position in space of the markers can be calculated geometrically.

Influences have also been drawn from other areas of mathematics and computer science. The popularisation of “impossible shapes” such as Penrose triangle in the 1950s sparked interest in the mathematical properties of drawings. One of the first uses of constraint programming in AI was

to reason about physically plausible 2-D figures. Edges and junctions were labelled according to wether they represented a convex, concave or occluding boundary. If a labelling consistent with a set of rules could be produced then the figure can be said to be a representation of a 3-D real shape.

The first steps toward the description of a modern computer vision system may be attributed to David Marr [58]. Marr was a neurobiologist who set out to describe how human vision could be defined as an information processing task. He argued that any such task required three components: the computational theory (which defines the problem and provides the motivation for its solution) the representation and algorithm - a model of the process which leads finally to an implementation.

Applying his theory to the human visual system he describes four stages in which visual information is processed, these are the retinal image, the primal sketch, the $2\frac{1}{2}$D sketch and finally the 3-D model.

In recent years researched has moved on from the first two stages to concentrate on the final stages. It is possible to infer shape from a variety of visual cues. Algorithms for 3-D reconstruction are often referred to as "shape from X" depending on the particular cue used.

Shape from shading aims to reconstruct a 3-D surface based on observed irradiance in a single image. By modelling the interaction between incident light and the surface[1] information about the shape can be inferred, although in general the problem is underconstrained (since for each observation two angles must be calculated to define the surface orientation). Additional constraints such as smoothness are thus required to obtain a unique solution [86]. Shape from shading has been shown to be effective for object recognition [87] although the applicability to general 3-D vision is limited by the requirement of untextured diffuse surfaces.

The appearance of an object's texture is strongly influenced by the surface orientation leading to the formlation of shape from texture. Geometric texture analysis requires the identification of repeating patterns which is not always possible. Frequency domain algorithms such as [69] rely on changes in the power spectrum that indicate variation in surface orientation. Unlike shape from shading the problem has sufficient mathematical constraint to provide a unique solution however a fixed repeating texture must be used.

Human binocular vision provides the inspiration for shape from stereo. Stereo algorithms take a pair of images and use the displacement of corresponding image features in order to estimate depth

[1] For example using Lambert's law, which states that the intensity of reflected light is proportional to the angle between the surface and the light source.

(distance from the camera). The output is often in the form of a depth map, which contains a depth value for each pixel in the input images [74].

A related field to shape from stereo is shape from motion (more commonly referred to as structure from motion, or multi-view stereo). As the name suggests, movement of either the object or camera provides the information for reconstruction via the influence of perspective. A wide variety of algorithms have been presented for this problem and these can be broadly classified as either 2-D to 3-D, in which image measurements are used to infer 3-D information (similar to photogrammetry), or 3-D to 2-D, in which a 3-D shape is created and refined to match the images.

Shape from shading, shape from texture and shape from stereo only produce a $2\frac{1}{2}$D representation, meaning some parts of the scene are not present in the model. One goal of 3-D computer vision is to obtain complete 3-D models of real world objects and scenes. Such models form the backbone of a number of applications, including:

- Simulation, in which models are used to study new scenarios or provide realistic training that would otherwise be expensive or impossible to undertake (for example space missions).
- Modelling, whereby experimental designs can be evaluated or explored visually without the need to create expensive fabrications.
- Virtual Reality, immersion in a 3-D environment where a user is free to move around and interact. Forms of VR are commonly available for both educational and recreation purposes.
- Augmented reality, where the real and virtual worlds are combined. These techniques are often used to provide special effects in films such as The Matrix.

In many of the applications mentioned above the models are required to be photorealistic, that is views rendered of the model must be photographic in appearance, if not completely indistinguishable from photos.

This thesis describes a method to obtain photorealistic 3-D models from scenes containing multiple objects, using only a set of images captured from various positions around the scene.

Chapter 2 presents a survey of literature on the subject, starting with the foundations of computer vision and then considering early work in 3-D modelling followed by detailed studies of relevant strands.

Chapter 3 concerns creating and improving 3-D representations of real scenes using image sets. The theory behind image formation is explained in detail as this forms the foundation of much of the following material. Probabilistic space carving forms the initial step of the modelling process, followed by the fitting of smooth surfaces to the space carving result. A novel method of updating these surfaces by examining image disparities is presented. This method avoids the need to approximate the form of the 2-D distortion that arises when the initial surface is incorrect. Finally Experiments are presented using two challenging real world data sets and the results analysed and conclusions drawn.

This work is built upon in Chapter 4, which principally addresses a common problem whereby objects close to each other in the scene become joined together in the reconstruction. A solution is proposed based on the observation that an edge in 3-D (where the surface disappears from view) must have a corresponding edge in the images thus the artificial joining edge can be identified and removed. A investigation is conducted into the conditions under which this is possible, using synthetic data. These results are confirmed by further experiments using the same data as the previous chapter and conclusions are drawn about the effectiveness and limitations of multi-object 3-D reconstruction from images.

Finally, Chapter 5 summarises all of the results obtained, and discusses possible applications of the work.

Chapter 2

Literature Review

2.1 Computer Vision

One of the earliest publications in the field of computer vision was David Marr's seminal work on visual processing [58]. Marr argues that the human visual system is best expressed as an information processing task. He goes on to state that any such task requires analysis at two levels, that of the computational theory, which is concerned both with what is computed, and importantly also why it is computed. The level below this is the realm of the how, where the algorithms that accomplish the task belong. Marr uses the analogy of fourier analysis, the definition of the fourier transform is separate from the fast fourier transform algorithm. Fully understanding the latter requires a full understanding of the former. Thus each of the stages of vision are always described in reference to their place in the overall system.

The motivation, the 'why' part of a visual system, is to extract a representation that useful to the observer, and which does not contain extraneous or redundant information. As for the 'how', Marr identifies four stages characterised principally by the represention and type of data that is manipulated. Namely these are; the retinal image, the primal sketch, the $2\frac{1}{2}$D sketch and finally the 3-D model. The first stage is analogous to signal processing whereby the raw data is captured and transformed as necessary. The primal sketch contains some higher level structures, which might be simple features identified in an image. These features are used to infer 3-D information in the next step. Marr coined the term $2\frac{1}{2}$D to refer to a representation that is labelled with some relative 3-D relationships but does not constitute a complete 3-D description. Finally the full 3-D model is

described using an absolute frame of reference.

2.2 Stereopsis

Stereopsis refers to the problem of inferring depth from pairs of images taken from side by side viewpoints. It corresponds to the $2\frac{1}{2}$D sketch in Marr's vision hierarchy, and work in this area was clearly inspired by the human visual system. The technique principally revolves around the problem of correspondence, once the location of a point is known in both images the depth can be estimated by the degree to which this location differs.

A detailed catalogue of the range and breadth of approaches to stereopsis can be found in [73]. Only methods which take two images and return a depth estimate for each point in the images (dense estimates) are considered, thus they share a "computational theory" in Marr's analysis. Differences are limited then to the type of algorithm used to solve the problem. The paper describes this computational theory as consisting of four steps: matching cost computation, cost aggregation, disparity computation / optimisation, and disparity refinement.

The methods surveyed are grouped according to the algorithms applied to each of these steps for example, what cost function is used for matching (i.e. how do you determine how good a given match between two image regions corresponding to the same 3-D feature is). This provides a means to group methods together and reveals two broad categories of stereo vision algorithms, local and global. Local methods such as [10, 36, 92] are based on computing costs across small image windows and simple aggregation methods. Global methods such as [5, 14, 59, 72] tend to pose the stereo problem in an optimisation framework where the global cost of all matches is minimised. The performance of global methods is shown to be better on the whole.

In addition to this the authors also make available reference implementations of each of the main algorithmic steps covered, as well as datasets with ground truth depth labelling. The paper is thus a very important contribution to the field.

2.3 Structure from Motion

Research into Structure from Motion SfM grew out of work on stereo vision. Whilst results from stereo algorithms can be very good, they are still unsuitable to certain applications, due to the fact

that even a dense depth map is only a $2\frac{1}{2}$D representation. Often depth values can only be interpreted as relative measurements thus not corresponding to a complete 3-D representation. In addition to this there are some ambiguities that cannot be resolved when only considering two images, for example certain parts of the scene will only be visible in one of the pair of images, due to occlusion. Depth cannot be estimated for these regions.

In order to overcome this shortcomings structure from motion generalises stereo vision such that arbitrary motion of a camera (or a set of many static cameras) is used to calculate 3-D information.

Early approaches to structure from motion were based on analytical geometry. Two views of the same scene are related be a quantity known as the fundamental matrix, which describes the implicit geometry [57]. Use of the fundamental matrix became popular particularly for studying the uncalibrated case since it is dependant on 2D image observations alone. From this matrix 3-D shape can be recovered up to a projective transformation. Errors in the 2D measurements often cause serious distortion in the resultant shape. This can be improved by using three views. The three view equivalent of the fundamental matrix is known as the trifocal tensor, first presented in [38]. The trifocal tensor also overcomes scaling problems, when the scene points lie close to a certain plane.

Similar formulations have been derived for the case of four images [39], but beyond this direct calculation of structure and motion becomes very difficult. So far nobody has presented a closed form solution to the geometry of five images. Further research was based either on approximations or nonlinear solutions to the problem.

Within the category of approximate solutions live a family of so called “batch” algorithms. Accurate correspondence is vital to any structure from motion algorithm. Correspondence between images taken from significantly different viewpoints is very difficult to achieve without some user intervention and this prompted the use of video sequences taken from a moving camera, with a small timelapse between frames. This produces a set of images with limited spacial transformations allowing salient features to be more easily tracked through the sequence.

Another advantage to video is the large degree of redundancy in the data can make up for errors due to mistracked features and the simplifying assumptions that have to be made. The ability to make use of this large number of images is what distinguishes a batch algorithm. The first such algorithms were based on restricting the camera motion to a planar or linear trajectory to reduce the number of degrees of freedom in the motion equations [80]. This occurs is at the expense of both generality (precisely controlling the camera trajectory is difficult in practice) and the accuracy of the

resultant model since ambiguities can occur in such limited camera trajectories.

In [81] a technique known as the factorisation method was developed to extract shape information from arbitrary camera motions. Using an orthographic camera model the projection of all scene points from all views can be expressed as a single overconstrained matrix equation, $\mathbf{W} = \mathbf{MS}$, where $\mathbf{W}$ contains the observations, $\mathbf{M}$ represents the camera motion and $\mathbf{S}$ the objects shape. This equation can be solved for the shape and motion components by factorisation, using reliable and fast numerical algorithms such as the singular value decomposition (after overcoming certain ambiguities in the solution).

The accuracy of this approach is limited by the assumption of orthographic projection, which fails to take into account any perspective effects such as the fact that objects appear larger when closer to the camera, and surfaces appear skewed when viewed from oblique angles. So whilst arbitrary camera motion is allowed in practice the method is limited to scenes of compact objects in which the camera to object distance remains approximately constant.

This issue was addressed in [65] where the authors construct a similar linear equation from a first order approximation of the full perspective projection. The paraperspective projection preserves some of the key features of perspective projection, namely that objects get smaller the further they are from the camera and become skewed when viewed from oblique angles. The only feature not present is the tendancy for parallel lines to appear to converge as they get further from the observer. The authors show how this more complicated projection model can be transformed to fit into exactly the same matrix equation as before. This constitutes a very elegant solution to the structure from motion problem.

The authors also present a confidence weighted factorisation algorithm whereby the decomposition yields the lowest residual weighted by measurement confidence values. This allows reliability estimates for tracked features to be incorporated, and occlusion to be implicitly modelled by assigning zero confidence values to features when they cease to be visible. All in all the method they propose is a very elegant solution to the problem, and allows large numbers of images to be processed simultaneously.

These factorisation methods are ultimately limited to single objects, since the necessary approximations are only valid close to objects centroid - collection of objects have a common centroid that may be far from each object. In the appendix [65] describes a post processing step to iteratively refine the shape and motion estimates using a gradient decent algorithm to minimise the error between

the predicted and observed image measurements.

The factorisation method works with point features tracked in images and thus yields a 3-D point cloud as the representation of the scene. Whilst this is useful for certain tasks a representation more recognisable to humans is preferable. In addition to this point features in images can be confused for one another as they lack many distinguishing features, prompting research into methods which use higher level features. Lines are the obvious choice after points and algorithms to locate and extract line features from images such as the Hough transform [42] have been for a long time.

In [68] a linear method to recover shape in the form of 3-D line segments from multiple images is presented. A linear solution is permitted due to the use of an affine camera model, again an approximation of the full perspective projection. Due to this limitation the extra information that may be obtained from comparing line segments (difference in length, orientation, location etc.) is not utilised.

Better results are obtained by minimising an error function based on the area between predicted and observed line segments [79]. This paper uses a similar approach to the nonlinear post-processing step from [65]. A cost function is minimised by a multi-step iterative algorithm whereby the unknown variables are divided into four sets representing the line orientations, line positions, camera rotation and camera positions. Local optimisations are carried out on each set (which are re-initialised if necessary) prior to a global minimisation of the cost function.

Alternatives to the analytical approach have been explored including the use of statistical techniques which are better able to cope with the noise inherent in image data. In [28] shape and motion are calculated from image measurements within Bayesian framework. Under certain assumptions Bayes theory is able to make the best use of the available data. One of these assumptions is that the posterior probability distribution is known. As this is not true in general, the authors use a Markov chain Monte Carlo sampler to estimate the posterior of the problem. The results are shown to be robust in the presence of tracking errors, however the scaled orthographic projection model is used which limits the generality of this approach, and the use of point features only is limiting as discussed above.

The statistical formulation was extended in [21] to work without any correspondence information. This completely removes the problems of inaccurate 2-D features and allows the algorithm to

work in the absence of any pre-processing steps to track and match image features.

The algorithm works by finding the maximum likelihood estimation over all assignments of shape points to image features. Statistical sampling (again using the MCMC method) is used to generate a set of virtual measurements which are then fed into an integration step as if they were a unique structure from motion problem. The Expectation-Maximisation framework provides good convergence properties for the iteration. Results on real images are presented and whilst promising, the number of and type of features is limited reducing the usefulness of this approach outside of situations where correspondence is exceptionally difficult.

The line and point based 3D representations mentioned so far, whilst suitable for many applications (such as camera position estimation, robot navigation etc.) fall someway short of photorealism. This goal requires a shape representation that is dense, i.e. contains no gaps. The following sections describe work undertaken to provide such representations.

2.4 Volumetric Methods

Methods to estimate the volume of space occupied by an object have existed for a long time in computer vision [48] and is a research area that has attracted a significant amount of interest. In [53] a novel approach is presented to estimate the shape of an object directly from the input images, provided the outline, or silhouette, has been defined. The silhouette is simply a binary image in which each pixel is labelled as being part of either the object, or the background, generated either by hand or by automatic subtraction of a known or constant colour background. The method is explained by means of a geometric formulation whereby the silhouette from each image is back-projected from the corresponding camera position (which must be known a priori) forming a set of conics. The intersection of these conics defines a convex volume known as the visual hull.

There are usually many different shapes that can be consistent with a given visual hull, the paper presents a detailed analysis of the conditions under which the minimal hull is obtained. Methods are presented to distinguish between ‘hard points’ which lie on the surface of the object and are thus constrained within all consistent shapes, and ‘soft points’ that mark the extremes of the visual hull.

Many shape from silhouette algorithms employ a discrete representation of the volume (similar to voxels which are covered below). The drawback of this approach is that the accuracy is dependant on the resolution of the model. In [13] the authors note that much of the space is wasted when high

resolution grids are used due to the uniform sampling. Instead they propose a non-uniform grid composed of tetrahedrons and concentrated in the vicinity of the visual hull. This grid is based on the Delaunay triangulation to sampled points on the visual hull. The results produce pleasing surfaces which are more detailed but don't require a correspondingly larger amount of space to store than other methods.

A method to estimate both shape and motion of curved surfaces from silhouettes is proposed in [32]. The notion of frontier points (referred to as hard points in the previous paper) provides a means to solve a correspondence problem via the RANSAC (random sample consensus) method. A specialised voting scheme enforces the geometric redundancy in the image set and allows the estimation of camera positions which have consistent corresponding frontier points.

Their method is interesting in that it requires almost no a-priori information about the scene, although it does assume that a set of curves corresponding the object's outline can be segmented from the images, or provided by the user. Their formulation also permits images to be added sequentially the the input data in order to incrementally refine the representation. This "coarse to fine" approach is common to computer vision algorithms where potentially large datasets are involved.

A related method to shape from silhouette, is shape from shadow [70]. Whilst silhouettes can be thought of as the shadow of an object cast onto a plane by a lightsource aligned with the camera, [70] demonstrates that this can be extended to the case of shadows cast onto arbitrary surfaces and the object casting shadows onto itself. The incident illumination must be a point source and the origin with respect to the scene must be known.

From an initial coarse shape estimate, modelling takes place by means of a carving methodology whereby parts of the shape which are inconsistent with the observered shadows are removed, or carved away. This continues until no further inconsistencies remain (note that this doesn't necessarily mean that the object's true shape has been reached). The authors provide a proof of correctness that relies on conservative estimates of shadow regions, that is in the detection of shadows false negatives may occur but never false positives.

Whilst this is an interesting approach it is heavily constrained in the conditions under which it can perform. However, shadow constraints have been successfully combined with shape from shading algorithms which have similar prerequisites for simple, known illumination.

Carving methods based on shadows and silhouettes have difficulty when objects with complex

surfaces and concavities are present. A family of alternative methods have been proposed to carve shapes using primitives called voxels. Voxels are the 3-D analogue of pixels, where a 2-D array of coloured pixels can represent an arbitrary image, a 3-D lattice of coloured voxels can represent an arbitrary shape, limited only by the size and number of voxels used.

Voxels have long been used to visualise dense 3-D information present in medical images from CT and MRI scanners. Creating a voxel model from a set of images was first addressed in [75]. The algorithm is based around the concept of photoconsistency. Under the assumption that objects obey Lambert's law and reflect light equally in all directions, the appearance of a voxel should be the same in all the images in which it is visible. Thus starting with an array of voxels that completely encloses the scene, any voxels which project to image regions with different colours cannot correspond to parts of the scene and may be removed (marked as transparent). The algorithm is formulated to provide photorealism by design - if each of the output voxels are consistent, then together their appearance will match the input images.

The difficulty with this approach arises due to visibility - if another object or part of the same object is in front in a particular image photoconsistency with that image is not required. Determining whether a voxel is visible will depend on the existence of other voxels. For this reason the authors place a restriction that the candidate set of voxels and the camera positions must be separable by a plane. This restriction, known as the ordinal visibility constraint ensures that for a given pair of voxels A and B it is not possible that A occludes B in one image while B occludes A in another. Thus the volume comprising the voxels can be processed from front (nearest the cameras) to back, now when the photoconsistency of a voxel is evaluated the opacity of all voxels which could occlude it will have been decided.

The algorithm presented in [75] performs well under this condition however it is not a fundamental restriction for all voxel carving methods. This was shown in two papers published around the same time which presented algorithms which, by means of multiple sweeps of the volume, were able to overcome the ordinal visibility constraint. In [50] the Space Carving algorithm is presented. At its core is a plane sweep method of evaluating consistency that is applied in each of the six possible directions. Only a subset of cameras behind the current sweep plane is used to evaluate consistency of a given voxel. The ability to adopt arbitrary camera configurations was demonstrated by means of a synthetic scene involving a room with an open door. Multiple cameras were placed both inside and around the outside of the room. Real image sequences from which results were presented in the

paper were significantly less complex that this, however.

An alternative approach was presented in [20]. Generalised Voxel carving is a conservative method whereby a mapping from each image pixel to a set of candidate voxels which may have contributed to that pixel is maintained. For this purpose two data structures are proposed, the item buffer and layered depth images. The former making the most efficient use of memory whilst the latter sacrifices memory usage in order to gain speed (the item buffers update visibility information less frequently, and whilst this does not affect the correctness of the algorithm, it means that voxels can remain in an undetermined state for longer, increasing the number of consistency evaluations).

This means that visibility is determined exactly, using every possible image, which ought to produce better results than Space Carving. This time results from real images consisting of more that one object are presented. Whilst the reconstructions are incomplete an advantage over the space carving algorithm is evident.

One problem with these methods is that real surfaces are rarely perfectly Lambertian, thus genuine voxels will not appear to be exactly the same colour in each image, thus a threshold is used. If the threshold is set too high the model will expand and detail will be lost. The situation is much worse when the threshold is set too low, however. If part of the forground is deemed inconsistent, anything behind is also likely to be deemed inconsistent as it will considered to be unoccluded. This creates a knock-on effect, ultimately leading to holes being carved right through the model.

A method for automatically determining the threshold is presented in [47]. The idea of hard/frontier points is once again employed to solve the threshold problem. Recall that such points occur when the object shape touches the visual hull and thus their location can be relied upon. By examining the photoconsistency of these points (which are assumed accurate) an estimate of the variation in colour due to non lambertian reflectance is reached, and a value of the threshold sufficient to take account of this variation can be set. This method can still suffer from threshold problems if many surfaces with different properties are present, and it doesn't provide any means to deal with situations that cause a very high threshold to be set.

A better solution is presented in [55] which allows the threshold to be altered after the carving of the volume is performed. Instead of recording whether or not a certain voxel exists in the model, their algorithm calculates the lowest threshold that would allow that voxel to be photoconsistent. Thus the results of applying multiple thresholds are "embedded" in the same voxel set. Upon completion

of the carving the user can then choose the appropriate threshold, or at determine a trade off between accuracy and completeness that is appropriate for the application without having to reprocess any of the voxels

Another novel solution to this problem is presented in [15] which frames the problem of estimating the existence of voxels (and thus the shape of the scene) in a statistical framework. As in [28], Bayes' rule is used to decide the existence of each voxel from the available data by marginalising the existence of possible occluding voxels. Some approximations are necessary to render this calculation tractable. The full set of visibility configurations are not examined for each voxel but rather the most probable case is chosen from the best 1-view, 2-view, ... n-view configurations.

Once likelihoods have been calculated for all voxels, a concrete voxel model can be created by calculating the voxels most likely to be responsible for each image pixel. This guarantees no holes appear in the model since every pixel is accounted for, although it does require the volume from which the voxels are drawn to completely enclose the scene. The real images used by the authors have the background manually removed for this reason, effectively creating an additional constraint in the form of a shape from silhouette approach.

The paper presents results obtained from a synthetic scene as well as images of real objects. The reconstructions produced are very convincing although the objects are shot against a black background and thus some of the accuracy may be accounted for by silhouette extrusion alone.

All of the voxel colouring processes described so far require the camera positions corresponding to each input image to be known in advance. This is typically achieved by rotating the object on a turntable in front of a fixed camera. [22] presents an algorithm which, by using a generic shape model which is successively refined, extracts the camera positions for a voxel reconstruction step.

A recent new approach to the space carving methodology was presented in [67]. The method is semi-supervised in the sense that an example silhouette is provided to the system by the user (it should be noted that many other algorithms require a complete set of silhouettes so this isn't considered a serious disadvantage). Once the algorithm is initialised there are no other parameters that to be carefully set.

From the initialisation a probabilistic set of silhouettes are created for the other images. Carving is achieved by calculating posterior probabilities that a voxel exists and propagating this information in an evidence combining setting. Instead of a regular grid a projective voxel lattice is used which ensures the voxels project to a similar area in the images even for parts of the scene that may be

further away from the cameras. Experiments on real world data demonstrates the effectiveness of the silhouette expansion step.

2.5 Surface Reconstruction

Voxel based methods are ultimately limited by the assumption of Lambertian reflectance. Since voxels represent a point sample of space, they have no orientation and therefor models which include a term based on the angle between a surface and light source to account for specular reflections (shinyness) cannot be used. In addition to this voxels are generally considered independently of eachother, making it hard to impose any global constraints. In order to more accurately represent opaque solid objects a closed 3D surface is required.

The factorisation method described in section 2.3 was applied to the surface reconstruction problem in [1]. Like many early surface-based approaches algebraic surfaces such as polynomials are used. These representations are however rather inflexible in this domain, their ability to represent arbitrary shapes is severely limited by the maximum number of terms and degree of the polynomial.

Mesh, or polygon, representations very popular in computer graphics due to the ease with which they can both be rendered and created. Meshes consist of a series of connected vertices which describe a piecewise linear surface. It is possible to create a mesh from a set of isolated points provided by another algorithm. The crust algorithm [2] performs this operation using the medial axis transform. Successful operation requires the points to be sufficiently dense otherwise there may be ambiguities in the triangulation. The crust algorithm was used as a post processing stage to generate complete surface representations from voxel sets in [23].

When using meshes, attention must be paid to the distribution of the constituent vertices and their connectivity, to prevent innefficient or degenerate meshes (where polygons intersect each other). In [25] a framework is presented to deform and evolve a surface based on partial differential equations. Many operations are required to maintain the integrity of mesh based representations in this framework, including edge swapping, splitting and merging, and the application of a Laplacian smoothness operator to ensure numerical stability.

A contrasting approach comes in the form of implicit surfaces whereby the surface is described by the zero level set (that is, the set of points for which $f(x) = 0$) of a function. This formulation allows interesting mathematical properties (such as smoothness) to be enforced as well as automat-

ically handling any changes in topology which might occur as a surface is manipulated to fit the image data.

In [82] the authors define such a function as the sum of a set of simple radial basis functions (RBFs). They also demonstrate how such surfaces can be created from surface points and how specifying exterior points allows a greater control of the resultant implicit surface. Whilst their work was aimed toward creating 3D surfaces by hand, the same approach has been applied to the problem of reconstructing continuous surfaces from laser scan data, which typically consists of many points in space (a point cloud). The dense nature of the point cloud removes the need for carefully placed external constraints and results in very detailed reconstructions [16].

More recently, in [23] RBF surface reconstruction was applied to the results of volumetric algorithms. Here the positions of the solid voxels are used as the point cloud and thus attention must be paid to the increased level of noise associated with this type of data. The paper provides a comprehensive background into the use of RBFs and other surface construction and spatial smoothing methods, and demonstrates how the principals may be applied to smooth voxel data. The paper concludes that interpolants which minimise several orders of smoothness improve both the quality of the reconstruction and the numerical stability of the results. Results are presented of fitting surfaces to voxel sets produced by the generalised voxel colouring algorithm [20].

Direct fitting of an implicit surface to image data is addressed in [27]. The authors proceed by defining a cost functional based on consistency with the input images. From this cost functional a set of PDEs (partial differential equations) are obtained through the Euler-Lagrange equation. These PDEs are used to evolve a surface toward the minimum of the cost functional using the level set framework [63]. Although the authors present an elegant derivation from a strong theoretical standpoint, few results are presented and no quantitative analysis of the performance of their method is presented.

RBF modelling of surfaces is also used in the related field of computer graphics as an efficient means to represent smooth surfaces. In [17] methods of fitting RBFs to range data in order to smooth noise and interpolate across gaps using polyharmonic splines is presented. Compression in the space required by the representation is provided by means of a greedy algorithm which iteratively increases the number of basis functions until a sufficient closeness to the data is achieved.

Results are provided on several well known objects such as the Stanford Buddha and dragon datasets, as well as range data of medium and large scale structures. Whilst the range data contains

noise, it is orders of magnitude better ordered than typical voxel carving data so these techniques may not be applicable to the 3-D modelling from images domain.

2.6 Photorealistic Object Modelling

This section discusses a number of papers published with the goal of moving toward photorealistic 3-D representations of the real world (or part thereof). The majority of cases concentrate on single objects in isolation.

As discussed previously, when dealing with single compact objects the visual hull provides a very useful constraint of the shape of an object. In [46], a stochastic algorithm is presented to improve surface provided by the visual hull and recover a representation of the objects colour/texture in the form of a texture map (a 2D image that is stretched over a polygon mesh).

At all times the object silhouettes are treated as hard constraint, by fixing in place the frontier points where the actual shape grazes the visual hull. As before these located by determining where the visual hull is photoconsistent. The model is refined by applying free form deformations at vertices sampled at random, weighted by their reprojection error. These deformations pull the model toward the true surface by performing a linear search for the lowest error. The texture is then recalculated using the current shape estimate. The algorithm successfully recovers the concavities missing from the visual hull although overall the models lack detail - the way in which the initial mesh is deformed prevents the final shape from increasing significantly in complexity. The authors note that the quality of the results are also dependant on the object having significant texture (that is, variations in colour across the surface).

A common thread in a large number of the 3-D vision algorithms presented so far is the formulation of reconstruction as an optimisation problem, where a shape is sought which has a maximal consistency with the input images. Usually the consistency error function that is to be minimised is too complex for a global minimum to be found efficiently. Optimisation by graph cuts, popular in stereo vision has also been applied to the problem of 3-D surface reconstruction in a method presented in [83].

Graph cuts allow the global minimum of a certain class of cost functions defined over a set of binary variables to be found in fast polynomial time. This is a significant improvement over exponential time algorithms. In a graph cut optimisation problem the variables are represented as

nodes in a weighted graph with two special nodes, the sink and source. Optimisation is achieved by finding the minimal cut, that is, a partition of the nodes into two groups (each containing either the sink or source) which minimises the sum of the weights crossing the cut. In order to represent a 3-D surface, each variable/node is a voxel with a value of [0,1] depending on whether the space occupied by the voxel is solid or empty. The minimal cut then separates the solid inside from the empty outside and thus represents the surface of the object.

In [83] a suitable cost function is proposed which maximises both smoothness and photosconsistency, however the graph weights cannot be modified during the optimisation so the photoconsistency of each point cannot be updated to reflect changes in visibility (recall that points are only required to be photoconsistent if they are unoccluded). The authors solve this problem by introducing the notion of approximate visibility: given an initial estimate of the shape (such as the visual hull), the visibility of points within the shape is assumed to be the same as the visibility of the closest point on the initial surface. Provided the real surface is close to this estimate this assumption holds but for extreme viewing angles.

One problem the authors recognise with this scheme is that the graph cut optimisation produces the surface with the lowest total cost. This can result in thin structures being truncated as although the cost per unit area is low, the large areas push the total cost up. To combat this the authors introduce a "ballooning term" which penalises small volumes and appears to solve the problem although they concede that this parameter has to be set carefully by hand. Another limitation of their approach is that to reduce the number of nodes in the associated graph, the final surface is constrained to lie a within a certain distance from the initial outer surface, limiting the amount of carving the algorithm can perform.

Another method, presented in [33], also uses graph cuts to carve models however their approach differs in several important areas. Rather than just using the visual hull as an initialisation, silhouette consistency is enforced as a hard constraint throughout the modelling process just as in [46]. This helps prevent protrusions being truncated as in the previous case, as these structures often contribute to the visual hull, being the most outward located parts of a surface. It is noted that frontier points will not occur sporadically but will comprise a series small arcs across the surface, referred to as "rims". These are identified using dynamic programming to find the shortest path along which the an image discrepancy measure, defined as the photoconsistency over a small neighbourhood, is minimised.

Another significant contribution is an iterative local refinement procedure, performed after carving the visual hull using graph cuts, in which each vertex of in the model is moved inward or outward according to the derivative of the image discrepancy function in addition to smoothness and silhouette consistency forces. This process is similar in spirit to [46] it differs in the criteria used to move each vertex and the fact that the vertices begin much closer to the ground truth. This step is intended to recover fine details in the surface, and thus the fact that the surface doesn't move very far during this procedure ensures stability in the mesh. In a series of steps the vertices are repeatedly subdivided increasing the resolution of the model until each triangle projects to an area approximately the size of a pixel in the images.

The models obtained from their method are indeed very detailed however the images must be captured under carefully controlled conditions to facilitate extraction of the silhouettes, and there are still limits to the amount of deviation from the visual hull that can be tolerated.

A recent paper [51] proposes an alternative way to use graph cuts which permits the use of a photometric minimisation without the potential loss of fine structures. Instead of constructing a network for the entire shape, the optimisation proceeds in an iterative fashion by carving in a series of concentric bands. This allows for much more accurate surface normal estimation (as the area being carved is much closer to the most recent normal estimates) which in turn results in a more accurate photo consistency term. Results presented on the same datasets reveal an increase in both accuracy and completeness of the models when compared to the algorithms of [33] and [83].

A related technique to these papers is presented in [49]. Here a global minimisation is achieved by means of a convex functional, the key contribution of this paper being this convex formulation. Hard silhouette constraints limit the range of possible functions. What is unique about this method is that exact silhouette and visibility constraint are applied negating the need to biases such as the ballooning term of [83] in the minimisation. The formulation also allows the algorithm to be parallelised and implemented on commodity graphics hardware where the authors report solutions can be obtained in under a minute.

So far all the algorithms presented have assumed the target object to be Lambertian, occasionally taking steps to handle some amount of specularity (for example using thresholds, or selecting the subset of cameras to use for a particular point to avoid oblique angles [33]).

Attempts have been made to not only explicitly account for non-lambertian reflectance but to also

recover the reflection properties themselves. An example of this idea can be found in [90]. Their approach is based around the bi-directional reflectance distribution function (BRDF) for specular surfaces. The BRDF describes how much light is reflected as a function of the incidence angle and viewing direction, however, rather than derive both the lighting conditions and surface properties, the authors define a view independent reflectance map (VIRM) which describes their combined effect, and is sufficient to predict the observed appearance of the surface.

Optimisation is performed alternately on the shape and VIRM using the Levenberg Marquardt algorithm to minimise the least-squares error between the predicted and observed reflectance. A multi-scale coarse to fine approach and surface initialisation are required for the algorithm to run efficiently.

This technique is effective in capturing the characteristics of specular surfaces and provides convincing novel views, however it relies on the assumption that the object is constructed from a single material. The method also ignores the possibility of self-shadowing or interreflection between parts of the same object. Results were demonstrated on highly specular surfaces, for which other techniques would be expected to perform very poorly. It is not clear whether this approach offers any benefits when the objects are only slightly non-lambertian.

2.7 Multiple Object Scenes

A strong pattern which emerges from the previous sections is that the current state of the art of image based reconstruction has predominantly focused on single objects. Most work extracting 3-D information from multiple objects concerns the case of a fixed [video] camera and independently moving objects [45, 3, 62]. The purpose of such systems lies in surveillance and autonomous navigation applications. This is a different problem domain to photorealistic modelling although some similarities exist.

It is possible to argue that at the highest level there is no difference between a scene consisting of a single object and one in which several objects appear, all connected by the floor, or a table, which is itself an object and thus part of the scene. The distinctions between objects are artificial - as far as the reconstruction is concerned they might as well be glued down. However, the intuitive human classification of multiple objects does hold true in the sort of datasets used by researchers. In the case of single object algorithms presented in the previous chapter, shapes are usually compact,

with a plain or uncluttered background which is easily distinguished from the object of interest.

Some of the problems associated with multiple object scenes are addressed in [9]. Multiple depth maps are used to produce an entity known as the reduced depth hull of a scene containing multiple objects. Due to the use of expensive active scanning methods to produce the depth maps the work is not directly applicable to the topic i hand. However, the authors do introduce a more rigorous notion of what a constitutes a complex multi-object scene, as one for which there exists "a plane whose intersection with the scene consists of more than one connected region".

Explicit reference to multi-objects is made in [66]. A complete method for 3-D scene recovery is presented which first calculates the camera positions by means of point tracking across images and self calibration. Next, pairs of consecutive images are used as stereopairs and dense depth maps are computed. The key contribution here is the fusing of depth maps together into a single complete 3-D surface. The use of a stereo vision as an intermediate step whilst places a requirement for the camera positions to be very close together, does provide the means to model multiple objects without explicit initialisation.

Another key contribution towards multi-object modelling is that their method attempts to identify specular surfaces. One characteristic of multi-object scenes is the increased likelihood that there will be one or more non-lambertian surface somewhere in the scene. The authors attempt to identify such surfaces by classifying values at the tail ends of the appearance distribution as outliers. Results are presented on several real datasets consisting of images of the exterior of a building and remains of a Roman building.

Two interactive methods to model multi-objects scenes are presented in [76]. The first operates on a set of panoramic images and requires users to select a set of primitives (points, lines, planes) which are used to build a set of geometric constraints which provide the basis for the reconstruction. This process may be repeated in order to refine all or part of the model.

The second method makes use of an alternative representation for multiples scenes in the form of individual image layers at various depths. This is based on the assumption that individual objects can be approximated by a flat "cardboard cutout". The user interactively selects image segments corresponding to individual planar regions and then the camera pose and depth are estimated by exploiting redundancies in the images. Results are presented on real scenes both indoor and outdoor with effective models being created in the case of the first method in the presence of largely planar shapes (the interior of an office).

Several real world datasets are used and the authors provide a quantitative evaluation using ground truth data. The method is however tailored toward the sort of data expected for this application (modelling buildings) and may not generalise to other scenarios.

More research on modelling multiple object scenes has been undertaken at the very other end of the spectrum - where techniques are employed to attempt to model entire cities in some cases. In [30] a process is described to acquire models of a large scale urban environments using a pair of laser rangefinder devices mounted on the back of a truck. One is oriented vertically, to capture the depth profile of the city facade, the other one horizontally, pointing near ground level to aid in registering the vertical stripes. In addition to this the path of the vehicle is calibrated using aerial photographs.

A very similar vehicle mounted laser rangefinder system is used in [12] to create urban models. In this case the authors use the Global Positioning System (GPS) in order to obtain the trajectroy of the vehicle. Importantly, the paper also addresses issues associated with manipulating and rendering the data that has been acquired. As the size of scene being modelled grows, so does the rendering time. However, only some of the model will be visible at any time so the authors perform an octree decomposition of the 3D data, recording the opacity of the leaf nodes, and then render the visible parts of the scene in a front to back ordering. Another issue with highly detailed models is that the data is often too large to fit in the RAM of the machine. The authors also address this, using a statistical model of visibility from which data is put in a priority queue to be prefetched, thus hopefully avoiding delays caused by reading data from a hard disk. The authors report real-time rendering speeds, making the system suitable for virtual reality.

An approach using images alone is presented in [56]. The method is based on aerial photographs, extracted line features are interpreted as depth discontinuities (i.e. the boundaries of buildings where the height rises swiftly from the ground plane). This information is combined with dense depth estimates obtained from a traditional stereo vision approach. The key contribution is the fusing of this data into a representation based on geometric primitives (planes and surfaces of revolution) to form a polyhedral model, using the graph cuts global optimisation procedure. Several real world datasets are used and the authors provide a quantitative evaluation using ground truth data. The results are impressive although the method is however tailored toward the sort of data expected for this application (modelling buildings) and may not generalise to other scenarios.

Many of these methods covered in this section make extensive use of highly accurate laser

scanning devices. Such devices are priced in the tens of thousands of pounds whereas consumer digital cameras are available for under a hundred pounds. Modelling multi-objects scenes using only images captured from relatively inexpensive cameras remains very much an open topic.

2.8 Conclusion

The field of computer vision, and specifically 3-D vision has come a long way since the foundations were laid in the late 70s and early 80s. Direct analytical and geometric methods have given way to iterative optimisation approaches. Many different ways of representing scenes have been proposed, from simple point and line clouds to volumes to various methods to describe complete 3-D surfaces. The prevailing philosophy is that reconstruction takes the form of multi-step procedure whereby an initial representation is progessively improved in a number of stages. This permits different representations to be used at each stage each reflecting a particular trade off between accuracy speed or ease of implementation.

One very frequently occurring theme is use the visual hull to provide initialisation, due to the ease at which it can be computed for arbitrary shapes from a set of input images. Reliance on the visual hull can be limiting, however as it's closeness to the true shape can be dramatically reduced in the case of multiple objects which variously occlude each other and the centre of the scene. It can also be argued that if your scene is, for example, the interior of a room, no visual hull exists as everything is of interest and there is no "background".

Any task involving 3-D modelling from images inevitably requires the ability to process a large amount of data, as well as being able to cope with errors and omissions in the data. Modelling multiple object scenes is an area for which this is especially true. For this reason statistical methods will be important as will any approaches such as voxel carving which can operate without a good initial shape estimate. Implicit surfaces represented using Radial Basis Functions provide means smooth errors and also manage the data that has to be recorded by means of approximation and interpolation. These will be the areas which are explored and studied in the remainder of this thesis.

Chapter 3

Modelling and Correcting 3-D Scenes Using RBFs

3.1 Introduction

The vast majority of recent research into photorealistic reconstruction from images has focused on the case of isolated objects. The results that have been obtained so far are good [33], however for several applications we would like to be able to reconstruct scenes consisting of many objects/surfaces. This is a significantly more difficult problem, not just because of the increased complexity of the models, but for many other reasons detailed the following.

Input images to object modelling algorithms are usually captured under laboratory conditions, using a static camera in front of which the object revolves on a computer controlled turntable [33]. This allows the camera positions relative to the object to be known to a high level of precision. For the case of medium to large scale scene reconstruction, images will probably be captured using a handheld camera, the positions and orientations of which will be unknown. Even though many good algorithms for camera motion recovery have been presented [29] there is always some discrepancy between the results and ground truth. Other factors such as lighting are also much easier to control in the laboratory setting.

Recall from section 3.2.2 that the visual hull is provided by the intersection of many silhouette cones. It is effectively an outer-bound on the shape and is often used to provide an initial estimate for the surface [46]. For a wide variety of objects used in reconstruction the visual hull is very close

to the true surface. When the aim is to reconstruct a scene which surrounds the camera, there will be no silhouettes and thus it is not possible to calculate the visual hull and use this valuable constraint.

For some objects, image based reconstruction is simply not possible. This may be due to extremely complex topologies, or the result of semi-transparent, translucent or highly specular surfaces. For this reason objects used for reconstruction are carefully chosen. However, when reconstructing an entire scene, the inclusion of such objects may be unavoidable. Windows, for example, exhibit a large degree of specularity and are generally impossible to avoid when reconstructing buildings. Any practical scene reconstruction algorithm must not let these objects degrade the reconstruction of other parts of the scene, as frequently happens with voxel carving algorithms [20].

Finally, the images captured could potentially feature areas which are far away from the cameras and/or area of interest, e.g. in outdoor scenes in which the sky is visible. Regardless of wether the final reconstruction is intended to include these areas they must still be taken into account. With the exception of [78] the vast majority of current algorithms assume the scene lies entirely within a bounding box [75][15][33].
The scene reconstruction problem may be stated as follows:

> Given a set $\mathcal{P}$ of cameras which observe a scene simultaneously[1] and a set of corresponding images $\mathcal{I}$, construct a 3D surface $\mathcal{S}$, representing the scene, which minimises the difference between $\mathcal{I}$ and the projection of $\mathcal{S}$ from $\mathcal{P}$.

I chose to use radial basis functions for scene reconstruction due to their ability to interpolate when data is missing and the ability to incorporate information with varying reliability into a single framework, which also allows the level of detail to vary locally. All of these properties are essential to the scene reconstruction problem. Radial basis functions have been used for function approximation and are well established in the fields of pattern recognition and AI, where they are often used as processing units in neural networks.

Despite their many flaws voxel carving algorithms posses a key advantage over visual hull based techniques in that they can operate with no initialisation, that is, no a-priori knowledge about the scene. I will follow the approach outlined in [23] which describes fitting an implicit surface to a voxel dataset using RBFs. Note that in contrast to [23] where this is performed as a final step, I treat this procedure as an initial step towards recovering a complete model of a scene. Subsequent steps

[1] this is equivalent to a single moving camera which observes a static scene

will refine this surface by comparing its appearance in different images, under the principal that the true surface should recreate the input images exactly.

The content of this chapter is as follows. Section 3.2 begins by describing a mathematical model of the image formation process and explains how that can be used to infer 3-D information from 2-D images. Next the probabilistic space carving algorithm[15] is outlined followed by modelling implicit surfaces with RBFs. A novel method for correcting errors in such surfaces using disparities calculated from the input images is discussed in Section 3.3. Results from experiments using real data are presented and discussed in Section 3.4. Finally conclusions are drawn in Section 3.5.

3.2 Representing and Modelling 3-D Scenes Using Radial Basis Functions

3.2.1 Projection and Image Formation Models

In order to solve the problem stated above it is necessary to have a mathematical model of how a 3D scene is converted into a 2D image by the camera. The pattern of light recorded on the sensor[2] can be attributed to the action of two processes, defined by the radiometric and geometric models.

The radiometric model describes the interaction of incident light with the object surface, classified by its reflectance function. The true physical process is complicated and thus it is assumed that the surfaces of all objects in the scene are diffuse and obey Lambert's law - that is they reflect light equally in all directions.

Perspective projection is used by the space carving method described in section 3.2.2 (to project voxels into the images) and by the surface refinement procedure described in section 3.3 thus it is important to begin with an explanation of how it operates.

An image is the result of intersecting a ray from each scene point with the image plane (where the sensor is located). The perspective projection is based on a pinhole camera, a simple device which focuses light through a small aperture onto the image plane. Instead of tracing rays through the aperture, the model places a virtual image plane in front of the camera. The only difference is that images are flipped about the horizontal axis (see figure 3.1).

The image plane coordinates of a projected point $\mathbf{x} = [x, y, z]^{\mathsf{T}}$ can be calculated using similar

[2] I will assume throughout that a digital camera is used.

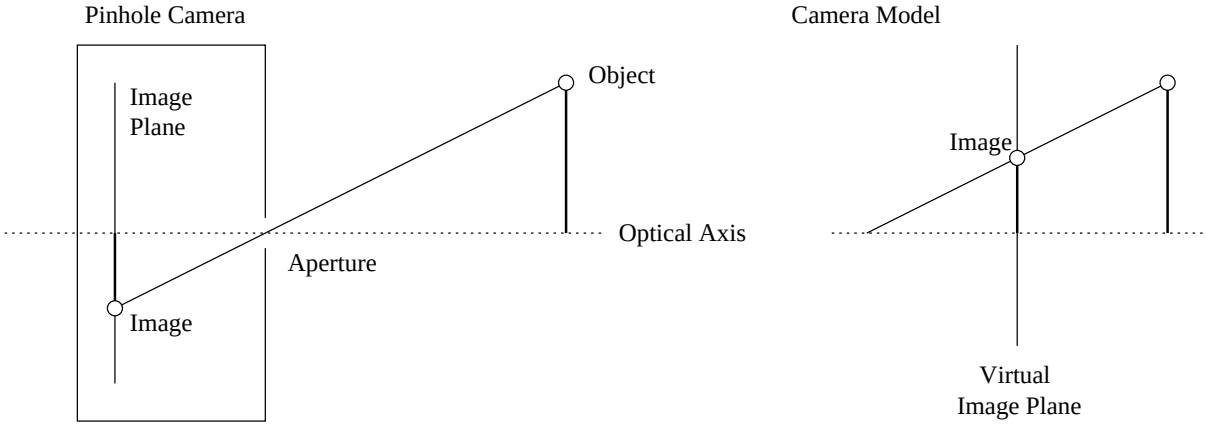
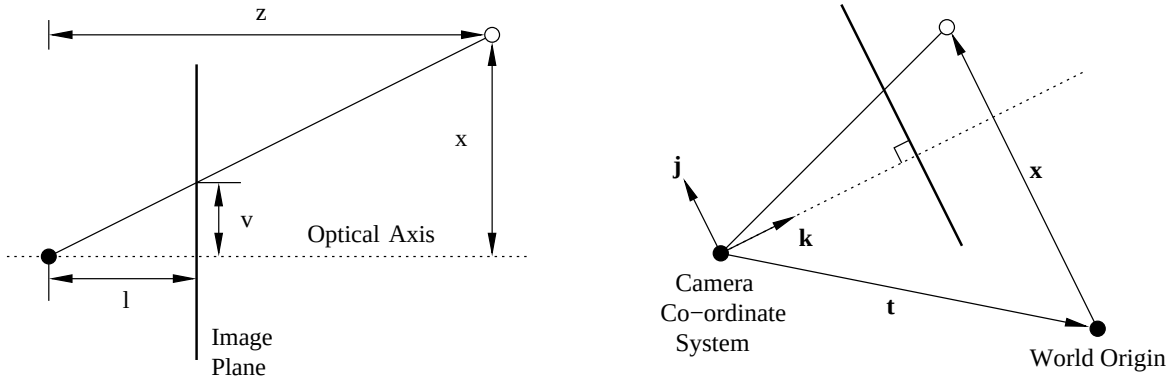


Figure 3.1: The pinhole camera and associated model



(a) (b)

Figure 3.2: (a) Projection geometry with similar triangles (b) the world centred view.

triangles:

$$u = l\,\frac{x}{z} \qquad\qquad v = l\,\frac{y}{z} \tag{3.1}$$

As several images must be used by the reconstruction algorithm, all scene points and camera positions must be specified relative to the world origin. The direction of the camera is given by three mutually orthogonal unit vectors **i**, **j**, and **k** which represent the orientation of the image plane. The

position of a point relative to the camera is found by adding world origin, $\mathbf{t} = [t_x, t_y, t_z]^\mathsf{T}$. The projection of point $\mathbf{x}$ is now given by

$$u = l\,\frac{\mathbf{i}\cdot\mathbf{x}+t_x}{\mathbf{k}\cdot\mathbf{x}+t_z} \qquad\qquad v = l\,\frac{\mathbf{j}\cdot\mathbf{x}+t_y}{\mathbf{k}\cdot\mathbf{x}+t_z} \tag{3.2}$$

Image plane coordinates must be transformed into pixel coordinates. Since we measure pixel coordinates from the top left corner, the u and v values are normalised by subtracting the coordinates of the image centre, $[u_0, v_0]$. Cells in the camera's sensor may not be exactly square and so a scale factor s is applied to the horizontal coordinate (the vertical scale factor can be fixed as 1 since it is co-linear with the focal length).

$$u = l\,\frac{\mathbf{i}\cdot\mathbf{x}+t_x}{\mathbf{k}\cdot\mathbf{x}+t_z} + u_0 \qquad\qquad v = sl\,\frac{\mathbf{j}\cdot\mathbf{x}+t_y}{\mathbf{k}\cdot\mathbf{x}+t_z} + v_0 \tag{3.3}$$

this can be expressed as a matrix equation in homogenous coordinates

$$\begin{bmatrix} u' \\ v' \\ h \end{bmatrix} = \begin{bmatrix} l & 0 & u_0 & 0 \\ 0 & sl & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} & \mathbf{i}^\mathsf{T} & & t_x \\ & \mathbf{j}^\mathsf{T} & & t_y \\ & \mathbf{k}^\mathsf{T} & & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.4}$$

$$\begin{bmatrix} u' \\ v' \\ h \end{bmatrix} = \mathbf{P} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \tag{3.5}$$

where $u = u'/h, v = v'/h$. The matrix $\mathbf{P}$ fully specifies the transformation from 3D into 2D coordinates and for this reason it referred to as a "camera".

3.2.2 Probabilistic Space Carving

The probabilistic space carving algorithm [15] is a variant of the voxel carving approach, in which voxel occupancy is determined by utilising statistical methods to overcome the uncertainty in localisation of voxels, avoiding the need for a specific, pre-defined threshold for voxel occupancy. In common with other methods a cuboid surrounding the scene is discretise into a 3D array of voxels [75]. In [15], each voxel is represented by a spherical Gaussian distribution in a 3D colour space. The output $\mathcal{V}$ is a subset of these voxels. In probabilistic space carving, Bayes' theorem is used to calculate the likelihood that each voxel is part of the scene (and should be included in $\mathcal{V}$) given the input data

$$P(\mathbf{x} \in \mathcal{V} \mid \mathcal{I}, \mathcal{P}) = \frac{P(\mathcal{I}, \mathcal{P} \mid \mathbf{x} \in \mathcal{V}) P(\mathbf{x} \in \mathcal{V})}{P(\mathcal{I}, \mathcal{P} \mid \mathbf{x} \in \mathcal{V}) P(\mathbf{x} \in \mathcal{V}) + P(\mathcal{I}, \mathcal{P} \mid \mathbf{x} \notin \mathcal{V}) P(\mathbf{x} \notin \mathcal{V})} \tag{3.6}$$

where, in absence of any data, the prior probabilities are assumed to be $P(\mathbf{x} \in \mathcal{V}) = P(\mathbf{x} \notin \mathcal{V}) = 0.5$. Other values may be used to bias the resultant voxel model.

The difficulty in this approach comes from the relationship between voxels which arises due to occlusion. Calculating $P(\mathcal{I}, \mathcal{P} \mid \mathbf{x} \in \mathcal{V})$ and $P(\mathcal{I}, \mathcal{P} \mid \mathbf{x} \notin \mathcal{V})$ requires the existence of all voxels which might occlude the voxel at $\mathbf{x}$ to be marginalised. This is generally intractable as there are $O(2^{n^3})$ cases (where n is the dimension of the bounding cuboid). This may be reduced by the observation that only voxels that lie on the line connecting $\mathbf{x}$ and camera centre will affect it's visibility, however this still leaves $O(2^n)$ visibility configurations (where n is the number of images). To obtain a practical solution, a local threshold γ is introduced and only cameras with likelihood greater than γ are included in the calculation. The value of γ is varied to find the most probable visibility configuration from the best 1-view, 2-view, ... n-view configurations. Further efficiency gains are obtained by processing voxels in a fixed ordering from front to back (just as in [75]) as visibility evaluations may be cached.

Once likelihoods have been calculated for all voxels, a solid model can be created by calculating the voxels most likely to be responsible for each image pixel. To do this, new views are rendered corresponding to each of the input images. For each pixel, a ray $\mathcal{R}$ is defined containing the set of voxels which intersect the line formed by back-projecting the the pixel into the volume. For each ray, the voxel with the highest likelihood, $P(\mathbf{x} \in \mathcal{V} \mid \mathcal{I}, \mathcal{P})$ is added to $\mathcal{V}$, yielding a complete model

which when projected matches the input images as closely as possible. Each voxel is characterised by a 6 element vector containing its position in space and three colour co-ordinates.

3.2.3 3-D Surface Reconstruction

3.2.3.1 Fitting Implicit Surfaces using RBFs

The voxel model $\mathcal{V}$ obtained from probabilistic space carving is noisy and contains many discontinuities and other artifacts which are undesirable in a 3D scene model. Radial basis functions allow both approximation and interpolation of data as well as smoothness and regularity constraints to be imposed. The general form of a function f represented as a sum of radial basis functions is

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i \phi(|\mathbf{x} - \mathbf{c}_i|) + F(\mathbf{x}) \tag{3.7}$$

where there are n radially symmetric basis functions, ϕ, each with weight w and centre $\mathbf{c}$. $F(\mathbf{x})$ is a polynomial which spans the null space of ϕ. If the basis function ϕ is positive-definite, $F(\mathbf{x})$ is a constant.

The form of this function is derived from the study of scattered data interpolation, whereby a function is sought to match a sparse set of observations. This problem is ill posed since there are an infinite number of functions which could account for the observations. To obtain a solution it is assumed the unknown function is smooth. In [35] it is shown that equation (3.7) is obtained from variational principals as the minimum of

$$H[f] = \beta[f] + \frac{1}{\lambda} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 \tag{3.8}$$

where $\beta[f]$ is the smoothness functional, y_i is the observation at point $\mathbf{x}_i$ and λ is a parameter which weighs smoothness against data closeness - defined as the sum-of-squares error in the approximation, f. The basis function ϕ depends on the smoothness functional, examples are given in the next section.

If each radial basis function centre, $\mathbf{c}$ is chosen to coincide with the location $\mathbf{x}$ of an observation, then the weights may be calculated by a linear system of equations

$$\begin{bmatrix} \phi(r_{11})+\lambda_1 & \cdots & \phi(r_{1n}) & 1 \\ \vdots & \ddots & \vdots & \mathbf{1} \\ \phi(r_{n1}) & \cdots & \phi(r_{nn})+\lambda_n & 1 \\ 1 & \mathbf{1} & 1 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_n \\ p_0 \end{bmatrix} = \begin{bmatrix} f(\mathbf{c}_1) \\ \vdots \\ f(\mathbf{c}_n) \\ 0 \end{bmatrix} \tag{3.9}$$

where $r_{ij} = |\mathbf{c}_i - \mathbf{c}_j|$ and λ terms added to the diagonal to allow approximation by letting f deviate from the observations [84].

An implicit surface $\mathcal{S}$ is described by the zero level set of some real-valued function f on $\mathbb{R}^3$, $\mathcal{S} = \{\mathbf{x} \mid f(\mathbf{x}) = 0\}$. For example, the function $f(\mathbf{x}) = |\mathbf{x}| - 1$ describes a sphere of radius 1 ($|\cdot|$ represents the Euclidean norm on $\mathbb{R}^3$). The surface function is sometimes called a signed distance function as negative values represent the inside of the object and positive values represent the outside (note this is not a metric distance).

For 3D implicit surface reconstruction, this signed distance function can be approximated by RBFs. The observations are a set of points which lie on the 3D surface and so have value zero (these will be referred to as surface constraints). In addition to this, a small number of external or internal constraints must be specified (with positive or negative values) to provide orientation to the surface and also prevent the trivial solution that $f(\mathbf{x}) = 0$.

In [23] the co-ordinates of voxels are used as surface constraints. It is not feasible to use all of the voxels in $\mathcal{V}$ for this, some method of selecting a subset is required. A regular subsampling of the discrete volume that $\mathcal{V}$ is drawn from, cannot be used, as the matrix in equation (3.9) becomes singular if the centres are colinear. For this reason a Poisson sphere random sampling scheme (the 3D analogue of the Poisson disc [52]) is used. This is an iterative procedure. At each step a voxel is chosen at random from the set of remaining voxels. The position of this voxel is used as the centre for a radial basis function. A sphere of radius ρ is centred on this location and all voxels inside the sphere are removed from contention. Another voxel is chosen and the process continues until there are no voxels remaining. This creates an approximately uniform distribution of RBF centres across the surface of the voxel model and ensures no two spheres can overlap. Exterior constraints may easily be generated by finding regions of empty space between the voxels and camera positions.

We would like to be able to use as much of the voxel data as possible. To this end, equation (3.9) can be formulated as an overconstrained system allowing more surface constraints to be specified

for the same number of RBFs. However, a value of the surface function that is close to zero doesn't necessarily correspond to a point in space that is close to the surface, hence the minimal sum-of-squares solution to the equation will not yield the surface that best fits the constraints. Any non-linear method to minimise the Euclidean distance between the constraints and surface would almost certainly be infeasible due to the extremely high dimensionality of the problem - for most scenes there will be thousands of weights that would have to be optimised simultaneously.

A better approach is to use an averaging scheme as this will not increase the number of terms in equation (3.9).Due to the one to one correspondence between the spheres and basis functions the distribution of voxels within the sphere can be used to determine the precise location of the RBF centre. So instead of just choosing the voxel in the centre of the sampling sphere, the mean of the coordinates of all voxels inside the sphere is calculated before they are removed from the selection pool. A more robust scheme is to replace the mean with the vector median, defined as the point to which voxels in the sphere have the smallest Euclidean distance [4], as this will remove some of the noise present in $\mathcal{V}$.

Outliers in $\mathcal{V}$, voxels which do not correspond to actual surfaces, will create errors in the reconstructed surface. These occur as a result of noise in the images, deviations the assumption of lambertian reflectance and camera calibration errors. The photoconsistency measure used to decide whether a voxel is carved is compromised by these deviations and due to the fact that higher voxels (which are closer to the cameras) are processed first, this can cause floating voxels to appear above the surfaces and in the gaps between objects. As the formulation of (3.9) allows the value of λ to be set for each surface constraint in turn, the voxel likelihoods computed by the probabilistic space carving algorithm can be used to weight the contribution of each constraint, under the assumption that the most strongly photoconsistant voxels should be favoured:

$$\lambda_n = \frac{\eta}{P(\mathbf{x}_n \in \mathcal{V}|\mathcal{I}, \mathcal{P})} \tag{3.10}$$

where η is a scaling parameter dependant on $\phi(0)$.

3.2.3.2 Choice of Basis Function

The RBF framework permits the use of any basis function which is positive definite or conditionally positive definite (this requirement ensures the matrix in equation (3.9) is non-singular). There are many such functions which have been applied previously in different situations. Thin plate splines often used in 2D applications have been shown to perform very poorly in 3D [61]. A common choice is the Gaussian: $\phi(r) = e^{-r^2/\sigma^2}$ where r is the radius $|\mathbf{x} - \mathbf{c}|$ and σ the width. This function is widely used in radial basis function networks, a technique used in pattern recognition. It was also one of the first basis functions to be used to create implicit surfaces, it's use in this field originally inspired by electric field potentials [8]. It has been noted, however, that when applied to surface reconstruction, the Gaussian has a tendancy to both oversmooth and also lead to gaps in the surface - instead the use of a multi-order basis function was proposed in [23]

$$\phi(r) = \frac{1}{4\pi\delta^2 r}\left(1 + \frac{we^{-\sqrt{v}r}}{v-w} - \frac{ve^{-\sqrt{w}r}}{v-w}\right) \tag{3.11}$$

$$v = \frac{1+\sqrt{1-4\tau^2\delta^2}}{2\tau^2} \qquad w = \frac{1-\sqrt{1-4\tau^2\delta^2}}{2\tau^2} \tag{3.12}$$

This function, derived in [18], imposes a combination of first, second and third order smoothness

$$-\delta\Delta f + \Delta^2 f - \tau\Delta^3 f = 0 \tag{3.13}$$

determined by varying the parameters τ and δ. This approach allows more detail to be retained in reconstructions without sacrificing smoothness.

For these basis functions, each evaluation of the surface function $f(\mathbf{x})$ requires $O(N)$ evaluations of the basic function, thus calculating the entire surface $\mathcal{S}$ at resolution R is $O(NR^3)$. The Gaussian and other similar basis functions fall away sharply as the radius r increases. This observation lead to the use of compactly supported basis functions which evaluate to zero for all values of r above a certain threshold (the radius of support) e.g.

$$\phi(r) = \begin{cases} (1-r)^4(4r+1) & \text{if } r < 1 \\ 0 & \text{if } r \geq 1 \end{cases} \tag{3.14}$$

here, the radius of support is 1 although r can be scaled to give any desired radius of support [61]. The advantage of this is clear - when evaluating $f(\mathbf{x})$ only the basis functions whose centre lies within a certain distance of $\mathbf{x}$ need to be summed. In addition to this, the matrix in equation (3.9) becomes sparse and this can be exploited to provide a more efficient solution to the system using the LU decomposition [61]. The time taken to evaluate the surface is thus dependant on the average density of the functions with respect to the radius of support. Reducing the radius of support allows more surface constraints to be used, at the expense of smoothness. Equation (3.14) was originally derived in [85] as the minimum degree polynomial for interpolation of a 3D function with C^2 continuity, which is guaranteed to be positive definite.

Unfortunately due to their very nature, compact functions have a limited ability to fill the gaps between RBF centres. It would be possible to combine compactly supported and non-compactly supported functions although the system matrix would no longer be sparse, limiting the number of basis functions that could be used. It is also unclear how the mathematical properties of the surface will be altered.

3.3 Updating Surfaces Using Image Disparities

Recall from Section 3.2.3.1 that the voxel model often contain many spurious results principally due to uncertainty in the illumination and camera positions. Whilst small scale protrusions and gaps are smoothed/interpolated over by the RBFs, larger scale deviations (in the form of large numbers of voxels in the wrong place due to being photoconsistent by chance) or parts of objects that are missing (due to over carving when changes is illumination violate photoconsistency) will be propagated through to the RBF surface. Thus some method of improving the surface using information from the input images is required. One disadvantage of implicit surfaces compared to meshes is that is not possible to locally adjust the surface directly, since it is described by the combined effect of many basis functions. However, the surface is constrained to pass close to the centres thus these points can be moved based on the local appearance of the surface. Once all the centres have been moved the

weights may be recalculated using equation (3.9).

Photoconsistency plays a central role in many algorithms for 3-D reconstruction. The photoconsistency of point $\mathbf{x}$ is usually defined as the variance in colour of the pixels p that it projects onto, in the images in which it is visible: $\mathrm{var}(\{p \mid p = \mathcal{I}^i_{\mathbf{Px}}\})$. Recall from section 3.2.1 that Lambertian reflectance was assumed. Lambert's law is a simplification and most surfaces exhibit some level of specular reflections, so that their appearance varies depending on the viewing angle. Because of this a threshold is required to decide whether a pixel is photoconstent.

Relying on photoconsistency alone can cause errors due to the possibility that arbitrary points will project to pixels of the same colour to within the threshold, especially if certain colours occur frequently in an image. Looking at the photoconsistency of all points contained within a small textured patch of the surface is far less likely to succumb to this ambiguity - provided there is sufficient variation in the colour across the patch.

In this section we assume the surface around each basis function may be locally approximated by a square planar patch (a valid assumption when the size of the patch is small), visible in two images. A plane in 3-D induces a homography $\mathbf{H}$ between two images [40], that is $\mathbf{H}$ is a mapping that transforms one image of the plane to match the appearance of the plane in the other image. As the surface is unlikely to be planar on a large scale, this assumption is only valid within small image windows around the projection of the RBF centre.

3.3.1 Homography Estimation by Block Matching

When the plane tangent to the RBF surface is not aligned with the true surface, $\mathbf{H}$ wont map one window onto the other precisely leaving a disparity between the resulting transformed windows. If this disparity can be measured and then taken into account it is possible to recover the parameters of this plane and thus know the location of a point on the true surface onto which the basis function may be constrained to lie. The following presents a method to recover the disparity by finding local offsets which maximise the consistency between windows.

The *observed* texture of the patch in image $\mathcal{I}^i$ is defined by projecting the four corners of the patch into the image, by using equation (3.3) (see figure 3.3).

The influence of perspective can be removed by interpolating between the four corners of the projection, rectifying to an $n \times n$ square orthogonal view of the patch $\mathcal{I}'^{i}$. Texture consistency, T, can now be evaluated as average variances of corresponding pixels across all rectified image regions $\mathcal{I}'^{i}$.

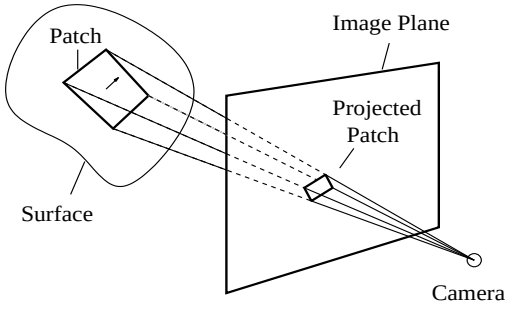
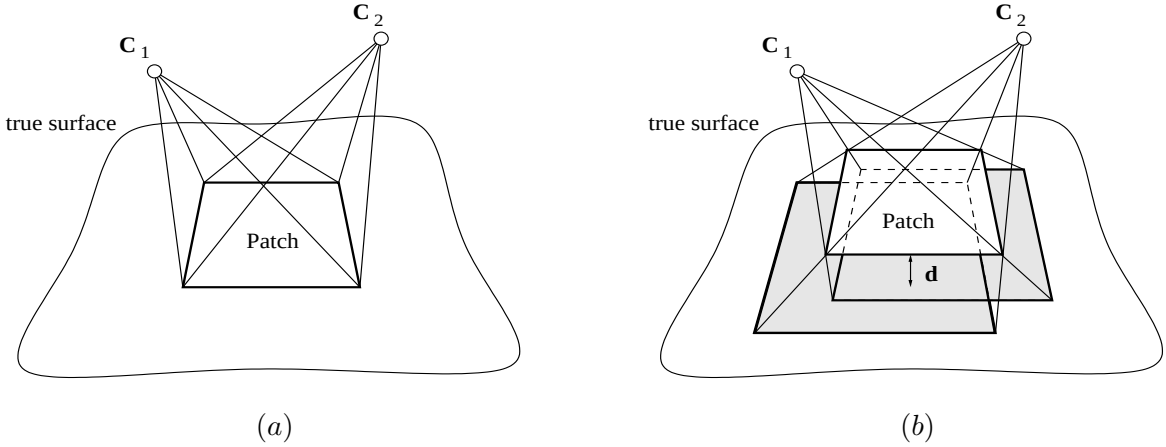


Figure 3.3: The projection of a surface patch

$$T = \frac{1}{n^2} \sum_{j=1}^{n} \sum_{k=1}^{n} \mathrm{var}(\{\mathcal{I'}^{i}_{jk}\}) \tag{3.15}$$

Observed textures are inconsistent when there is a misalignment between the surface $\mathcal{S}$ and the true surface observed by $\mathcal{P}$ (figure 3.4). Rather than using the consistency alone to guide an optimisation algorithm, it is to compute a geometric solution to the displacement from the true surface (as mentioned above), from correspondences between pairs of observed textures. Let us assume a patch is observed as $\mathcal{I'}^{1}$ and $\mathcal{I'}^{2}$ by two cameras $\mathbf{P_1}, \mathbf{P_2} \in \mathcal{P}$ with associated positions t_1 and t_2.



(a) (b)

Figure 3.4: (a) The patch is aligned with true surface (b) the patch is misaligned by a displacement, $\mathbf{d}$, due to errors in $\mathcal{V}$

Finding such correspondences is a fundamental problem in the field of wide baseline stereo [71].

Due to the potentially large distances between camera positions viewpoint invariant statistics about surface textures are used in matching. However if the displacement between the patch and true surface is small the transformation may be approximated by translation, a block matching algorithm can be used as in motion estimation [43]. The block matching scheme allows for changes in perspective between images by breaking up the image into smaller sections. The local change across the smaller sections is less pronounced.

The first observed texture image $\mathcal{I}'^1$ is broken into a set of smaller blocks of $q \times q$ pixels, and for each block the translation vector $\mathbf{b} = [b_1, b_1]$ is found which minimises the error:

$$\epsilon = \sum_{j=1}^{q} \sum_{k=1}^{q} (\mathcal{I}'^1_{j+b_1,k+b_2} - \mathcal{I}'^2_{j,k})^2 \tag{3.16}$$

Since the perspective projection preserves straight lines these vectors should form a regular field allowing the disparity to be removed.

The observed detail may be significantly reduced in patches viewed from oblique angles and also any deviation from planarity will be more pronounced. For this reason camera positions are given a score χ based on the sum of the angle between the surface, $\mathbf{x}$ and camera position, $\mathbf{t}$, and the area of the projection

$$\chi = \frac{\overrightarrow{\mathbf{n}} \cdot (\mathbf{t} - \mathbf{x})}{|\mathbf{t} - \mathbf{x}|} + \alpha \left| (u_4 - u_2)(v_3 - v_1) - (v_4 - v_2)(u_3 - u_1)) \right| \tag{3.17}$$

where u_1, v_1, u_2, v_2 etc. are the horizontal and vertical image coordinates of the corners of the projection (in a clockwise direction) and $\overrightarrow{\mathbf{n}}$ is the surface normal. The parameter α weighing between the two measures is kept small, as the area of the projection will be strongly related to the angle between the surface patch and the camera, and this term is only included to penalise cameras which are significantly further from the scene.

From these scores the two best positions may be selected to compute disparities, however camera positions which are too close together will not provide enough disparity to extract reliable information. This may be remedied by enforcing a minimum value for the angle formed between the centre of the patch and the two camera positions.

Even when these conditions are met this method may still fail. Even though each block is less sensitive to changes in perspective than the window as a whole, when the disparity between windows is large the method will fail. Making the blocks smaller reduces their succeptibility to this problem however the smaller a block is the more likely it is that a false match will be obtained, making it very hard to get a reliable result.

3.3.2 Texture Matching Under Perspective Projection

The block matching scheme presented above often fails in the presence of strong perspective effects between images. Several methods have been presented to match textures between images under the assumption of affine projection [7, 54]. The affine approximation is only valid locally which can cause these methods to give innaccurate results. While useful for finding initial matches when camera positions are unknown, they are not suitable for accurate correspondence between image regions. This section presents a method that explicitly takes perspective into account by first rectifying the two windows reduce the disparity to a linear one which may be accurately estimated.

Let $\mathbf{P}$ and $\mathbf{P}'$ be two 3×4 matrices which describe the camera projection from 3-D coordinates to homogenous image coordinates. Let $\mathbf{y} = [u, v, 1]^T$ be the projection of a point in the patch from the first camera and $\mathbf{y}' = [u', v', 1]^T$ be the corresponding point from the second camera. These points are related by $\mathbf{y}' = \mathbf{H}\mathbf{y}$. If the patch belongs to plane ψ, where $\mathbf{z}^T\psi = 0$ for all the points $\mathbf{z}$ which lie on ψ, the formula for $\mathbf{H}$ as given in [40] is :

$$\mathbf{H} = \mathbf{A} - \mathbf{a}\mathbf{v}^T \tag{3.18}$$

where $\mathbf{A}$ and $\mathbf{a}$ are a 3×3 matrix and a 3×1 vector, respectively, given by :

$$[\mathbf{A} \mid \mathbf{a}] = \mathbf{P}' \begin{bmatrix} \mathbf{P} \\ 0\,0\,0\,1 \end{bmatrix}^{-1} \tag{3.19}$$

and $\mathbf{v}$ is the vector given by the following expression :

$$\begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix} = \left(\begin{bmatrix} \mathbf{P} \\ 0\,0\,0\,1 \end{bmatrix}^{-1} \right)^T \psi \tag{3.20}$$

Given a point in one image the position corresponding point in another image can be constrained to lie on a line known as the epipolar line. Epipolar lines depend only on the imaging geometry and not the shape of the scene so it is possible to transform the images to correspond to a pair rotated 'virtual cameras' whose epipolar lines are all horizontal and co-linear. This process is known as rectification and is often performed as an initial step in stereo algorithms [34].

Let $\mathbf{R}$ and $\mathbf{R}'$ be the rectifying 3×3 matrix transformations. The rectified images of the patch are now related by

$$\mathbf{R}'\mathbf{y}' = \mathbf{H}_R\mathbf{R}\mathbf{y} \tag{3.21}$$

As the epipoles are now horizontal $\mathbf{H}_R$ is guaranteed to map v-coordinates to the same value in each pair and is thus of the form

$$\mathbf{H}_R = \begin{bmatrix} s & k & t \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.22}$$

where s, k and t, correspond to scaling, skew and translation, respectively (all in the u direction). To calculate these parameters, the images of the patch are divided into l rows of pixels. When considering a single row of pixels, the skew translation act together to produce a single horizontal offset, o, since the v coordinate of each pixel is the same.

The normalised cross correlation is computed between each pair of rows at different scale and offset values, and the values which result in the lowest score (best match) are recorded. As the scale should be the same for all rows, s is taken to be the median of the values found for each row. Any values significantly outside the median are deemed to be errors and discarded. Using the offsets from all rows, the skew and translation parameters k and t can be calculated by solving a linear system

$$\begin{bmatrix} k \\ t \end{bmatrix} = \begin{bmatrix} v_1 & 1 \\ \vdots & \vdots \\ v_l & 1 \end{bmatrix}^{-1} \begin{bmatrix} o_1 \\ \vdots \\ o_l \end{bmatrix} \tag{3.23}$$

where v_l is the v coordinate of row l. To improve the robustness further, the rows with the greatest residuals are removed from Equation (3.23) and k, t recalculated. This process is repeated until

convergence. Now $\mathbf{H}$ can be calculated from $\mathbf{H}_R$ taking into account the rectifying transformations :

$$\mathbf{H} = \mathbf{R}'^{-1}\mathbf{H}_R\mathbf{R} \tag{3.24}$$

With $\mathbf{H}$ at hand we calculate the vector $\mathbf{v}$ from (3.18) and consequently the location of the plane ψ that should contain the basis function using (3.20). The corrected position μ' for the basis function centre μ is calculated as :

$$\mu' = \mu + \overrightarrow{\mathbf{n}} \frac{\mu^{\mathsf{T}}\psi}{\overrightarrow{\mathbf{n}}^{\mathsf{T}}\overrightarrow{\mathbf{n}}} \tag{3.25}$$

where $\overrightarrow{\mathbf{n}}$ is the surface normal to ψ. The line by line matching based on cross-correlation requires that there must be significant detail in order to find the offsets uniquely. Thus some basis functions are not updated by this procedure. Additionally, false matches may be obtained due to noise in the images or patches which span the boundary of an object. A limit is placed on the maximum distance that a centre can move in order to prevent this from causing further errors in the surface.

The method works on pairs of images. Like the block matching scheme it is detrimental to use cameras which view the patch from an oblique angle, so a similar quality measure is used to ensure good camera positions:

$$\chi = \frac{\overrightarrow{\mathbf{n}} \cdot (\mathbf{t} - \mathbf{x})}{|\mathbf{t} - \mathbf{x}|} + \alpha|\mathbf{t} - \mathbf{x}| \tag{3.26}$$

With the only difference being that the distance from the surface patch is used directly instead of the patch area in the image.

Although the method can only operate on pairs of images, it may still benefit from the availability of other images. This happens by finding the top three ranking cameras according to Equation (3.26) and estimating the correction using images from cameras 1 and 2, 1 and 3, 2 and 3. In the absence of errors in the matching, each of these runs should produce the same answer, thus if one differs from the others it is discarded as an outlier. If all three differ then the updating of that centre is abandoned.

Some of the basis function centres will converge towards neighbouring locations on the 3-D surface causing singularity in the system matrix in equation (3.9). If multiple basis functions occur in the immediate neighbourhood of each other, only one will be preserved while the others will be removed.

3.4 Experimental Results

This section presents the results obtained from applying surface reconstruction, and investigating texture consistency using two sets of real images of multi-object 3-D scenes.

3.4.1 Experiment Setup and Data Capture

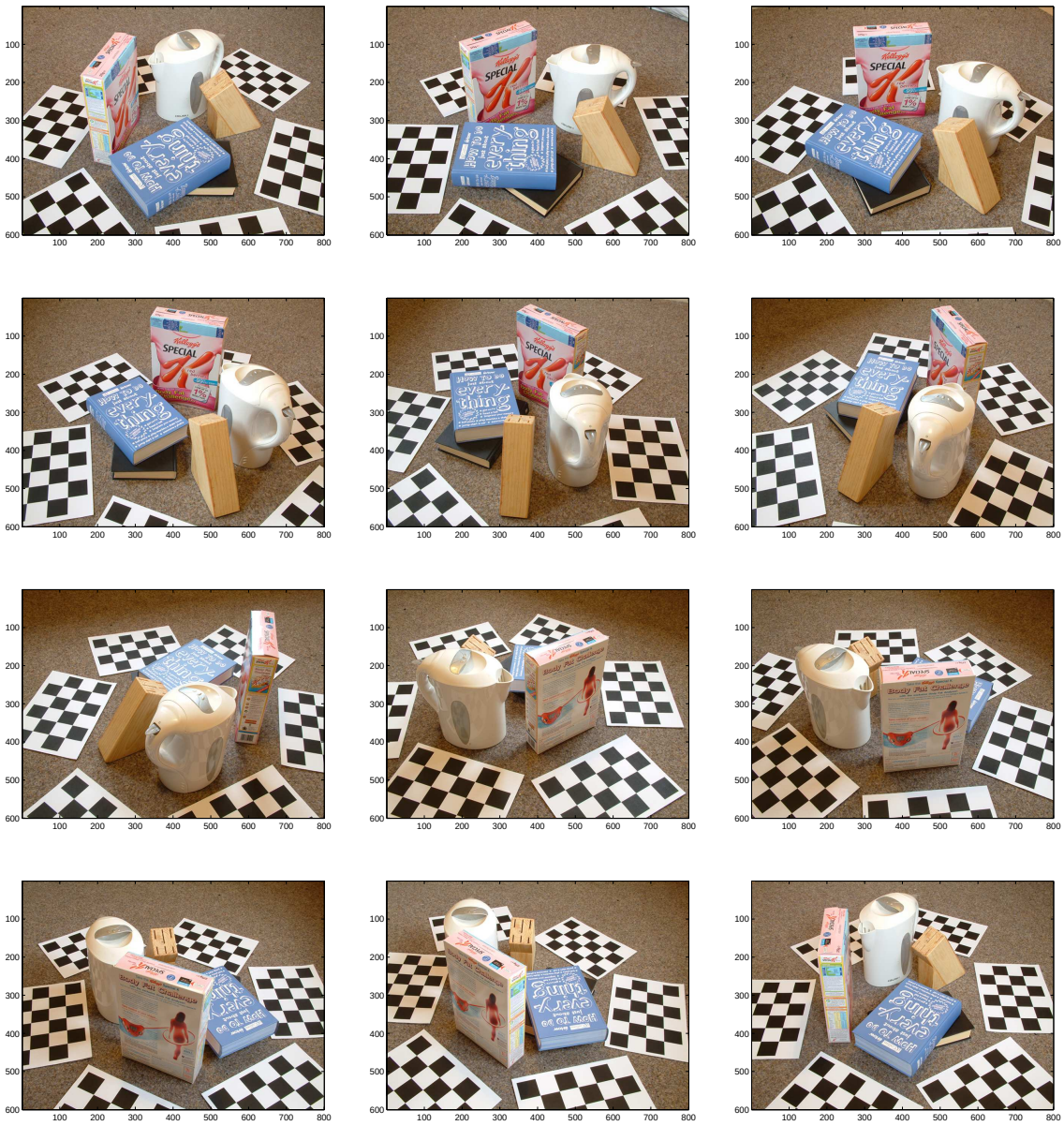


Figure 3.5: Original input images for scene 1

The first scene was captured using a 3-megapixel digital camera which had been calibrated according to the method outlined in [41]. For this sequence I placed 2D targets around the objects to provide reliable features from which to calculate the camera positions (note that it is possible to do this using features of the scene itself, automatically extracted and tracked [29]). In total 12 images were taken from positions surrounding the scene at approximately equal intervals (see Figure 3.5). The scene comprises five objects; a cereal box, a kettle, a wooden knife-block and a stack of two books.

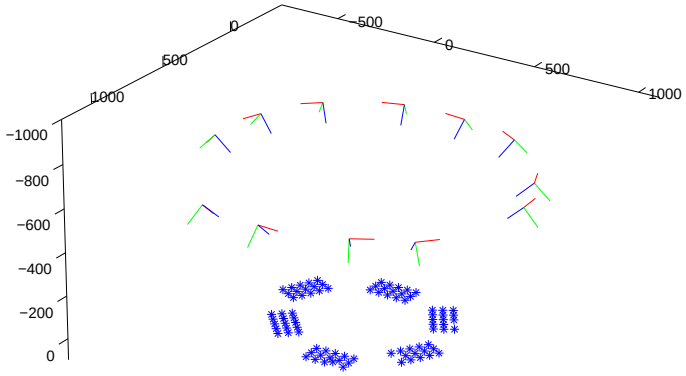
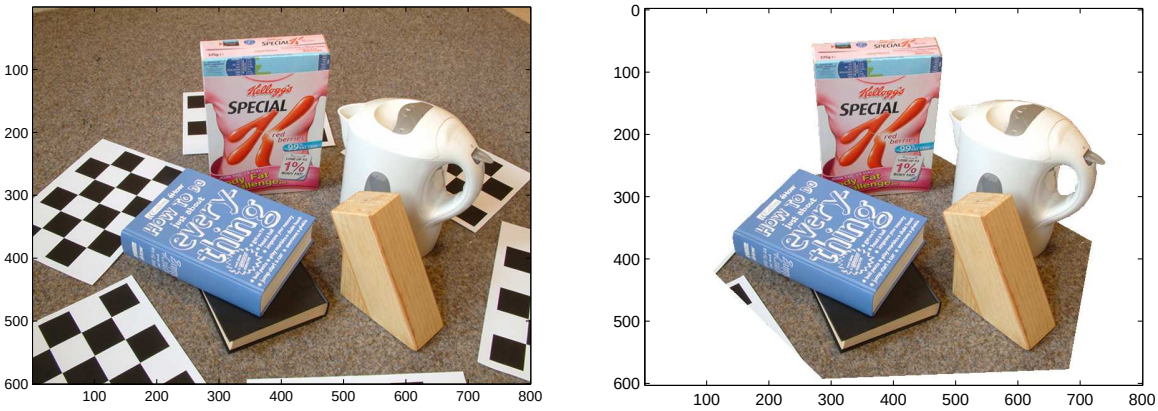


Figure 3.6: Recovered camera positions and tracked points from the first scene

Initial camera positions were calculated using the fast 8-point algorithm [40], operating on pairs of images. The position of each camera was refined in turn using a nonlinear minimisation procedure based on the Levenberg/Marquardt algorithm [65]. Figure 3.6 shows the recovered camera axes and the 3-D positions of the tracked features.

The background in each image was manually removed to reduce the size of the volume required to enclose the scene (figure 3.7 (b)). This segmentation defines an outer bound on the shape of the scene, known as the visual hull [48]. Figure 3.8 demonstrates that whilst the visual hull defines the tall objects (the kettle and cereal box) well, it provides very little detail in the centre of the scene as many objects are occluding each other. Modelling using contours is studied more extensively in chapter 4.

The second scene was captured using an 8 megapixel digital SLR camera and 50mm lens, which



(a) (b)

Figure 3.7: Scene 1, (a) original image, (b) segmented image

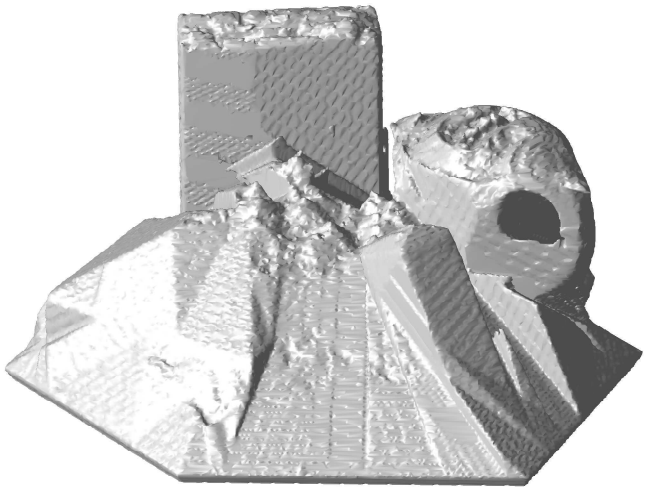


Figure 3.8: Visual hull of the segmented input images

was again calibrated according to the method outlined in [41]. In total 16 images were captured, again with approximately even spacing (see Figure 3.9). This sequence was captured in a room with black walls so there was no need for manual segmentation. This scene comprises four objects; a bag of flour, an owl statue, a cardboard box and a running shoe.

Camera positions were calculated by manually tracking a number of features, and then refining the positions automatically. Figure 3.10 shows the recovered camera axes and the 3-D positions of the tracked features.

Figure 3.9: Original input images for scene 2

To provide ground truth a scan of the scene was acquired using a Cyberware 3030 laser rangefinder device. Figure 3.11 shows the resulting model. The scan head rotates about the scene on a cylindrical trajectory and is thus only able to capture the outward facing side of the objects.

3.4.2 Probabilistic Space Carving

From the input images and camera positions associated with the first scene a voxel model was created using the probabilistic space carving algorithm [15]. The resolution of the model was $337 \times 284 \times 426$ voxels.

Figure 3.12 shows two views of the reconstructed voxels from the image set shown in Figure 3.5. In general the shape of the scene is represented well (due in a large part to the constraint provided by background segmentation). The coloured model looks fairly accurate, however this is somewhat deceiving. By design the space carving algorithm aims to replicate the input images as well as possible, rather than provide the most accurate representation of the scene. Looking at the set of

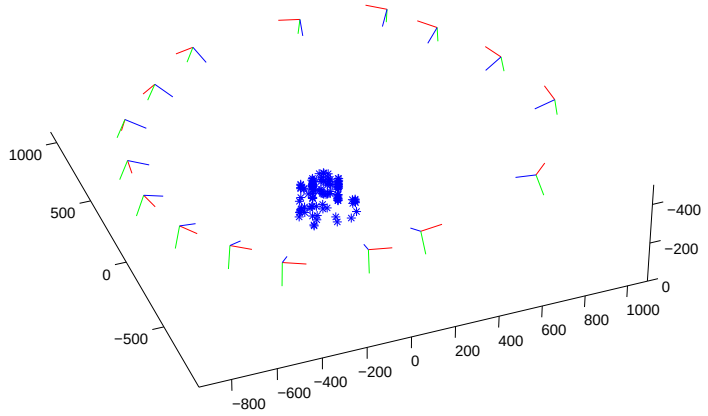


Figure 3.10: Recovered camera positions and tracked points from the second scene

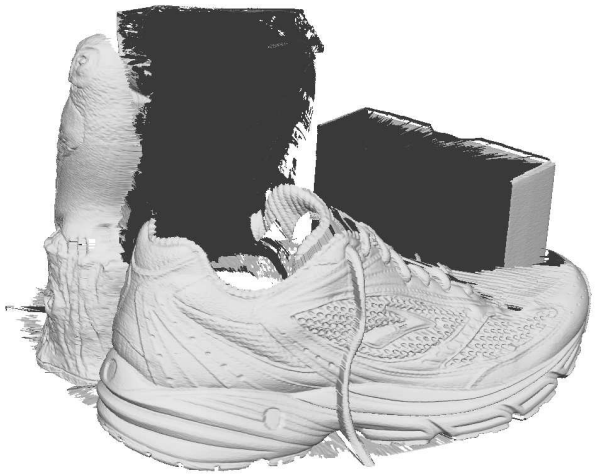


Figure 3.11: Ground truth from laser scan from the second scene

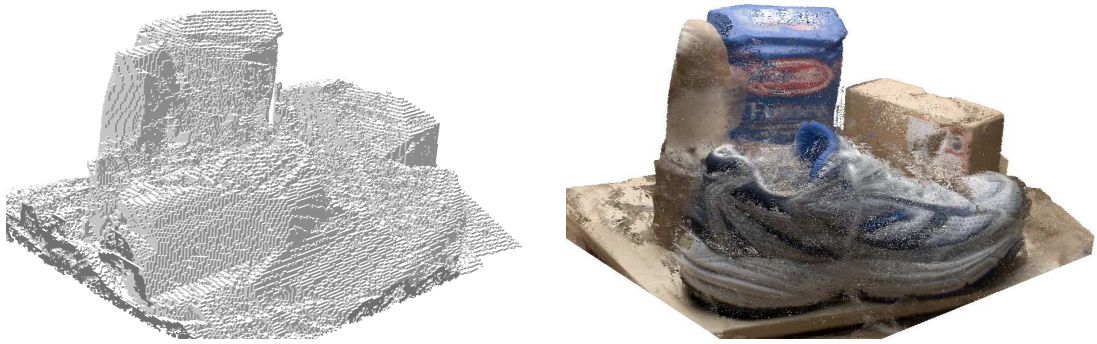
voxels directly shows the noisy reconstruction typical of space carving algorithms.

Figure 3.13 shows the results of applying the same process to the second scene. The voxel model resolution was $426 \times 250 \times 414$. Again the uncoloured rendering shows a very similar pattern of noise.

In addition to the noise both models contain many ‘floating’ voxels that are not connected to anything, particularly in the centre of the scene where the true surface lies a long way below the visual hull.

(a) (b)

Figure 3.12: Scene 1 (a) raw voxel geometry (b) colour image rendered from voxel model



(a) (b)

Figure 3.13: Scene 2 (a) raw voxel geometry (b) colour image rendered from voxel model

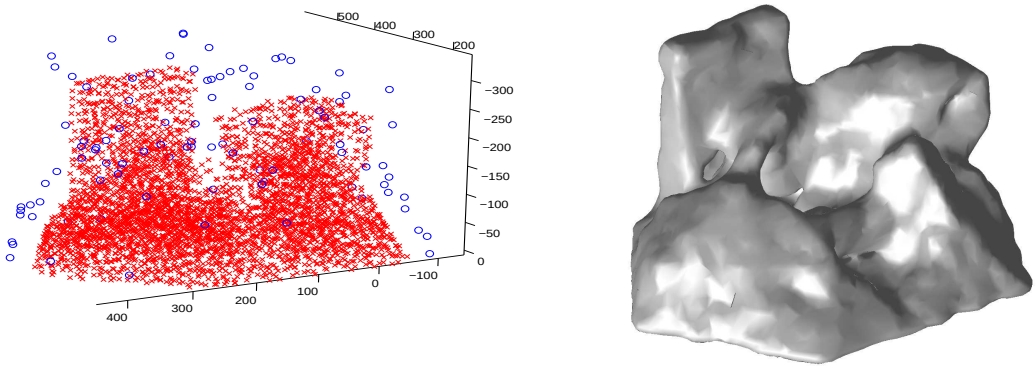
The reason for this stems from the way the voxels are processed in layers starting with the layer closest to the cameras. If a voxel in a higher layer happens to be photoconsistent by chance it will be added to the model in preference to voxels lower down which represent actual parts of the scene. Note this processing order is necessitated by the need to model occlusions and is a common feature of space carving algorithms.

3.4.3 Generating Surface Constraints

This section presents the results of extracted a candidate set of surface constraints from a voxel model and fitting an RBF surface to those constraints. Using the Poisson Sphere sampling scheme

of section 3.2.3.1 a total of 3800 constraints were generated for the first scene. In addition to this 290 exterior constraints were generated by dilating the model (i.e. the model is made successively fatter by adding extra voxels around each voxel). The surface points of the dilated model are then sampled, yielding a set of points which are guaranteed to be a minimum distance from the original surface. Figure 3.14 (a) shows these points as red crosses in the case of surface constraints and blue circles for the exterior constraints.

The weights were calculated from equation (3.9) and the function $f(\mathbf{x})$ evaluated over a regular 3D grid. These values allow a mesh to be generated using the marching tetrahedrons algorithm [24], which searches for places on the grid where the function changes sign. Figure 3.14 (b) shows the resultant surface.



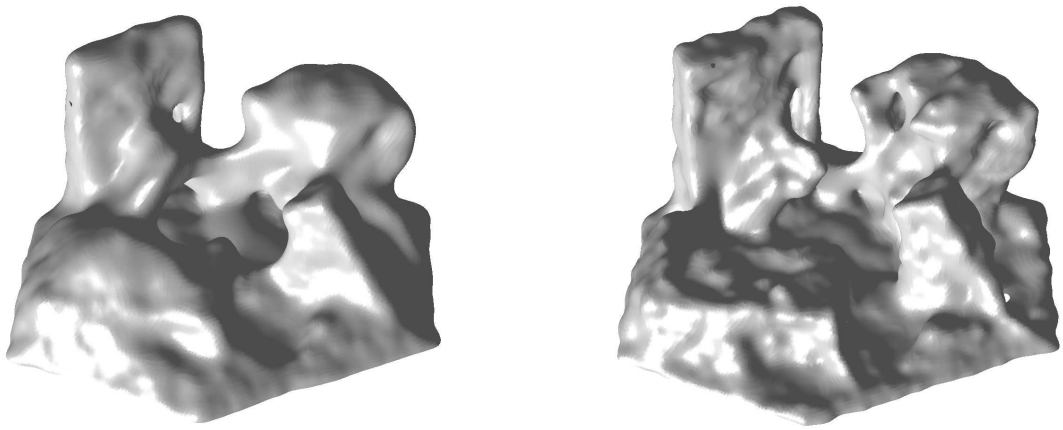
(a) (b)

Figure 3.14: (a) The set of basis functions modelling the surface [red crosses] along with the exterior constraints [blue circles] (b) the resultant implicit surface using these centres

Basing surface constraints on the voxels allows errors in the voxel model (described in the previous subsection) to be propagated through to the RBF surface.

The surface when forced to comply to constraints based on floating voxels exhibits bridges connecting certain objects together in addition to spurious protrusions. This problem is made worse by the fact that floating voxels are more likely to be chosen by the Possion sphere sampling scheme. As many will have no neighbours within the radius of the sampling sphere, they can never be eliminated and will eventually be selected.

Recall from Section 3.2.3.1 that the RBF formulation allows for λ, the 'confidence' value (which determines how faithful to the constraint the surface will be) to be set per basis function. Figure 3.15



(a) (b)

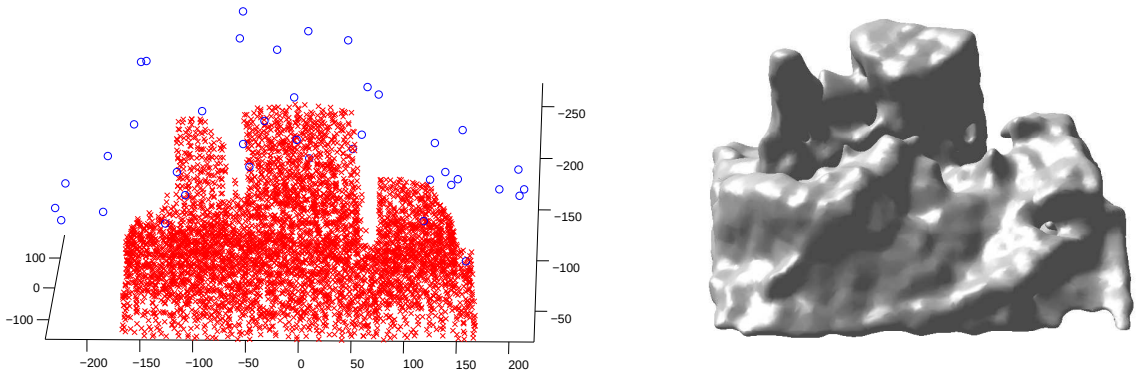
Figure 3.15: (a) Surface obtained by varying λ (b) Surface obtained by removing floating voxels prior to generating surface constraints.

(a) shows the results obtained when λ was set according to the voxel likelihoods obtained from space carving. There is still some bias in the reconstruction as some floating voxels occur far from the real surface and exert an influence even when their confidence weighting is low. The effectiveness of the scheme is limited by the fact that some floating voxels will have a high confidence value, as they can be strongly photoconsistent. Part of the outcome of this procedure can be attributed to the fact that it results in a lower λ value on average, which results in a general smoothing effect.

Figure 3.15 (b) demonstrates an alternative solution obtained by simply eliminating voxels which are not connected to the main surface, prior to generating surface constraints. This gives more detailed results and is effective provided the proportion of voxels removed is low and they are not so tightly clustered that they actually occlude part of the model which would otherwise not be photoconsistent. Note that the erroneous protrusion from the cereal box is made up of tightly clustered voxels and thus represents a more serious deficiency of the voxel carving approach.

Whilst removal of floating voxels works well for the scene in question, it may cause problems in certain cases. For example it is possible to have an object that is made up from a cloud of voxels, which whilst disconnected from each other, are sufficient in density and number to resemble a solid object when viewed. Thus whole objects or parts of objects may be deleted by this method meaning that care must be taken.

Applying the same method to the second scene (whose image set is shown in Figure 3.9) yielded



(a) (b)

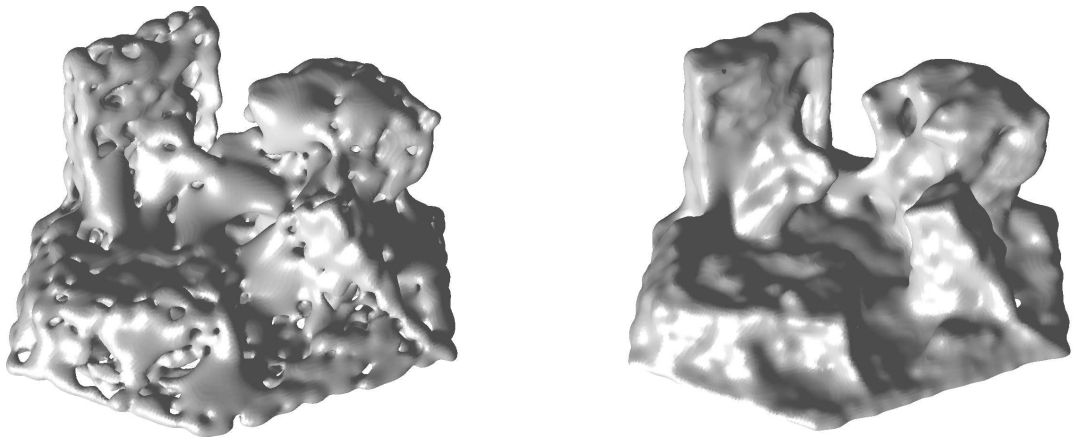
Figure 3.16: (a) The set of basis functions modelling the surface [red crosses] along with the exterior constraints [blue circles] (b) the resultant implicit surface using these centres

a set of 4150 surface constraints and 68 external constraints (again represented as red crosses and blue circles respectively in Figure 3.16 (a)). The resultant RBF surface after floating voxel removal is shown in Figure 3.16 (b). The surface result is less recognisable than the result from the first scene. This is due in part to the fact that the objects are closer together allowing the smoothing of the RBF surface to create bridges between objects. Also the shoe is a more complex shape than objects from the first scene. The taller structures (owl statue, bag of flour) are modelled best. This can be explained by the action of occlusion reducing the accuracy of the voxel model (the fewer images a voxel is visible from, the easier it is to pass the photoconsistency test). In addition to this the taller objects stand clear of the others and are mostly seen in front of a black background which reduces the possibility for false photoconsistency compared to the lower objects which sit in front of a multi-coloured background.

3.4.4 Compact Versus Non-compact Basis Functions

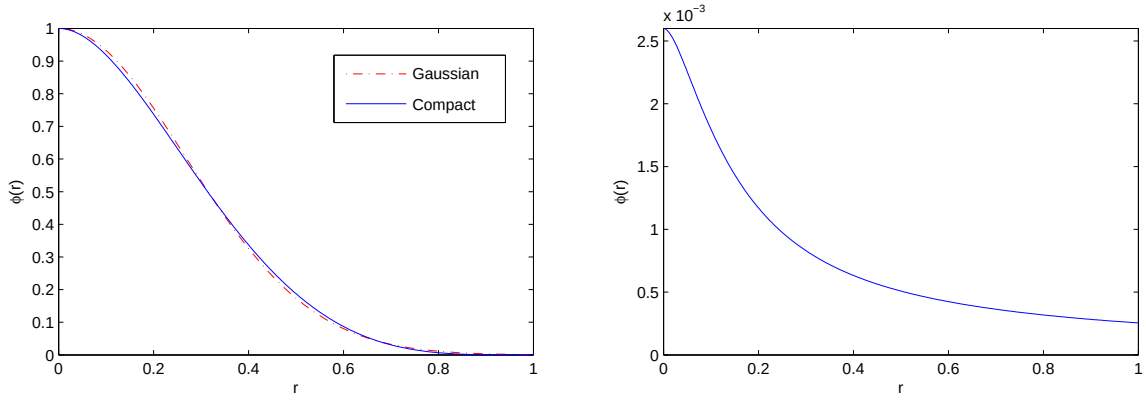
Using compact radial basis functions offers considerable gains in efficiency as only a subset of the functions need to be evaluated at each point. To take advantage of this, the parameters for each basis function are stored in an open hash table.

Figure 3.17 shows reconstructions obtained from compactly supported basis function (a) and



(a) (b)

Figure 3.17: Surfaces obtained from (a) compactly supported basis functions (b) multi-order basis functions



(a) (b)

Figure 3.18: (a) Profiles of the Gaussian and compactly supported basis function (b) profile of the multi-order basis function

multi-order basis function (b). The compact function given in Equation (3.14) is very similar to the Gaussian, see figure 3.18 (a), and thus shares many of the undesirable properties when used in reconstruction. Comparing it to the multi-order basis function, (3.11), provides some insight into the reasons behind this. The peak of the Gaussian is much wider giving poor definition to the surface and it falls away much more quickly resulting in gaps forming between constraints. However, evaluating the surface function created from the multi-order basis function took over three hours whereas the surface created from the compactly supported function took only 30 minutes.

The performance of the multi-order basis function depends on the values of parameters δ and τ,

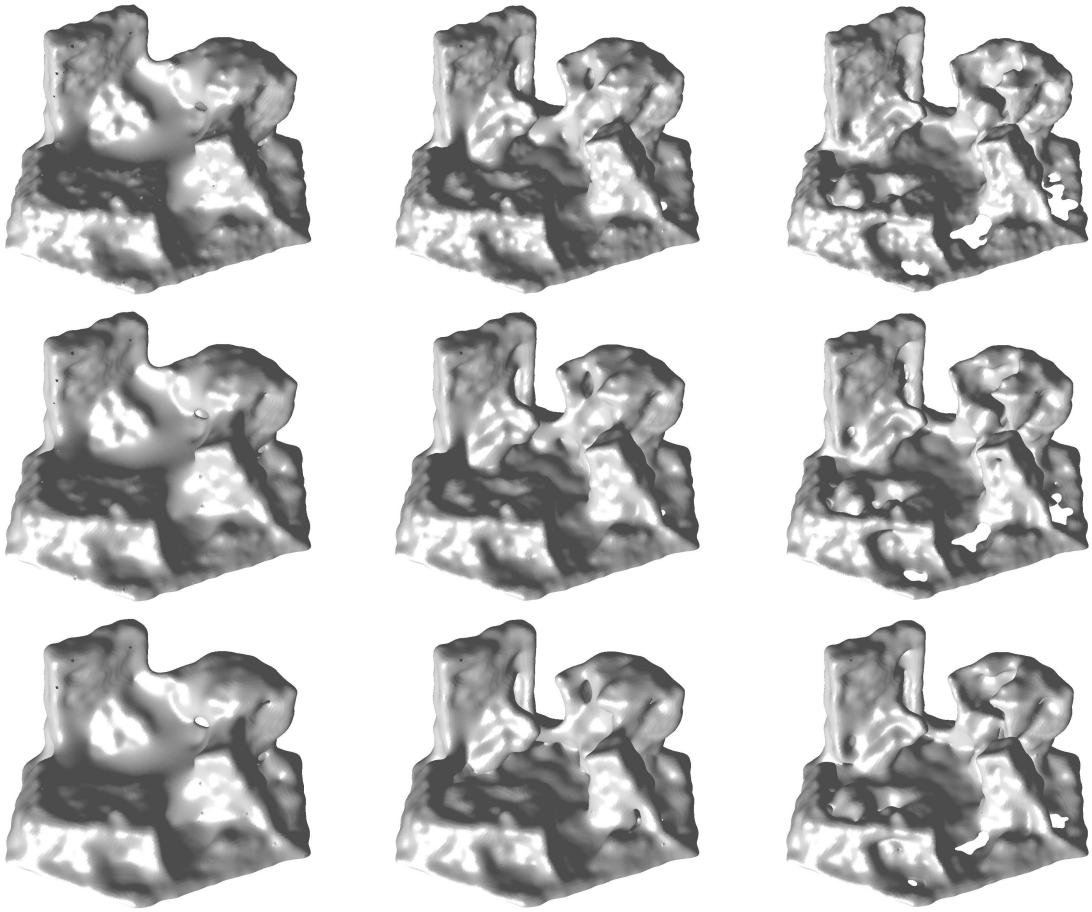


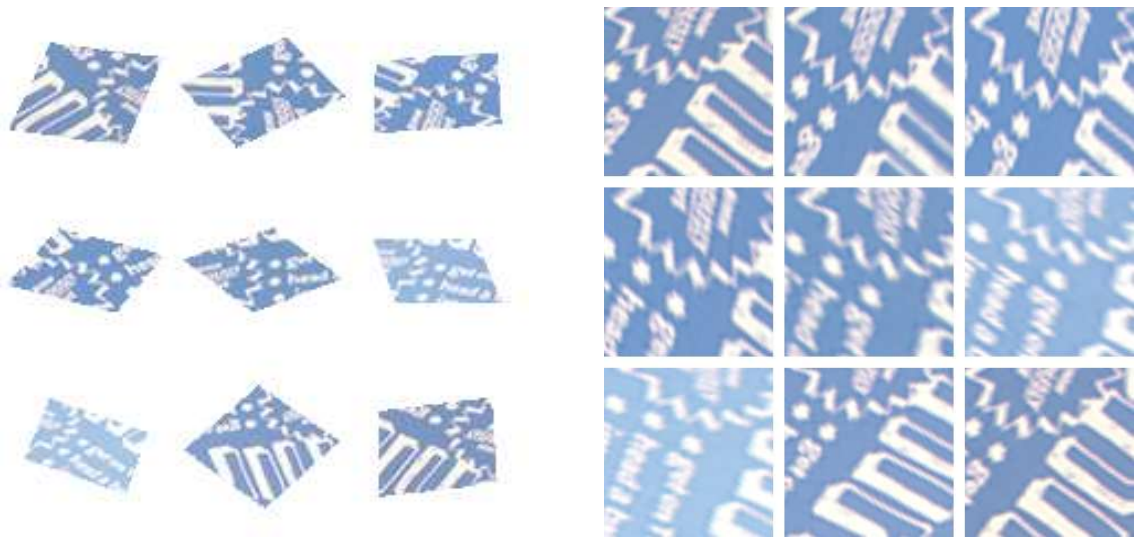
Figure 3.19: Reconstructions using the multi-order basis functions with different smoothing parameters. First column, $\delta = 5$, second column $\delta = 15$, third column $\delta = 25$. First row $\tau = 0.005$, second row $\tau = 0.015$, third row $\tau = 0.025$.

defined by equation (3.13). Figure 3.19 demonstrates the influence of the smoothing parameters δ and τ. The first order smoothing parameter, τ, has the greatest effect, if it is set too low the individual objects become fused, while if it is set too high parts of the surface become collapsed. Values of $\delta = 15$, $\tau = 0.015$ produce the best results.

3.4.5 Estimating Disparities by Block Matching

Figure 3.20 (a) shows the observed texture of a patch located on the surface of a book, in the 9 images in which it is visible, whilst figure 3.20 (b) shows the result of rectifying these image regions.

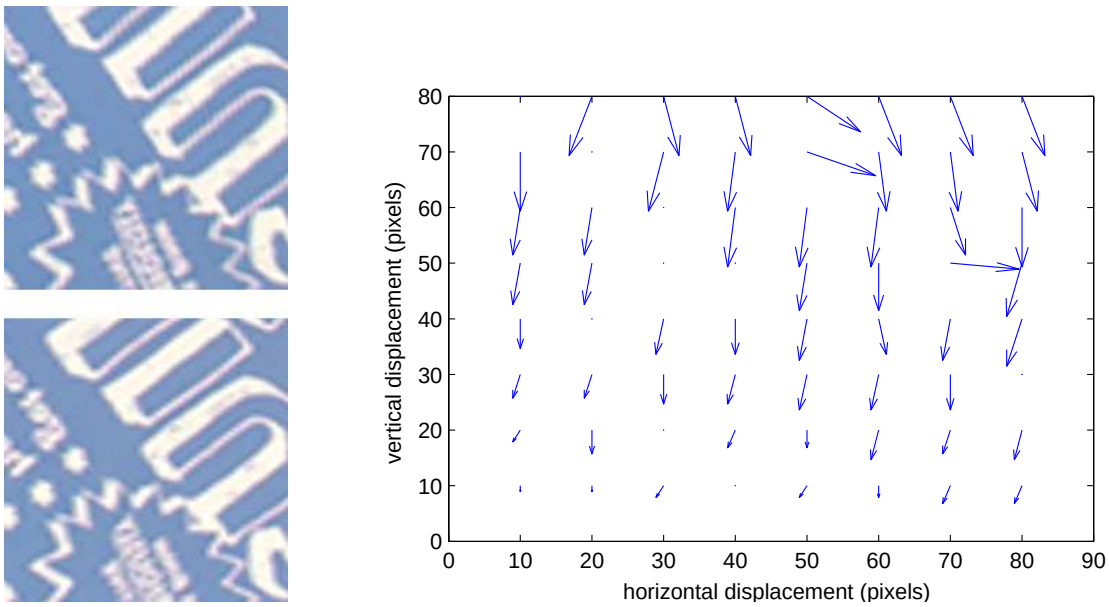
It can now be seen that the surface generated from the voxels is not quite aligned with the true surface, hence the slight variation in the observed textures. Also notice how even though the book



(a) (b)

Figure 3.20: Image regions corresponding to the projection of a surface patch (a) before and (b) after normalising

cover is not particularly shiny, it still exhibits a degree of non-lambertian reflectance, as demonstrated by changes in the shade of blue in the different images.



(a) (b)

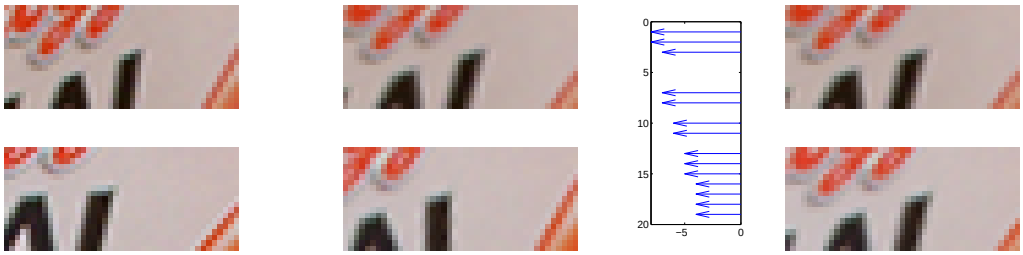
Figure 3.21: Results from section 3.3.1 (a) Two rectified texture observations (b) vector field showing displacements evaluated by block matching

(a) Initial image pairs of the projected patches.

(b) Epipolar rectification given by $\mathbf{R}$ and $\mathbf{R}'$.

(c) Offset vectors.

(d) Aligned projected patches



(e) Initial image pairs of the projected patches.

(f) Epipolar rectification given by $\mathbf{R}$ and $\mathbf{R}'$.

(g) Offset vectors.

(h) Aligned projected patches

Figure 3.22: Results from the method described in section 3.3.2, pairs of patches and their correction.

Figure 3.21 demonstrates the results of applying block matching to a pair of observations. If the variance in pixel intensities within a block falls below a certain threshold then matching is not attempted. There are some outliers in the vector field due to the fact that small blocks can sometimes generate spurious matches. These may be rejected for example if the length and orientation for a vector deviates more than a certain percentage from its neighbours.

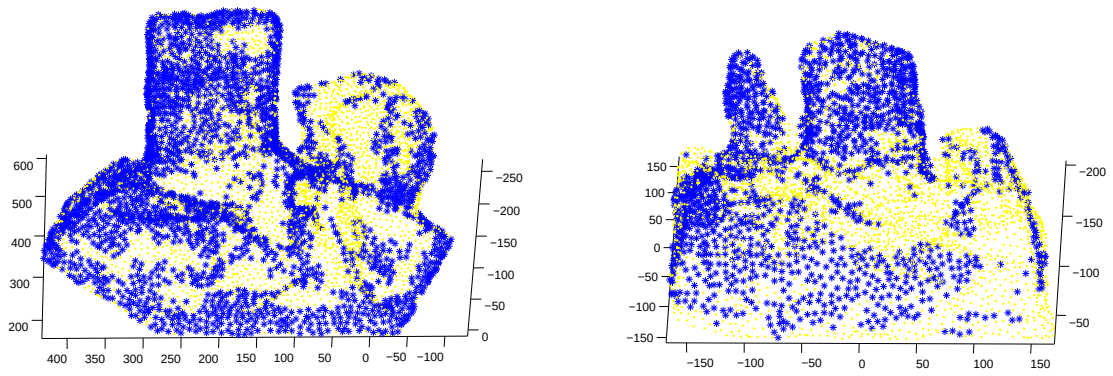
3.4.6 Texture Matching of Surface Patches

Section 3.3.2 describes a method to calculate the disparity between two images of a surface patch, induced by the changing camera geometry. Surface patches located at RBF centres and the initial estimated orientation provided by the derivative of the implicit surface function f, from Equation (3.7). From image windows centred on the surface patches, more accurate values for the position and orientation of the patch are calculated by matching the textures.

Figure 3.22 shows two cases of surface texture matching. The first column figure 3.22 (a) & (e) shows the image regions which correspond to the projection of a point on the surface. Note the difference in appearance of the same point in the two images. Figure 3.22 (b) & (f) show the same image regions after rectification by matrices $\mathbf{R}$ and $\mathbf{R}'$. The texture now lines up vertically (i.e. the top of the 'o' is the same height in both images etc.) 3.22 (c) and (g) show the offset vectors for each row, detailing the horizontal shift and skew. Finally figure 3.22 (d) & (h) show the results after the patches have been aligned by applying transformation $\mathbf{H}$ to the bottom image.

3.4.7 RBF Surface Updating

The texture matching and homography estimation proceedure was applied to each basis function in both scenes to try and calculate a more accurate position. Once the basis function centres have been moved, the weights are recalculated using Equation (3.9), and a new updated surface is obtained.



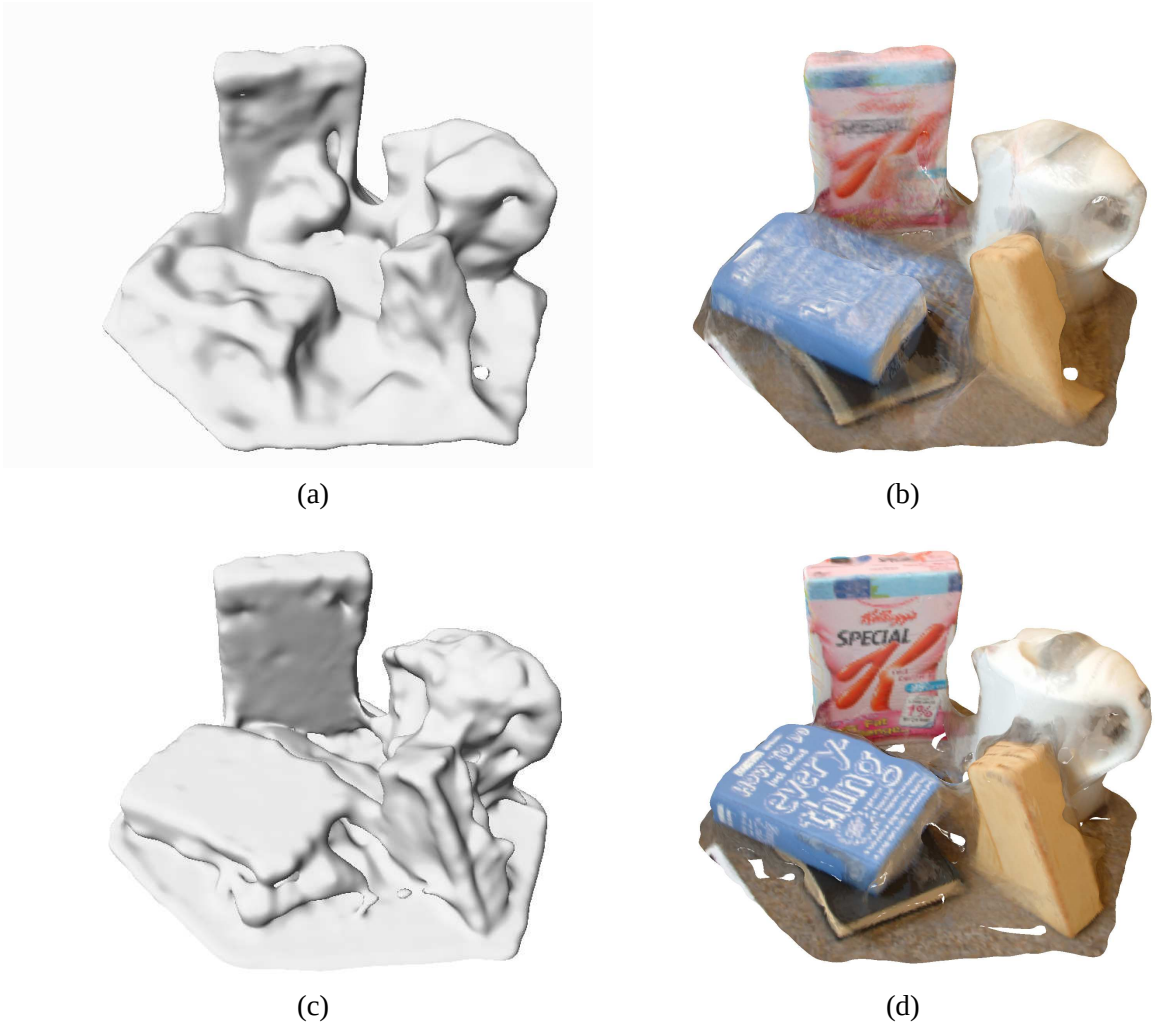
(a) (b)

Figure 3.23: Successfully updated centres, shown as blue stars in (a) scene 1, and (b) scene 2. Yellow dots represent centres which were not moved.

Recall from Section 3.3.2 that there must be sufficient surface texture to get a reliable match. In the first scene 69% of patches met this criteria, and accurate matches were obtained for 46% of the total number of centres. For the second scene 75% had sufficient detail and 35% resulted in accurate matches. Figure 3.23 shows the centres which were updated (shown in blue) and those which were not (shown in yellow). In both cases the centres that were updated are not evenly spread but clustered around textured areas and strong edges.

Despite containing more textured patches, the overall number of accepted matches was lower in the second scene. This is due to the character of the textures present on some of the objects. The owl statue featured mostly complex noisy patterns, whereas the shoe had a fine self repeating texture, both of these cases present difficulties to the matching algorithm. Several potentially correct matches were discarded due to inconsistency in the results.

3.4.7.1 Qualitative results



(a) (b)

(c) (d)

Figure 3.24: Qualitative results, scene 1. Top row: initial surface, bottom row: updated surface

Textured surfaces renderings were produced by projecting each point on the RBF surface into each of the input images. The colours of the corresponding pixels are then averaged, and the resultant

colour is applied to that surface point.

If the RBF surface is misaligned, a collection of image points representing one point on the real surface will back-project to several different points on the RBF surface. The effect of this is that the image points become spread out and the surface texture will appear blurred in areas where the model is incorrect. Thus the degree of local blurring is an indication of how accurate the surface is. This is most noticeable where there is sharp detail in the original scene, such as text.

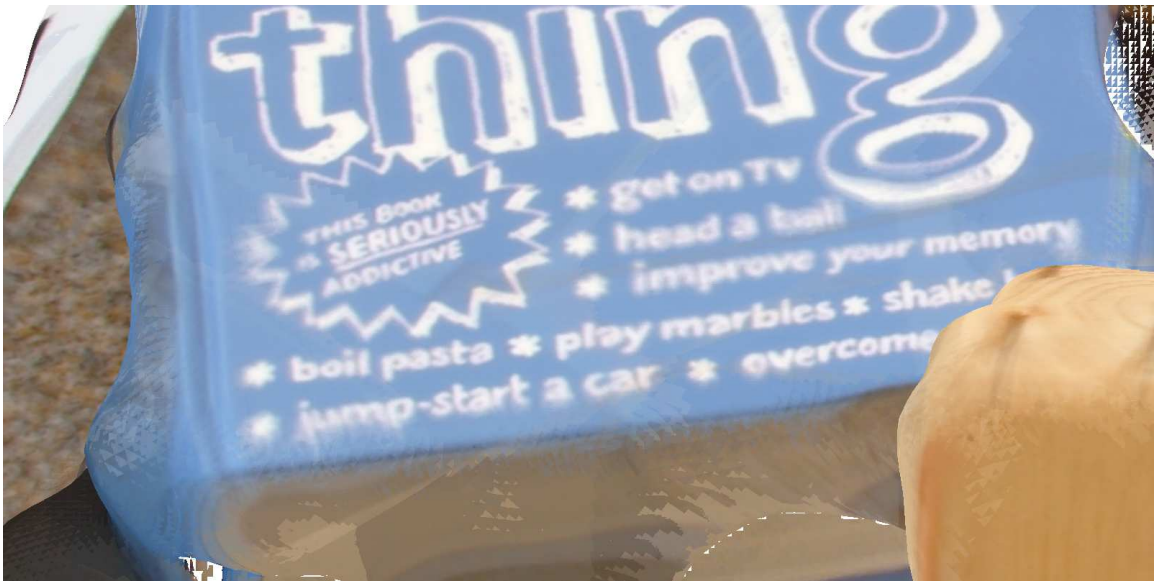
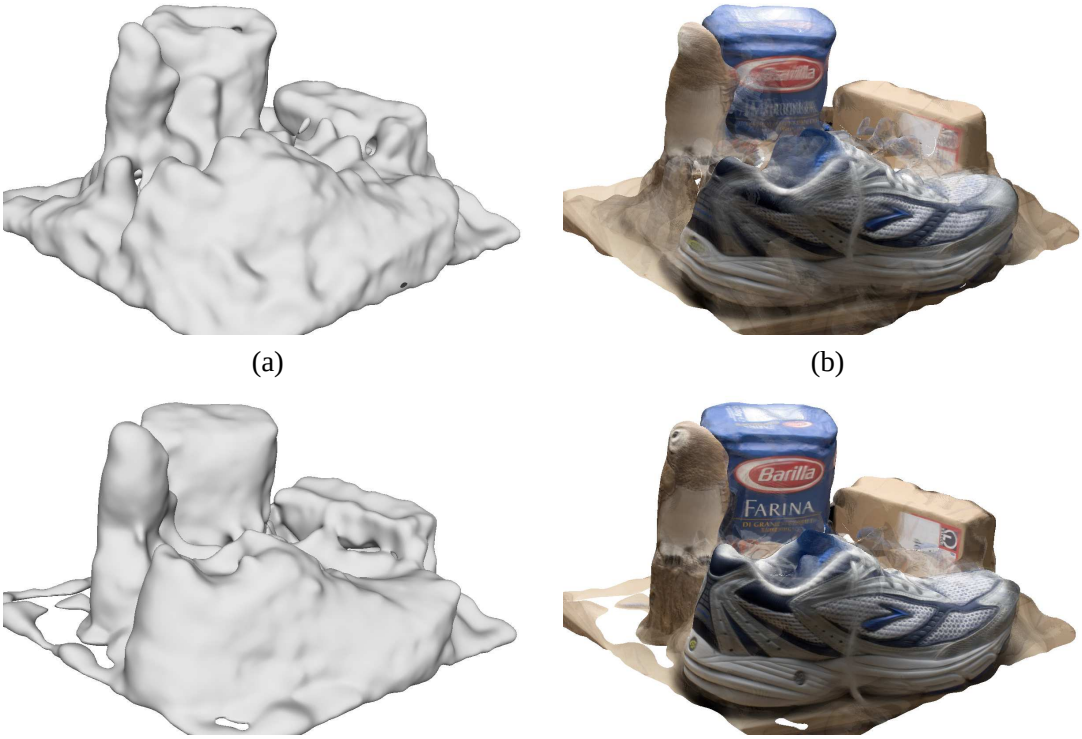


Figure 3.25: Closeup of the coloured rendering showing legibility of small text

Figure 3.24 shows the initial and updated surface in both shaded and coloured formats for the first scene. Certain improvements are clear from the shaded surface alone. The book surface is flatter and also extends to the correct position in the updated surface. Parts missing from the right hand edge of the knifeblock have been recovered, and the large protrusion from the cereal box has been completely eradicated. In the coloured renderings the text on the surface of the book is vastly more legible after updating indicating very good alignment between images. The closeup shown in Figure 3.25 proves that even the very small text at the bottom can be read. The same is true of the cereal box. A slight improvement in the surface texture is visible everywhere. Certain defects remain in the updated surface. There is a large concavity in the left side of the kettle, inherited from the voxel model, this has not been corrected due to the lack of texture on the kettle surface. The kettle and knifeblock remain joined together for the same reason. In some places holes have

developed in the floor of the model, probably due to the addition of extra external constraints.



(a) (b)

(c) (d)

Figure 3.26: Qualitative results, scene2. Top row: initial surface, bottom row: updated surface

Figure 3.26 shows the initial and updated surface in both shaded and coloured formats for the second scene. Whilst the improvements visible in the shaded rendering are perhaps not as striking as the previous case, a large spurious structure in the bottom left has been removed and the back of the shoe and owl statue now extend to the correct position. The coloured view shows the full extent to which the shape owl statue has been recovered. In addition to this there are clear improvements to the text on the bag of flour indicated that the surface has been pulled into the correct position.

The generation of holes is more serious in this dataset, particularly in the bottom left corner. This is due to the fact that a lot of centres were moved but not replaced as there was insufficient texture information to go on. One drawback of this updating approach is that there is no constraint to require the model to be complete (i.e. free of holes!)

3.4.7.2 Quantitative results

In order to provide quantitative assessment of the improvement in the first scene in the absence of ground truth, I note that the real surface of the blue book is almost perfectly planar. I then measure its deviation from the planarity in the estimated 3-D model. This deviation, measured in millimetres, was estimated for the voxel model, the initial RBF surface, as well as for the surface updated according to the algorithm provided in Section 3.3.2 and is shown in the first column of Table 3.1. Gaps in the surface and floating voxels result in a large error in the voxel model. These are smoothed in to a bumpy but essentially flat surface by the RBFs. Correction based on texture yields a final error of just over 1mm.

For the second dataset ground truth is available in terms of a laser scan, taken at the same time as the input images were captured. The laser scanner provides depth information in terms of the distance from the central axis of rotation for a 360° arc surrounding the scene. The depth map error is weighted by the distance from this axis (as more distant values in the depth map correspond to larger areas of the surface) and is calculated as:

$$DE = \sqrt{\frac{\sum_{i=1}^{n} d_i (d_i - \hat{d}_i)^2}{\sum_{i=1}^{n} d_i}} \tag{3.27}$$

where $\hat{d}_i$ is the value for a specific point on the model and d_i is the corresponding value measured by the laser, n represents the number of surface locations for which we have a valid depth information. The measurements for each stage in the reconstruction are provided in the third column of Table 3.1. Again presence of floating voxels results in a large initial error which is reduced by the subsequent RBF surface fitting. The updating procedure reduces this error further. It is important to consider that this error is the average figure for the entire surface which is why it is higher than the planarity error of the previous scene.

Another measure of the accuracy of the reconstruction can be obtained by comparing the input images with coloured renderings from a matching viewpoint. As before the colour of each point on the 3-D surface is obtained by averaging the colours of all its corresponding projection pixels from the original images. I evaluated the PSNR between the projected scene using $\{\mathbf{P}_i | i = 1, \ldots, n\}$ and

Method	Scene 1		Scene 2	
	Planarity error (mm)	PSNR (dB)	DE (mm)	PSNR (dB)
Voxel carving	11.6	13.40	25.23	9.485
Initial RBF	3.63	14.14	18.24	13.02
Updated RBF	1.04	17.45	14.76	15.93

Table 3.1: Surface geometry and PSNR errors for the 3-D scene reconstructions from images.

the original images. The PSNR is calculated as:

$$PSNR = \frac{10}{n}\sum_{i=1}^{n}\log_{10}\left[\frac{g\ 255^2}{\sum_{j=1}^{g}(I_{ij}-\hat{I}_{ij})^2}\right] \tag{3.28}$$

where $\hat{I}_i$ represents the projection of the reconstructed textured 3-D surface from camera i, n represents the number of images and g represents the total number of pixels in each image. The PSNR values are provided in columns two and four in Table 3.1. The figures follow the same general trend of improvement. For comparison purposes, the PSNR for JPEG images with medium compression is 25 Db. The PSNR figures aggregate the errors over the entire image and can therefor hide large errors than occur over only part of the image. They are best considered in conjuction with the qualitative results as shown in Figure 3.24.

3.5 Conclusion

In this chapter a method was presented for representing 3-D scenes with several objects from multiple images. An implicit function model using RBFs is initialised using the voxel representation provided by the space carving algorithm. The resulting RBF model solves certain problems related to the uncertainty in the voxel estimation and provides a smooth representation of the surface. Errors are caused in the resulting surface due to various uncertain factors such as illumination variation, complex object shapes or occlusions. I consider a set of patches in 3-D, where each patch is associated with an RBF function. I propose a method for refining RBF centres by using correspondences of images, representing the projections of the same patch from the 3-D scene, along epipolar lines.

Experiments are conducted on two real multi-object datasets. Both of these datasets contain ob-

jects which would present serious problems to traditional object modelling algorithms. The methods outlined in this chapter were applied to the datasets, each yielding three representations, the initial voxel mode, the initial RBF surface and the RBF surface after updating from image information. Colour surfaces representations are obtained in all cases

The space carving results demonstrate the level of noise and uncertainty that occurs in the voxel model due to the lack of any global constraints. The initial RBF surface is smoother but is inconsistent with the input images in several areas. The updated surface is far superior with a large proportion of basis function centres moved to lie on the correct surface. In general the various objects are modelled well and are easily identified in the representation. Numerical evaluations are provided with respect to both geometry and texture information for the two sets of images. The geometry reconstruction is evaluated by employing a planarity test in the first image set and by comparing with the ground truth provided by a 3-D laser scanner in the second image set. The colour information estimation is evaluated using the PSNR when backprojecting the information from the 3-D surface onto the image planes. The numerical results confirm what is seen qualitatively, with large leaps between the voxel model and initial RBF surface, and smaller but still significant improvements made after updating.

Whist the improvements over the initial surface are significant, there are still errors in the reconstruction. These are mainly in the form of adjacent objects which are spuriously joined by the interpolating action of the RBFs. This occurs in areas which lack sufficient texture to obtain reliable matches and hence find the correct positions for the RBF centres. Correcting these errors forms the main topic of the following chapter.

Chapter 4

Contour Based Correction of 3-D Scenes

4.1 Introduction

This chapter builds upon the work of Chapter 3 and also complements it. Whereas in Chapter 3 the existence of surface detail or patterns (specifically edges) was a requirement for obtaining good 3-D information, this work focuses mainly on homogenous areas without significant detail. The boundaries of such areas form a set of image edges which are used to correct the scene.

The majority of the work here is based on the following observation. For each edge in 3-D (either an object boundary or a 'rim', where the surface disappears from view) there should be a corresponding edge in the observed image. This will be true except in rare cases in which an object sits in front of another object with exactly the same colour and shading.

It should thus be possible to detect any gross errors in the recovered shape by comparing the predicted and observed edges. This is specifically important for the case where two objects are erroneously joined together or fused. There should be no observable image feature corresponding to the part of the shape which joins the objects.

The outline or contour of a shape also provides some information about it's shape in 3-D so it should also be possible to correct the shape using the same contours. Methods of 3-D modelling using this information are usually referred to as "Shape from Silhouette" [48] or "Shape from Contours" [91], and have been around for a long time in Computer Vision. In a common with voxel carving, Shape from Silhouette also works on the principal of cutting away areas that are inconsistent with the images. In this case inconsistency occurs when part of the shape projects onto the

image in a way that falls outside of the contour.

Sections 4.2 and 4.3 demonstrate how this theory can be adapted to the case of correcting large shape errors and separated objects that have become joined by the RBF modelling process. A statistical formulation of the 3-D scene updating methodology is provided in Section 4.2. Such a 2-D to 3-D approach requires performing many low level vision tasks such as segmentation, edge detection etc. Here we are not particularly interested in segmenting whole objects or image understanding tasks, but merely in finding the boundaries of homogenous image regions (edges). Here the term homogenous applies to appearance, thus areas of homogenous texture are just as important as areas of homogenous colour.

Texture classification and segmentation is studied in the field of video compression where regions containing arbitrary texture are removed from the compression process and later synthesised during decompression [11]. Several methods are used, but most are based on collecting local statistics concerning the variation in image brightness levels and then using standard clustering algorithms. In this work two segmentation methods are used, one supervised, utilising support vector machine classifiers, and one unsupervised, utilising mean shift clustering.

A method for transforming segmented images into continuous object contours is also presented. It is based on research using active contours to fit shapes to image data for medical imaging algorithms [60]. Given a suitable initialisation a contour evolves as it is acted upon by forces derived from the image, and subject to limits on elasticity and smoothness. The (incorrect) 2-D contour given by projection of the current 3-D surface is used as initialisation. This also helps ensure only locally relevant image edges are used. With this contour information, the RBF model is corrected to be consistent with the input images uniting the methods discussed previously into a novel contribution to multi-shape modelling. Problems associated with the interpolating and smoothing properties of RBF implicit surfaces are addressed using constraints with negative weights.

A detailed analysis is undertaken in Section 4.4 in which the factors influencing the potential of contour based methods are studied. It is shown that in certain circumstances disparity between the correct and incorrect contours is too small and makes detection or correction of the error impossible. Full analysis of the key variables and their contribution is provided, used to provide recommendations on how to capture image sequences to maximise the chances of success when modelling scenes with multiple objects which are located closely to each other.

In section 4.5, results of the methods discussed so far are presented using real image data and

both qualitative and quantitative evaluations are performed. Finally the conclusions of this chapter are drawn in section 4.6.

4.2 Probabilistic Updating of 3-D Scenes Using Contour Consistency

In this chapter it is assumed that we already have a 3-D reconstruction of a scene consisting of several objects constructed from a set of images taken from various angles. The procedure used was described in Chapter 3 and employs space carving followed by surface modelling using radial basis functions.

In this study I will show how the 3-D scene representation can be improved by using additional information extracted from the input images. Such information may consist of colour, texture, corners or boundaries of objects in the scene. This section describes a formal statistical framework for reasoning about such information which will be used as the basis to derive concrete algorithms.

In the following it is assumed that the input images contain objects which are characterised by well defined colours and textures while separated by boundaries. When the scene contains multiple objects such as the case considered in this thesis, some of these may become merged by the resulting surface if they are positioned close together.

Let us assume that in addition to the input images $\mathcal{I} = \{I_i | i = 1, \ldots, n\}$ there is a corresponding set of silhouettes. In several papers it was shown that a set of silhouettes can be used to model the 3-D scene using a union of cone volumes [48]. These methods provide good reconstructions except when the objects contain extensive concave regions. The extraction of such silhouettes is usually based on the existence of a plain background. Silhouette updating has been used for single object 3D reconstructions in several studies [46]. In the following I investigate how problems with merged objects can be detected and corrected using segmented contours in a similar approach with that of shape from silhouettes.

Let us denote the set of contours as $\mathcal{C} = \{C_i | i = 1, \ldots, n\}$ extracted from the input images $\mathcal{I}$. Contours can be extracted directly from images using various means [64][31]. The contours are assumed known and represent a segmentation of the images into distinct 3-D objects. The situation when two or more distinct objects are merged into the resulting 3-D scene is investigating in the following. Such errors can occur when two or more objects are close to each other resulting in the RBF surface interpolating across the gap between the objects.

We assume that the scene $\mathcal{S}$ contains distinct objects denoted as $\{\mathbf{A}, \mathbf{B}\} \in \mathcal{S}$ which are close to each other and could occlude each other in several images. The background is assumed as an object of the scene as well. The process of space carving and the surface fitting may fail to identify them as separate objects. The first process may produce a large number of floating voxels in the area between the two objects, blurring the boundary between them, while the second stage may merge both objects with the same surface, failing to represent the existing gap in between the two objects. A probabilistic assessment of whether we have two objects merged into one or a single independent object may be obtained from evaluating the following :

$$P(\mathbf{A})P(\mathbf{B}) > P(\mathbf{A} \cup \mathbf{B}) \tag{4.1}$$

where the left side term represents the probability of having two separate objects and the right side term represents the probability of having a single object. These probabilities are never explicitly estimated in the contour based updating method, however the concept of deciding between two hypotheses (that there exist two separate objects or that there really exists only one object) forms the foundation of the method, the steps of which are summarised in the following.

By comparing coloured renderings of the scene with the input images, anomalies can easily be detected. Objects suspected of being joined can be segmented from the rest of the scene so that their 3-D contour is known. This segmentation is achieved by removing the ground plane which connects the free standing objects. In the case of images both colour and texture features (texels) are considered for characterising the objects. Such features have been used before for image segmentation [11] as well as have been considered for the MPEG-4 coding standard [6].

The following steps are used for checking the contour consistency:

- Threshold the scene $\mathcal{S}$ in order to separate the objects.
- Use either unsupervised or supervised segmentation as follows:
  - Unsupervised segmentation - Segment each image using the mean-shift by considering the colour and texels as features and consequently define the object boundaries.
  - Supervised segmentation - Consider one image or more for training and sample values of pixels for various objects of interest. Employ supervised training by using Support

Vector Machines (SVM) for learning the given data set and define the object boundaries. Usually it is assumed to have two objects and their background.

- Model the boundaries of objects using snakes.
- Project the contour of the 3-D objects $\mathcal{C}_{\mathcal{S}}$ using the projections matrices $\mathcal{P} = \{\mathbf{P}_i \in i = 1, \ldots, n\}$ onto each image from the given set $I_i, i = 1, \ldots, n$.
- Compare the segmented contours from 2-D images with those resulting from projection from the 3-D scene and detect inconsistencies.

The concept of this approach consists of detecting merged objects by identifying inconsistencies in the object contours, as they are detected in 2-D from the given images, after using either supervised or unsupervised segmentation, and the contours resulted from projecting the object surface from the 3-D scene. In the following section the feature selection, the supervised and unsupervised segmentation, the contours extraction and their modelling by snakes are described in detail.

4.3 Contour Extraction for Model Correction

In order to detect disparities between 3-D object shapes and 2-D shapes we have to detect the contours of objects. In the 3-D scene $\mathcal{S}$ the objects are easily segmented by thresholding the scene by assuming that we have a set of objects lying on a flat surface. A more complex 3-D clustering approach, involving spatial coordinates can be used for mode complex scene assumptions.

I use a statistical approach to object contour extraction from images. In the following the approach used for the object contour extraction from the given 2-D image set $\mathcal{I}$ is explained. The 3-D scene modelling will be verified and corrected wherever necessary by checking the consistency of the contours obtained by projecting the object surfaces extracted from the 3-D scene $\mathcal{S}$ with those extracted from the original image set $\mathcal{I}$.

The object contours are extractred from the images by employing two different approaches: supervised and unsupervised object segmentation. In both approaches the same set of features is used to characterise the objects. The supervised approach uses training based on object samples from one or more of the images and afterwards classifies the image content in objects by using Support Vector Machines (SVM) [64]. The unsupervised training employs clustering in the given

set of images by using the mean-shift algorithm. The clustering corresponds to a segmentation of the image in objects. In both cases single continuous contours of the objects are extracted and modelled using snakes. Each of these steps are described in detail in the following.

4.3.1 Feature Selection

As in other studies [11], it is considered that objects are characterised by colour and texture these as features are used for segmenting the images and for afterwards extracting the object contours. The primary features used in segmentations are the three colour channels, red green and blue.

Other colour spaces such as L*a*b* [26] have been proposed to better characterise the perception of colour by the human visual system. These colour spaces do not add any information since they are just a coordinate transform of the same unlying data. For this reason I use the camera's native RGB colour space. In addition to colour I found it useful to consider texture information to provide means with which to distinguish between objects of similar colour but contrasting textures.

To provide an estimate of the local image texture a formula used in the Harris corner detector [37] provides a rough measure of texture which is good enough in most circumstances. It gives a large response for grainy textures which are uniform in every direction, a reduced response for more stripy textures with anisotropic variance in colour and a very low response for untextured surfaces. Let us consider the location of a pixel $[u, v]^\mathsf{T}$ in one of the input images. The measure is defined as:

$$\mathrm{tx}(u, v) = \det(\mathbf{M}) - (k \,\mathrm{trace}^2(\mathbf{M})) \tag{4.2}$$

where $\mathbf{M}$ is the 'structure matrix' which describes the local image gradients around the point $[u, v]^\mathsf{T}$, defined as:

$$\mathbf{M} = \begin{bmatrix} \sum_{u,v \in \mathcal{W}} (I_v(u, v))^2 & \sum_{u,v \in \mathcal{W}} I_u(u, v) I_v(u, v) \\ \sum_{u,v \in \mathcal{W}} I_u(u, v) I_v(u, v) & \sum_{u,v \in \mathcal{W}} (I_v(u, v))^2 \end{bmatrix} \tag{4.3}$$

$\mathcal{W}$ represents the window which is weighted by a Gaussian and I_u, I_v represent the image gradient in the u, v direction. The parameter k determines how the response varies with respect to the strength of the gradients in orthogonal directions. If the gradient is strong in both directions the first term

will dominate, however if the gradient is significantly stronger in one direction the second term will dominate yielding a negative response. As this formulation can result in extreme values which throw off segmentation algorithms, the output is clamped to a specific interval $[tx_{min}, tx_{max}]$. The contribution of the texture response is weighted by a parameter ζ. Each image pixel is then mapped into a 4-D feature space as $\mathbf{z} = [r, g, b, \zeta tx]^{\mathsf{T}}$.

4.3.2 Contour Extraction

In order to extract contours we use object segmentation in all images from the input set $\mathcal{I}$. Two probabilistic methods of image segmentation are discussed in this section, supervised and unsupervised. Both methods use the same feature vector $\mathbf{z}$ as described above.

4.3.2.1 Unsupervised Image Segmentation

Unsupervised segmentation consists of estimating the most likely partition of the image without using *a priori* knowledge. Usually this consist of applying a clustering method. I choose to use the mean shift segmentation method due to its easy usability and proven success in characterising clusters well [31].

The mean shift algorithm classifies based on identifying local maxima in the density estimation of the sample points. It is unsupervised in that no a-priori knowledge is required about the number of classes or their distributions. In this case we are trying to segment a pair of objects from the background so the number of classes, 2, is known, however the contour fitting procedure described in this section is robust with respect to oversegmentation - provided the area between the objects is segmented correctly.

The mean shift procedure for clustering (and hence classification) is based around kernel density estimation of the feature space, which is defined as follows:

$$f_d(\mathbf{z}) = \frac{c}{nh^d} \sum_{i=1}^{n} k\left(\frac{\mathbf{z} - \mathbf{z}_i}{h}\right) \tag{4.4}$$

where there are n feature vectors z, of dimension d (in our case $d = 4$). $k(\cdot)$ is the radially symmetric kernel function (I will use the multivariate normal kernel), the parameter h defines the width of the

kernel, and is referred to as the bandwidth. The term c is a constant that ensures the kernel integrates to 1.

The mean shift procedure iteratively updates the centre of the cluster using the derivative of the density function. The vector which points in the direction of greatest increase in density is given by :

$$\mathbf{m}(\mathbf{z}) = \frac{\sum_{i=1}^{n} \mathbf{z}_i g(||\frac{\mathbf{z}-\mathbf{z}_i}{h}||^2)}{\sum_{i=1}^{n} g(||\frac{\mathbf{z}-\mathbf{z}_i}{h}||^2)} - \mathbf{z} \tag{4.5}$$

where $g(\mathbf{z}) = -k'(\mathbf{z})$. The classification algorithm is iterative, starting with a random placing of seeds, sufficient in number so that kernels overlap ensuring all populated areas of the feature space are covered. The steps used are as follows:

1. Compute the mean shift $\mathbf{m}$ for the current window
2. Translate the window according to the mean shift vector $\mathbf{z}^{t+1} = \mathbf{z}^t + \mathbf{m}(\mathbf{z^t})$
3. repeat until convergence

The stationary points are then the centres of the classes and all feature vectors within the basin of attraction are assigned to that class. The output of the algorithm is then a mapping from image pixels to classes. Due to the fact that no spatial information from the image is used, there is no guarantee of compactness of the resulting classes. Utilising texture information helps with this in areas where pixel colours tend to vary a lot from their neighbours. However, the goal of the segmentation is to produce a set of edges which correspond to genuine changes in surface colour or topology. With this in mind I use a post processing step to enforce regularity in the data.

The classes are recomputed to be contiguous sets of pixels (originally two pixels with similar features on opposite sides of the image could be assigned to the same class). This is achieved by sweeping the image and using the union find algorithm [19]. Initially each pixel belongs to a unique set of size one. The pixels are then examined in order and if a pixel belongs to the same class as it's neighbour, the union of the containing sets is taken. When this has completed each set represents a contiguous group of pixels with the same original class assignment. These sets then become the new classes.

This procedure results in the creation of many very small classes. Any classes with a population below a given threshold are merged into the largest neighbouring class.

The bandwidth parameter defines the sensitivity of the algorithm and indirectly determines the number of clusters found. Although it requires no training, a disadvantage of the mean shift algorithm compared with other algorithms is that it can be difficult to set the bandwidth in order to obtain a good segmentation for some scenes. Various statistical based algorithms can be used to choose the bandwidth for the mean shift algorithm [77].

4.3.2.2 Supervised Image Segmentation

A supervised image segmentation method is able to overcome certain problems associated with parameter selection by using a training set of pixels explicitly labelled in two classes as the foreground and background in the image. This section describes such a method which is based on support vector machines (SVM) [64].

SVMs classify data points by constructing an optimal separation plane by identifying support vectors which lie along the boundaries of the two classes. The optimal plane is the one which maximises the separation between the two classes (the intuition is that this property will help the classifier generalise well). SVM was shown to provide very good supervised classification results. As this is a supervised method, a training set is created by sampling n pixels creating feature vectors consisting of colour components and texels $\mathbf{z}_i, \ldots, \mathbf{z}_n$ from one or more images from our set $\mathcal{I}$ and labelling them with $y_i \in \{-1, 1\}$, $i = 1, \ldots, n$, each label corresponding to an object class as foreground and background, respectively.

We want to find the maximum-margin hyperplane which divides the points having $y_i = 1$ from those having $y_i = 0$. Any hyperplane $\mathbf{w}$ can be written as the set of points $\mathbf{z}$ satisfying:

$$\mathbf{w}^{\mathsf{T}}\mathbf{p} = b \tag{4.6}$$

where b is a constant. The vector $\mathbf{w}$ is a normal vector: it is perpendicular to the hyperplane. The parameter $\frac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin along the normal vector $\mathbf{w}$. We want to choose the $\mathbf{w}$ and b to maximise the margin, or distance between the parallel hyperplanes that are as far apart as possible while still separating the data. These hyperplanes can be described by the equations

$$\mathbf{w}^{\mathsf{T}}\mathbf{z} - b = 1 \tag{4.7}$$

and

$$\mathbf{w}^{\mathsf{T}}\mathbf{z} - b = -1 \tag{4.8}$$

Formally, the SVM is the solution to the following quadratic programming optimisation problem:. Mimimise in the given feature space of $\mathbf{w}$ and b the following

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \tag{4.9}$$

such that for all data we have

$$y_i(\mathbf{w}^{\mathsf{T}}\mathbf{z}_i - b) \geq 1 \tag{4.10}$$

That is, the value of $\mathbf{w}$ which maxmises the separation, provided all training points in the two classes lie on the correct side of the plane. As the points in the 4-D feature space described above are unlikely to be linearly separable, the problem is transformed by replacing dot products with a kernel function which maps the points into a higher dimensional space. This allows the algorithm to fit the maximum-margin hyperplane in the transformed feature space. The Gaussian RBF kernel is used for this purpose as it produces the best results:

$$k(\mathbf{w}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{w} - \mathbf{z}\|^2}{2\sigma^2}\right) \tag{4.11}$$

where σ represents the Gaussian spread. This kernel replaces the dot products $\mathbf{w}^{\mathsf{T}}\mathbf{z}$ from equations (4.7) and (4.10).

Once the classifier is trained, the image segmentation is performed by calculating where each pixel falls with respect to the boundary in the kernel feature space and labelling them accordingly. The classified image is post-processed using the same steps as the unsupervised case, to produce contiguous classes with no holes.

4.3.2.3 From Edges to Contours

The next part of the contour correction algorithm generates an improved contour estimate. Using the segmentations produced so far, class boundaries in the images are used to generate an edge map for each image. To join this edge map into a continuous contour the initial estimate is refined by using Active Contours [88] (better known as 'snakes'). This technique has been used for segmenting organs in medical images by applying snakes on a plane extracted from a volumetric dataset, such as an MRI scan [89]. Starting within an initial estimate, snakes iteratively evolve a contour using a combination of different forces. External forces push the snake towards image features, such as edges, whereas internal forces regulate the shape of the snake, enforcing a sort of smoothness known as elasticity which prevents the snake from bending too much and over fitting to the data. A typical external force would be

$$E(u,v) = G_\sigma(u,v) * \mathbf{I}_{I,u}v \tag{4.12}$$

Where $\mathbf{I}$ is the image, G is the 2-D Gaussian function and the width σ controls how far away from the edge the force will be felt by the snake.

In this case the a variation of the classical snakes algorithm is used [88]. This variant uses the gradient vector flow as the external force, and possesses the property that it is able to expand into heavily concave regions, which is a necessary condition to be able to separate objects. The snake is a curve $\mathbf{s}(q)$ parametrised by $q \in [0,1]$. It is updated by solving the following equation

$$\mathbf{s}_t(q,t) = \alpha\mathbf{s}''(q,t) - \beta\mathbf{s}''''(q,t) + \nabla E \tag{4.13}$$

where $\mathbf{s}_t$ represents the snake as a function of time, t, by means of a discrete iterative algorithm. Constants α and β represent the weights of the second and third derivatives of the snake function and ∇E is the gradient of (4.12).

The algorithm takes as input a binary edge map (obtained from the boundaries between classes provided by either the supervised or unsupervised segmentation methods, described in sections 4.3.2.1 and 4.3.2.2) and an initial contour. Again the initial contour used is that of the current

(presumably fused) shape, projected into 2-D. This is treated as a reasonably good estimate of the objects' true contours, initialising the snake's position.

Instead of using segment boundaries as edges, an simple edge detection algorithm could have been used which would have negated the need for complex segmentation techniques. However, if there were any gaps in the detected edges the snake could have expanded into one of the objects instead of stopping on the boundary. Another problem is that edge detection often gives spurious results in the presence of textures, a problem which texture aware segmentation resolves.

One drawback of the Snakes approach is the inability to change the topology of the contour as it evolves. This means that even in the case that the objects in question are clearly distinct in a given view, the algorithm will only produce a single contour, with a joining line of unit width. A post processing step is required to detect this configuration and separate the result into two contours.

4.3.3 Shape Correction Using Object Contour Consistency in 2-D and 3-D

The visual hull [48], denoted by $\mathcal{H}$, is the outer bound of the scene shape based on it's appearance in several images, and may be defined as follows:

$$\mathcal{H} = \{\mathbf{x} | \forall_{i \in 1..n} \, \mathbf{P}_i \mathbf{x} \in \mathcal{C}_i\} \tag{4.14}$$

Informally, if a point is within the visual hull then it's projection falls within the scene silhouette in every image. The visual hull can be calculated as the intersection of the backprojected silhouette cones corresponding to the given set of images. The visual hull is used for shape representation from silhouette algorithms.

The concept of the visual hull also applies to individual components of the scene - single objects or groups of objects. It is by considering the visual hull of objects that the surface can be corrected using the improved contours as calculated in the previous section.

When objects are joined in the scene representation, the surface extends beyond the visual hull of the objects. Correction can then be applied by moving the basis function centres to lie on the visual hull (just as centres were moved to fit certain planes based on image block matching, as described in the previous chapter, see section 3.3.2).

Centres to be moved must lie on or be close to the original merged surface. This check is important as centres are allowed to lie outside the visual hull if they are part of other objects. The following updating formula is used for the selected RBF centres :

$$\hat{\mathbf{c}}_i' = \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{argmin}} \|\hat{\mathbf{c}}_i - \mathbf{x}\| \tag{4.15}$$

where $\|\cdot\|$ denotes the Euclidean distance. Thus the basis function centres are forced to be located on the visual hull $\mathcal{H}$ while producing a minimal change to the given scene surface.

Sometimes moving the centres does not result in the expected improvement in modelling accuracy. This occurs when the correction requires creation of a depression in the surface, or separation of two objects. In this case RBF interpolated surface will often flatten the surrounding area in order to preserve smoothness. To remedy this problem when centres are moved, extra 'external' constraints with negative weights are created in their place. Recall from section 3.2.3.1 that such constraints are usually only required to provide orientability to the surface. However, their property of enforcing emptyness in their vicinity helps force the surface to obey the necessary depression or separation.

4.3.4 Applicability to Other Representations

It is only the last step of the procedure described above which relates specifically to the radial basis function modelling. The same contour correction methodology can be applied to many different shape representations. For example, using a voxel representation you can simply eliminate any voxels which lie outside the true visual hull in any image.

For mesh based representations vertices can be moved using the same formula given above for RBFs, equation (4.15). However, care must be taken not to produce a degenerate mesh (e.g. one in which triangles intersect each other) is not produced. It may be necessary to iteratively move the vertices with remeshing in between steps, or use a complex post processing step. This highlights a key advantage of radial basis function surfaces in that guaranteeing orientability and non-degeneracy are trivial.

4.4 Analysis of Object Separation in Multiple Object Scenes

4.4.1 Introduction

This section presents an analytical study of the factors which influence the ability of multi-object 3D reconstruction algorithms to correctly distinguish between separate objects. Detection of spuriously connected objects relies on disparity between the 2-D outlines of an object observed in the image and the outlines you would expect to get if the 3-D model was accurate (predicted contours). The error between pairs of countours will signal that a case of fused objects exists and guide the reconstruction accordingly.

It is the variables which determine this error which will be investigated here in a simplified environment. Despite these simplifications a closed form solution for the this error with respect to each variable is likely to be intractable so a simulation is used to provide empirical data about the relationships.

4.4.2 Experiment Design

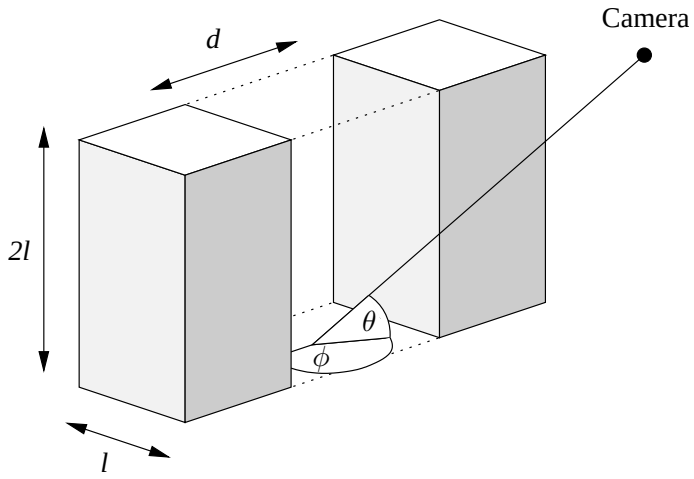


Figure 4.1: Key variables under investigation

Two object types are considered, in order to evaluate the degree to which shape influences the error. These two types are a pair of cuboids with square base of side length l and height $2l$, and a pair of cylinders of diameter l and height $2l$. The shapes stand upright on an horizontal plane (see Figure 4.1) separated by a gap of size d (the distance between object centres is $d + l$).

There are two scenarios for each object type, one in which there are two distinct objects, and one in which the objects are joined by filling in the gap between them (see Figure 4.2) for an example of this in the case of cylindrical objects.

(a) (b)

Figure 4.2: Two cases considered (a) Cylinders joined (b) Cylinders separated

Recall from section 3.2.1 that there are six extrinsic camera parameters, three of which correspond to the position in space of the centre of projection and three angles which determine camera orientation. The camera's position in space can be expressed in spherical coordinates $[\phi\ \theta\ r]$, this is convenient since we wish to fix r, the distance from the centre of the scene, and allow the elevation angle, θ, and azimuth angle, ϕ to vary. The camera is assumed to be always pointing toward the centre of the scene with no tilt/roll, which determines the remaining three extrinsic parameters.

The shapes are similar to shapes present in real scenes (particularly the cuboids). The size of the objects as well as the distance between them, and camera positions were all chosen to closely approximate the arrangement in one of the real scenes I used. Figure 4.3 shows the objects to scale, along with a selection of 16 camera positions with the same elevation angle.

2-D projections of these scenes are obtained using the intrinsic camera parameters as measured in section 3.2.1, thus at least geometrically resembling the images you would get from a real system though they lack noise, illumination variation, shadows etc. As there is no background, the object contours are easy to extract.

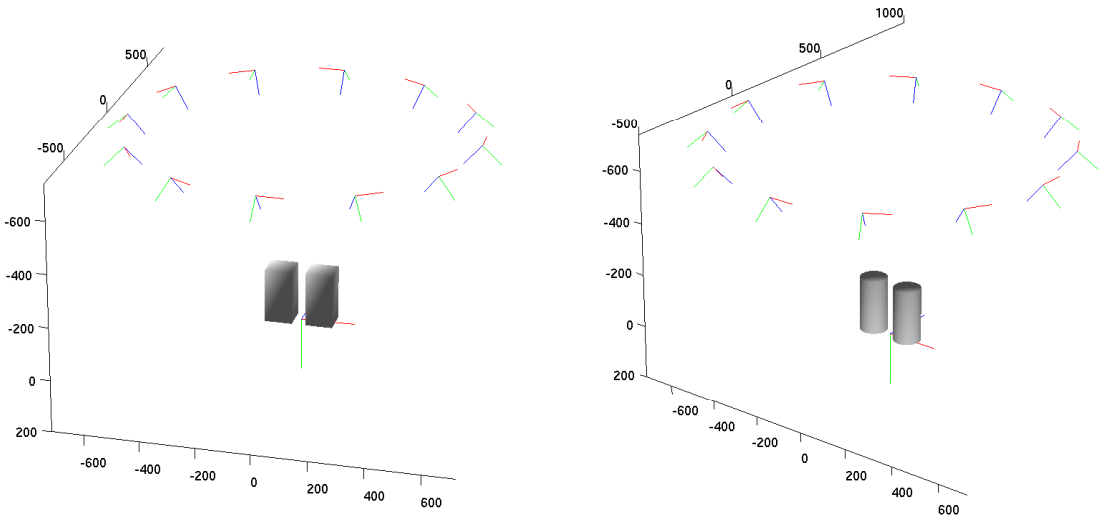


Figure 4.3: Synthetic Scenes, overlaid with the camera positions

4.4.3 Analysis

This section explains the various factors which were investigated and discusses the results of the simulation.

The ability of a system to distinguish between two distinct objects and a single large object will be judged according to the size of the error measured between contours obtained from the fused object scene with respect to the separate object scene. Two methods of calculating this error are considered here. The first is the Hausdorff distance [44] where $\mathcal{F}$ represents the set of points $\mathbf{f}$ that comprise the fused contour, and $\mathcal{S}$ the set of points $\mathbf{s}$ belonging to the separate object contour, is defined as

$$e_{\mathrm{H}}(\mathcal{F},\mathcal{S}) = \max_{f\in\mathcal{F}} \min_{s\in\mathcal{S}} ||f-s|| \tag{4.16}$$

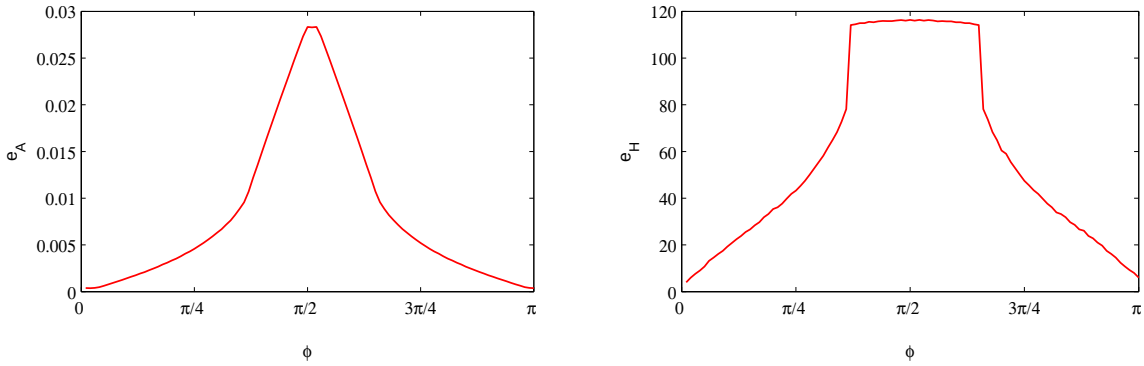
The shortest distance is found between every pair of points from F and S, and the greatest of these distances is returned. The Hausdorff measure can be thought of corresponding to the size of the largest divergence between the two contours. If they are everywhere reasonably close to each other the Hausdorff distance will be small, whereas if one moves further away at any point this will result in a large error.

The other measure considered is the difference in the area enclosed by each contour. If $\mathcal{F}^{+}$ and

$\mathcal{S}^+$ represent the set of points that fall within each of the contours, this corresponds to the symmetric difference, normalised by the cardinality of $\mathcal{S}^+$.

$$e_{\mathrm{A}}(\mathcal{F}^+, \mathcal{S}^+) = \frac{|(\mathcal{F}^+ \cup \mathcal{S}^+) \backslash (\mathcal{F}^+ \cap \mathcal{S}^+)|}{|\mathcal{S}^+|} \tag{4.17}$$

This is the difference between the area of the combined contours and the area of the overlap between the contours. Again contours which are always close to each other will result in a small value of the error. Cases in which one contour significantly overlaps the other will result in a greater error. However, if the area of the overlap is small relative to its size (for example if it forms a thin peninsular) this will not result in a large error - in contrast to the Hausdorff distance.



(a) Area Difference (b) Hausdorff Distance

Figure 4.4: Comparison of error measures for the cuboid scene

Figure 4.4 shows the two error measures for the cuboid scene when varying the azimuth angle θ. Initially viewing the shapes end on, there is very little difference between the outlines of the pair of cuboids and the large fused cuboid. As the camera pans around, the disparity becomes more visible, reaching a peak when the camera is perpendicular to the gap between the objects and falling again as the camera views them end on once again.

The Hausdorff distance is more descriptive in terms of topological differences, which accounts for the large spike as soon as the gap between objects becomes visible. The surface difference is more sensitive to general changes in contours and will thus be used for the other experiments in this section. However, the Hausdorff distance is more reliable in real world scenarios where noise is present due to a variety of factors beyond control, as we shall see in the next experimental result.

4.4.3.1 Influence of Object Shape

In this experiment I will study how the shape of the object affects the error which is observed. I set the distance between the objects equal their width ($d = l$). The elevation θ was fixed at 0.85 radians (again corresponding to values found in the real scenes) and ϕ varied from 0 to π (a full revolution was not necessary as the second half would be identical to the first). Figure 4.5 shows the results of this experiment, using the area error measure from equation (4.17).

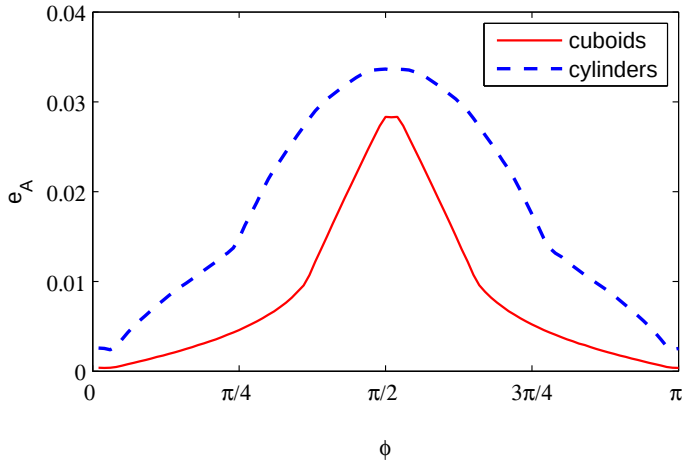
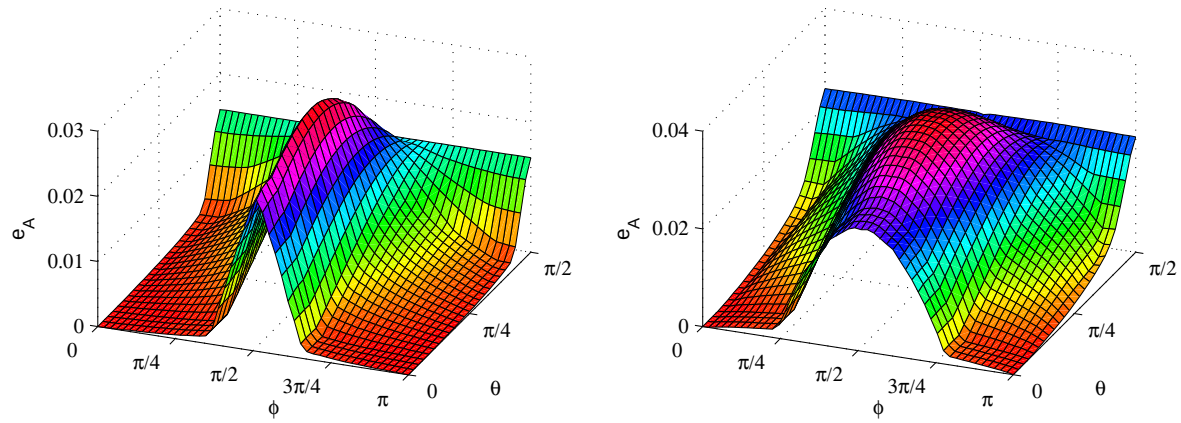


Figure 4.5: Error measure as a function of shape

It can be observed from Figure 4.5 that in the scene with cylindrical objects, disparities between the fused and separate case are detectable from a wider range of camera positions, hence the wider peak to the curve. However the shape of both lines are not quite similar, so in the real world noise may mask this trend. Also apparent from the figure is that the error is higher for the cylinders at every point. This can be explained by the fact that fusing adds more in terms of volume to the cylinders than it does to the cubes. The fused and separate cylinder scenes are thus more 'different' from each other than is the case with cuboids.

4.4.3.2 Varying the Camera Elevation

In this experiment the elevation angle ϕ is varied, in addition to θ. Recall from section 4.4.2 that in this simulation camera positions have only two free parameters. Clearly the viewing angle is a very important factor in the ability to correctly distinguishing between compound and separate objects. Distance between the objects is kept at l. Figure 4.6 shows a plot of area error against ϕ and θ.



(a) Cuboids (b) Cylinders

Figure 4.6: Error with respect to camera angles

It is easiest to interpret this figure with respect to ϕ. Assuming the standard configuration of a set of inward pointing cameras surrounding the scene, the surface represents error profiles for several different choices of elevation. There is little change to the shape of the peak (and thus ability to differentiate separate and joined objects) until $\phi = \frac{3\pi}{8}$ when the error starts to increase for all values of θ. It is easy to see why this occurs by considering the most extreme case, where $\phi = \frac{\pi}{2}$. In this configuration, the gap is always visible as the camera is directly above the objects and thus θ no longer has any effect. The result of increased elevation and thus independence from θ is that separation becomes possible for fewer cameras/images and is thus more reliable. The corollary of this is that adding a single high elevation image is likely to lead to a large improvement in the result by revealing the gap between the objects. This assumes that most real world scenes will consist of tall objects on a ground plane, for other arbitrary configurations the issue is more complicated and there may be no way to gaurantee good separation of objects. Also, the availability of high elevation images depends on the scene, for example it is relatively easy to achieve for a collection of small objects, however if your scene consists of buildings, for example, it may be very difficult to acquire images from above.

4.4.3.3 Evaluating the Number of Cameras Necessary

In this experiment the goal is to determine how the number of cameras used influences the observed error. This variable is very important in determining the performance of any multi-view 3D recon-

struction technique as increasing the number of cameras will generally improve the result at the expense of additional processing requirements.

Again a circular configuration at fixed elevation (ϕ = 0.85 rad) is assumed. To manage the complexity I will also assume that for a given n camera configuration the positions are evenly spaced. However, there may still be differences between the errors observed for two different evenly spaced n camera configurations with different θ offsets. With this in mind, for each value of n, I will record the error measured between the fused and separate case for all possible offsets, and calculate the minimum and maximum errors that arise. These extremes represents the best and worse case for detecting the separation for each configuration. The results are shown in Figure 4.7. The figure clearly demonstrates the need to consider all possible evenly spaced n camera configurations due to the fact that the best case results are almost totally invariant to n. This is because deciding whether two objects are joined can be achieved with as few as one or two cameras - if they happen to be in exactly the right position to observe the gap between the objects.

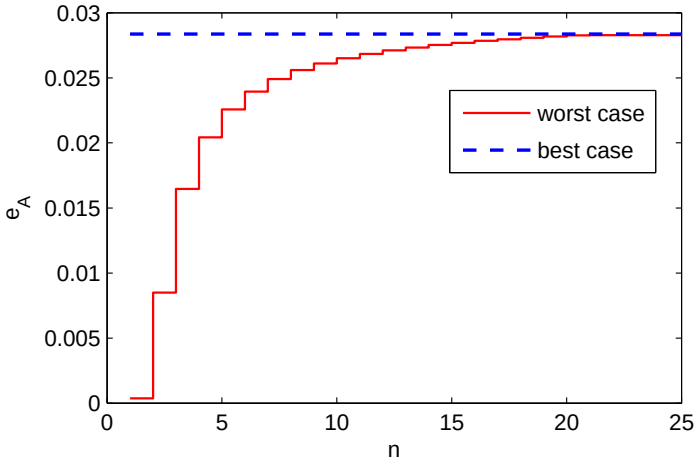


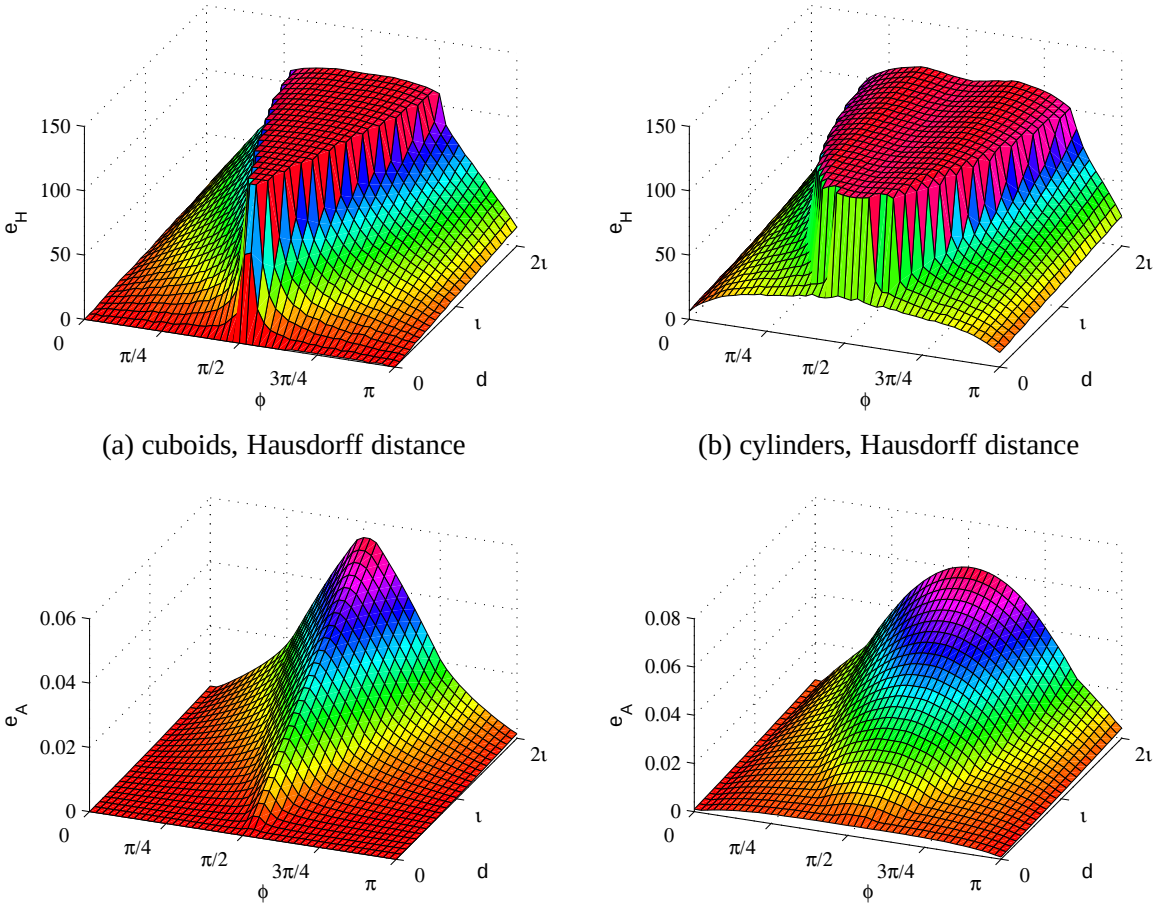
Figure 4.7: Error plotted against number of cameras

The shape of the worst case performance line fits with what you would expect given that there are only a certain number of angles for which the gap is visible. When the number of cameras is sufficient to guarantee that at least one camera falls within this zone, the worst case performance suddenly picks up. After this point there is little improvement (with respect to identifying fused objects) when using additional cameras, as illustrated by Figure 4.7, where the worst case performance curve converges asymptotically toward the best case as n increases.

4.4.3.4 Evaluating the Object Separation Distance

The distance d between the objects is another factor that would be expected to have a large influence on the result, especially considering the extreme case when the separation is almost zero there is a miniscule difference between the joined and unjoined cases.

In this experiment d is varied by moving the objects away from the origin by some distance. A circular configuration of cameras at elevation 0.85 is used as before. A plot of both area error and Hausdorff distance against θ and d is shown in Figure 4.8.



(a) cuboids, Hausdorff distance

(b) cylinders, Hausdorff distance

(c) cuboids, area difference

(d) cylinders, area difference

Figure 4.8: Error plotted against inter-object distance

Again the general trend is as expected, smaller gaps are harder to identify by the given array of cameras. The shape of the Hausdorff distance plots (a) and (b) clearly shows the width of the peak getting smaller as the gap is reduced. The area error plots (c) and (d) show that the profile of

the two different shaped objects follows the same trend as in Figure 4.5 with the cylindrical shapes yielding a wider more gently sloping peak. Also note that with the cylindrical objects the error never disappears, even when $d = 0$, as there is a difference between a scene with two cylinders touching and the same two cylinders joined by extending to the sides (please refer to Figure 4.2).

The degree to which performance suffers for small separations hints at fundamental limitations of any multiple object 3D reconstruction algorithm. A small increase in error is likely to be undetectable in the presence of noise thus when the gap is small the chances of being able to detect and resolve this in the model are slim, even when many images are used. This also has implications for the recovery of fine details that are part of a given object.

4.4.4 Conclusion

This section presented experiments to determine how the observable error associated with incorrectly joining a pair of objects varies depending on: shape, angle, number of cameras and separation distance.

The shape of the objects determines how quickly the error rises when the gap between the objects starts to become visible, and thus the range of angles for which the error is above a given threshold. From this we conclude that it is easier to detect that two cylinders have become joined as opposed to two cuboids.

The azimuth angle θ is primarily responsible for variations in the observed error since it directly affects whether or not the gap between the objects is visible (this trend is shown in all experiments). As the elevation angle ϕ increases, so does the number of views which display a high error, again as it controls whether the space between the objects is visible. Higher angles are thus preferable for this task. It's worth noting that for other tasks lower elevations are more helpful due to the larger baseline [71]. However since almost any high angle is able to observe the gap, the addition of a small number of images taken from above will in most cases lead to a large improvement in results.

A larger number of cameras will on average lead to a greater detection rate, however a very small number of cameras can still be effective if they are in fortunate positions. The mechanism for this is the same as before and centres around when the gap between the objects is visible in different amounts. There is a hard limit for the number of camera positions, beyond which adding additional cameras leads to no further improvement.

Finally the separation distance also presents limits to detection as both the visibility angles and

difference in projection of the contours shrink when d is reduced.

The results from these experiments help set out the operating limits for any multi-object 3-D recovery algorithm, and suggest a reasonable number of cameras that should be for the task. As expected closely placed objects are unlikely to be separable unless there is an extensive collection of images available. However, overhead cameras can be very effective for separated free standing objects.

Due to the fact that occlusion by other objects does not occur in the simulation, the performance on more complex collections of objects will be worse than the cases analysed in this study. The correct demarcation of individual objects is a very difficult problem in practice and in some cases complete scene recovery will be impossible for any realistic number of cameras external to the scene.

In the following section we will analyse the 3-D reconstruction of a real scene with several objects using multiple images.

4.5 Experimental Results

This section presents the results of experiments carried out to test the algorithms presented in Section 4.3, using real data. The dataset used is of the first object collection Figure 3.5 from the previous chapter. The original input images and camera calibration information were used. The RBF surface after correspondence based based updating (see Section 3.3.2) provided a starting point.

This dataset was chosen as it exhibits a very good example of objects which are spuriously fused together in the reconstruction. The knife block and kettle are almost completely joined, due to their proximity, and as a result of errors in the voxel model. These errors themselves stem from the similarity in colour between the carpet and knifeblock, and the fact the gap between the objects is only visible in two views due to occlusion from the cereal box. The second dataset did not contain any fused objects after the disparity based updating procedure.

Figures 4.9 and 4.10 show the input data. Figure 4.9 shows two of the input images (which are identical to those used in the previous chapter) whilst Figure 4.10 (a) shows the RBF surface (after disparity based updating), and Figure 4.10 (b) shows the fused object pair itself after segmentation from the rest of the scene. The pair of objects were segmented by thresholding the scene to remove the ground plane .

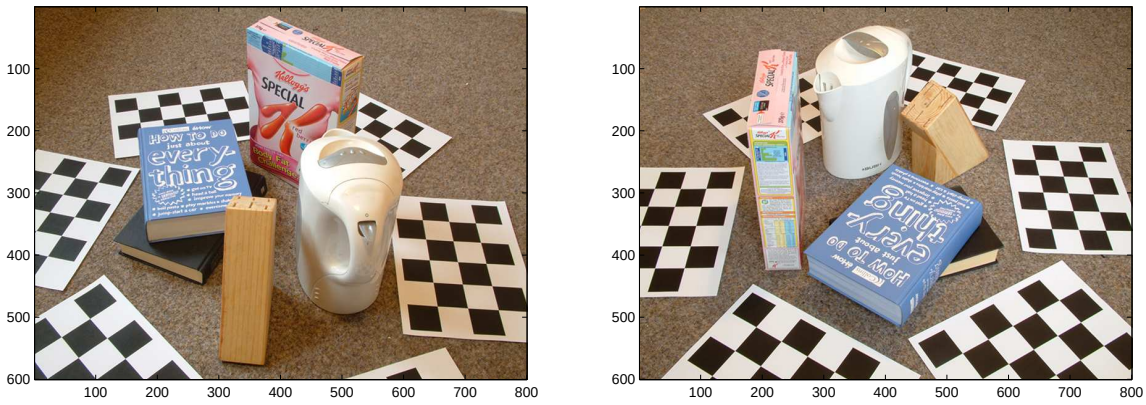
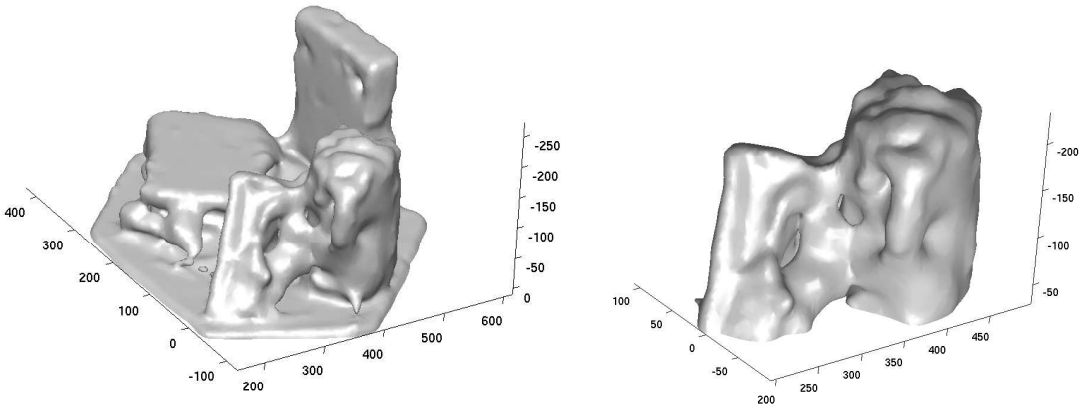


Figure 4.9: Two of the twelve input images



(a) (b)

Figure 4.10: The input surface before (a) and after (b) the problem objects have been segmented

4.5.1 Feature Selection

The texture filter described by Equation (4.2) from the previous section was applied to the input images in order to provide additional information for segmentation. This is combined with the three colour channels to give a four dimensional feature space. The influence of the texture channel is controlled by scaling parameter ζ, for example a very large weighting value has the tendancy to allow texture to dominate the segmentation. A setting of $\zeta = 2$ (meaning texture was twice as important as any single colour channel) was found to give good results across all of the images. The width of the filter, σ was set to 2 pixels and the output was clamped to the interval [0 2000], and then normalised to 255.

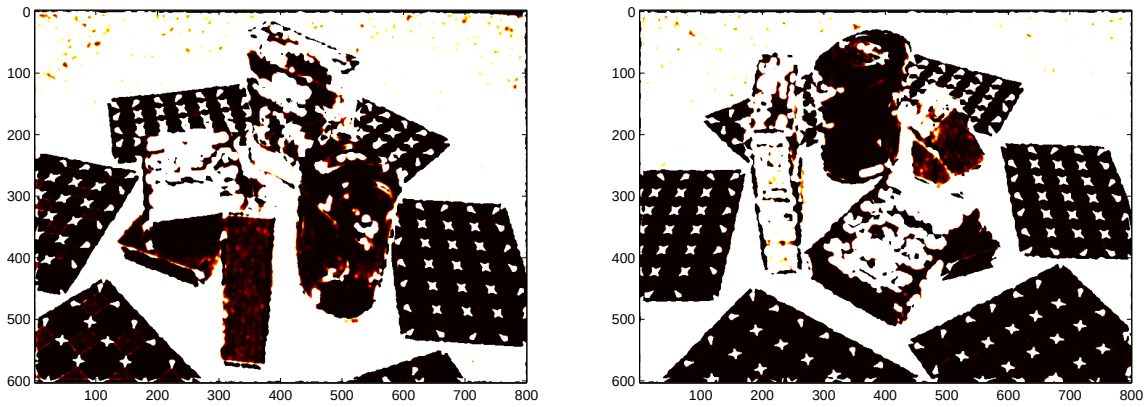


Figure 4.11: Texture response in two images

Figure 4.11 shows the response of the texture filter in 2 views ranging from 0 (black) to 255 (white). The filter clearly distinguishes between the carpet and wood grain of the knife block. It also exhibits a high response around very strong edges as can be seen on the blue book cover text. This is an undesirable effect as strong edges and grainy textures are very different in character. However in this case the results of contour fitting are not significantly compromised due to the fact that the new contour is constrained to lie within a certain distance of an initial estimate.

4.5.2 Unsupervised Segmentation

In this experiment the supervised segmentation method based on mean shift clustering [31], described in section 4.3.2.2, was applied to the feature set described above. The implementation con-

tains code for mean shift clustering by Bryan Feldman, `http://www.bryanfeldman.com/`.

As this segmentation method is based on an unsupervised classifier the number of classes is assumed to be unknown. The sensitivity, and hence the number of classes that result, is based on a parameter of the density estimation step called the bandwidth.

If the value is too low, the filter is too sensitive and oversegmentation occurs, if the value is set too high undersegmentation occurs. Generally oversegmentation is not a problem unless the area between the two objects becomes split into two or more classes. This will result in spurious edges being passed to the contour fitting step. Undersegmentation can be tolerated provided that the objects in question are assigned to different classes to each other, and to the background! A bandwidth of 25 was found to be effective across all images.

Figure 4.12: The unsupervised segmentation results for one of the input images

Figure 4.12 shows the results of the segmentation for various input images. Segments have been coloured according to the average colour of pixels belonging to each segment. As can be observed from the figure, the algorithm segments well the various different objects in the scene. Key edges are preserved despite the oversegmentation of the objects (particularly the kettle). Even though the image is oversegmented in the sense that each object is made up of several segments, the set of segments belonging to each object are fully disjoint and thus the object boundaries are not compromised.

4.5.3 Supervised Segmentation

In this experiment the supervised segmentation method based on Support Vector Machines described in section 4.3.2.1, was applied to the feature set obtained in section 4.5.1. The implementation contains code for Support Vector Machine classification by Gavin Cawley, from `http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox`.

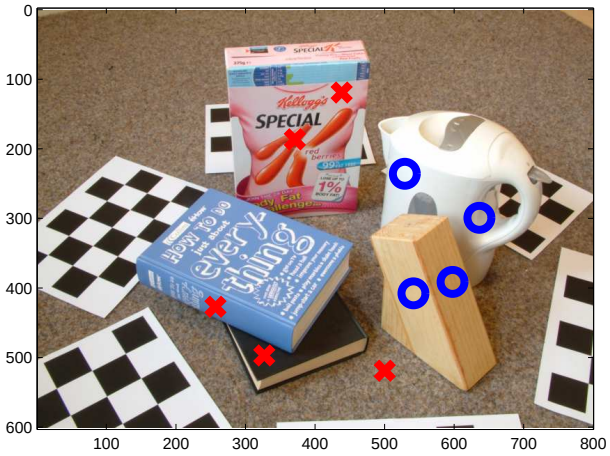
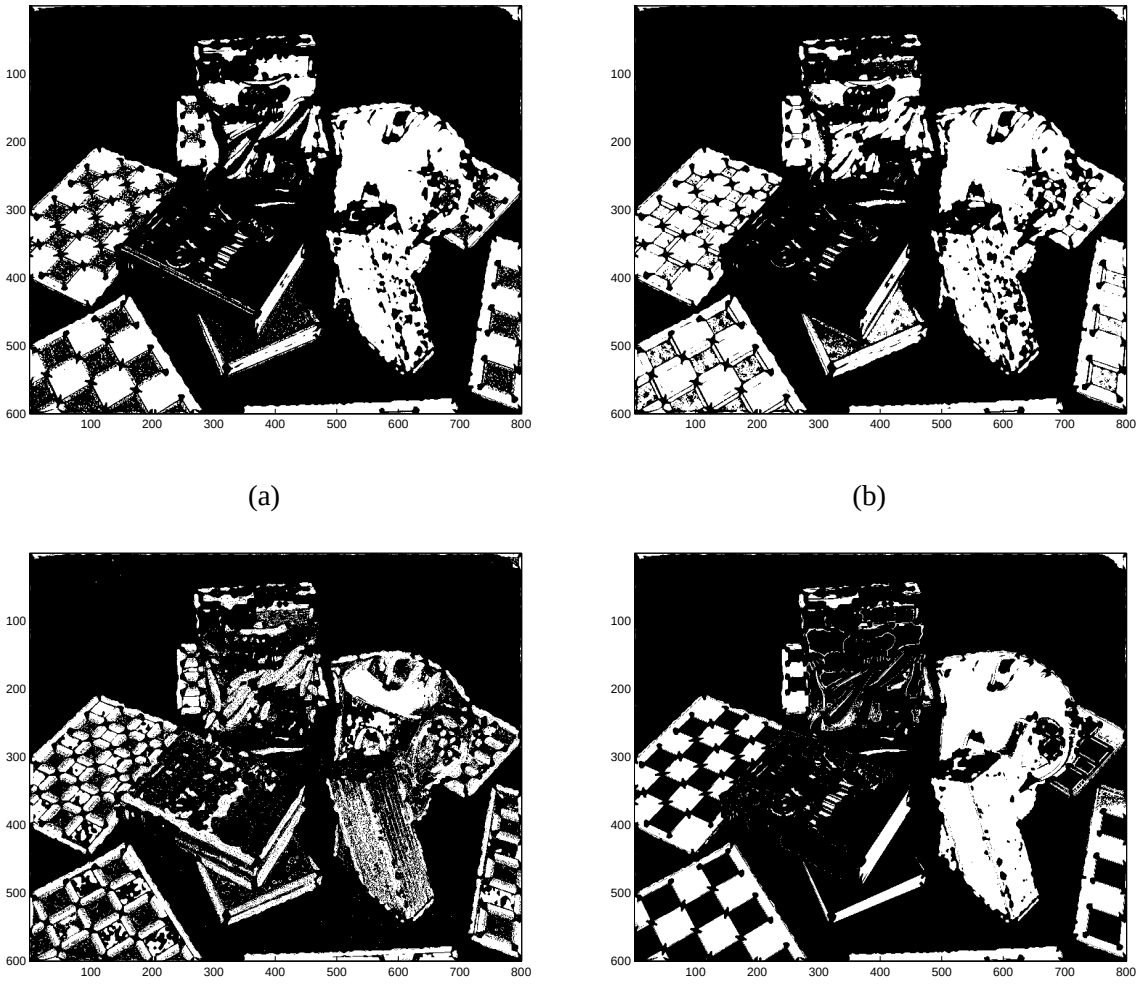


Figure 4.13: The image used to train the algorithm overlaid with samples belonging to class 1 (red crosses) and samples belonging to class 2 (blue circles)

This method is supervised and thus requires training. Image 3, shown in figure Figure 4.13, was used as training data. Samples belonging to class 1 are marked in with red crosses and samples from class 2 with blue circles. In total two points from each object and one or two points from the each background object were sampled giving training set of size 11. This image was specifically chosen as it doesn't show the gap between the objects so that no samples are taken from this area to avoid biasing the result.

Figure 4.14 shows the results of applying different kernels to the SVM algorithm. White represents class 1 (the objects) and black represents class 2 (the background). The linear kernel performed better than expected, the various colours and textures are well separated in the feature space. The polynomial kernel (b) gave very similar results with the main difference being the inability to distinguish tones close to black.

The b-spline kernel gave the worst results which is somewhat surprising, failing to distinguish between the forground and background classes at all. Interestingly the Gaussian RBF kernel (d)



(a) (b)

(c) (d)

Figure 4.14: Results of applying different kernels, (a) linear (b) polynomial (c) bspline (d) gaussian RBF

gave the best results, this is essentially the same kernel that performed poorly at surface modelling.

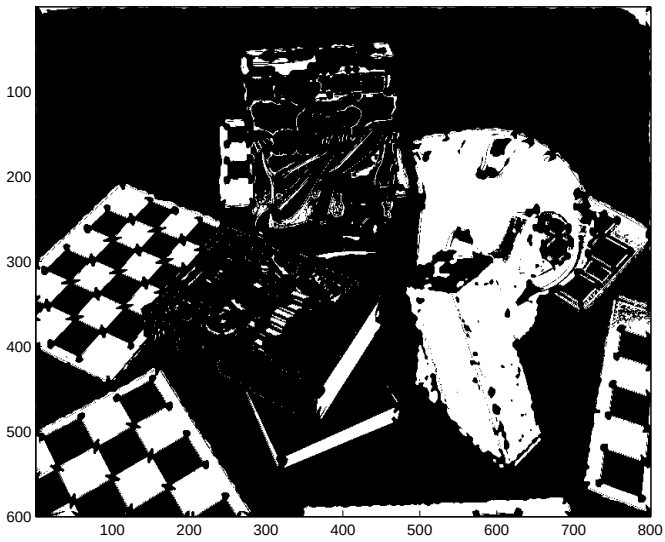


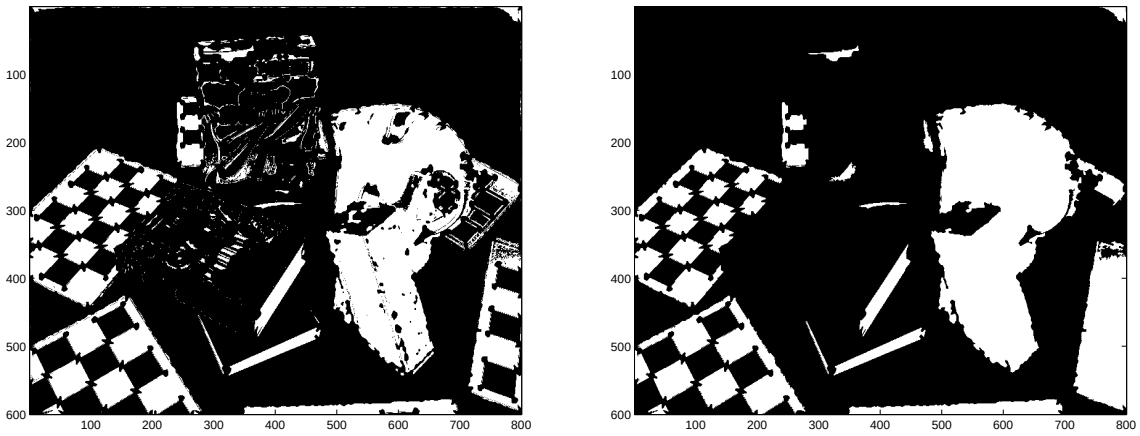
Figure 4.15: Final segmentation result

Figure 4.15 shows the final result using the Gaussian kernel for image 3 (which corresponds with the results from the unsupervised method, Figure 4.12). Parts of the background are falsely classified as the objects due to containing exactly the same colour and texture properties as the objects in question. However since the boundary of the objects is distinct in most cases and due to the search area restrictions this is not a problem, and in general the segmentation is successful. One problem area compared with the unsupervised results is the top of the knife block, which is assigned to the background. This region is very similar in colour (but not texture) to the carpet.

4.5.4 Generating Contours

The contour fitting method described in section 4.3.2.3 is applied to the both the unsupervised and supervised segmentation results in this experiment. The implementation contains code for the Gradient Vector Flow snakes algorithm by Chenyang Xu and Jerry Prince, `http://iacl.ece.jhu.edu/projects/gvf`. The elasticity, rigidity and viscosity parameters were set to 0.05, 0, and 1 respectively. As binary edge maps are used (in contrast to edge maps with varying intensities) the snake is not overly sensitive to these settings as, for example it doesn't have to overcome weaker 'false' edges or be resistant to gaps in the edge map. Elasticity needs to be

sufficient to allow the snake to fully explore concavities in the contour. Snake settings were kept the same for the unsupervised and supervised datasets. The projection of the fused object RBF surface (see Figure 4.10 (d)) is used to initialise the snake (see Figure 4.17 (b)). A search radius of 50 pixels is applied to prevent the snake from moving too far away.



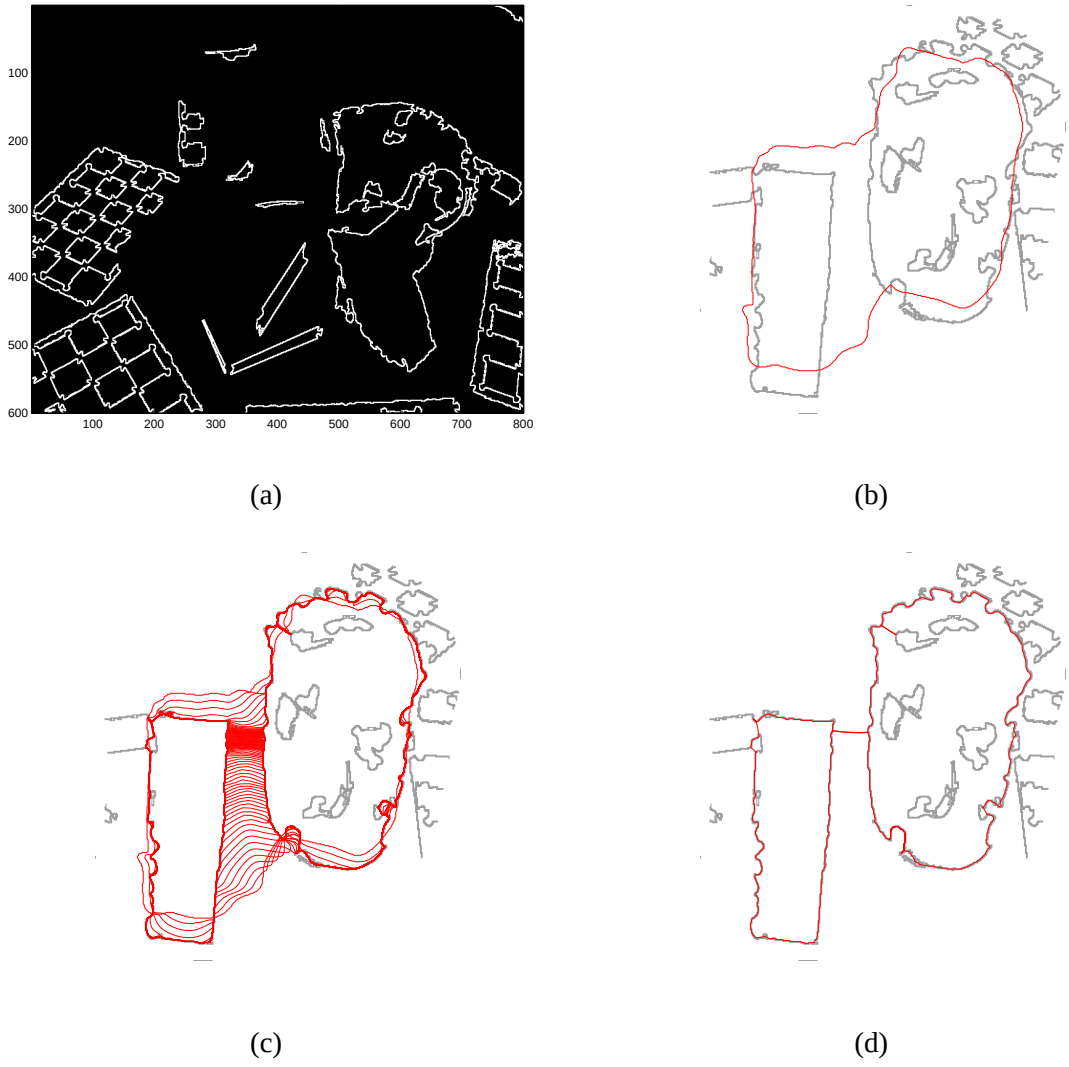
(a) (b)

Figure 4.16: Merging of small segments (a) before (b) after

Before the edge map is obtained there is a filtering stage which removes classes with fewer than 100 pixels and assigns their pixels to the largest adjacent class. This helps remove some noise since such small clusters are unlikely to be significant. Figure 4.16 shows the results before and after this operation on the supervised segmentation data. As can be seen from the figure the output is visibly cleaner. Many of the falsely classified background areas are corrected, for example the cereal box and the book text. Similarly, falsely classified areas of the knifeblock and particularly the kettle are corrected in this step.

Figure 4.17 (a) shows the resultant binary edge map (before the search radius restriction was applied). Figure 4.17 (b) shows the initial state, the thin red line represents the initial location of the snake whilst the edges within range are shown in grey. (c) represents the progress of the snake as it evolves from the initial contour estimate to fit the edge data shown in intervals of 15 iterations. (d) shows the final result the snake converged on. A limit of 750 iterations total was required to allow the snake enough time to fully extend into all concavities.

Performance of the snake was very good in all cases. Comparisons between the initial and final



(a) (b)

(c) (d)

Figure 4.17: Progress of the snakes algorithm (a) input edge map (b) snake initialisation (c) progress in intervals of 15 iterations (d) final snake result

contours, overlaid with the image data are shown in Figure 4.18 for the unsupervised segmentation and in Figure 4.19 for the supervised segmentation. Note that due to the thresholding of the ground plane, the initial contours (red) do not extend to the bottom of the objects.

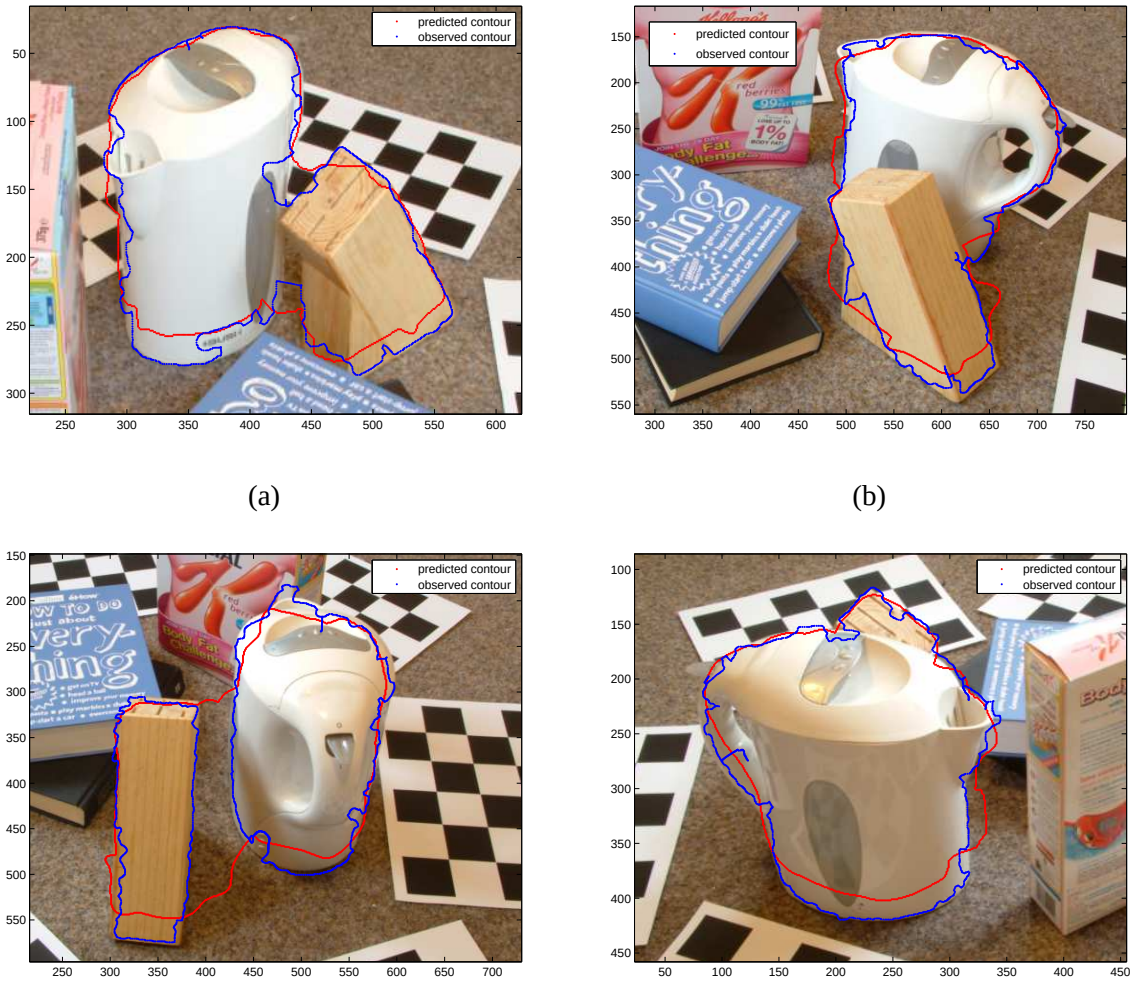


Figure 4.18: Initial and final contours for various images using unsupervised segmentation

In general the results from the unsupervised segmentation were better. There were fewer parts of the objects missing (false negatives) and fewer areas where the contour extends beyond the objects (false positives). Figure 4.20 shows this trend numerically, contours extracted from image data are compared to the ground truth (obtained by hand) using the Hausdorff distance. The performance of the supervised algorithm is slightly worse in almost all images.

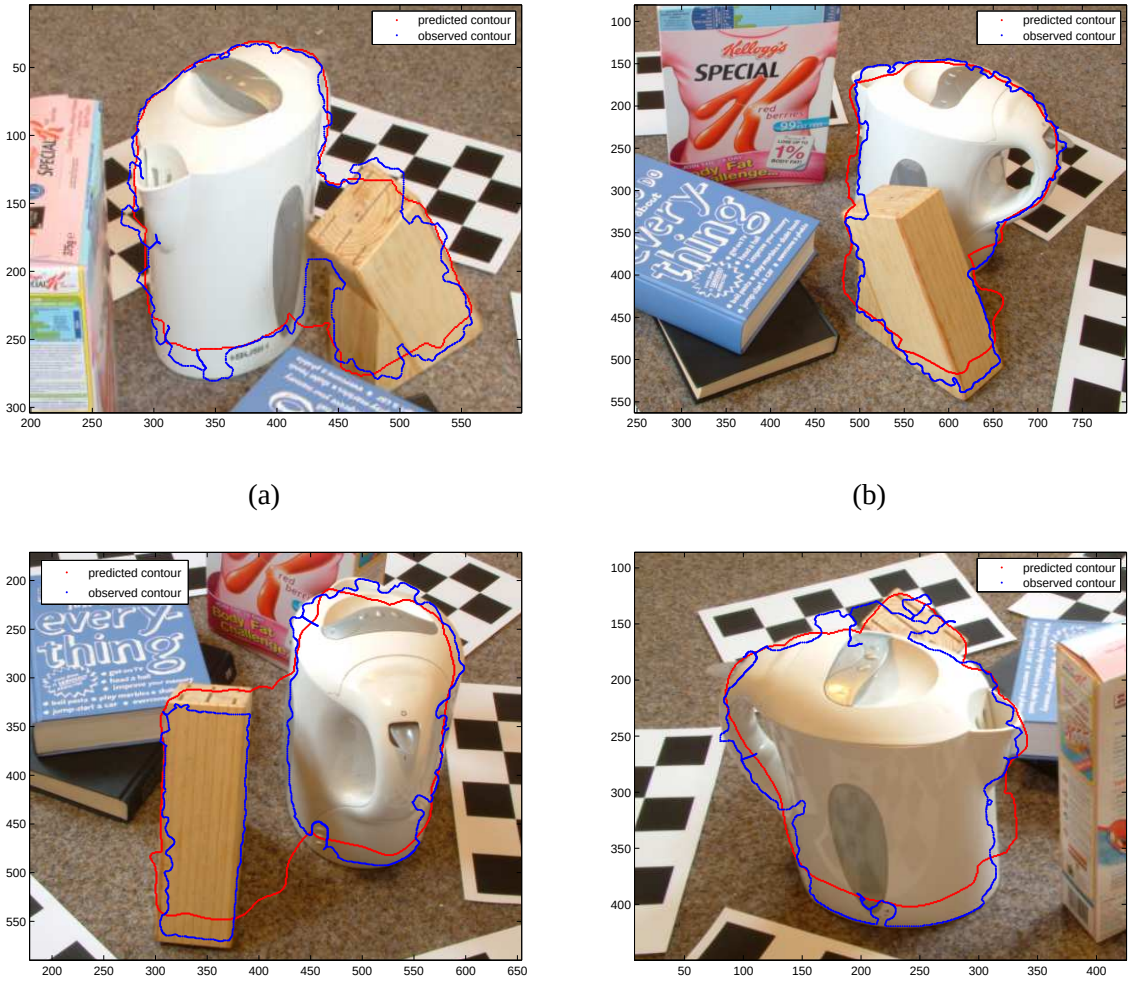


Figure 4.19: Initial and final contours for various images using supervised segmentation

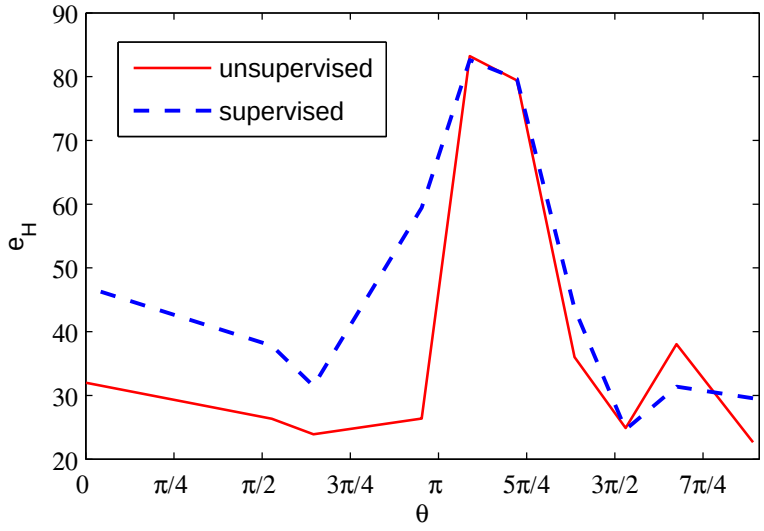


Figure 4.20: Numerical accuracy of extracted contours as evaluated against the ground truth using the Hausdorff distance

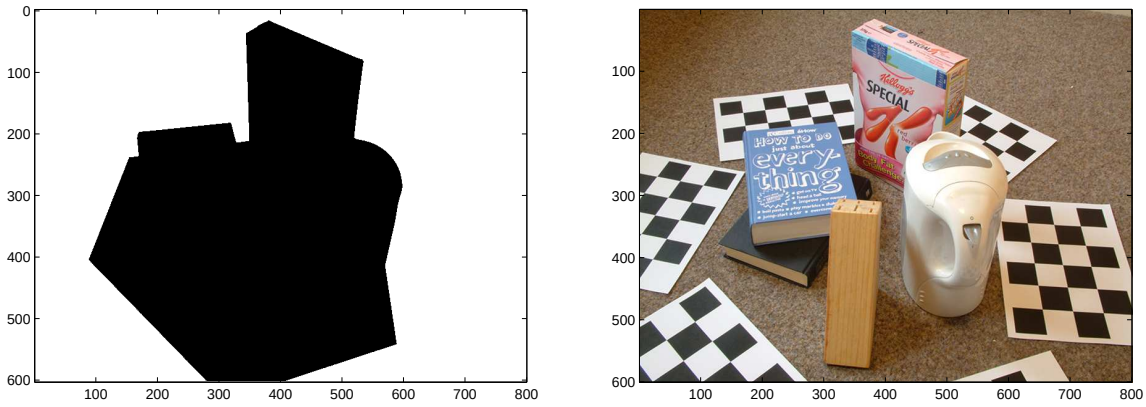
This can be explained by the fact that the two class model used in the supervised case is too restrictive for this application. It doesn't matter that background objects are assigned to many different classes, likewise it is not essential that the objects in question are uniquely segmented. There could be many classes as part of each object - the only important fact is that class boundaries correspond to genuine edges in the image.

4.5.5 RBF Surface Correction Using Contours

In this experiment the extracted contours are used to correct the input RBF surface (see Figure 4.10 (c)) using the methods described previously in Section 4.3.3.

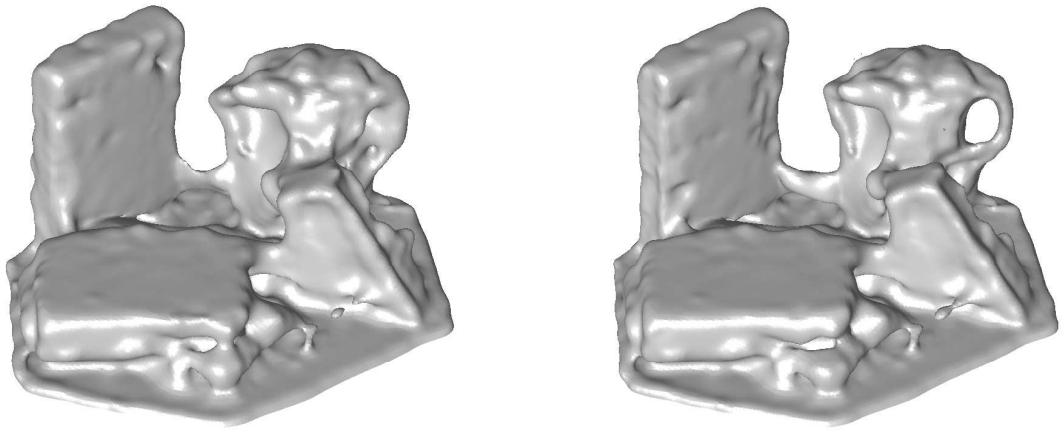
Recall from Section 3.4.1 that a manual segmentation was performed on the images to ensure the scene was contained within a bounding volume for the voxel carving algorithm. This segmentation removes the background and thus provides a silhouette of the scene in each image, as shown in Figure 4.21. These silhouettes may be treated as contours and used to correct the RBF surface, using the contour updating method. No initial contour is required since we are dealing with the entire scene, every basis function centre is checked for consistency.

The results are shown in Figure 4.22. The most obvious improvement is in the handle of the



(a) (b)

Figure 4.21: (a) segmented input image (b) scene silhouette

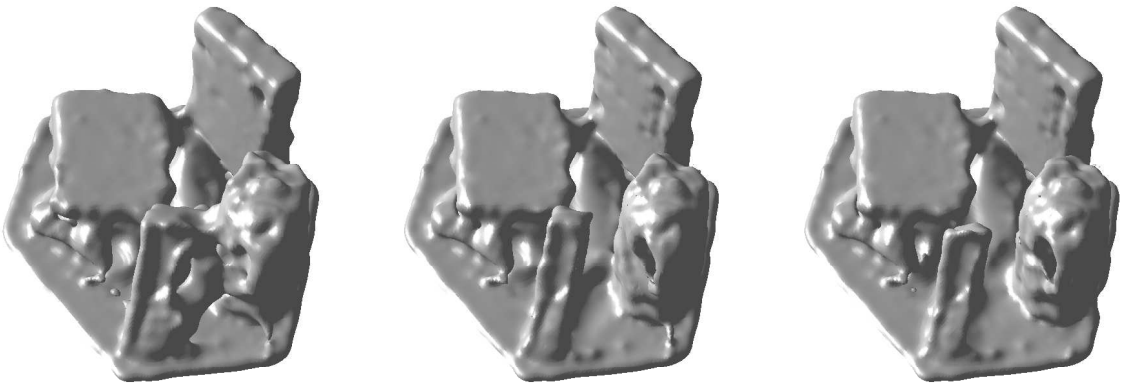


(a) (b)

Figure 4.22: silhouette based correction (a) before (b) after

kettle. This had previously been smoothed over by the RBFs but movement of centres away from the problem area and the creation of negative constraints have allowed the recovery of this thin structure. This demonstrates that with accurate data, the smoothing tendencies of RBF surfaces can be overcome to recover fine details. There is also a slight reduction in a bridge between the kettle and cereal box. Elsewhere in the model the RBF surface closely follows the silhouettes so no further improvement occurs.

Bulding upon this result the contours extracted by both the unsupervised and supervised segmentation algorithms were applied in turn. The same initial 3-D segmentation (see section Section 4.5) was used in both cases to limit the basis functions that were checked for contour consistency. The results are shown in Figure 4.23 and compared to the surface before silhouette correction.



(a) (b) (c)

Figure 4.23: Correction results (a) initial surface before correction (b) surface after contour correction with the unsupervised method (c) surface after contour correction with the supervised method

There is clearly a large improvement in the area between the kettle and knife block, with both methods achieving a complete separation of the two objects. Initially the appearance of the surface is very similar between the unsupervised and supervised results, however it should be noted that the upper part of the knife block is falsely truncated by in the supervised result. This can be explained in reference to the contours; in Figure 4.19 (c) the extracted contour misses the top of the knife block due to misclassifying the whiter tone of the wood as background. Compare this to Figure 4.18 (c). This error is sufficient to move the centres too far down the knifeblock, reducing its height.

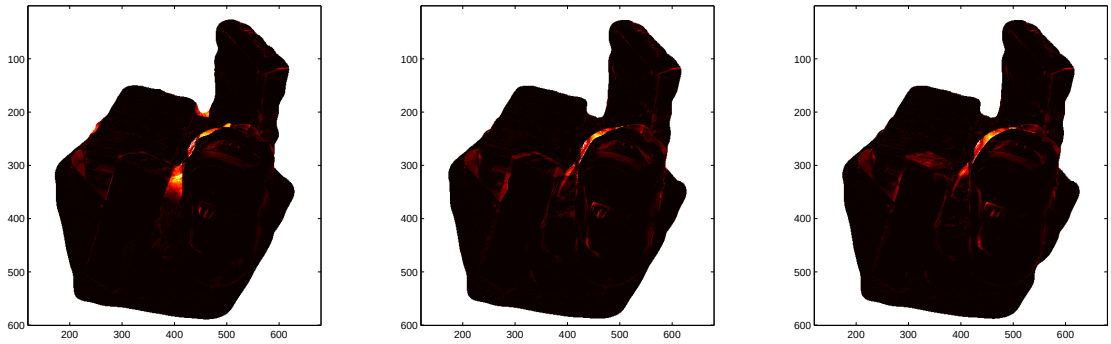
Figure 4.24 shows the coloured results for the same three surfaces. As expected there is an

(a) (b) (c)

Figure 4.24: Coloured renderings from (a) initial surface before correction (b) surface after contour correction with the unsupervised method (c) surface after contour correction with the unsupervised method

improvement after contour correction with objects now visible through the gap between the kettle and knife block. The only clearly visible difference between the results from the two segmentation methods is the top of the knife block, which is missing in the supervised result, confirming what was observed from the 3-D surfaces.

It is possible to quantify the differences between the coloured surface renderings and one of the input images by considering the squared errors or a per pixel basis. Plots of these errors for the three surfaces are shown in Figure 4.25



(a) (b) (c)

Figure 4.25: squared pixel errors between an input image coloured renderings from (a) initial surface before any correction (b) surface after contour correction with the unsupervised method (c) surface after contour correction with the unsupervised method

Model	PSNR (dB)
Initial surface	11.17
Surface after contour correction with the unsupervised method	13.91
Surface after contour correction with the supervised method	13.75

Table 4.1: PSNR when comparing rendering images with an input image window

The error is greatest when part of the black book is seen through the gap. In other areas the brown texture of the carpet is very similar to the wood of the knife block yielding a smaller error. The area in between the objects shows a considerable improvement, becoming much more consistent with the input image. The area at the top of the image between the cereal box and book also shows a large improvement, this time due to use of the scene silhouettes.

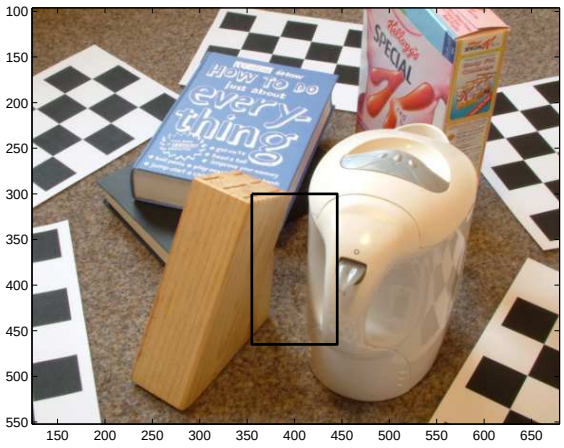


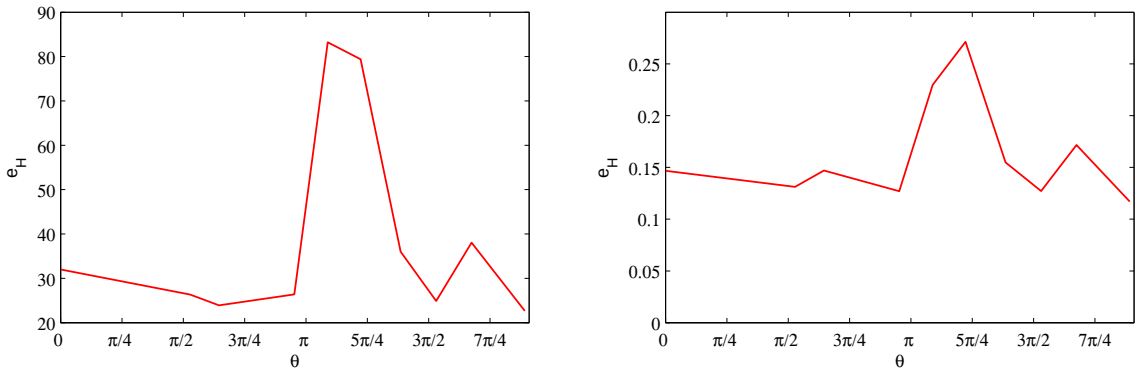
Figure 4.26: The window over which the PSNR between the original and rendered images was computed

The input image was specifically chosen to display the gap between the objects. Even with this consideration, the overall reduction in error is quite small. However, concentrating on the area between the objects, and calculating the PSNR between the coloured renderings and input images across the window highlighted in Figure 4.26 shows that a quantifyable improvement does result, as shown in Table 4.1.

The values show a clear improvement from the uncorrected surface and give a slight advantage to the unsupervised algorithm. Note that the window doesn't fully cover the truncated part of the knifeblock so this difference is on top of the differences already discussed.

4.5.6 Fused Object Disparity

Finally this section presents an experiment into the effectiveness in detecting fused objects using real data. This provides a companion to Section 4.4 which investigated the factors which influence observed contour errors on synthetic data. Figure 4.27 shows the disparity between the contours extracted from the image data and the 'predicted' contours obtained from the uncorrected RBF model using the Hausdorff and area measures defined in Section 4.4.3. The extracted contours were from the unsupervised method.



(a) Hausdorff distance (b) Area difference

Figure 4.27: Numerical discrepancies between the predicted and observed contours

As demonstrated by Figure 4.27 (a), the real data displays almost exactly the same trend as the simulation for the Hausdorff distance, with a peak centred on the angles for which the gap is visible. However when using the area measure (b) the peak is drowned out by noise as small area errors along the contours length add up to become similar in magnitude to the error caused by the incorrect topology. The Hausdorff distance only looks for the greatest deviation between a pair of inputs and so avoids this error aggregation, making it more suitable for use on real data.

Figure 4.28: More views of the unsupervised contour correction result

4.5.7 Final Results

This section presents some more views of the most accurate result obtained (using the unsupervised contour correction method). Coloured renderings from each of the 12 original camera positions are shown in Figure 4.28, a larger version of one of the images, along with the corresponding input image is shown in Figure 4.29.

Figure 4.29: Closeup of one of the final views (bottom), with corresponding input image (top)

4.6 Conclusion

A method was presented to correct RBF scenes using contours extracted from image data, particularly in the case where objects have become undesirably joined by the RBF modelling procedure. This method complements the approach described in the previous chapter by utilising areas with constant colour or texture to correct errors. In addition to this a study is presented into the factors and ultimate limits of such techniques.

The approach fits into a statistical framework for maximising the likelyhood of accurate correspondence between 2-D and 3-D contours, given the image data (see section 4.2). In section 4.3 two methods of extracting contours were presented. Both jointly utilise colour and texture information. The first unsupervised method benefits from a simpler initialisation but can perform poorly when the parameters are not properly set. The second supervised approach is more robust in this respect however some user input is required (in the form of selecting regions to use as training data).

Experiments were presented using real image data (see section 4.5). The object outlines were extracted accurately in most cases, but more importantly the shapes were completely separated by the approach in spite of occasional errors. Some of these errors result from similar colours being present in the background. Performance of the two segmentation approaches were similar however the numerical results indicate an advantage to the unsupervised approach. This may be attributed to the fact that oversegmentation that it may produce is not usually a problem, and the flexibility this adds improves reliability.

The improvement in the model is clear from looking at textured representations from cameras in which the gap between the objects is visible. Before correction the textures are blurred and correspond to different regions in different images. After correction the appropriate background texture is applied to the area between the objects. This improvement is also confirmed numerically by summing the errors between synthesised images of the scene and the corresponding input images.

The theoretical study provided a number of interesting results (see Section 4.4). The shape of the objects determines how quickly the error rises when the gap between the objects starts to become visible, and thus the ease of which an image based algorithm can separate them.

The azimuth angle is primarily responsible for variations in the observed error since it directly affects whether or not the gap between the objects is visible, however as the elevation angle ϕ increases, so does the number of views which display a disparity. Elevated views are thus preferable

for this task.

A greater number of cameras will on average lead to a greater detection rate, however the best case performance indicates that a small number of cameras can be effective if they are in the right position. There is also a practical limit for the number of cameras, beyond which adding additional cameras leads to no further improvement.

Finally, the separation distance also presents limits to detection as both the visibility angles and difference in projection of the contours shrink when the separation distance is reduced. This indicates that closely placed objects are unlikely to be separable unless there is an extensive collection of images available.

Overall the study confirms that modelling multi object scenes is a significantly more difficult problem than the case of single objects which has predominantly been studied previously.

Further work in this area would include a method to automatically detect and segment problem surface regions, or a more sophisticated segmentation approach to reliably find image edges corresponding to all object boundaries. Such a scheme might work best in an iterative manner where 3-D information is fed back into the segmentation/edge detection algorithm to incrementally improve the correspondence between 2-D and 3-D edges.

Finally, increases in the accuracy of the model will at some point require more detailed, higher resolution 3-D models, particularly in the presence of sharp surface features or edges. This might be achieved by either compactly supported basis functions to allow many more centres to be used, or anisotropic basis functions which support tighter curvatures in certain directions (this is particularly useful for representing hard edges for example). Alternatively hybrid representations may be developed to account for both smooth and angular surfaces.

Chapter 5

Conclusion

This chapter presents the conclusions that may be drawn from this thesis. Section 5.1 details the novel contributions made herein. Section 5.2 discusses potential applications of the work. Section 5.3 presents a critical analysis of some of the weaknesses in this approach to 3-D scene modelling, which are then addressed in section 5.4 which discusses future directions for this research.

5.1 Contributions

The first novel aspect of this research is the systematic approach designed specifically for the modelling of multiple object scenes. As discussed in section 3.1 this problem is significantly more difficult than than the case of modelling single objects and thus requires different methods for its solution. Initialisation is a much bigger problem since the shape of the scene is considered to be unknown. Also there is an increased likelihood that the actual shape varies considerably from the visual hull or approximation of the object(s) as a sphere.

The next novel contribution is the method to update radial basis function models. Traditionally radial basis functions have been used to approximate functions given observations and to fit surfaces to sets of observations. In this work they are also used as an intermediate representation for surface refinement instead of just as a final destination. Previously polygon meshes and volumetric methods (which involve updating a 3-D grid of values) have been used almost exclusively for this purpose. Evolving the RBF surface entails more than just moving the points used to fit the surface, as this will often result in concavities being smoothed over. In addition to moving the RBF centres appropriately, extra centres are created which are constrained to force the implicit surface function to take a positive

value therefor causing the space to become empty.

Two methods are provided to calculate how to adjust the RBF surfaces based on the image data, which is regarded to be the only information available about the scene. The first contains a novel method to match surface textures between perspective views of a surface patch (under the assumption that the surface can be locally approximated as planar). This method reduces the matching problem to a linear search (conducted at difference scales) whilst still taking the full perspective projection into account

The second method utilises classification and segmentation algorithms which have been published previously. However it is novel in that it forms a compliment to the texture matching method and in that it extends the concept of silhouettes into a multi-object domain. Whereas the previous disparity based updating method only works in the presence of strong textures, the segmentation based contour correction works best in the absence of surface textures. The two methods together form a novel scheme that takes full advantage of all available image information.

5.2 Applications

There are many potential applications to this technique of modelling multiple object scenes from images. Digital 3-D objects are used in many areas such as films, computer games, educational/learning tools and interactive mapping. It is not intended that this method would speed up the digitization of objects by processing several objects at once - it would be more accurate to model each in isolation and then re-pose them in 3-D if necessary. Instead the goal is to demonstrate the feasibility of modelling more complex scenes that would take a long time to either model by hand or digitize object by object.

Such scenes could represent the interiors of rooms, collections such as museums or historical sites so that people may visit them in a virtual environment. The ability to handle scenes containing multiple objects hints at possibilities in the field of large scale recovery of urban environments. Interest in this area has increased due to the popularity and widespread use of technologies such as Google streetview. Currently Google streetview contains very little 3-D information (not counting the trajectory of the vehicle used to capture the images), consisting as it does of a series of individual 360 degree panoramas. However future versions will be able to take advantage of multi-object reconstruction from images in order to provide a richer true 3-D experience.

The multiple object modelling methodology presented here also achieves compression by removing the redundancy present in sequences of images of the same scene. If coloured output is not required, the RBF model achieves a very high compression ratio, generalising the data contained in tens of megabytes of image data down to around 6000 integer coordinates in three dimensions (the RBF centres) and 6000 floating point weights, which together take up approximately 40 kilobytes.

If colour/texture information is also required the space used increases substantially as values have to be stored for each point on the surface. The precise amount depends on the resolution at which this is achieved, however. Matching the image resolution would still result in a reduction of storage as the corresponding parts of each image are only stored once. Whilst this is not a practical alternative to standard image compression in most situations as fields such as 3-D television mature, compressing and coding 3-D models efficiently will gain importance.

5.3 Critical Analysis

This section covers some shortcomings of the methods presented in this thesis. Firstly it is assumed that both camera positions and the projection parameters are known in advance. This is fairly common amongst 3-D reconstruction algorithms, and there are reliable methods to obtain this calibration information. However, the precise internal camera parameters may change slightly from their calibrated values when a particular image sequence is captured. The focal length, for example, changes slightly depending on where the lens is focused, even for a non-zoom lens.

All of the methods presented in this thesis rely at some level on the assumption of Lambertian reflectance. That is, that the appearance of a point on a surface does not vary with the viewing angle. Almost all real surfaces deviate from this to some degree. Significant deviations cause serious problems as they result in false matches.

There is room for improvement in the contour extraction methods presented in section 4.3. Supervised methods in general are undesirable in this field as the ultimate goal is for a system which operates without any human interaction (since humans are often a lot better at solving high level vision problems). Also the separation into a distinct segmentation and contour fitting by snakes steps is not strictly necessary.

Radial basis functions have many advantages, principal amongst which is the fact that they guarantee smoothness of the surface in the first three derivatives. However, this very smoothness

makes it very difficult to represent rough surfaces and sharp edges. By scaling down the RBFs more detail can be represented but this requires more centres. There is a practical limit on the number of RBF centres that can be used as the time and space requirements to calculate the weights are $O(n^2)$ in the number of centres.

Finally the space containing the scene must fit within a bounding cuboid. This clearly limits the application to large scale scene recovery (such as urban environments) and breaks the condition that no a-priori knowledge about the scene is required.

5.4 Further Work

In order to address the main shortcomings identified previously the following further work should be carried out. In order to address the problem of calibration errors, recalibration could take place at each step, using the image data from the scene directly. This could be achieved by varying the paramters which describe the camera's interal characteristics, recomputing the shape and examining the error with respect to the images.

Although voxel carving methods are restricted to the Lambertian reflectance - due to the fact that voxels have no orientation - RBF surfaces do, so it would be possible to take non-uniform reflectance into account. This would be achieved by modelling the BDRF, the bi-directional reflectance distribution function. In some cases the problem may be under-constrained or there may be ambiguities. This could cause image regions that do not correspond to the same part of a 3-D surface to appear to match, for some BDRF. However it would be possible to model the BDRF across the surfaces with RBFs as well, in order to take advantage of smoothness to remove ambiguities by assuming the surface reflectance properties are locally similar.

The contour correction process would benefit from a more sophisticated colour/texture segmentation algorithm which could potentially also take non Lambertian reflectance into account, as an intermediate 3-D model of the scene is available at this stage. Segmentation/contour fitting with snakes could be combined into a single step using a snake energy function that takes colour and texture information into account, forgoing the need for separate stages.

There are several problems with RBF surfaces mentioned above which could be addressed with more research. Firstly there is the problem of smoothness, which could be overcome by utilising a mesh based representation in addition to the RBF surface. One of the disadvantages of polygon

meshes compared to RBFs is the need to constantly check for degenerate meshes when altering the surface and the inability to cope with topological changes. However after refinement, the surface model is quite close to the true surface so these disadvantages are no longer significant. Thus it would be possible to convert the RBF surface into a mesh and evolve the mesh using an error function based on the difference between the rendered surface and input images. This final post processing step would allow fine details and sharp edges to be recovered.

Another problem is that the number of RBFs is limited by the need to calculate all weights simultaneously. Attempts to use compactly supported basis functions remove this requirement resulted in poor performance in experiments (see section 3.17). Further research into a radius of support which is great enough to take advantage of smoothing and hole filling, whilst still being finite, would allow more basis functions to be used in order to represent more complicated scenes.

This would fit naturally with a scheme to warp the space that the model occupies in order to allow far off objects to be represented, but in less detail. This is acceptable as further away objects will be smaller in the images and feature less perspective change so their modelling will not be as accurate. This non-uniform space would forgo the need to keep the scene within a bounding cuboid and allow extensive scenes such as large urban areas to be modelled.

List of References

[1] Pedro M. Q. Aguiar and José M. F. Moura. Three-dimensional modeling from two-dimensional video. *IEEE Transactions on Image Processing*, 10(10):1541–1551, 2001.

[2] Nina Amenta, Marshall W. Bern, and David Eppstein. The crust and the beta-skeleton: Combinatorial curve reconstruction. *Graphical Models and Image Processing*, 60(2):125–135, 1998.

[3] S. Araki, T. Matsuoka, N. Yokoya, and H. Takemura. Real-time tracking of multiple moving object contours in a moving camera image sequence. *IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS E SERIES D*, 83(7):1583–1591, 2000.

[4] J. T. Astola, P. Haavisto, and Y. Neuvo. Vector median filters. *Proceedings of IEEE*, 78:678–689, 1990.

[5] S.T. Barnard. Stochastic stereo matching over scale. *International Journal of Computer Vision*, 3(1):17–32, 1989.

[6] S. Bauer, J. Kneip, T. Mlasko, B. Schmale, J. Vollmer, A. Hutter, and M. Berekovic. The mpeg-4 multimedia coding standard: Algorithms, architectures and applications. *J. VLSI Signal Process. Syst.*, 23(1):7–26, 1999.

[7] A. Baumberg. Reliable feature matching across widely separated views. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1. IEEE Computer Society; 1999, 2000.

[8] James F. Blinn. A generalization of algebraic surface drawing. *ACM Trans. Graph*, 1(3):235–256, 1982.

[9] A. Bogomjakov and C. Gotsman. Reduced depth and visual hulls of complex 3D scenes. In *Computer Graphics Forum*, volume 27, pages 175–182. Blackwell Science Ltd, Osney Mead, Oxford, OX 2 0 EL, UK,, 2008.

[10] R.C. Bolles, H.H. Baker, and M.J. Hannah. The JISCT Stereo Evaluation'. In *Image Understanding Workshop: Proceedings of a Workshop Held in Washington, DC April 18-21, 1993*, page 263. Morgan Kaufmann, 1993.

[11] M. Bosch, Fengqing Zhu, and E.J. Delp. Spatial texture models for video compression. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 1, pages I –93–I –96, 16 2007-Oct. 19 2007.

[12] G. Boström, M. Fiocco, D. Puig, A. Rossini, João G. M. Gonçalves, and Vítor Sequeira. Acquisition, modelling and rendering of very large urban environments. In *3DPVT*, pages 191–198. IEEE Computer Society, 2004.

[13] E. Boyer, J.S. Franco, and G.I. Rhône-Alpes. A hybrid approach for computing visual hulls of complex objects. In *Ieee Computer Society Conference On Computer Vision And Pattern Recognition*, volume 1. IEEE Computer Society; 1999, 2003.

[14] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1222–1239, 2001.

[15] Adrian Broadhurst, Tom Drummond, and Roberto Cipolla. A probabilistic framework for space carving. In *ICCV*, pages 388–393, 2001.

[16] J. C. Carr, R. K. Beatson, B. C. McCallum, W. R. Fright, T. J. McLennan, and T. J. Mitchell. Smooth surface reconstruction from noisy range data. In Matt Adcock, Ian Gwilt, and Yong Tsui Lee, editors, *Proceedings of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia 2003, Melbourne, Australia, February 11-14, 2003*, pages 119–126. ACM, 2003.

[17] JC Carr, RK Beatson, JB Cherrie, TJ Mitchell, WR Fright, BC McCallum, and TR Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76. ACM New York, NY, USA, 2001.

[18] Fang Chen and David Suter. Multiple order laplacian splines - including splines with tension. Technical Report MECSE 1996-5, Dept. of Electrical and Computer Systems Engineeringm, Monash University, July 1996.

[19] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. Introduction to algorithms, 1990.

[20] W. B. Culbertson, T. Malzbender, and G. Slabaugh. Generalized voxel coloring. *Lecture Notes in Computer Science*, 1883:100–??, 2000.

[21] F. Dellaert, S.M. Seitz, C. Thorpe, and S. Thrun. Structure from motion without correspondence. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE Computer Society; 1999, 2000.

[22] Sainz Bagherzadeh Department, M. Sainz N. Bagherzadeh, and Key Words. Carving 3d models from uncalibrated views. In *Proceedings of the 5th IASTED International Conference Computer Graphics and Imaging*, pages 144–149. Press, 2002.

[23] Huong Quynh Dinh, Greg Turk, and Gregory G. Slabaugh. Reconstructing surfaces by volumetric regularization using radial basis functions. *IEEE Trans. Pattern Anal. Mach. Intell*, 24(10):1358–1371, 2002.

[24] A. Doi and A. Koide. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE TRANSACTIONS on Information and Systems*, 74(1):214–224, 1991.

[25] Y. Duan, L. Yang, H. Qin, and D. Samaras. Shape reconstruction from 3D and 2D data using PDE-based deformable surfaces. In *European Conference on Computer Vision*, pages Vol III: 238–251, 2004.

[26] M.D. Fairchild. *Color appearance models*. Wiley, 2005.

[27] Olivier D. Faugeras and Renaud Keriven. Variational principles, surface evolution, PDEs, level set methods, and the stereo problem. *IEEE Transactions on Image Processing*, 7(3):336–344, 1998.

[28] DA Forsyth, S. Ioffe, and J. Haddon. Bayesian structure from motion. In *Int. Conf. on Computer Vision (ICCV)*, pages 660–665, 1999.

[29] J. M. Frahm, K. Koser, and R. Koch. Pose estimation for multi-camera systems. In *German Pattern Recognition Symposium*, pages 286–293, 2004.

[30] C. Fruh and A. Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60(1):5–24, October 2004.

[31] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

[32] Y. Furukawa, A. Sethi, J. Ponce, and DJ Kriegman. Robust structure and motion from outlines of smooth curved surfaces. *IEEE transactions on pattern analysis and machine intelligence*, 28(2):302–315, 2006.

[33] Yasutaka Furukawa and Jean Ponce. Carved visual hulls for image-based modeling. In *European Conference on Computer Vision*, 2006.

[34] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.

[35] Michael Jones Federico Girosi and Tomaso Poggio. Priors stabilizers and basis functions: From regularization to radial, tensor and additive splines. Technical Report CBCL-75, MIT Artificial Intelligence Laboratory, June 6 1993.

[36] M.J. Hannah. Computer matching of areas in stereo images. 1974.

[37] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[38] R. I. Hartley. Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 22(2):125–140, March 1997.

[39] R. I. Hartley. Computation of the quadrifocal tensor. In *European Conference on Computer Vision*, page I: 20, 1998.

[40] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[41] Janne Heikkila. Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1066–1077, 2000.

[42] P V C Hough. Methods and means for recognising complex patterns. *U.S. Patent 3 069 654*, 1962. HOUGH62.

[43] Yu-Wen Huang, Ching-Yeh Chen, Chen-Han Tsai, Chun-Fu Shen, and Liang-Gee Chen. Survey on block matching motion estimation algorithms and architectures with new results. *J. VLSI Signal Process. Syst.*, 42(3):297–320, 2006.

[44] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*, 15(9):850–863, 1993.

[45] M. Irani, B. Rousso, and S. Peleg. Detecting and tracking multiple moving objects using temporal integration. In *European Conference on Computer Vision*, pages 282–287. Springer, 1992.

[46] J. Isidoro and S. Sclaroff. Stochastic refinement of the visual hull to satisfy photometric and silhouette consistency constraints. In *International Conference on Computer Vision*, pages 1335–1342, 2003.

[47] John Isidoro and Stan Sclaroff. Contour generator points for threshold selection and a novel photo-consistency measure for space carving. Technical report, December 06 2003.

[48] J.J. Koenderink. What does the occluding contour tell us about solid shape. *Perception*, 13(3):321–30, 1984.

[49] K. Kolev and D. Cremers. Integration of Multiview Stereo and Silhouettes Via Convex Functionals on Convex Domains. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, pages 752–765. Springer, 2008.

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

[51] A. Ladikos, S. Benhimane, and N. Navab. Multi-View Reconstruction using Narrow-Band Graph-Cuts and Surface Normal Optimization.

[52] A. Lagae and P. Dutre. A comparison of methods for generating Poisson disk distributions. *status: accepted*.

[53] Aldo Laurentini. How far 3D shapes can be understood from 2D silhouettes. *IEEE Trans. Pattern Anal. Mach. Intell*, 17(2):188–195, 1995.

[54] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using affine-invariant regions. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings*, volume 2, 2003.

[55] Carlos Leung, Ben Appleton, and Changming Sun. Embedded voxel colouring. January 01 2003.

[56] Z. Lukas, B. Joachim, K. Konrad, and B. Horst. Fusion of feature-and area-based information for urban buildings modeling from aerial imagery. In *Proceedings of the European Conference on Computer Vision*, 2008.

[57] Q. T. Luong and O. D. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–75, January 1996.

[58] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman, New York, 1982.

[59] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *Journal of the American Statistical Association*, pages 76–89, 1987.

[60] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: a survey. *Medical image analysis*, 1(2):91–108, 1996.

[61] Bryan S. Morse, Terry S. Yoo, David T. Chen, Penny Rheingans, and Kalpathi R. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Shape Modeling International*, pages 89–98, 2001.

[62] Shohei Nobuhara and Takashi Matsuyama. Deformable mesh model for complex multi-object 3d motion estimation from multi-viewpoint video. *3D Data Processing Visualization and Transmission, International Symposium on*, 0:264–271, 2006.

[63] S. Osher and J. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[64] J.C. Platt. Fast training of support vector machines using sequential minimal optimization. 1999.

[65] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. *Lecture Notes in Computer Science*, 800:97–110, 1994.

[66] Marc Pollefeys, Reinhard Koch, Maarten Vergauwen, and Luc J. Van Gool. Metric 3d surface reconstruction from uncalibrated image sequences. In *SMILE*, pages 139–154, 1998.

[67] S. Prakash and A. Robles-Kelly. A semi-supervised approach to space carving. *Pattern Recognition*, 2009.

[68] L. Quan and T. Kanade. A factorization method for shape and motion from line correspondences. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages 803–808, 1996.

[69] Eraldo Ribeiro, Antonio Robles-Kelly, and Edwin R. Hancock. Detecting multiple texture planes using local spectral distortion. *Image Vision Comput*, 20(9-10):739–750, 2002.

[70] S. Savarese, H. Rushmeier, F. Bernardini, and P. Perona. Shadow carving. In *Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001. Proceedings*, volume 1, 2001.

[71] Frederik Schaffalitzky and Andrew Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *ICCV*, pages 636–643, 2001.

[72] D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. *International Journal of Computer Vision*, 28(2):155–174, 1998.

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

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

[75] Steven M. Seitz and Charles R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *CVPR*, page 1067. IEEE Computer Society, 1997.

[76] Heung-Yeung Shum, Richard Szeliski, Simon Baker, Mei Han, and P. Anandan. Interactive 3d modeling from multiple images using scene regularities. In *SMILE*, pages 236–252, 1998.

[77] MK Singh and N. Ahuja. Mean-shift segmentation with wavelet-based bandwidth selection. In *Sixth IEEE Workshop on Applications of Computer Vision, 2002.(WACV 2002). Proceedings*, pages 43–47, 2002.

[78] G. G. Slabaugh, T. Malzbender, and W. B. Culbertson. Volumetric warping for voxel coloring on an infinite domain. In *3D Structure from Multiple Images of Large-Scale Environments*, page 109 ff., 2000.

[79] C. J. Taylor, D. J. Kriegman, and P. Anandan. Structure and motion in two dimensions from multiple images: A least squares approach. In *Workshop on Visual Motion*, pages 242–248, 1991.

[80] C. Tomasi and T. Kanade. Shape and motion without depth. In *Image Understanding Workshop*, pages 258–270, 1990.

[81] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams: a factorization method: Full report on the orthographic case. Technical Report TR92-1270, Cornell University, Computer Science Department, march 1992.

[82] Greg Turk and James F. O'Brien. Modelling with implicit surfaces that interpolate. *ACM Trans. Graph*, 21(4):855–873, 2002.

[83] George Vogiatzis, Philip H. S. Torr, and Roberto Cipolla. Multi-view stereo via volumetric graph-cuts. In *CVPR*, pages 391–398. IEEE Computer Society, 2005.

[84] G. Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, 1990.

[85] Holger Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Adv. Comput. Math.*, 4(4):389–396, 1995.

[86] Philip L. Worthington and Edwin R. Hancock. New constraints on data-closeness and needle map consistency for shape-from-shading. *IEEE Trans. Pattern Anal. Mach. Intell*, 21(12):1250–1267, 1999.

[87] Philip L. Worthington and Edwin R. Hancock. Object recognition using shape-from-shading. *IEEE Trans. Pattern Anal. Mach. Intell*, 23(5):535–542, 2001.

[88] C. Xu and J. Prince. Gradient vector flow: A new external force for snakes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 66–71, 1997.

[89] A. Yezzi Jr, S. Kichenassamy, A. Kumar, P. Olver, and A. Tannenbaum. A geometric snake model for segmentation of medical imagery. *IEEE Transactions on Medical Imaging*, 16(2):199–209, 1997.

[90] Tianli Yu, Ning Xu, and Narendra Ahuja. Shape and view independent reflectance map from multiple views. In Tomás Pajdla and Jiri Matas, editors, *Computer Vision - ECCV 2004, 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV*, volume 3024 of *Lecture Notes in Computer Science*, pages 602–616. Springer, 2004.

[91] S.Y.K. Yuen. *Shape from contour using symmetries*. Springer, 1989.

[92] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. *Lecture Notes in Computer Science*, 801:151–158, 1994.