3D Object Reconstruction using Multiple Views

Donghoon Kim

Department of Computer Science and Statistics

University of Dublin, Trinity College

A thesis submitted to the University of Dublin, Trinity College in fulfillment of the requirements for the degree of

Doctor of Philosophy

September 2011

Declaration

Donghoon Kim

Dated: October 2011

Abstract

3D object modelling from multiple view images has recently been of increasing interest in computer vision. Two techniques, *Visual Hull* and *Photo Hull*, have been extensively studied in the hope of developing 3D shape from multiple views. These early methods have the advantage that they do not require pre-processing procedures such as feature selection and matching, which fail when images are of low resolution. One drawback of these two methods is their discrete formulation, which is demanding of memory and limits the type of optimisation methods that can be used. This study proposes a continuous formulation in contrast to the discrete formulations typical of these earlier methods, and aims to robustly reconstruct the 3D shape and colour of an object seen in a multi-view system. The use of a continuous formulation based on kernel density estimates enables us to define a gradient ascent algorithm (e.g. a mean shift algorithm) to recover the 3D shape and colour. Moreover, we propose to include prior information in this continuous modelling to improve the quality of the reconstruction. The proposed approach has several advantages: it is less memory demanding, the resulting algorithm is suitable for parallel processing, and it recovers concavities that are usually lost when estimating shape from silhouettes with the standard visual hull method.

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr. Rozenn Dahyot. I am hugely grateful for her wonderful guidance and encouragement over the past three years. Without her guidance this thesis would not be possible.

I also thank to Prof. Juneho Yi, formerly my Masters' degree supervisor. I learnt computer vision basics from him which helped me to successfully finalise this thesis.

Thanks to all in the GV2 group for being always kind and helpful. Especially I am grateful to Jonathan Ruttle for our collaboration.

This research has been supported by the "Irish Research Council for Science, Engineering and Technology in collaboration with INTEL Ireland Ltd: funded by the National Development Plan". The scholarship made possible my studies in Trinity College Dublin.

To my parents I wish to express my thanks for their support, love and sacrifices.

Lastly, I would like to give heartfelt thanks to my wife Hyungmin. She has lost a lot due to my research abroad. Without her encouragement, sacrifices and understanding it would have been impossible for me to finish this work.

Contents

List of Tables

List of Figures

Nomenclature

Acronyms

CAMShift Continuously Adaptive Mean Shift

DoG Difference of Gaussians

GPU Graphics Processing Unit

KDE Kernel Density Estimate

KNN K Nearest Neighbours

MVS Multi-view stereo

PCA Principal Component Analysis

SIFT Scale Invariant Feature Transform

SSD the Sum of Squared Differences

SURF Speeded Up Robust Feature

SVMs Support Vector Machines

Chapter 1

Introduction

Modelling an object from multiple views has turned out to be an essential requirement in areas as diverse as 3D photography, automatic modelling or virtual reality applications. Moreover, in recognition of the limitations of 2D information from images, there has been an increasing interest in 3D object modelling in the object recognition domain, especially for face recognition, which has helped to address difficulties caused by illumination and pose variations.

This thesis aims to automatically estimate a 3D model of an object of interest as accurately as possible using a multi-view system. *Visual Hull* has been widely used to compute a 3D shape of an object of interest from silhouettes of multiple views, and this concept has been further extended to *Photo Hull* to encompass colour information rather than binary silhouettes. Also, the Radon transform has provided a mathematical basis from which to estimate 3D shapes from 2D views in medical image processing, and in tomographic projection in particular.

Both *Visual Hull* and *Photo Hull* ideas have been proposed more than two decades ago. However these methods are still very much used today due to their simplicity because they can give an initial estimate of the 3D shape of the object in view for more advanced 3D reconstruction algorithms.

Both modellings (*Visual Hull* and *Photo Hull*) are formulated in order to optimise a discrete cost function. The modellings with discrete cost functions are usually memory demanding and can only be optimised by using extensive search algorithms. Moreover, *Visual Hull* provides only a convex approximation of the shape of an object. This results in a loss of the object's concavities beyond recovery. This thesis investigates new formulations for *Visual Hull* and *Photo Hull* that can address these limitations and improve the quality of object modelling.

1.1 Ideal Scenario

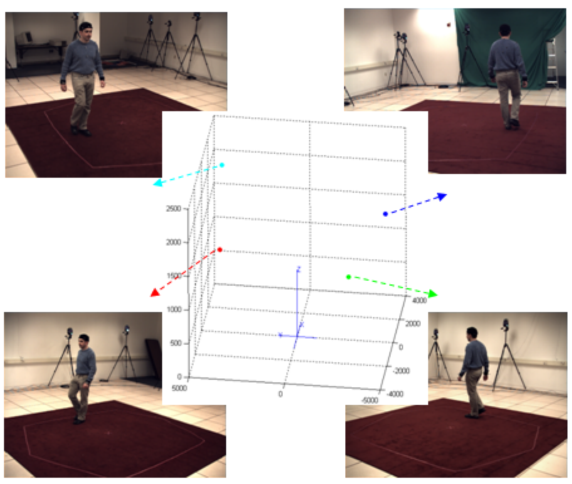


Figure 1.1: Multi-view system (71).

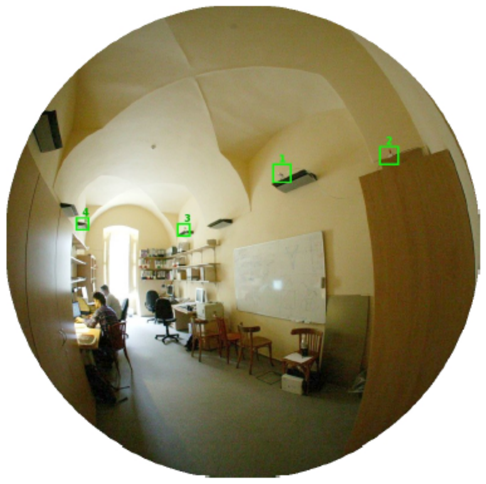


Figure 1.2: Multi-view system (88).

This thesis considers a multi-view system consisting of several synchronised static cameras over looking the same scene as shown in Figure 1.1 and 1.2. An object of interest, for instance the head of an individual, is visible from several camera views at different resolutions. We want to merge the information available from all cameras to recover the 3D shape of the object. This problem poses several challenges, including limited visibility (occlusions and different scales), pose variations, uncontrolled illumination, etc.

Zimmermann et al. (88) propose to track and to update sequentially 3D head shapes in a multi-view setting. They assume that some part of the head is visible to at least two cameras. Partial 3D reconstruction is then iteratively performed using stereo vision matching techniques. As the object moves in the scene, more views of the head can be merged in the 3D reconstruction model refining both the reconstruction and the tracking process. It should be noted, however, that the success of this approach depends on the robustness of the stereo matching process and the tracking accuracy.

In the same vein as Zimmermann et al. (88), the ultimate idea explored in this thesis is the possibility of merging sequential and/or simultaneous images as recorded by a multi-view system, in order to estimate a more accurate 3D shape of an object of interest. This system, if applied to the specific case of modelling facial features, has the potential of helping to identify people. For the sake of simplicity, we assume that the object to be reconstructed is rigid.

In general, to ensure the robustness of the stereo matching it is necessary to robustly extract feature points to match. The robustness of local feature detection was investigated at various scales (Appendix F). This preliminary work has shown that the local feature-based approach is not sufficiently robust to deal with objects of low resolution, as local feature detection and matching are not reliable. This suggests that it is difficult to estimate the 3D shape of an object based on matching several images with local features at the low resolution.

It is worth noting that this approach (i.e. stereo matching with local feature points) to recover depth information for 3D shape inference is a popular one, and leads to high quality reconstructions when the object image has a high resolution, and the cameras are close to each other such that stereo matching is effective. Earlier methods for 3D reconstruction, *Visual Hull* in particular, do not attempt stereo matching and therefore are not sensitive to its limitations.

1.2 Research Questions and Contributions

1. **How can multiple views information be merged without selecting local features to estimate the 3D shape of an object in view?**

   Because *Visual Hull* does not require feature selection and matching, this method is tested in Chapter 3. However, binary silhouette images are required as inputs. There are two major drawbacks of this approach:

   - the modelling is discrete, and
   - its computation requires a lot of memory.

   Therefore, we propose an alternative continuous formulation for *Visual Hull* and *Photo Hull* to recover the 3D shape and colour of an object seen in a multi-view system (Chapter 4). We can define a gradient ascent algorithm to estimate the 3D shape of an object from the formulation. The proposed method has two major advantages:

- less memory is required, and
- the method is suitable for parallel processing.

2. **How can prior information be integrated to estimate a more accurate 3D shape from multi-view images?**

   In Chapter 5, we investigate how to include prior information in continuous modelling so that a 3D shape can be more accurately estimated. We propose a prior that is invariant to scale variations, and show how to deal with rotations. The method we have used to approximate the prior optimally is based on the K-nearest neighbour algorithm. Incorporating this prior in our continuous modelling allows better 3D reconstruction. Particularly concave regions, which convex approximation methods such as the *Visual Hull* can not restore, can be reconstructed by using the proposed continuous modelling. Even if a small number of cameras is used, this proposed method of modelling will be able to reconstruct a 3D shape consistently.

1.3 Hypotheses in this Thesis

1. Silhouettes are assumed to be available: The silhouettes are known for 3D shape estimation based on silhouettes in Chapters 3, 4 and 5.

2. We have assumed an orthographic camera projection model in Chapter 4 and 5. This simplifying assumption has facilitated the statistical modelling of the likelihood that links the data (pixels) with the latent variable to be estimated (i.e. the 3D reconstruction). Note that recently, (64) have successfully extended this modelling to standard pin-hole cameras.

3. We have assumed that the object to reconstruct is rigid (Chapters 3, 4 and 5). This hypothesis is always true for any object when merging images recorded at the same time but not when sequential images are merged. For

instance, a face recorded from multi-view images at the same time could be regarded as a rigid object. However, the same face, recorded at different times, would probably appear with different facial expressions. In this case, the face is a non-rigid object and this is beyond the scope of this thesis.

4. We have assumed a basic knowledge of motion information in Chapter 3, 4 and 5. 3D motion estimation is not the focus of this research.

5. The cameras record synchronised images.

1.4 List of Publications

1. Donghoon Kim and Rozenn Dahyot, **Face Components Detection using SURF Descriptor and SVMs**, In International Machine Vision and Image Processing Conference, 2008.

2. Donghoon Kim and Rozenn Dahyot, **3D Head Reconstruction using Multi-camera Stream**, In Irish Machine Vision and Image Processing Conference, 2009.

3. Donghoon Kim, Jonathan Ruttle and Rozenn Dahyot, **3D Shape Estimation from Silhouettes using Mean-shift**, In IEEE International Conference on Acoustics, Speech and Signal Processing, 2010.

1.5 Summary of Chapters

This thesis is structured as follows:

- Chapter 2 provides a state-of-the-art literature review related to our research topic including 3D surface reconstruction and kernel density estimates. The 3D surface reconstruction in Section 2.1 covers not only shape from silhouettes and photo-consistency methods but also inverse Radon transform based methods. The kernel density estimate in Section 2.2 describes the concept of kernel density estimates with optimisation methods such as the mean shift algorithm and Newton's method.

- Chapter 3 explains how to estimate the 3D shape using voxels and refine it over time using motion information in multi-view video sequences.
- Chapter 4 gives the details of our new smooth kernel density estimates to reconstruct 3D shapes and illustrates their performance experimentally.
- Chapter 5 outlines the methodologies to model new posteriors computed with the kernel density estimates and a prior for the shape. Also 2D and 3D experimental results are provided to show the performance of the proposed framework.
- Chapter 6 concludes this research with a discussion of future research directions.

Chapter 2

Background and Related Work

The primary goal of this thesis is to automatically infer a 3D reconstruction (i.e. by volume or surface) of an object of interest when it is seen simultaneously by many cameras from multiple views. The state-of-the-art of the domain is presented in Section 2.1.

The proposed new framework, which is presented in Chapter 4, relies on a statistical modelling method that uses kernel density estimates. These are reviewed in Section 2.2 along with optimisation methods such as the mean shift algorithm.

2.1 3D Surface Reconstruction

A silhouette image is a binary image where the object of interest (foreground) is represented by pixels equalling one, whereas all other pixels are set to zero (background). Silhouette images can be computed using segmentation algorithms which can be made accurate in a controlled environment (e.g. using a blue-screen as the background). *Shape from silhouettes* is a popular technique used to estimate the 3D volume or surface of the object in view (21; 25). These are reviewed in Section 2.1.1.

Silhouette images can be difficult to compute in an uncontrolled environment. As an alternative, colour images that have additional photometric information, can also be used to infer the 3D shape of the object. *Shape from photo-consistency* methods based on this photometric information are described in Section 2.1.2.

Section 2.1.3 reviews methods using stereo matching to recover depth information from a pair of cameras and consequently merge these depth maps to infer the 3D surface of the object of interest.

2.1.1 Shape from Silhouettes

Shape from silhouettes methods are popular in computer vision because of their simplicity and their computational efficiency. The first known *Shape from silhouettes* method was proposed by Baumgart (2). Figure 2.1 illustrates an example of the reconstruction based on the intersection of three silhouettes recorded from different viewpoints (only the reconstruction of a slice of the object is illustrated). Each silhouette of the object creates a cone in the 3D world (i.e. locations inside each cone project onto the foreground of the silhouette image using the camera calibration parameters). The intersections of every cone from all camera views give an approximation of the 3D object volume.

The reconstructed volume approximates the real 3D shape of the object. The quality of this reconstruction depends on the number of camera views, their viewpoints and the complexity of the object. In particular, concave regions can not be observed in any silhouette; thus silhouette-based reconstructions are unsuitable for the reconstruction of object's concave regions. Laurentini (39) has defined *Visual Hull* as the best reconstruction that can be computed using an infinite number of silhouettes captured from all viewpoints outside the convex hull of the object.

The inverse Radon transform is another well-known technique in medical imaging for allowing the reconstruction of 3D volume from several projections. Section 2.1.1.3 shows how this method can be used to estimate the shape of a 3D object using silhouettes.

2.1.1.1 Voxel-based approaches

Volume-based approaches focus on the volume of the *Visual Hull* (39; 48; 59; 75). This volume is represented by voxels (i.e. volume element). Figure 2.2 illustrates the underlying concept of the volume-based approach from multiple cameras: the yellow grid is a horizontal slice of the voxels.

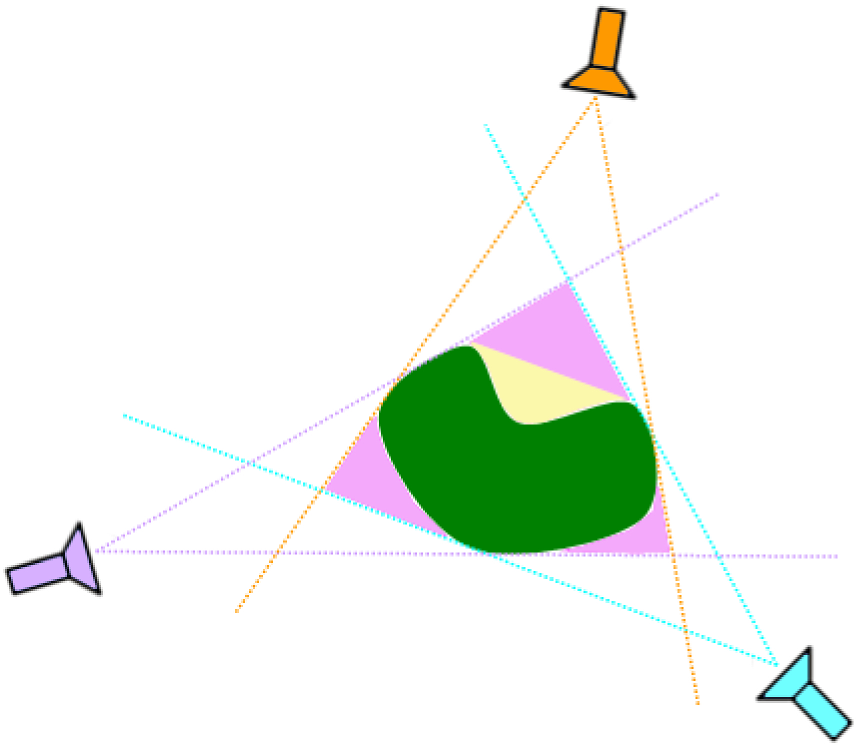


Figure 2.1: An example of the volume intersection from three views. The object (green) is approximated by the area intersected by the three cones.

The world volume can be considered to be composed of voxels each of which contributes to a pixel on the image planes of the cameras. The size of the cube defines a spatial enclosure of the voxel. The calibration parameters of each camera are needed to compute back-projection functions which allow the positions in 3D space to be mapped to the image planes. An early approach which used this representation was proposed by Martin and Aggarwal (48) using parallelepipedic voxels.

Voxel occupancy is the simplest voxel-based approach to estimate 3D shapes. This is generally a binary decision (e.g. voxel is occupied or unoccupied), though some methods include a real value of opacity. The set of occupied voxels represents the 3D volume.

A classic algorithm known as voxel carving computes this intersection by projecting each voxel to all viewpoints and discarding all voxels which fall outside any of the silhouettes. Szeliski (77) introduced the octree representation, which

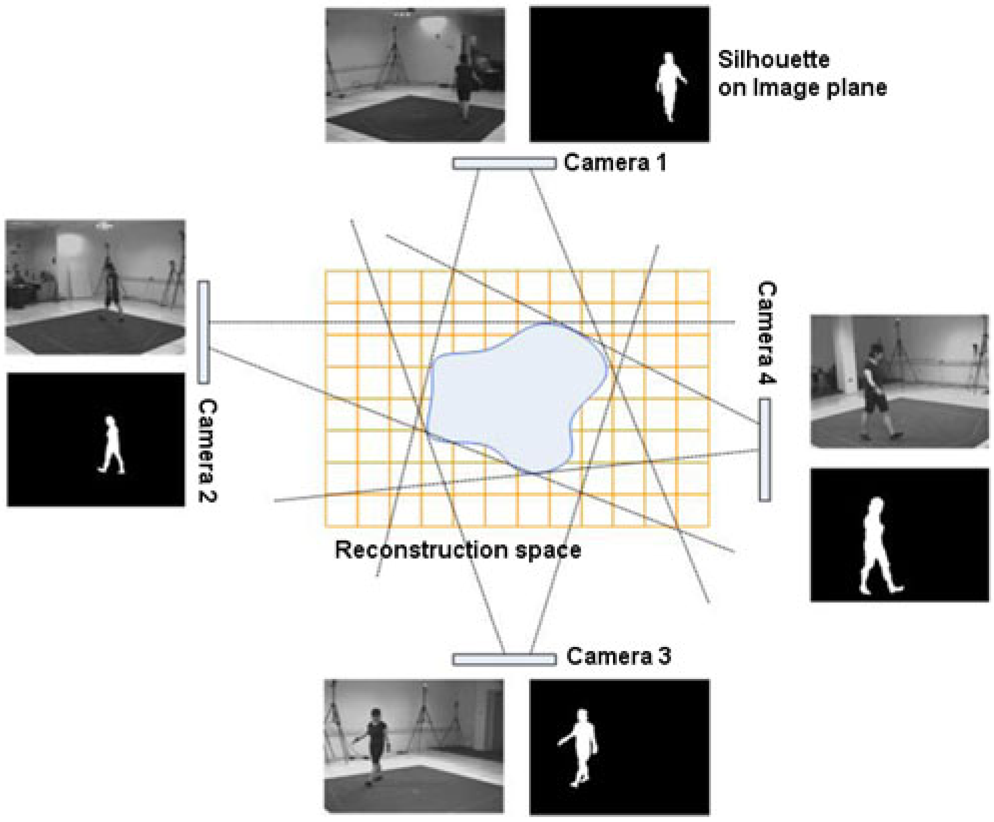


Figure 2.2: Shape from silhouettes using four views (Grey images are from HumanEva-II dataset in (71)).

is a tree of recursively subdivided voxels (Figure 2.3). Using the octree representation, the occupied voxels can be computed efficiently.

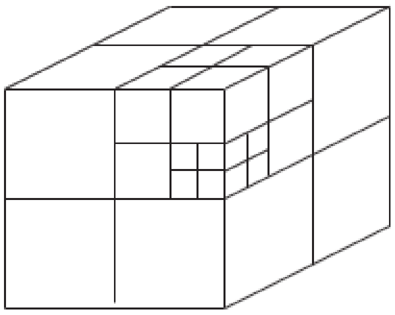


Figure 2.3: Octree representation (77).

In these voxel-based approaches, it is critical to robustly and accurately extract the silhouettes. Errors in the silhouette images lead to artifacts in the estimated visual hull. To relax the limitation of binary silhouettes, several methods have been proposed (25; 73). For instance, Snow et al. (73) substitute binary silhouettes with the difference between the intensity image recorded by the camera and the intensity image recorded by the same camera without the object (i.e. background image). This can be manifested as *fuzzy silhouettes* where high values indicate pixels more likely to be on the object whereas low values indicate that pixels are probably on the background. They formulate the voxel occupancy problem as an optimisation problem where the global minimum of an energy function is computed. The energy function contains a data term and a smoothness term. The data term specifies the likelihood of the voxel occupancy based on the fuzzy silhouettes. The smoothness term specifies the degree of smoothness of the labels in a neighbourhood of voxels. Their formulation can be regarded as a generalisation of silhouette intersections with two advantages: the silhouette is no longer a binary map, and a global spatial smoothness can be incorporated naturally.

A probabilistic representation of the problem has been proposed by Franco and Boyer (25). Instead of the binary decision of the voxel occupancy, the volume is represented by a grid of voxel occupancy probabilities.

In general, volume-based approaches are limited by the heavy computation and memory requirements. In addition, post-processing is required if a mesh representation of the surface is needed as opposed to a volume representation wih voxels. The mesh representation can be constructed using the Marching Cubes algorithm from the reconstructed voxels (45).

2.1.1.2 Surface-based approaches

Surface-based approaches aim to estimate a surface representation of the *Visual Hull* rather than a volume representation. The 3D surface of an object is reconstructed by analyzing the geometric relationship between the silhouette boundaries and the *Visual Hull* surface. These methods directly estimate a mesh representation of the 3D surface by computing the intersection of the cone surfaces associated with the silhouette edges (often based on the epipolar constraint) illustrated in Figure 2.4.

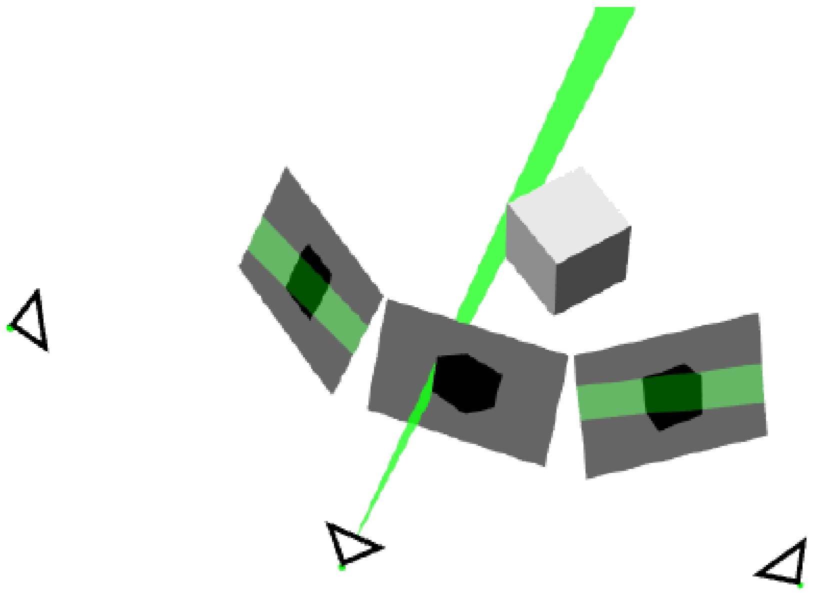


Figure 2.4: The face of a single silhouette cone is illustrated by the colour green (53).

An early attempt at direct polygonal intersection was introduced by Baumgart (2). Sullivan and Ponce (76) proposed the *polygonal Visual Hull* to initialize a triangular spline surface which is then subject to further refinement. However, the direct intersection of generalized viewing cones is neither efficient nor numerically stable.

Lazebnik et al. introduced a topological description of the contour generator for weakly calibrated cameras (40). This description has facilitated the computation of the *Visual Hull polyhedron*. Matusik et al. proposed an efficient algorithm for computing the *polyhedral Visual Hull* directly from silhouettes (53). Their algorithm is capable of generating mesh in real time in the case of a few cameras. However, it may still suffer from numerical instability if more cameras are introduced.

Boyer and Franco proposed a hybrid approach (7). Their algorithm first computes a set of surface points by intersecting the viewing lines in a surface-based fashion. Then, it subdivides the space into tetrahedrons by applying Delaunay tetrahedrization to those surface points. Finally, the visual hull is reconstructed by carving tetrahedrons volumetrically. The hybrid algorithm was designed to avoid artifacts of discretisation by using irregular volumetric cells. In a separate attempt, Cheung et al. (12) adopted colour information not only to locate exactly the points tangential to the surface along each viewing edge but also to estimate rigid motion in video sequences. More recently, Franco and Boyer (25) proposed an algorithm to compute polyhedral visual hulls based on the observation of the incidence relationships between primitives on polyhedron.

Surface-based approaches, which approximate the visual hull with polyhedrons, outperform the volumetric approaches in terms of accuracy and computational complexity. However, they lack robustness, because the calculation of intersections in the 3D space is sensitive to numerical instabilities, especially for complex objects in a low resolution environment. In addition, the mesh model they produce is usually composed of irregular triangles.

2.1.1.3 Inverse Radon Transform

Tomographic reconstruction is an important and active research topic in the field of medical image processing, for example Computed Tomography (CT) and Magnetic Resonance Imaging (MRI). The context of tomographic reconstruction is similar to the volume-based approach in the sense that both are back-projection techniques. However, a 3D medical image is represented in three dimensions as a stack of two-dimensional images reconstructed from tomographic projections. Each slice in the stack is calculated by the Inverse Radon transform.

The Radon transform was first introduced by Johann Radon in 1917 and referred to as the x-ray transform or the projection transform. He showed how to describe a function in terms of its (integral) projection. Radon transform is the mapping of a function on to its projection. The inverse Radon transform is the reconstruction of the function from the projections. The projection can be expressed as a simple line integral or ray sum of the activity distribution along a line which passes through the object. For a 2D distribution $f(x, y)$ on a single cross-sectional plane as shown in Figure 2.5, the line integral at an arbitrary angle

θ is expressed as

$$g_\theta(t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\ \delta(x\cos\theta + y\sin\theta - t)\ dx\ dy, \tag{2.1}$$

where $g_\theta(t)$ is the ray sum along the line which passes through the object and reaches detector t at the angle θ. The function $g_\theta(t)$ is referred to as the Radon

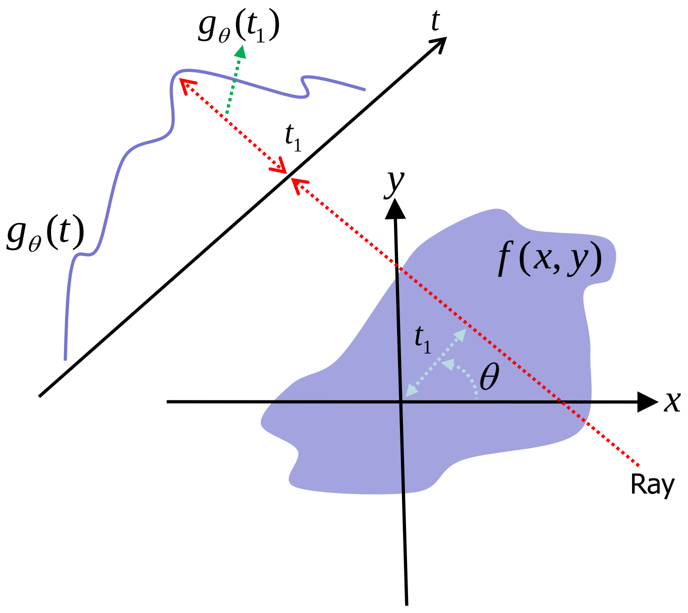


Figure 2.5: The Radon transform $g_\theta(t)$ of a distribution $f(x,y)$.

transform of the function $f(x,y)$.

The Fourier slice theorem relates the 1D Fourier transform of projection data $g_\theta(t)$ to the 2D Fourier transform of the object $f(x,y)$ evaluated along a radial line in Fourier space. Let $F(u,v)$ be the Fourier transform of $f(x,y)$:

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\ \exp^{-\jmath 2\pi(ux+vy)}\ dx\ dy \tag{2.2}$$

And let $G_\theta(\omega)$ be the 1D Fourier transform of $g_\theta(t)$

$$G_\theta(\omega) = \int_{-\infty}^{\infty} g_\theta(t)\ \exp^{-\jmath 2\pi\omega t}\ dt \tag{2.3}$$

By substituting equation 2.1 into equation 2.3, $G_\theta(\omega)$ is given by:

$$\begin{aligned}
G_\theta(\omega) &= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\ \delta(x\cos\theta + y\sin\theta - t)\ dx\ dy \right) \exp^{-\jmath 2\pi\omega t}\ dt \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y) \left(\int_{-\infty}^{\infty} \delta(x\cos\theta + y\sin\theta - t)\ \exp^{-\jmath 2\pi\omega t}\ dt \right)\ dx\ dy \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\ \exp^{-\jmath 2\pi\omega(x\cos\theta + y\sin\theta)}\ dx\ dy \\
&= F(u,v) \text{ for } u = \omega\cos\theta,\ v = \omega\sin\theta \\
&\equiv F(\omega,\theta)
\end{aligned} \tag{2.4}$$

This result is illustrated in Figure 2.6. If $F(\omega,\theta)$ denotes the values of $F(u,v)$ evaluated along a line at the angle θ with the u axis in the frequency domain, then it is equal to the Fourier transform $G_\theta(t)$ of the projection $g_\theta(t)$.

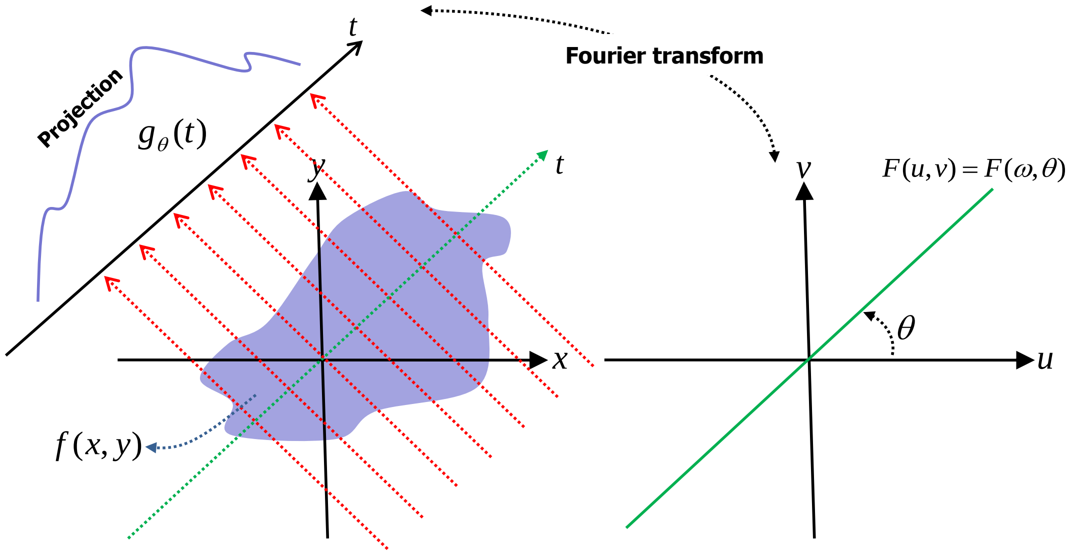


Figure 2.6: The Fourier slice theorem: The Fourier transform of the projection $g_\theta(t)$ at angle θ gives the values of $F(u,v)$ evaluated along a line at an angle θ in the uv plane.

Theoretically, the object can be fully reconstructed by a simple inverse 2D Fourier transform of its 2D Fourier transform if the 1D transform of projection data is given at enough radial and angular samples. This implies that:

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u,v)\ \exp^{\jmath 2\pi(ux+vy)}\ du\ dv \tag{2.5}$$

By converting the Cartesian coordinates (u, v) into polar coordinate (ω, θ) in the frequency domain, it follows that:

$$\begin{aligned} f(x,y) &= \int_0^{2\pi}\int_0^{\infty} F(\omega,\theta)\ \exp^{\jmath 2\pi\omega(x\cos\theta+y\sin\theta)}\ \omega\ d\omega\ d\theta \\ &= \int_0^{\pi}\int_0^{\infty} F(\omega,\theta)\ \exp^{\jmath 2\pi\omega(x\cos\theta+y\sin\theta)}\ \omega\ d\omega\ d\theta \\ &\quad + \int_0^{\pi}\int_0^{\infty} F(\omega,\theta+\pi)\ \exp^{\jmath 2\pi\omega(x\cos(\theta+\pi)+y\sin(\theta+\pi))}\ \omega\ d\omega\ d\theta \\ &= \int_0^{\pi}\int_{-\infty}^{\infty} F(\omega,\theta)\ |\omega|\ \exp^{\jmath 2\pi\omega(x\cos\theta+y\sin\theta)}\ d\omega\ d\theta \\ &= \int_0^{\pi}\left(\int_{-\infty}^{\infty} G_\theta(\omega)\ |\omega|\ \exp^{\jmath 2\pi\omega t}\ d\omega\right)\ d\theta \end{aligned} \tag{2.6}$$

Equation 2.6 can be re-expressed as:

$$f(x,y) = \int_0^{\pi} Q_\theta(x\cos\theta + y\sin\theta)\ d\theta, \tag{2.7}$$

where

$$Q_\theta(t) = \int_{-\infty}^{\infty} G_\theta(\omega)\ |\omega|\ \exp^{\jmath 2\pi\omega t}\ d\omega. \tag{2.8}$$

In equation 2.8, $Q_\theta(t)$ is the 1D inverse Fourier transform of $G_\theta(\omega)|\omega|$ which is the 1D Fourier transform of the projection $g_\theta(t)$ filtered by ramp filter $|\omega|$. Therefore, $Q_\theta(t)$ is the filtered projection data. Equation 2.7 is a *back-projection* process which projects the filtered projection back to the object domain. Back-projection can be viewed as *smearing back* the projection data along the ray from which it came. By using the filtered back-projection (FBP) algorithm, one can reconstruct the object from the projection data.

An explicit and computationally efficient inversion algorithm exists for 2D Radon transforms called FBP (47; 61). Tomographic reconstruction reconstructs 3D volume from density data, and silhouette images are a very crude approximation of it. Consequently, similarly to *Visual Hull*, using the inverse Radon transform on silhouette images of an object taken from different perspectives, enables the reconstructions of an approximation of the 3D shape. This has been proposed by Pintavirooj and Sangworasil for inferring 3D shapes from silhouettes (58) and it is used for comparison with the new approach in Chapter 4.

2.1.2 Shape from Photo-Consistency

Instead of using binary silhouettes, photo-consistency approaches consider photometric information which compute sets of photo-consistent voxels called *Photohull*. This method assumes a Lambertian surface and constant illumination, which means a valid point on the scene surface appears with the same colour over all visible images.

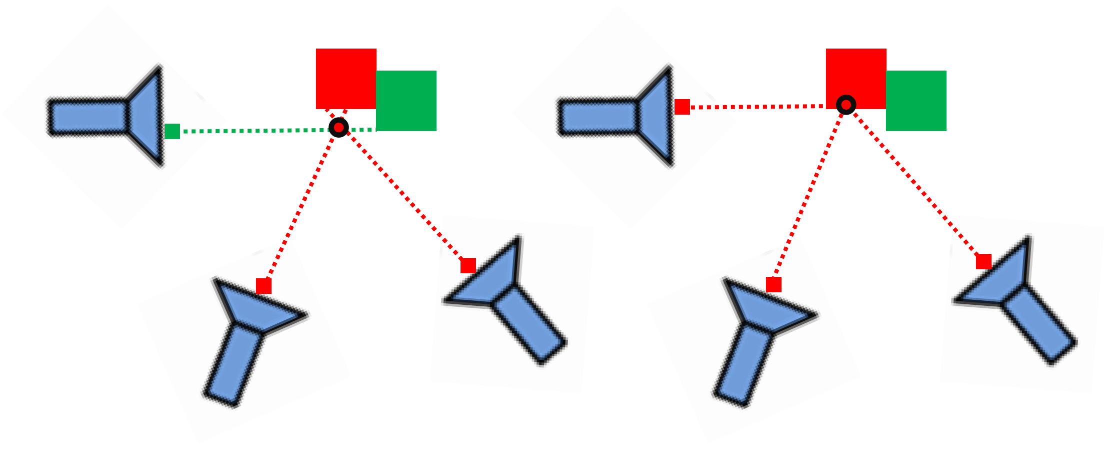


Figure 2.7: Left: A candidate surface point visible but not photo-consistent in the left view (Green colour is back-projected to the image plane of the left view). Right: A candidate surface point visible and photo-consistent in all three views (Red colour is back-projected to all image planes).

Figure 2.7 illustrates the photo-consistency of a 3D surface point in a multi-view system. If the 3D candidate point is not photo-consistent with one of the camera views, it is discarded. Otherwise (if the 3D point is photo-consistent with all views), the point is regarded as a surface point. This filtering method is also known as *space carving*.

Photo hull uses colour information of the images as constraints and builds a volumetric model that is photo-consistent with all the input images (38; 68). The process starts from an initialised 3D volume that contains the scene as an unknown sub-volume. The volume is generally a large cube and is divided into a set of small voxels. The photo-consistency-based algorithm iteratively computes the photo-consistency of each of the surface voxels. Each voxel is projected onto images which are visible to it and these projections are compared. If these projections are consistent in colour, this voxel is classified as a true surface voxel and kept in the *Photo hull*. The algorithm stops when all surface voxels are consistent with the input images. Slabaugh et al.(72) provides a detailed survey on photo-consistency-based reconstruction up to 2001. More recently Anwar and Ferrie (1) have proposed to improve this approach by taking into account camera calibration errors and partial emptiness of the surface voxels.

Due to the richer information available, *Photo hull* constitutes a tighter estimate of the actual shape than the *Visual Hull*, allowing the appearance of concavities on the reconstructed shape. However, it is impractical for colour information to be consistent across different cameras due to varying positions of lighting sources as well as different characteristics of sensing images such as white balance, exposure time, auto-focus, etc. Therefore, for some voxels, colour consistency can be ineffective as a means of surface verification.

2.1.3 Multi-view stereo

Multi-view stereo (MVS) methods have received lots of attention recently, producing a variety of reconstruction algorithms. Multi-view stereo methods calculate correspondences across images to recover depth maps, then the depth maps are merged to reconstruct the 3D structure. A general strategy is to divide the reconstruction process into two stages. The first stage consists of the estimation of a series of depth maps using stereo pairs of the input images. Then the second stage features the combination of the depth information into a global surface estimation, making use of registration and regularisation techniques.

Seitz et al. (67) investigated the state-of-the-art in MVS up to 2006 as well as creating a website (66) which evaluates various methods in terms of accuracy and completeness. The latest MVS methods have been introduced in Middlebury evaluation (66). In this section, two leading methods (8; 27) used in this evaluation are described.

Bradley et al. (8) propose a method to produce accurate 3D results based on depth map estimation and integration with greatly reduced computation time. The overall flow is described in Figure 2.8. The algorithm has two overall stages:

1) binocular stereo on image pairs and 2) surface reconstruction. The binocular stereo stage creates depth maps from pairs of adjacent viewpoints. The depth maps calculated by a *scaled-window matching* technique are converted to 3D points and merged into a single dense point cloud. Then a hierarchical vertex clustering approach is used to eliminate some noise points. Finally, the processed point cloud is triangulated. The resulting meshes contain holes in regions occluded from the cameras, and each mesh has a different connectivity.

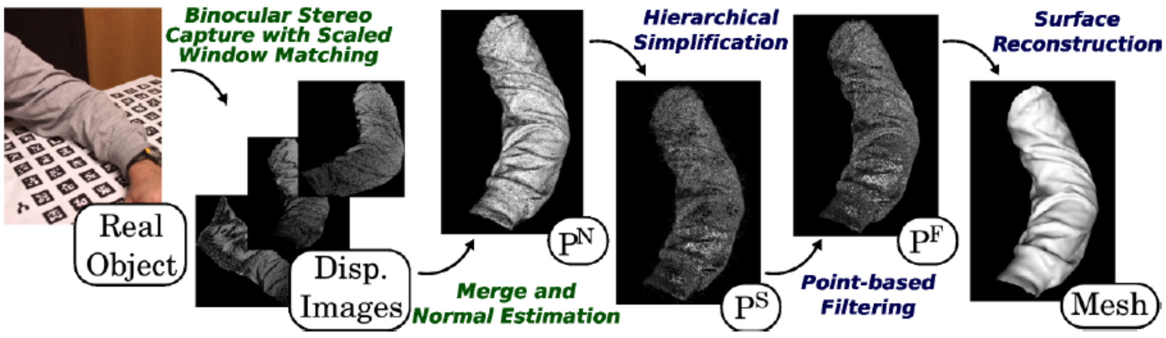

Binocular Stereo Capture with Scaled Window Matching
Real Object
Disp. Images
Merge and Normal Estimation
P^N
Hierarchical Simplification
P^S
Point-based Filtering
P^F
Surface Reconstruction
Mesh

Figure 2.8: Acquisition flow (8)

Furukawa and Ponce (27) proposed a region growing approach for MVS that propagates a surface out from initial seed points. Their approach broadly consists of three stages: 1) a patch-based MVS algorithm, 2) the conversion of the patches into a polygonal mesh model using *Poisson Surface Reconstruction* algorithm and 3) the polygonal mesh refinement via an energy minimisation approach regarding photometric discrepancy and geometric smoothness. The result of 1) a patch-based MVS algorithm is a dense collection of small oriented rectangular patches. The collection of the patches is first detected by Harris and difference-of-Gaussian operators. Expanding and filtering the detected patches are then followed. Each patch is defined by its centre and unit normal vector to represent a local tangent plane approximation of a surface shown in Figure 2.9. The patch based method is limited in its ability to calculate local region correspondences in poor-texture surfaces or sparse input images.

This is a general outline of some of the disadvantages of the MVS approaches. Correspondence methods for the depth map calculation may be more effective when views are close to each other. Additionally, correspondences should be kept

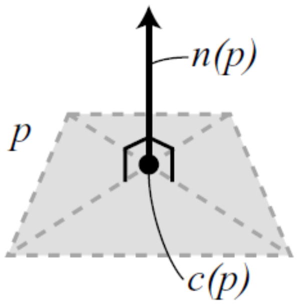


Figure 2.9: Definition of a patch (27)

over many views even if there are significant changes in viewpoint. Moreover, it is difficult to deal with occlusion differences between views.

2.1.4 Remarks

In this section, various 3D surface reconstruction methods have been discussed including *Visual Hull*, *Photo Hull*, inverse Radon transform as well as Multi-view stereo. The volume-based approach in Section 2.1.1.1 focused on the volume of the visual hull which is discretised as voxels. In general, this approach is weakened by a heavy computation and memory requirement. In Section 2.1.1.2, the surface-based approach reconstructs a surface representation of the visual hull. This method requires less computation and memory than the volume-based approach. However, the intersection in the 3D space is sensitive to numerical instabilities, especially in complicated objects. The Radon transform described in 2.1.1.3 has a continuous formulation, however its calculation applied to digital images is discrete.

Section 2.1.2 describes the *Photo Hull* approach which is based on colour consistency between images from multiple views. The underlying assumption regarding the colour consistency may not be practicable in real environments owing to uncontrolled lighting conditions and different characteristics of sensing images.

Section 2.1.3 introduces multi-view stereo approaches which require the estimation of depth information based on correspondences of stereo pair images.

These approaches need relatively higher quality images and closer adjacent views to resolve the corresponding problem than aforementioned methods.

These approaches are still very much used at least as an initial step in more advanced methods. Both *Visual Hull* and *Photo Hull* can be understood as estimating a 3D histogram describing the probability of a point in space being part of the object. These methods use discrete objective functions, which demand a memory requirement and optimisation performed with an exhaustive search.

The following section reviews Kernel density estimates that are continuous functions and Chapter 4 will show how these estimates are used to propose new continuous objective functions for *Visual Hull* and *Photo Hull*.

2.2 Kernel density estimates (KDEs)

A kernel density estimation is conceptually introduced in Section 2.2.1 along with two optimisation methods for the kernel density estimator and its applications in computer vision, which are described in Section 2.2.2.

2.2.1 Kernel density estimate of probability density functions

A kernel density estimate, also called Parzen window method (56), is a *non-parametric* density estimate for the probability density function of a random variable. It was first introduced by Rosenblatt (63). If the observation data points are $[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]$, which are independently and identically distributed (i.i.d), then the kernel probability density estimation of a random variable $\mathbf{x}$ is defined as follows:

$$\hat{f}(\mathbf{x}) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \tag{2.9}$$

where K is a kernel function, h is its bandwidth and $\int K(t)dt = 1$ to ensure that the estimate of $f(\mathbf{x})$ integrates to 1.

In the kernel density estimation, an optimal approximation has two main requirements: an appropriate kernel function and bandwidth. The most frequently selected kernel function is the Gaussian kernel:

$$K(u) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}u^2\right). \tag{2.10}$$

Table 2.1 also shows several kernel functions.

Table 2.1: Various Kernel Functions

Kernel Functions	$K(u)$	Range
Uniform	$\frac{1}{2}$	for $\lvert u\rvert \le 1$
Triangle	$(1-\lvert u\rvert)$	for $\lvert u\rvert \le 1$
Epanechnikov	$\frac{3}{4}(1-u^2)$	for $\lvert u\rvert \le 1$
Quartic(Biweight)	$\frac{15}{16}(1-u^2)^2$	for $\lvert u\rvert \le 1$
Triweight	$\frac{35}{32}(1-u^2)^3$	for $\lvert u\rvert \le 1$
Gaussian	$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}u^2\right)$	
Cosinus	$\frac{\pi}{4}\cos\left(\frac{\pi}{2}u\right)$	for $\lvert u\rvert \le 1$
High Order	$0.375(3-5u^2)$	for $\lvert u\rvert \le 1$
Cauchy	$\frac{1}{\pi(1+u^2)}$	

The bandwidth is analogous to the interval of the bins in the histogram estimation of density function and it is an empirical choice in practice. If the bandwidth h is too large, the kernel density function results in over-smoothed estimation. It is better to extract a global maxima, but this can lose local details. Conversely, the density function becomes too rough if the smoothing parameter h is too small.

2.2.2 Finding Maxima of KDEs

Kernel density estimation methods for computer vision have attracted considerable attention within the past decade. Two approaches to find maxima of KDEs have been proposed and both are discussed here: the mean shift algorithm (Section 2.2.2.1) and the Newton-style algorithm (Section 2.2.2.2).

2.2.2.1 Mean shift algorithm

The mean shift algorithm is a non-parametric mode-seeking algorithm widely used in pattern recognition and computer vision. Fukunaga and Hostetler (26) developed the general form of the kernel gradient estimates and derived conditions on the kernel functions to assure that the estimates are asymptotically unbiased and consistent. They proposed the procedure of the mean shift based on seeking the density mode (peak). Cheng (11) later provided an appropriate generalisation of the mean shift algorithm. More recently, the mean shift has been applied to image segmentation (e.g. Figure 2.10), visual tracking (e.g. Figure 2.11), nonparametric density analysis, etc. (13; 14; 15; 28). The kernel density estimation can be efficiently combined with the mean shift algorithm to optimise for Gaussian kernel (11) and for any kernels with a convex and monotonically decreasing profile, such as the Epanechnikov kernel (14).

The mean shift is originally designed to find local modes. However, in many situations the global mode of a density function is the mode of interest. Shen et al. (70) proposed a novel global mode seeking mean shift with multi-bandwidths, termed annealed mean shift. Their algorithm converges to the global mode of the density function, regardless of the initialisation point. The annealed mean shift improves on the standard mean shift algorithm's accuracy and execution time. When the density has multiple modes, the annealed mean shift has better results. Also the annealed mean shift has a small number of iterations to convergence comparing to the standard mean shift algorithm.

2.2.2.2 Newton's method

Newton's method approximates roots of a function using the iteration formula

$$x^{(m+1)} = x^{(m)} - \frac{f(x^{(m)})}{f'(x^{(m)})}. \tag{2.11}$$

Therefore $x^{(m+1)}$ is the x-intersection of the tangent line to the function $f(x)$ at $x^{(m)}$. Under the condition, Newton's method can be guaranteed to converge

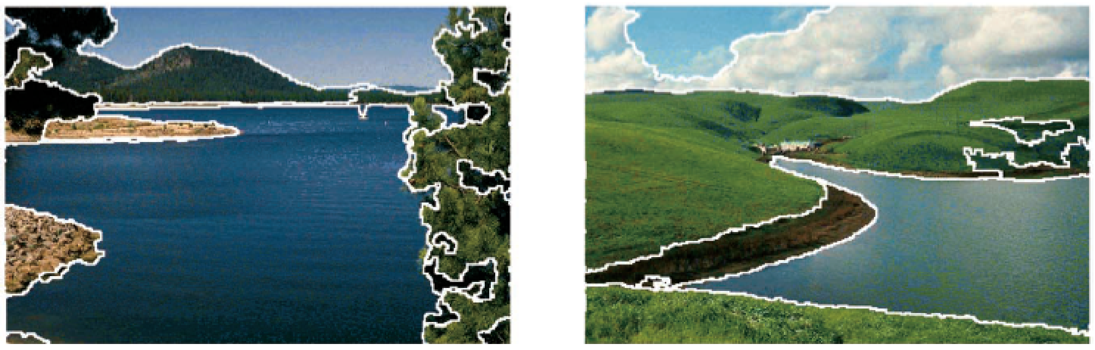


Figure 2.10: Kernel-based image segmentation using the mean shift algorithm (14).

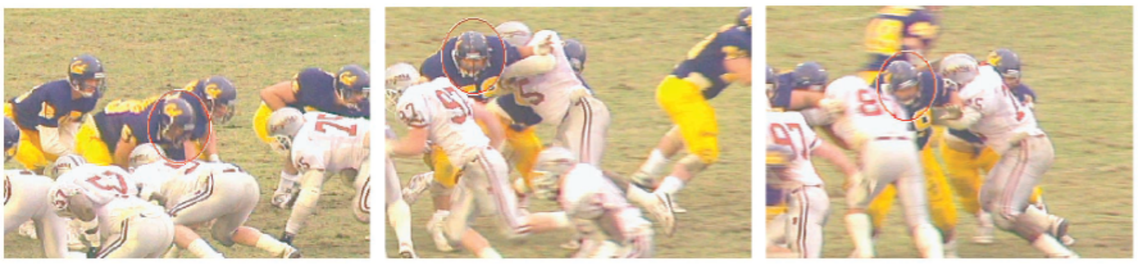


Figure 2.11: Real-time kernel-based colour tracking using the mean shift algorithm (15).

quickly to a root x_r of $f(x)$, as long as x_0 is sufficiently close to x_r. This approach can be used to find the roots of the derivatives of the kernel density estimate. In the multi-dimensional case as follows, starting from an initial guess, the iterative process is written as:

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - \mathbf{H}^{-1}\nabla f(\mathbf{x}^{(m)}), \tag{2.12}$$

where $\mathbf{H}$ is the Hessian matrix of f at $\mathbf{x}^{(m)}$ and $\nabla f(\mathbf{x}^{(m)})$ is the gradient of f at $\mathbf{x}^{(m)}$.

Hager et al. (31) proposed a tracking method using multiple kernels to address invariance to rotation and scaling. The tracking equation is linearised and solved by the Newton-style iteration to simplify the optimisation procedure. It makes the tracking process faster. Fan et al. (24) further extend the multiple kernels to enhance the *kernel-observability* for articulated objects and complex motions. The SSD tracker was improved by using the observability theorem in (60).

Quasi-Newton gradient descent algorithms for finding maxima of KDEs have also been proposed (30; 43; 70). Liu and Chen (43) proposed a visual tracking technique using the Quasi-Newton algorithm which has better accuracy than mean shift optimisation. The Quasi-Newton algorithm was also used for kernel-based template alignment (30). This optimisation has been shown to be about twice as fast as the standard mean shift for data clustering (70).

2.2.3 Suitability of KDEs, Mean shift and Newton algorithms for parallel computing

Parallel implementations of the mean shift algorithm have been proposed on a Graphics Processing Unit (GPU) (23; 42; 74; 87). Li and Xiao (42) developed a parallel mean shift tracking on the GPU using CUDA, and Exner et al. (23) presented efficient GPU implementations for a continuously adaptive mean shift (CAMShift) algorithm. A parallel mean shift algorithm was implemented for 3D medical volume segmentation (87). Srinivasan et al. (74) provide a library (called GPUML) which is for a C/C++ and MATLAB interface for speeding up the computation of the weighted kernel summation and kernel matrix construction on GPU. From these implementations, it is possible to improve hugely the speed of the mean shift-based approaches.

2.3 Conclusion

In Section 2.1 we presented a review of *Visual Hull* and *Photo Hull* for 3D reconstruction among others. Both methods are in frequent use today, whether alone or as a pre-processing step to more advanced techniques. These two methods, the former using silhouette images and the latter using unsegmented colour images, have a formulation that models a discrete objective function similar to a 3D histogram. To limit the memory consumption of *Visual Hull* and *Photo Hull*, a reformulation of the problem is proposed in Chapter 4 using Kernel density estimates (KDEs). KDEs has been discussed in Section 2.2 along with several iterative optimisation techniques that are appropriate for parallel architecture.

Chapter 3

Voxel-based 3D reconstruction using sihouettes

This chapter presents a method of reconstructing a 3D shape from a sequence of images recorded by a multi-view system[1]. Indeed by combining silhouettes recorded at different points in time, more projections are available for the estimation of *Visual Hull*. This strategy has been used in Zimmermann et al. (88) to refine the 3D reconstruction of an object over time. This concept and its repercussions are discussed in this chapter under the following assumptions:

- The cameras are fixed standard cameras and their calibration parameters are known.
- The object of interest to be reconstructed is assumed to be rigid.
- The object of interest is in motion. This means that new views of the object can be captured from the cameras over time.
- The motion of the object of interest in the 3D space is assumed to be known. Standard motion tracking techniques can help to extract this information (19; 65).

[1]Part of this chapter was published in (36).

- All recorded images have been segmented (automatically or by hand) such that nearly perfect silhouettes are available. Various automatic techniques for background subtraction can be used to help this pre-process (44).

Firstly, section 3.1 presents in detail the algorithm to compute a voxel-based volume reconstruction from a set of silhouettes recorded at the same time. Section 3.2 explains how this initial estimate of the volume can be refined over time. The bulk of the description of this method, however, is featured in section 3.3, which presents two sets of results illustrating the approach:

- The first set is computed from the images available in the Middlebury database (as outlined in section 3.3.1). This database contains high resolution images of a rigid object recorded using a rotating platform and a digital camera, in a well controlled lighting environment.

- The second set of results has been computed using the Humaneva database (section 3.3.2). The object of interest is the head of a person (assumed to be rigid) who is walking around indoors and being filmed by four cameras. The resolution of the head in the images is low and this situation would be quite similar to what is recorded in video surveillance scenarios. Indeed voxel-based approaches to 3D reconstruction are more stable than surface-based approaches for low resolution images (see section 2.1.1.2).

3.1 3D Voxel Reconstruction

A voxel is a 3D cube used as a block element (or brick). The world volume can be considered to be composed of voxels, each of which contributes to a pixel (or pixels) in the images produced by cameras. The size of the cube defines a spatial enclosure of the voxel. The camera calibration parameters of each camera are required for computing back-projection functions which allow the positions in 3D space to be mapped to 2D images. A standard voxel-based 3D shape reconstruction algorithm is summarised in Table 3.1.

The world volume is split into voxels and classified into two categories: occupied or unoccupied. The set of the occupied voxels is an approximation of the

Table 3.1: The voxel-based 3D shape reconstruction algorithm. $I = \{I_1, I_2, \cdots, I_C\}$ is a set of colour images recorded at the same time, $S = \{S_1, S_2, \cdots, S_C\}$ is the corresponding set of silhouettes, and C is the number of cameras.

```
Step 1.   Initialisation: Divide the world volume observed by cameras
          into $N \times N \times N$ voxels, $v_n, n = 1, \cdots N^3$.
Step2.    Iteration:
            FOR $n = 1$ to $N^3$
              $counter = 0$
              FOR $c = 1$ to $C$
                PROJECT the voxel $v_n$ onto the $I_c$ image plane of camera $c$
                IF the projected point is inside the silhouette $S_c$
                  then increment $counter$ by 1
                END IF
              END FOR
              IF $counter = C$ then
                MARK $v_n$ as occupied
              END IF
            END FOR
```

visual hull of an object. In general, the estimated voxel based reconstruction is significantly larger than the actual size. This is one of the disadvantages of using voxels to represent a 3D shape.

The binary 3D array computed by the algorithm shown in Table 3.1 is indexed by the voxels $\{v_n\}_{n=1,\cdots,N^3}$ and labelled occupied (1) or unoccupied (0). It can be interpreted as an estimate of a probability density function of the 3D spatial position $\mathbf{x} = (x, y, z)$ to be in the volume of the object of interest, or not.

$$P(\mathbf{x}) \propto \begin{cases} 1 & \text{if } \exists n \in [1 : N^3] \text{ such that } \mathbf{x} \in v_n \text{ and } v_n \text{ is occupied;} \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

This representation is discrete and can be understood as a 3D histogram.

3.2 Refining the 3D reconstruction over time

Now we consider that sequences of images $\{I_1^t, \cdots, I_C^t\}_{t=1,\cdots,T}$ are collected from C cameras over the time period $t = 1$ to $t = T$. The silhouettes $\{S_1^t, \cdots, S_C^t\}_{t=1,\cdots,T}$ are segmented and at each time t, an estimate of the histogram $P_t(\mathbf{x}_t)$ can be computed using the algorithm in Table 3.1. Before these T estimates can be summed, the object motion that has occurred between different points in time needs to be compensated for refining the 3D reconstruction. We choose a reference time t_{ref} and the motion between any time t and t_{ref} is compensated for $P_t(\mathbf{x}_t)$. Once all estimates are motion compensated, each gives an estimate of the volume occupancy at time t_{ref} and the refinement is performed by averaging these T estimates. The algorithm used for refinement is summarised in Algorithm 1.

Algorithm 1 Refinement of the voxel-based reconstruction.

```
Initialisation: select reference time $t_{ref} \in [1 : T]$. The reference 3D array is set
to zeros $P_{t_{ref}}(\mathbf{x}_{ref}) = 0$.
for $t = 1 \rightarrow T$ do
    Compute $P_t(\mathbf{x}_t)$ from silhouettes $\{S_1^t, \cdots, S_C^t\}$ using algorithm in Tab. 3.1.
    Given the motion transformation $M_t(\cdot)$ between time $t$ and $t_{ref}$: $\mathbf{x}_t =$
    $M_t(\mathbf{x}_{ref})$, compute the motion compensated array $P_t(M_t(\mathbf{x}_{ref}))$
    Refine $P_{t_{ref}}(\mathbf{x}_{ref}) := P_{t_{ref}}(\mathbf{x}_{ref}) + P_t(M_t(\mathbf{x}_{ref}))$
end for
Compute $P_{t_{ref}}(\mathbf{x}_{ref}) := \frac{1}{T} P_{t_{ref}}(\mathbf{x}_{ref})$
```

The model for the 3D motion $M_t(\cdot)$ is explained in Section 3.2.1, and some information about the calculation of $P_t(M_t(\mathbf{x}_{ref}))$ is given in Section 3.2.2

3.2.1 Affine motion $M_t(\cdot)$

The object of interest is assumed to be rigid and the motion transformation between time t and t_{ref}, $\mathbf{x}_t = M_t(\mathbf{x}_{ref})$, is modelled with a 3×3 rotation matrix R_t and translation vector $\boldsymbol{\tau}_t$ as follows:

$$M_t(\mathbf{x}_{ref}) = \mathrm{R}_t \mathbf{x}_{ref} + \boldsymbol{\tau}_t, \tag{3.2}$$

The rotation matrix consists of three matrices with respect to the X, Y, Z axes, $\mathrm{R}_X, \mathrm{R}_Y, \mathrm{R}_Z$ defined as follows,

$$\begin{aligned} \mathrm{R}_X &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}, \\ \mathrm{R}_Y &= \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix}, \\ \mathrm{R}_Z &= \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \end{aligned} \tag{3.3}$$

where θ, ϕ, φ are the rotation angles around the X, Y, Z axes. If there is no motion, then the visual hulls are the same as the one computed at the reference time and the refinement does not improve the volume estimate over time. If there is motion, the estimated visual hull P_t at time t, should be rotated and translated back onto the same grid as the 3D histogram of the reference $P_{t_{ref}}$.

3.2.2 Computation details

The motion compensated histogram $P_t(M_t(\mathbf{x}_{ref}))$ is computed using trilinear interpolation of P_t onto the 3D reference grid. For each position $\mathbf{x}_{ref}$ on the reference grid, the transformed position $M_t(\mathbf{x}_{ref})$ may not directly index the array P_t but fall in between several cells of the array P_t. This is illustrated in Figure 3.1 where the point A indicates the position $M_t(\mathbf{x}_{ref})$ and the values of P_t are only available on the nearest eight corners $\{a_1, a_1, \cdots, a_8\}$. Therefore, the probability $P_t(A)$ of the point A can be computed using the probabilities of the eight corner points as follows:

$$P_t(A) = \sum_{i=1}^{8} \omega_i P_t(a_i), \tag{3.4}$$

where the weights $\{\omega_i\}_{i=1,\cdots,8}$ are calculated by the distance ratio between point A and point a_i.

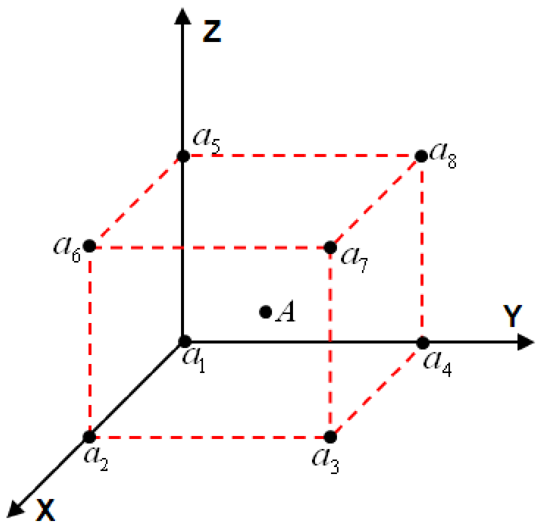


Figure 3.1: Eight corner points on a cube surrounding the point A.

3.3 Experimental Results

The method proposed here is evaluated using the Middlebury (67) and HumanEva-II datasets (71). The Middlebury dataset provides high resolution images and exact motion information. On the other hand, rough motion information has been extracted by hand for the HumanEva-II dataset. This dataset contains low resolution images which is a more realistic means of assessing the method as it will be used in genuine situations. More detail of the database is shown in Appendix A.

The evaluation measurement is based on the variation of the number of voxels as updating new views. In general, the lower the number of views, the bigger reconstruction results having a higher number of voxels. Therefore, the variation of the number of voxels is analysed to investigate how closely the reconstruction results are carved to be similar to the actual volume.

3.3.1 Middlebury Dataset

Experiments are carried out on real images from the Middlebury dataset to investigate the feasibility of this approach. 48 views are located horizontally around an object of interest, of which 40 are picked up. It is assumed that these 40 views

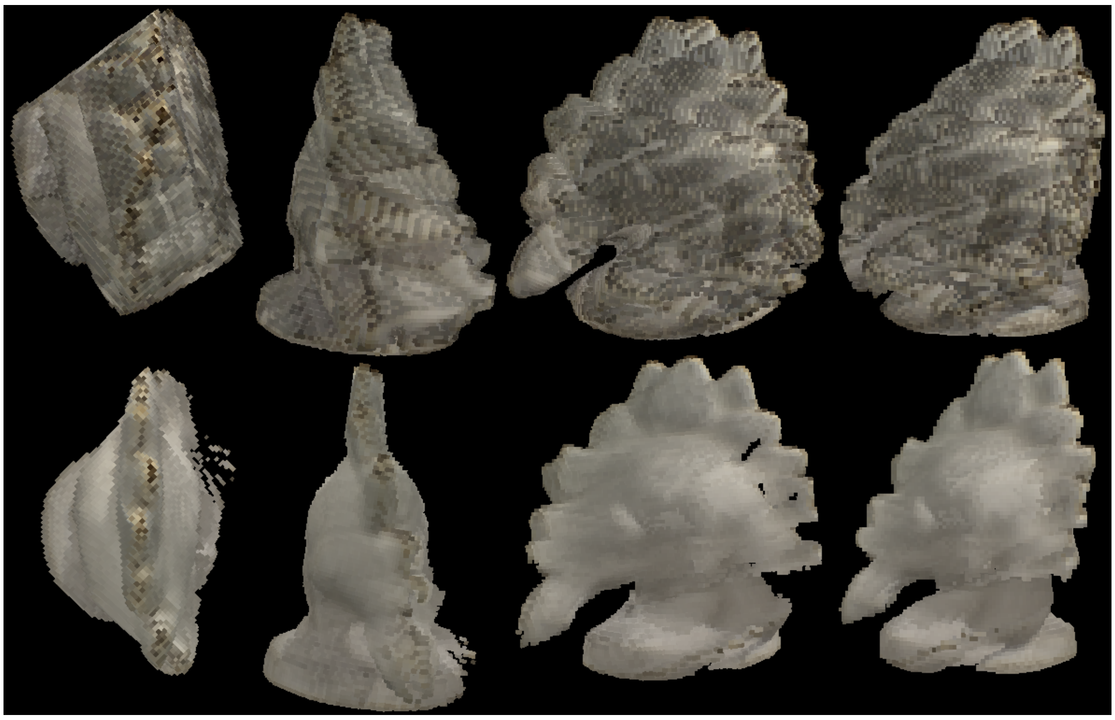


Figure 3.2: Different views of experimental results from Middlebury dataset (Top-row: a dinosaur result from ($t = 0$), bottom-row: a final dinosaur result from ($t = 10$)).

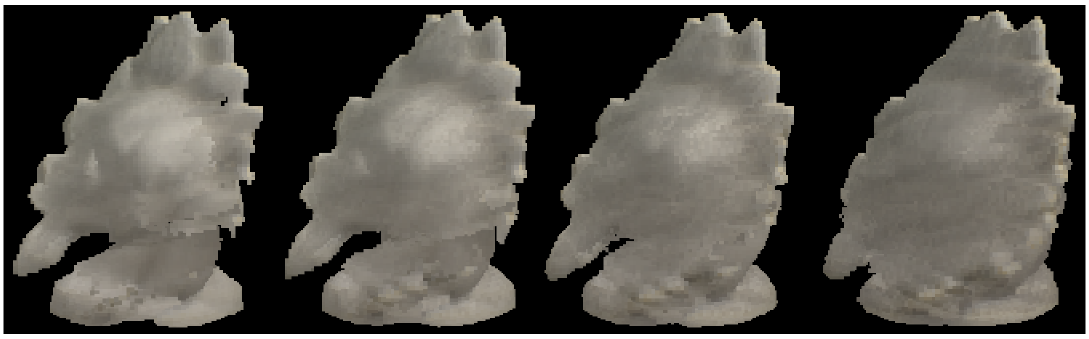


Figure 3.3: The final results with various thresholds and the voxels number from left $[0.975, 0.925, 0.875, 0.8]$ and $[85326, 107474, 124285, 150907]$.

have been recorded sequentially by 4 cameras from $t = 1$ to $t = 10$. The motion information is calculated using the locations of the cameras that take these 40 views. A reconstruction result from only 4 views ($t = 0$) is illustrated in the top row, Figure 3.2. The refinement is performed until ($t = 10$), at which all 40 views have been merged, and the final result is shown in the bottom row of Figure 3.2. There is no dedicated colour assignment procedure for voxels, and hence only the shape should be assessed. Several results computed with different thresholds are shown in Figure 3.3. It should be noted that this method is not unduly sensitive to the selection of threshold.

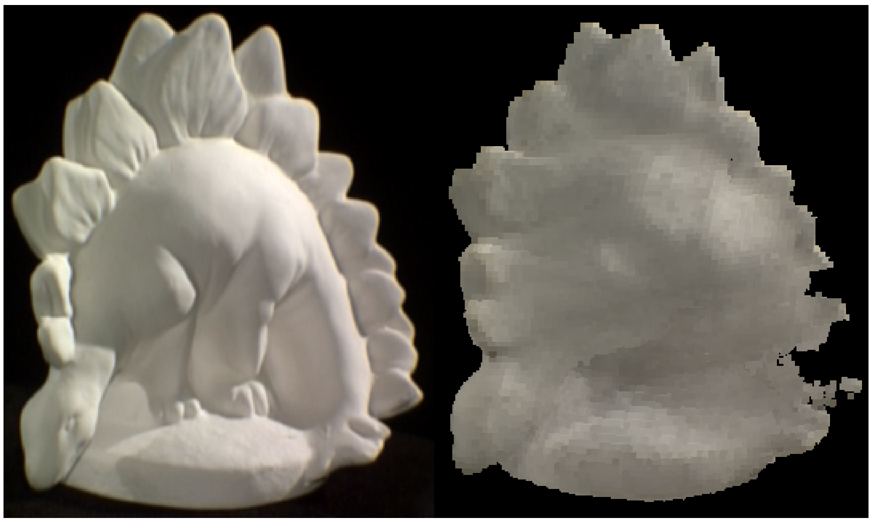


Figure 3.4: One of the dinosaur images (left) and the final reconstruction result from the threshold = 0.95 (right). The bottom right of the result is poorly reconstructed due to the error of the silhouette extraction. Indeed, shadows on the object makes the silhouette extraction hard.

The final result has 97966 voxels (the optimal threshold for the voxel probability = 0.95) carved from 158278 voxels illustrated in the right of Figure 3.4.

Figure 3.5 shows the comparison between results obtained using the proposed method, and those obtained by merging all 40 views as if they were recorded at the same time. Both results are very similar, and therefore this experiment illustrates the feasibility of estimating a 3D shape when 3D motion information is available.

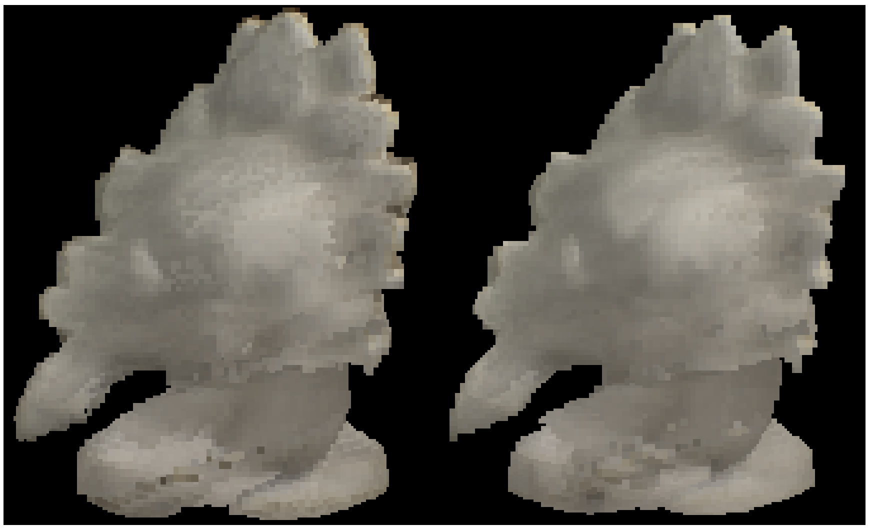


Figure 3.5: Comparison results from 40 simultaneous views (left) and the proposed method with threshold= 0.95 (right).

3.3.2 Humaneva-II Dataset

HumanEva-II dataset (71) is generated using a hardware synchronised system with four colour video cameras. This dataset is selected because it is more realistic for applications such as video surveillance, where images on the object of interest are in low resolution, and there are pose variations of the object over time. For example, even the OpenCV face detector (9), which has shown promise in the field of face detection, does not work well on these images because of the low resolution of the faces in the dataset (the size of faces roughly varies from 35×30 to 20×20). Figure 3.6 shows some images and the configuration of the data capture. The dataset is calibrated using the Matlab camera calibration toolkit (6), and contains two test sequences of different people with several actions: walking, jogging, boxing, and so on.

The walking part is used here to reconstruct the 3D head of the person in view. 16 frames are used, picked up from around 400 frames containing walking motion. These are illustrated in Figure 3.7. In selecting these 16 frames, the computation time is limited but the selected set is shown to represent various viewing angles of the head.

Figure 3.8 shows the variation of the voxel number labelled as a head when

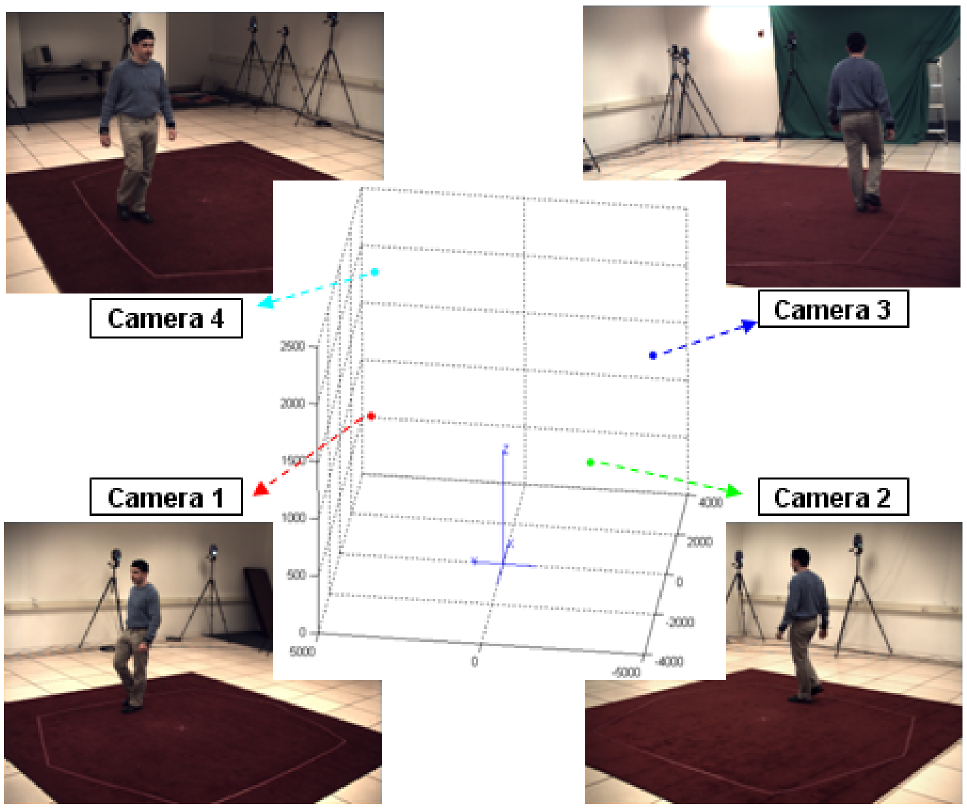


Figure 3.6: Plot for each camera and captured images in HumanEva-II.

accumulating information over time. The volume of the head is shown to decrease and then stabilise around time $t = 8$. This is because the recorded images from $t = 1$ to $t = 8$ are quite similar to the frames recorded between times $t = 9$ and $t = 16$. In other words, the images recorded from cameras C3 and C4 between $t = 1$ and $t = 8$ are similar to the images recorded by cameras C1 and C2 between times $t = 9$ amd $t = 16$ (see Figure 3.7). The volume increases slightly after the 8_{th} frame and decreases slightly after the 12_{th} frame, due to inaccuracies in the estimated motion information and the silhouette extraction. Although variation is minor after the 8_{th} frame, the final result at $t = 16$ is more realistic than any previous estimate (see Figure 3.9). Figure 3.9 illustrates a voxel representation of different views for comparing between frames including the reference of a 3D head at ($t = 1$) and the final result of a 3D head at ($t = 16$). In terms of the realism of the shape created, the voxel representation of the final result is well

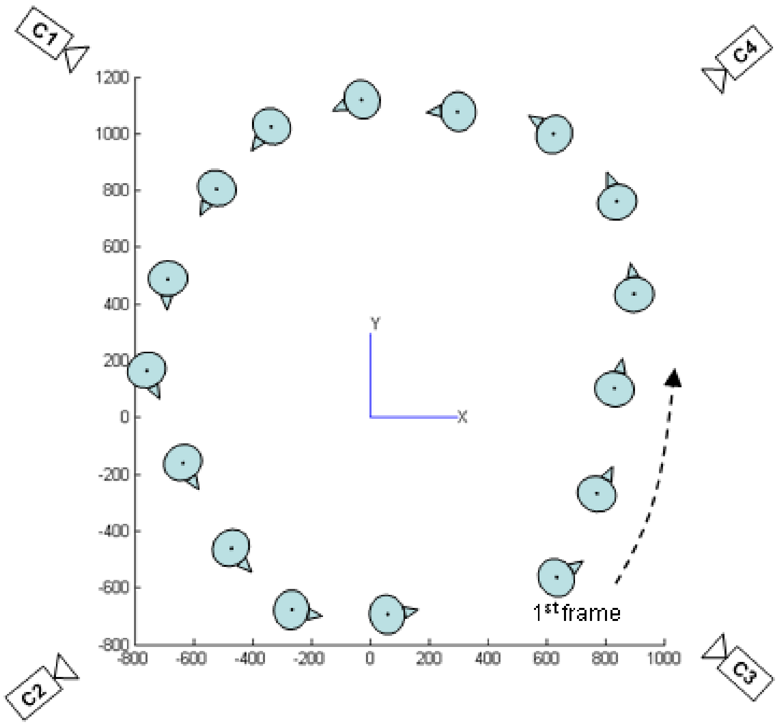


Figure 3.7: Walking footprints in the HumanEva-II (71).

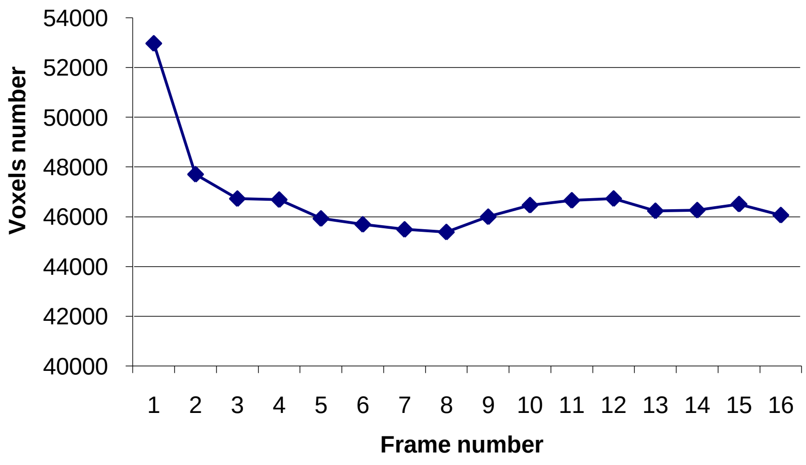


Figure 3.8: The variation of voxel number between frames (the probability threshold = 0.65).

carved and much more natural than the reference.

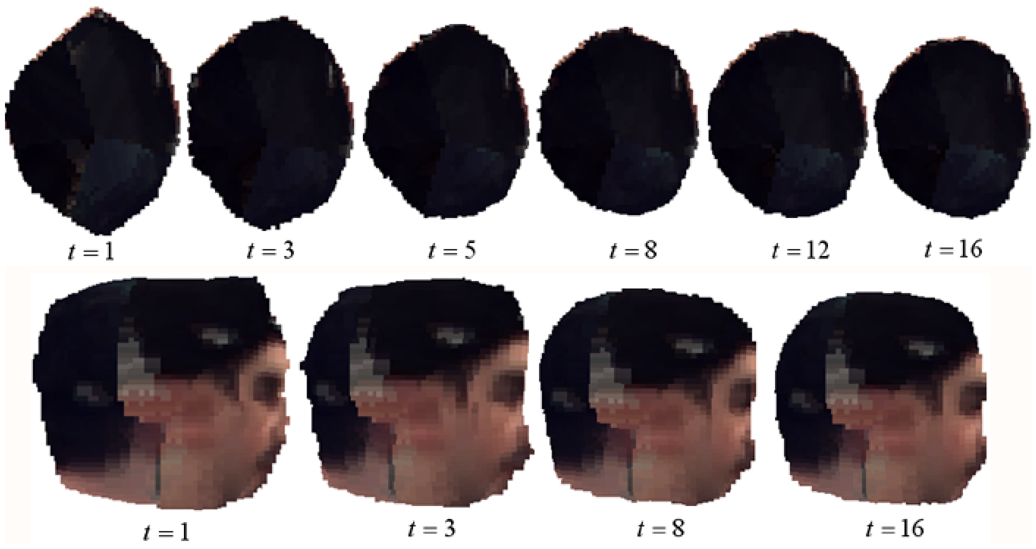


Figure 3.9: Voxel representation of top view (the first row) and side view (the second row) along with time t.

Figure 3.10 and 3.11 illustrate a comparison between a 3D head of the reference frame and a final 3D head result updated by 16 frames. The volume of the final head is much more condensed than the reference 3D head. There is no colour update, hence the texture of the head model has some artifacts.

Figure 3.12 illustrates the distribution of the voxel number along with its probability in the final head. The threshold is set between 0.6 and 0.7 and is an optimal choice in our experiments.

These values are comparatively smaller than the Middlebury dataset test in Section 3.3.1 due to less accurate motion and silhouette information. Voxels which have low probability are discarded by the threshold. The variation of the number of voxels with different threshold is shown in Figure 3.13. The point clouds of the 3D head models of the reference and final result are shown in Figure 3.14. Red (black) points correspond to occupied voxel centres of the reference frame, and green (grey) points come from the final result which is generated from the probability threshold, 0.65.

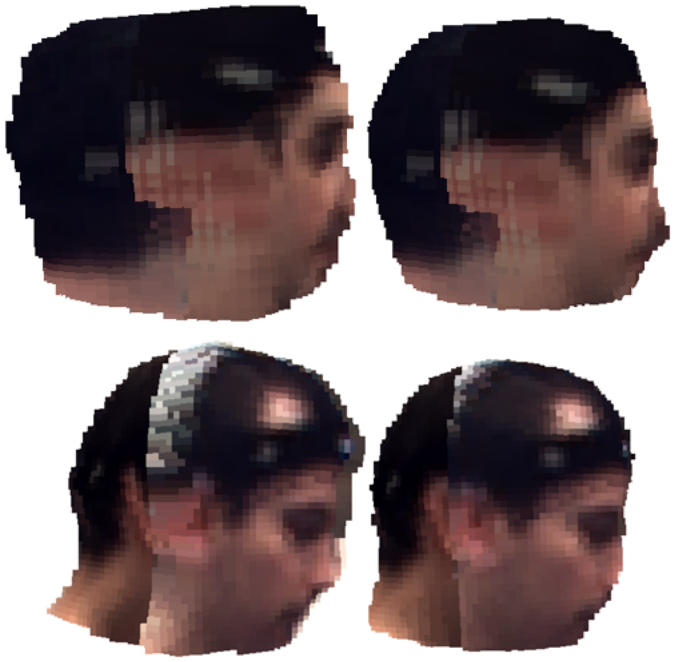


Figure 3.10: Side-view comparison between references (left column) and final results (right column).

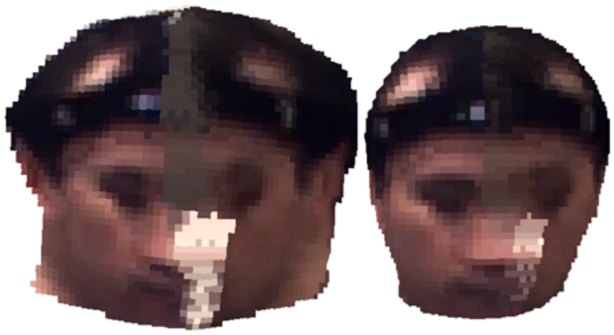


Figure 3.11: Frontal-view comparison between a reference and a final result.

3.4 Conclusion

This chapter has presented in further detail the shape from silhouette approach that estimates the visual hull using voxel-based representation. This approach can be viewed as the estimation of a discrete 3D histogram that is thresholded to find the estimated volume of the object of interest. When the motion of the

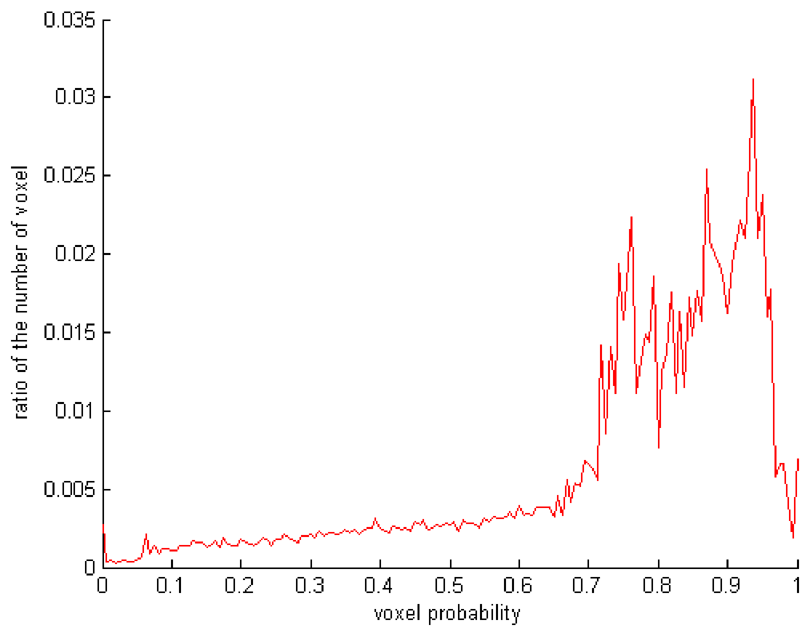


Figure 3.12: The distribution of the voxel number according to the voxel probability. This graph shows the actual volume is located between 0.7 and 1

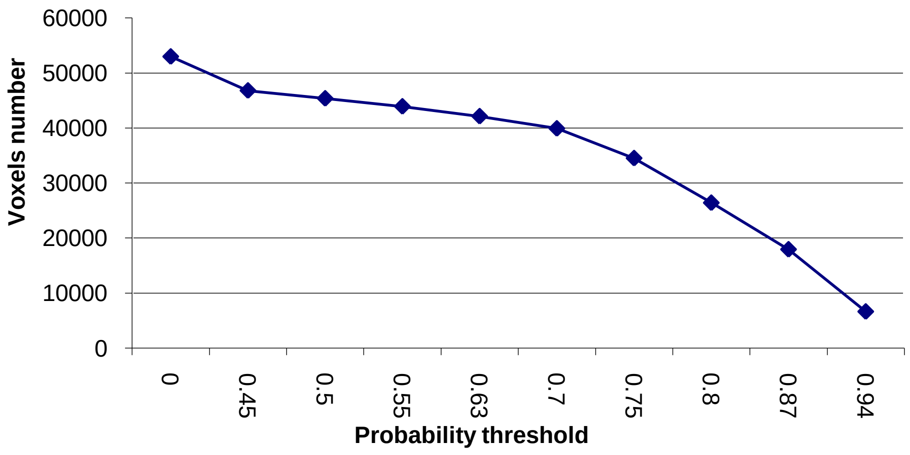


Figure 3.13: The variation of voxel number of the final result with different threshold.

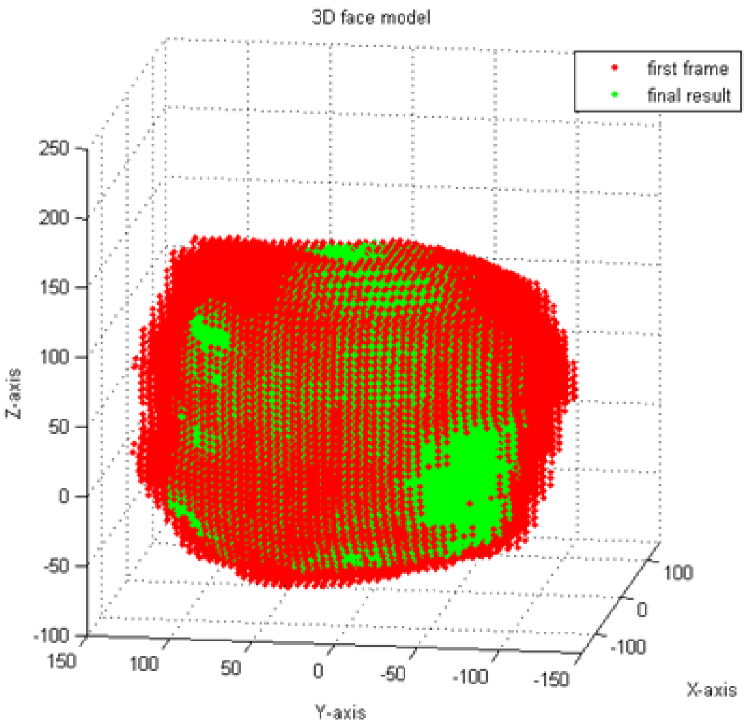


Figure 3.14: Plot of occupied voxel centres: red and green points come from the reference 3D head and the final result, respectively.

object is known in the 3D world, this approach can be refined up to the point where an infinite number of silhouettes is available and the estimate converges toward the true visual hull. Several disadvantages of this approach are:

1. The discrete nature of the histogram means that this approach places a high demand on memory;
2. The methods require silhouettes as inputs that can be difficult to compute automatically without error;
3. In practice, computer-aided methods for motion estimation will provide some rough 3D motion estimations, and these errors will also impact on the quality of the reconstruction.

The next chapter proposes to address the first two of these limitations: an explicit continuous formulation is introduced and colour information from the images is also used to remove the need for silhouettes as inputs.

Chapter 4

Kernel Density Estimate for 3D Shape Reconstruction

Having already discussed voxel-based 3D construction using silhouettes in Chapter 3, the purpose of this chapter is to give further detail on the proposed method being put forward in this thesis as an alternative to such voxel-based methods. In the previous chapter we alluded to the potential disadvantages of the shape-from-silhouette approach, highlighting the high memory demand placed by the discrete nature of the histogram, and the difficulty of accurately computing silhouette inputs. This chapter aims to address these limitations by discussing the possibility of introducing an explicit continuous formulation, and the use of colour information from the images to remove the need for silhouettes as inputs. In it, we propose explicit smooth functions that will first estimate the visual hull from silhouette images; this process will be outlined in Sections 4.1 and 4.2. 4.3 will then estimate the photo hull from unsegmented colour images.

It is assumed that multiple synchronised cameras capture images of a 3D object from different point of views (see Figure 4.1). For the sake of simplicity, the cameras are assumed to be orthographic cameras. Note however that the framework presented in this chapter is not limited to orthographic cameras, and has indeed recently been extended to apply to standard cameras as well (64).

Four new smooth kernel density estimates are introduced to reconstruct a 3D shape of interest in this chapter. First, like Pintavirooj and M. Sangworasil (58), we consider that a 3D object volume can be reconstructed by a stack of 2D

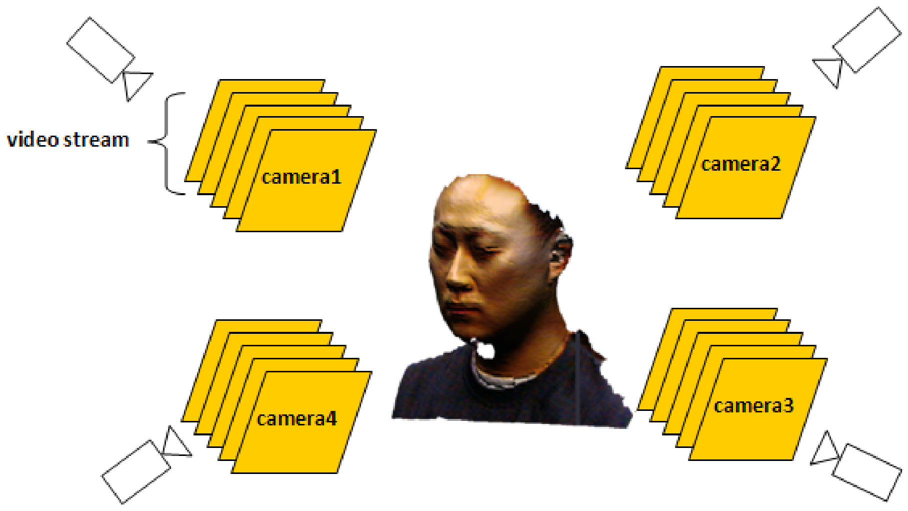


Figure 4.1: Multi-camera system.

slices. In this case the 3D reconstruction problem is reduced to the reconstruction of a 2D shape in each slice (Section 4.1)[1]. This kernel density estimate is then extended in order to carry out the 3D reconstruction directly without considering 2D slices. This is described in Section 4.2.

The first two kernel density estimates use silhouette images as inputs. Section 4.3 extends this framework to use colour information and infer the photo hull. In this case, colour images no longer need to be segmented and are used directly as inputs.

Mean shift algorithm is now a well known method for finding the maxima of kernel density estimates. An explanation is provided for the combination of the mean shift algorithm with the proposed kernel density estimates to find a 3D shape and its colour.

The rest of this chapter is organised as follows: The respective algorithms for the estimation of the most likely positions for the surface of the object in the 3D space are presented in Sections 4.1, 4.2 and 4.3. Section 4.4.1 proposes to use a simulated annealing approach for the bandwidth used as a temperature in order to avoid local maxima in the kernel density estimates. Section 4.5 illustrated

[1] This new method has been published in 2010 in the International Conference on Acoustics, Speech and Signal Processing (ICASSP) (37).

experimentally how well these methods perform, and a conclusion is provided in Section 4.6.

4.1 2D Kernel Density Estimate for Shape from Silhouettes

4.1.1 Hypotheses

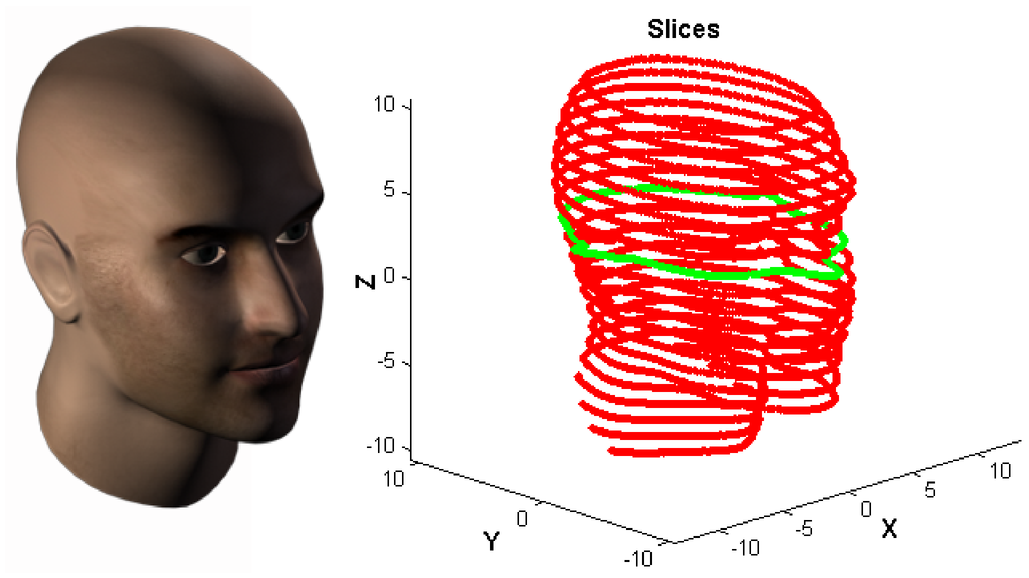


Figure 4.2: 3D head shape and 2D slices of the head (Green line illustrates a selected 2D slice).

A 3D shape is recovered by first reconstructing the 2D slices of an object from each line of the silhouette images. Each 2D slice is estimated and the 3D shape is represented by the set of the 2D slices. To record the 3D shape and define the 2D slices along the Z-axis, cameras are horizontally located around the 3D object (see Figure 4.15). A 2D slice of a 3D head is illustrated in Figure 4.2. For simplification purposes, the camera matrix is chosen as an orthographic projection: each foreground pixel on the 2D silhouette images is projected from the 2D image plane to 3D space as a ray using orthographic projection, as shown in Figure 4.3. Orthographic projection is an acceptable model for cameras when the object is close to the optical axis and the object's dimension is small in comparison to its distance to the camera.

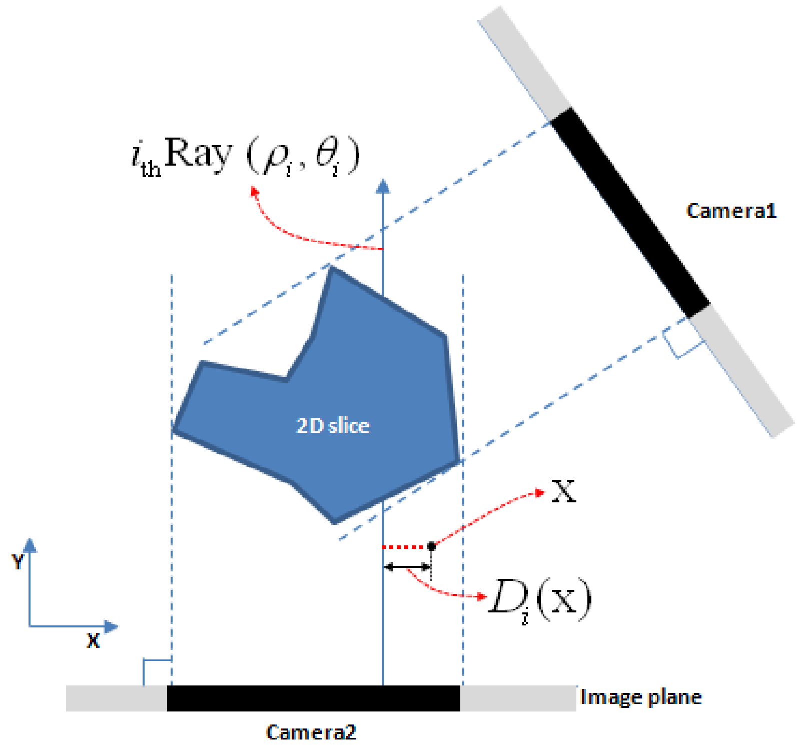


Figure 4.3: Orthographic projection and $D_i(\mathbf{x})$.

4.1.2 Kernel Density Estimate Function

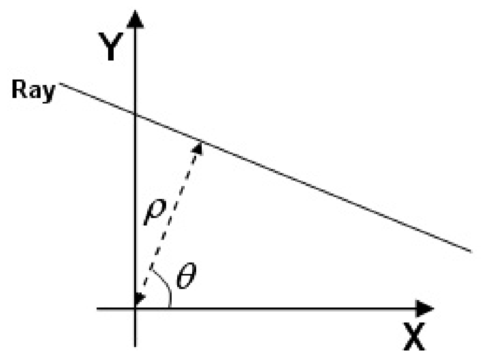


Figure 4.4: ρ and θ to define a ray from 2D image plane to 3D space.

We use a polar coordinate system to define the ray created by the foreground pixel i:

$$\rho_i = x \ \cos\theta_i + y \ \sin\theta_i \tag{4.1}$$

The concept of ρ and θ is illustrated in Figure 4.4. All parameters $\{(\rho_i, \theta_i)\}_{i=1,\cdots,n}$ are known: for all foreground pixels (n) in all image silhouettes from all camera views. Having only one ray (ρ_i, θ_i), our proposal is to model the probability density function of the random variable $\mathbf{x} = (x, y)$, representing the spatial position of the object in the 2D slice. If we want all possible positions to be exactly on the ray generated by (ρ_i, θ_i), then we could select the Dirac kernel as follows:

$$\hat{p}(\mathbf{x}|(\rho_i, \theta_i)) \propto \delta\left(\rho_i - x \ \cos\theta_i - y \ \sin\theta_i\right) \tag{4.2}$$

Instead, the Gaussian kernel is used:

$$\hat{p}(\mathbf{x}|(\rho_i, \theta_i)) \propto \frac{1}{\sqrt{2\pi}h} \exp\left(\frac{-(\rho_i - x \ \cos\theta_i - y \ \sin\theta_i)^2}{2h^2}\right) \tag{4.3}$$

This choice of the Gaussian kernel facilitates consideration of the positions close to the ray as potential positions for the object with a probability non-zero. Each foreground pixel now creates a fuzzy cylinder representing the object's possible positions, instead of a single line.

When considering the set of all rays generated by foreground pixels, $\{(\rho_i, \theta_i)\}_{i=1,\cdots,n}$ and assuming them equiprobable, then the following kernel estimate for $\mathbf{x}$ can be proposed:

$$\hat{p}(\mathbf{x}) \propto \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\sqrt{2\pi}h} \exp\left(\frac{-(\rho_i - x \ \cos\theta_i - y \ \sin\theta_i)^2}{2h^2}\right) \tag{4.4}$$

where n is the total number of foreground pixels in all camera views. Figure 4.5 illustrates the probability density generated by the equation (4.4) for a slice of a spherical object (e.g. soccer ball) reconstructed using 36 equally and horizontally spaced camera views, and compared with that estimated by the Inverse Radon transform approach. The Inverse Radon transform reconstructs a surface (or empty object), whereas the $\hat{p}(\mathbf{x})$ models a volume object (or full object). We want now to recover the positions $\mathbf{x}$ which maximize $\hat{p}(\mathbf{x})$. This will be performed using the mean shift algorithm.

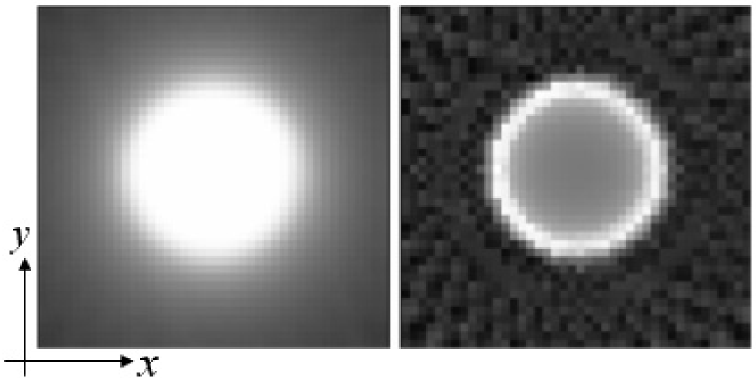


Figure 4.5: Probability density $\hat{p}(\mathbf{x})$ (left), the Inverse Radon transform (right).

4.1.3 Mean-shift Algorithm

The mean shift algorithm is performed by differentiating the kernel density estimate and equating the result to zero. The gradient of $\hat{p}(\mathbf{x})$ is:

$$\nabla\hat{p}(\mathbf{x}) =$$

$$\left[\begin{array}{l} \frac{\partial \hat{p}(\mathbf{x})}{\partial x} \propto \frac{1}{\sqrt{2\pi} n h^3} \sum_{i=1}^{n} D_i(\mathbf{x}) \;\cos\theta_i\; \exp\left(\frac{-D_i^2(\mathbf{x})}{2h^2}\right) \\ \\ \frac{\partial \hat{p}(\mathbf{x})}{\partial y} \propto \frac{1}{\sqrt{2\pi} n h^3} \sum_{i=1}^{n} D_i(\mathbf{x}) \;\sin\theta_i\; \exp\left(\frac{-D_i^2(\mathbf{x})}{2h^2}\right) \end{array}\right] \tag{4.5}$$

where, $D_i(\mathbf{x}) = (\rho_i - x\cos\theta_i - y\sin\theta_i)$ illustrated in Figure 4.3. Using the equation (4.5), starting from an initial position $\mathbf{x}^{(0)}$, the mean-shift iteration to converge towards the nearest local maximum, is then:

$$\mathbf{x}^{(t+1)} = \left(L(\mathbf{x}^{(t)})\right)^{-1} \cdot M(\mathbf{x}^{(t)}). \tag{4.6}$$

where $L(\mathbf{x})$ is a 2×2 matrix:

$$L(\mathbf{x}) = \sum_{i=1}^{n} \exp\left(\frac{-D_i^2(\mathbf{x})}{2h^2}\right) \times \left[\begin{array}{cc} \cos^2\theta_i & \sin\theta_i\;\cos\theta_i \\ \cos\theta_i\;\sin\theta_i & \sin^2\theta_i \end{array}\right] \tag{4.7}$$

and $M(\mathbf{x})$ is the 2×1 vector:

$$M(\mathbf{x}) = \sum_{i=1}^{n} \exp\left(\frac{-D_i^2(\mathbf{x})}{2h^2}\right) \times \left[\begin{array}{c} \rho_i\cos\theta_i \\ \rho_i\sin\theta_i \end{array}\right]. \tag{4.8}$$

From the starting position $\mathbf{x}^{(0)}$, the iteration in equation (4.6) is repeated until convergence. Figure 4.6 shows the mean-shift iterations from several starting points towards their closest maxima of the probability density.

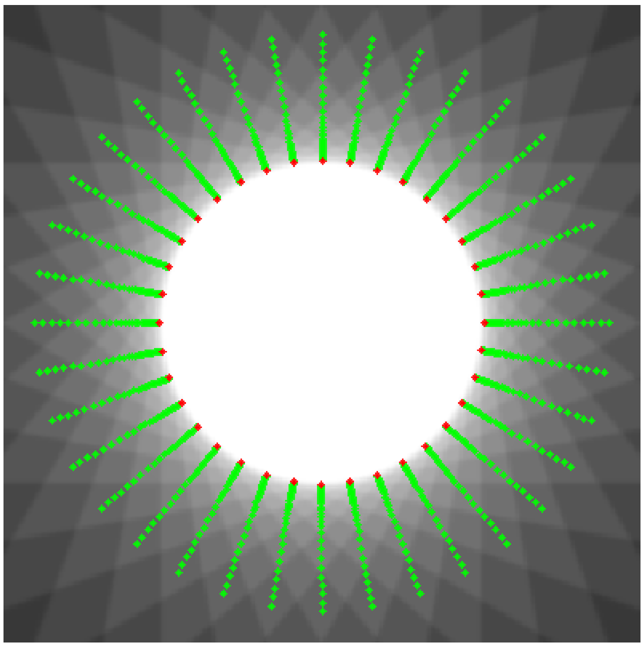


Figure 4.6: Mean-shift iteration in the case of 36 camera views (green: routes of iteratively moving the initial points towards the surface of the object by using the mean-shift equation, red: final estimated locations).

4.2 3D Kernel Density Estimate for Shape from Silhouette

In this section, a method to estimate 3D shapes from 2D silhouette images using a 3D ray and the mean shift algorithm is proposed. This method enables the estimation of a 3D shape directly, using any kind of camera settings (e.g. horizontal camera and spherical camera locations shown in Figure 4.15 and 4.16).

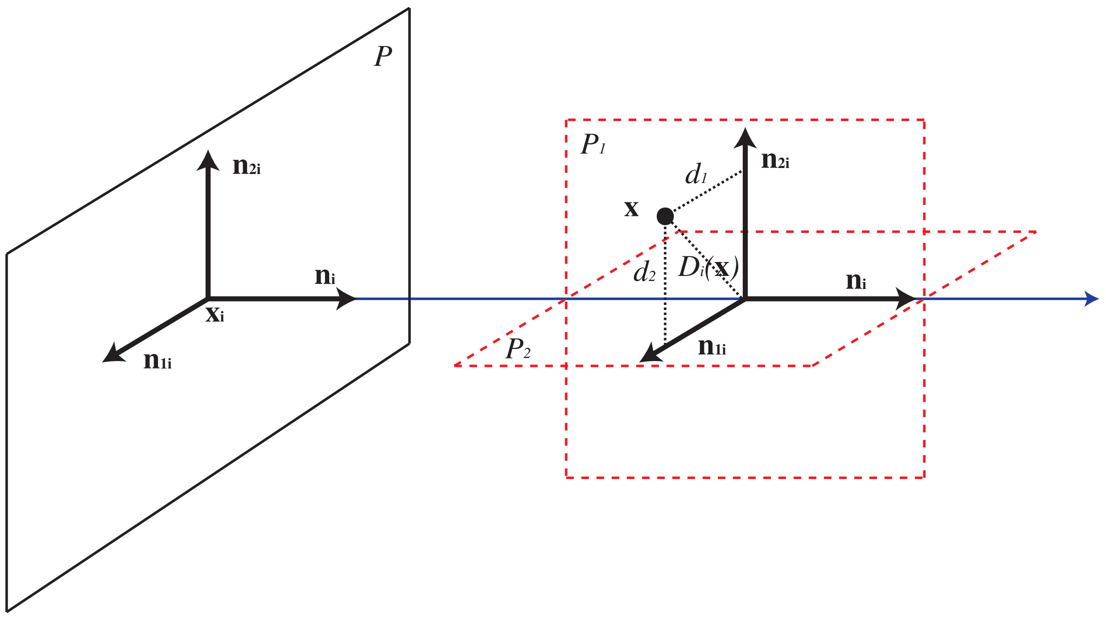


Figure 4.7: 3D ray modelled by two orthogonal planes. $\mathbf{x} = (x, y, z)$ is a random variable and $\mathbf{x_i}$ is one particular point (pixel position) belonging to P, P_1, and P_2. $D_i(\mathbf{x})$ is the shortest distance from the 3D ray.

4.2.1 Hypotheses and Notations

In order for the full 3D space to be considered for the 3D shape estimation, the image plane P is first defined in terms of its normal vector $\mathbf{n}_i$ and ρ_i. ρ_i is the shortest distance from origin to the plane.

$$P: \quad \rho - \vec{\mathbf{n}}\mathbf{x} = 0, \quad \mathbf{x} \in P. \tag{4.9}$$

where, $\mathbf{x}$ is the 3D pixel coordinate. Then, the 3D ray, which is generated by an orthographic projection, is defined by two orthogonal planes P_1 and P_2 which have normal vectors $\mathbf{n}_{1i}$ and $\mathbf{n}_{2i}$ as seen in Figures 4.7.

$$\vec{\mathbf{n}_i} = \begin{bmatrix} \cos\theta_i \cos\phi_i \\ \sin\theta_i \cos\phi_i \\ \sin\phi_i \end{bmatrix}, \quad \vec{\mathbf{n}_1}_i = \begin{bmatrix} -\sin\theta_i \\ \cos\theta_i \\ 0 \end{bmatrix}, \quad \vec{\mathbf{n}_2}_i = \begin{bmatrix} \cos\theta_i \sin\phi_i \\ \sin\theta_i \sin\phi_i \\ -\cos\phi_i \end{bmatrix},$$

where θ_i and ϕ_i are defined in the spherical coordinates of the plane shown in Figure 4.8. The ray projected from the foreground pixel can now be defined as the intersection of these two planes P_1 and P_2. All parameters $\{\theta_i, \phi_i, \mathbf{x}_i\}_{i=1,\cdots,n}$

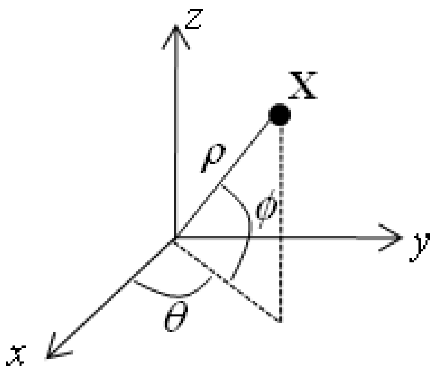


Figure 4.8: ρ, θ and ϕ

are known for all foreground pixels (n) in all image silhouettes from all camera views.

4.2.2 3D Kernel Density Estimate with Mean-shift

A probability density function of a random variable $\mathbf{x} = [x, y, z]$, which represents the spatial position of the object in the 3D space, is calculated by the shortest distance $D_i(\mathbf{x})$ shown in Figure 4.7 combining with a Gaussian kernel as follows:

$$\hat{p}(\mathbf{x}|\mathbf{n}_i) \propto \frac{1}{\sqrt{2\pi}h} \exp\left(\frac{-D_i(\mathbf{x})^2}{2h^2}\right) \tag{4.10}$$

with

$$\begin{aligned} D_i(\mathbf{x})^2 &= d_1^2 + d_2^2 \\ &= \left(\mathbf{n}_{1i}^T\,(\mathbf{x}-\mathbf{x}_i)\right)^2 + \left(\mathbf{n}_{2i}^T\,(\mathbf{x}-\mathbf{x}_i)\right)^2 \\ &= (\mathbf{x}-\mathbf{x}_i)^T\mathbf{n}_{1i}\mathbf{n}_{1i}^T(\mathbf{x}-\mathbf{x}_i) + (\mathbf{x}-\mathbf{x}_i)^T\mathbf{n}_{2i}\mathbf{n}_{2i}^T(\mathbf{x}-\mathbf{x}_i) \\ &= (\mathbf{x}-\mathbf{x}_i)^T(\mathbf{n}_{1i}\mathbf{n}_{1i}^T + \mathbf{n}_{2i}\mathbf{n}_{2i}^T)(\mathbf{x}-\mathbf{x}_i) \\ &= (\mathbf{x}-\mathbf{x}_i)^T\,\mathrm{A}_i\,(\mathbf{x}-\mathbf{x}_i) \end{aligned} \tag{4.11}$$

where

$$\begin{aligned} \mathrm{A}_i &= \mathbf{n}_{1i}\mathbf{n}_{1i}^T + \mathbf{n}_{2i}\mathbf{n}_{2i}^T \\ &= \begin{bmatrix} \sin^2\theta_i+\cos^2\theta_i\sin^2\phi_i & -\sin\theta_i\cos\theta_i+\sin\theta_i\cos\theta_i\sin^2\phi_i & -\cos\theta_i\sin\phi_i\cos\phi_i \\ -\sin\theta_i\cos\theta_i+\sin\theta_i\cos\theta_i\sin^2\phi_i & \cos^2\theta_i+\sin^2\theta_i\sin^2\phi_i & -\sin\theta_i\sin\phi_i\cos\phi_i \\ -\cos\theta_i\sin\phi_i\cos\phi_i & -\sin\theta_i\sin\phi_i\cos\phi_i & \cos^2\phi_i \end{bmatrix} \end{aligned} \tag{4.12}$$

The probability density function in the equation (4.10) can be generated when considering the set of all 3D rays as follows:

$$\begin{aligned}\hat{p}(\mathbf{x}) &\propto \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{2\pi}h}\exp\left(-\frac{D_i(\mathbf{x})^2}{2h^2}\right)\\ &\propto \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{2\pi}h}\exp\left(-\frac{(\mathbf{x}_i-\mathbf{x})^T\mathbf{A}_i(\mathbf{x}_i-\mathbf{x})}{2h^2}\right),\end{aligned} \tag{4.13}$$

where n is the number of whole foreground pixels in the selected camera views. A mean-shift iteration can be computed to estimate the 3D shape. From an initial point, $\mathbf{x}^{(0)}$, the mean-shift algorithm is computed until the nearest local maximum is reached:

$$\mathbf{x}^{(t+1)} = \left(L(\mathbf{x}^{(t)})\right)^{-1}\cdot M(\mathbf{x}^{(t)}), \tag{4.14}$$

where $L(\mathbf{x})$ is a 3×3 matrix:

$$L(\mathbf{x}) = \sum_{i=1}^{n}\exp\left(-\frac{(\mathbf{x}_i-\mathbf{x})^T\mathbf{A}_i(\mathbf{x}_i-\mathbf{x})}{2h^2}\right)\mathbf{A}_i$$

and $M(\mathbf{x})$ is the 3×1 vector:

$$M(\mathbf{x}) = \sum_{i=1}^{n}\exp\left(-\frac{(\mathbf{x}_i-\mathbf{x})^T\mathbf{A}_i(\mathbf{x}_i-\mathbf{x})}{2h^2}\right)\mathbf{A}_i\mathbf{x}_i^T.$$

4.3 Kernel Density Estimate for Shape from colour images

In this section, the 2D and 3D kernel density estimates described in the previous sections are extended to consider colour information. This concept is basically motivated by *Photo Hull* described in Section 2.1.2. Colour information is combined with the likelihood terms of the kernel density estimates and will help to improve accuracy performance [1].

[1]This work has been conducted in collaboration with Jonathan Ruttle.

4.3.1 Shape and Colour for Kernel Density Estimate

In this section, the extension of the 2D and 3D kernel density estimates is formulated to use colour information. In order to take colour into account, full *RGB* colour of the pixels is converted to chromaticity values because chromaticity red and green are more invariant to lighting conditions (4; 20). The converting equation is as follows:

$$r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B} \tag{4.15}$$

Two additional Gaussian probabilities of chromaticity red r and green g are added in the likelihood of the kernel density estimates. The colour kernel density estimate is then generalised as:

$$\hat{p}(\mathbf{x}, r, g) \propto \frac{1}{N} \sum_{i=1}^{N} \frac{1}{(2\pi)^{\frac{3}{2}} h_s h_r h_g} E_i(\mathbf{x}, r, g) \tag{4.16}$$

where, (h_s, h_r, h_g) are the bandwidths of the Gaussian kernels for the spatial and colour domains. $E_i(\mathbf{x}, r, g)$ is given by:

$$E_i(\mathbf{x}, r, g) = \exp\left(\frac{-D_i(\mathbf{x})^2}{2h_s^2}\right) \exp\left(\frac{-(r_i - r)^2}{2h_r^2}\right) \exp\left(\frac{-(g_i - g)^2}{2h_g^2}\right) \tag{4.17}$$

where, $D_i(\mathbf{x})$ of the 2D and 3D kernel density estimates are described in equations 4.5 and 4.11 respectively, and (r_i, g_i) are the (chromaticity-red and -green) of the foreground pixels.

4.3.2 Mean-shift Algorithm

Mean-shift iteration is operated by differentiating the kernel density estimate in equation 4.16 and equating the result to zero as follows:

$$(\mathbf{x}, r, g)^{(t+1)} = \left(L((\mathbf{x}, r, g)^{(t)})\right)^{-1} \cdot M((\mathbf{x}, r, g)^{(t)}). \tag{4.18}$$

In the case of the 2D kernel density estimate, $L(\mathbf{x}, r, g)$ is a 4×4 matrix:

$$L(\mathbf{x}, r, g) = \sum_{i=1}^{n} E_i(\mathbf{x}, r, g) \begin{bmatrix} \cos^2\theta_i & \sin\theta_i \ \cos\theta_i & 0 & 0 \\ \cos\theta_i \ \sin\theta_i & \sin^2\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.19}$$

and $M(\mathbf{x}, r, g)$ is the 4×1 vector:

$$M(\mathbf{x}, r, g) = \sum_{i=1}^{n} E_i(\mathbf{x}, r, g) \begin{bmatrix} \rho_i \cos\theta_i \\ \rho_i \sin\theta_i \\ r_i \\ g_i \end{bmatrix}. \tag{4.20}$$

In the case of the 3D kernel density estimate, $L(\mathbf{x}, r, g)$ is a 5×5 matrix:

$$L(\mathbf{x}, r, g) = \sum_{i=1}^{n} E_i(\mathbf{x}, r, g) \begin{bmatrix} [\mathbf{A}_i] & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.21}$$

and $M(\mathbf{x}, r, g)$ is the 5×1 vector:

$$M(\mathbf{x}, r, g) = \sum_{i=1}^{n} E_i(\mathbf{x}, r, g) \begin{bmatrix} [\mathbf{A}_i \mathbf{x}_i] \\ r_i \\ g_i \end{bmatrix}, \tag{4.22}$$

where $\mathbf{A}_i$ is defined in equation (4.12).

$(\mathbf{x}, r, g)^{(m)}$ is the current point with its colour and the point is moved to $(\mathbf{x}, r, g)^{(m+1)}$ as finding a higher probability.

4.4 Computational details

4.4.1 Bandwidths for Mean-shift

If only a small number of cameras is used, it is possible that the probability density is noisy (see Figure 4.9), and may have many spurious modes. In this case, the ordinary mean shift algorithm may get trapped in meaningless local maxima. To overcome this problem, we propose to use the mean shift algorithm with a simulated annealing scheme such as that used by Shen et al. (69).

Initially, the bandwidth starts relatively wide which results in a smoother probability density function with fewer local maxima. As the mean-shift point approaches the global maximum, the bandwidth is decreased in order to achieve the greatest possible accuracy. Such an approach allows this method to converge both robustly and quickly. The rate at which the bandwidth decreases from h_{max} to $h_{min} = 1$ is based on a geometric rate (18):

$$h_t = \alpha^t h_{max} \text{ until } h_t = h_{min} \text{ with } \alpha = 0.98, \tag{4.23}$$

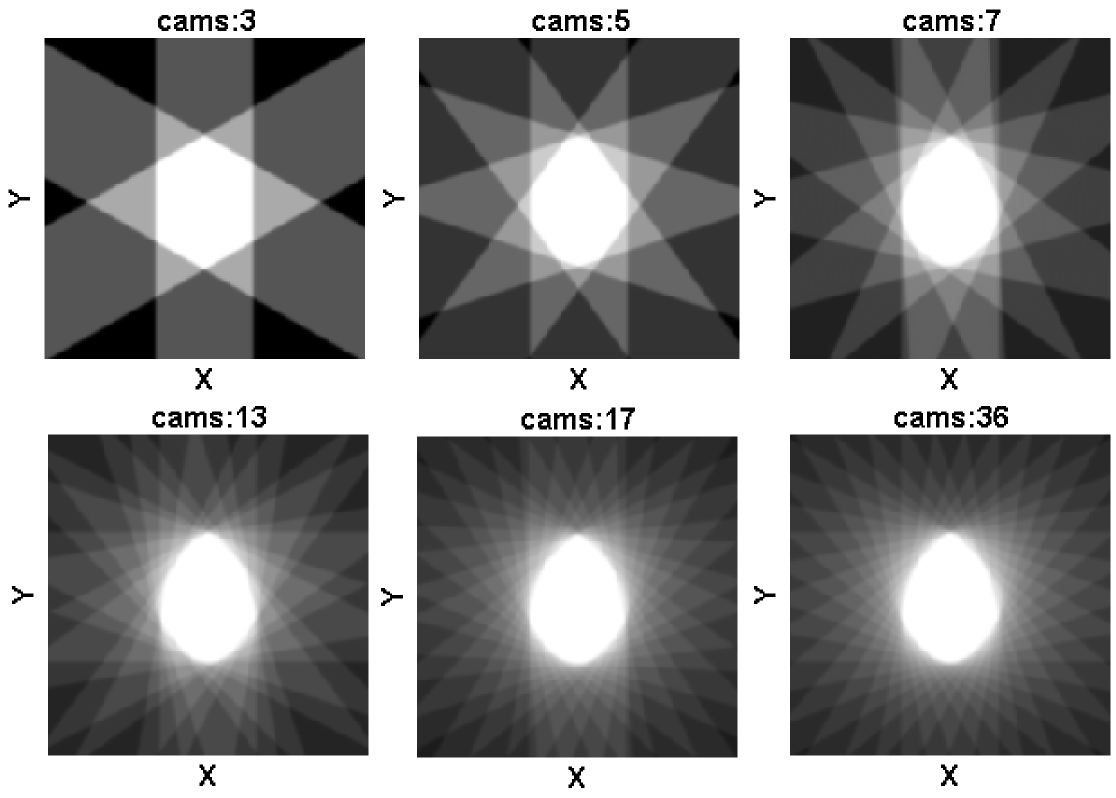


Figure 4.9: 2D KDE probability density variations of the selected slice in Figure 4.2 as changing the number of cameras.

The minimum bandwidth reflects the uncertainty on the pixel resolution $h_{min} = 1$. $h_{max} = 10$ has been chosen experimentally. In the case of fewer cameras being used, the mean-shift iteration may be trapped in a local maximum even if the simulated approach described above is used. These local maxima can be avoided by re-increasing the bandwidth, since the value of the density on the object $\max \hat{p}(\mathbf{x})$ is a known value, which can be calculated by the number of maximum intersecting rays.

Several guess points are randomly created in the spatial domain around the object, and these are moved using the simulated mean shift algorithm until convergence. The contour of the object is then inferred by connecting the closest points.

4.4.2 Alternative Method to Meanshift

As an alternative to the mean shift iterations used to infer the shape when computer memory is available, the kernel density estimates can be computed on a regular grid spanning the 3D space. Positions on the grid having a density value above a certain threshold are then kept, giving a likely indication of where the object is located. It is then possible to place the point around the outside of the object as illustrated in Figure 4.10. The more points that are sampled, the more computation time and memory are required. However, if the sampling is too sparse, then there is a risk of only areas of low probability being covered, and the object being missed. Figure 4.11 shows a set of result points which are sampled densely to estimate 180 points with threshold=0.8 and 36 camera views.

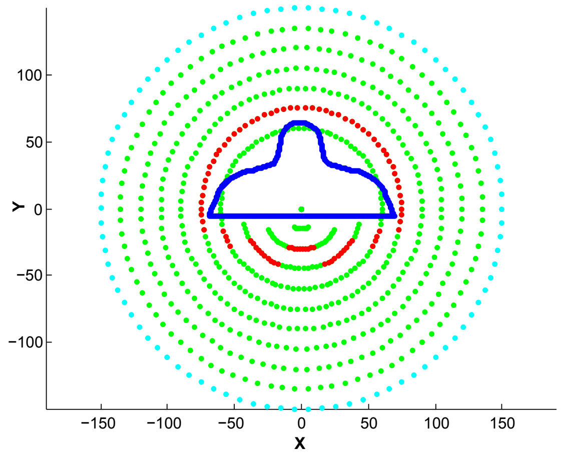


Figure 4.10: Using a regular sampling grid to reconstruct the object: result (red), initial points (cyan), sampled points (green), and ground-truth (blue).

4.5 Experiments

Section 4.5.1 presents the experimental settings to evaluate the methods. Qualitative and quantitative assessments are presented respectively in sections 4.5.2 and 4.5.3.

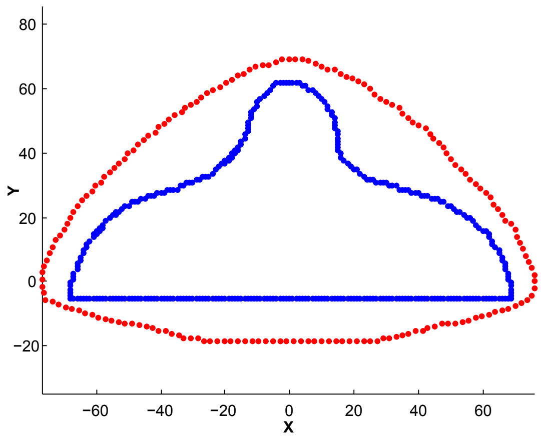


Figure 4.11: Final result (red) (using threshold=0.8 and 36 camera views) and ground-truth (blue).

4.5.1 Database

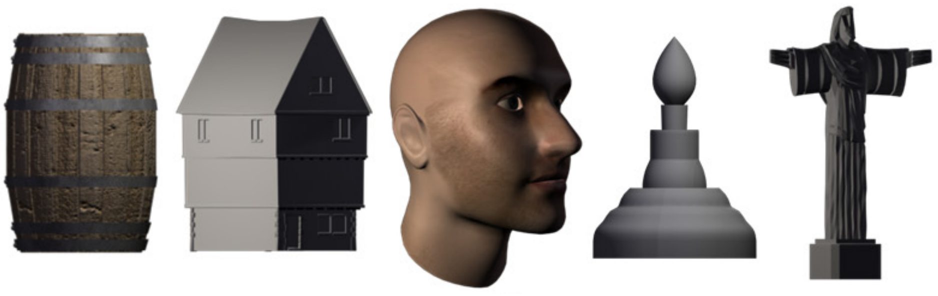


Figure 4.12: 3D objects in the database.

To test our methods, a database with a ground truth was generated using Autodesk 3ds Max (see examples Figure 4.12). The database consists of ten mesh objects (barrel, bulb, head, house, hydrant, lamp, push-pin, sculpture, service bell and warning marker). Each object was orthographically projected into 360 image planes equally spaced horizontally around the object (equivalent setting as a turning table in Pintavirooj and Sangworasil (58)) as well as spaced spherically. Different views of a head from the horizontal and spherical camera settings are illustrated in Figure 4.13 and 4.14. The camera locations are illustrated in Figure 4.15 and 4.16. The proposed 2D kernel density estimates and the Inverse

Radon transform are tested using the database of horizontal different views. The proposed 3D kernel density estimates are validated using both horizontal and spherical views.

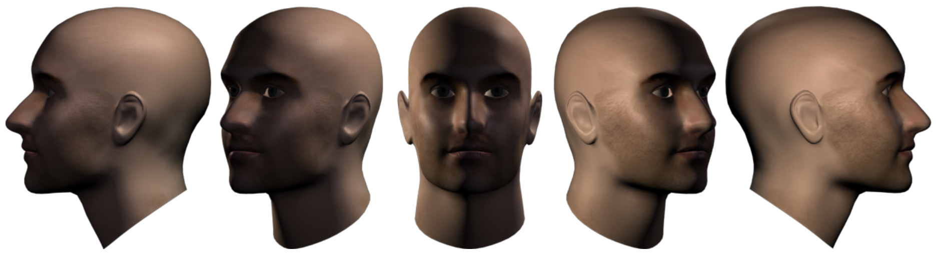


Figure 4.13: Horizontal different views of a head $[0°, 45°, 90°, 135°, 180°]$.

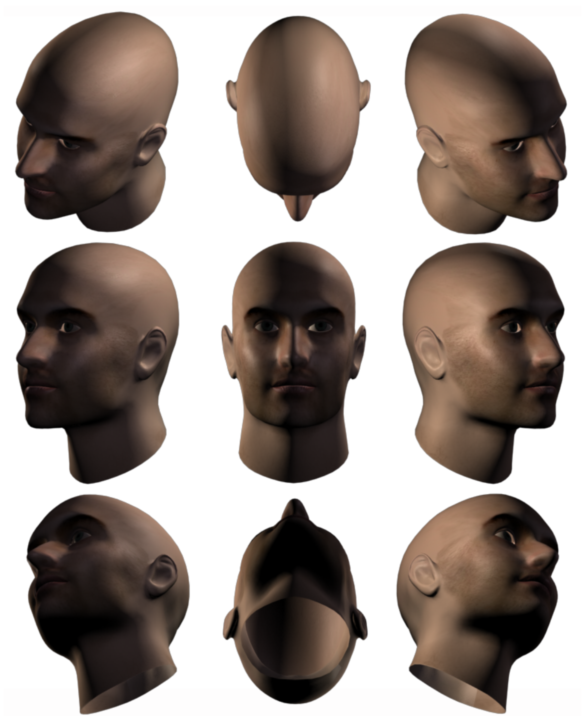


Figure 4.14: Spherical different views of a head.

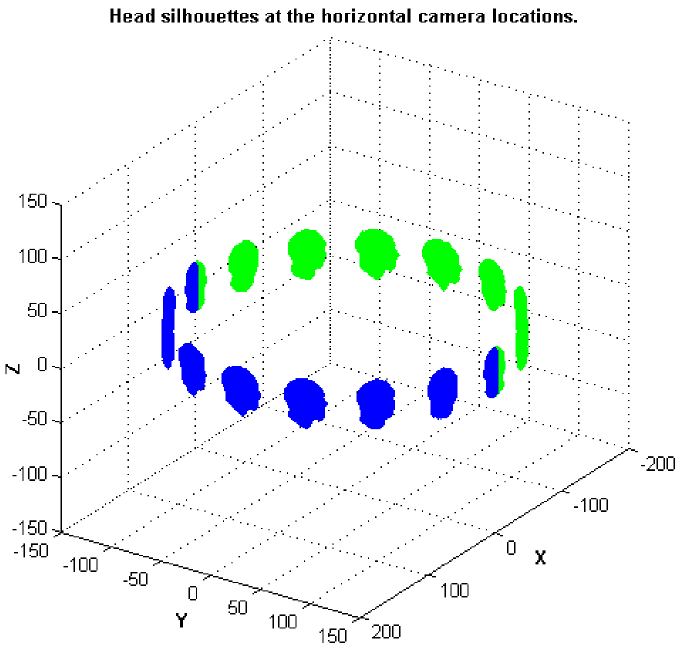


Figure 4.15: Horizontal camera locations(blue: $x \geqq 0$, green: $x < 0$).

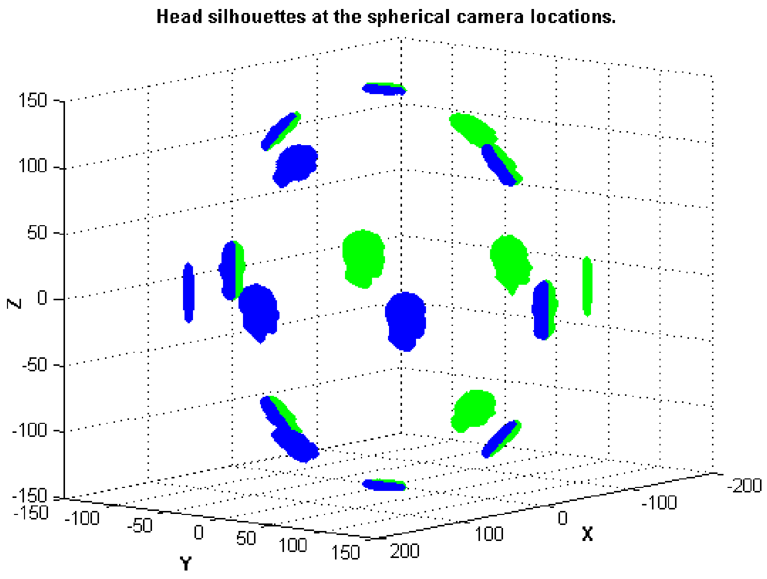


Figure 4.16: Spherical camera locations(blue: $x \geqq 0$, green: $x < 0$).

4.5.2 Qualitative evaluations

The four kernel density estimates are carried out using silhouettes and colour information from different camera settings. The number of horizontal and spherical cameras varies from 4 to 36 and from 6 to 38, respectively.

2D kernel density estimates, as described in Section 4.1, use horizontally located cameras. Initial 2D points (x, y) in the 2D slices are randomly created around the object of interest and then iteratively move toward the contour of the 2D slice of the 3D object using the mean shift algorithm. Figure 4.17 presents the

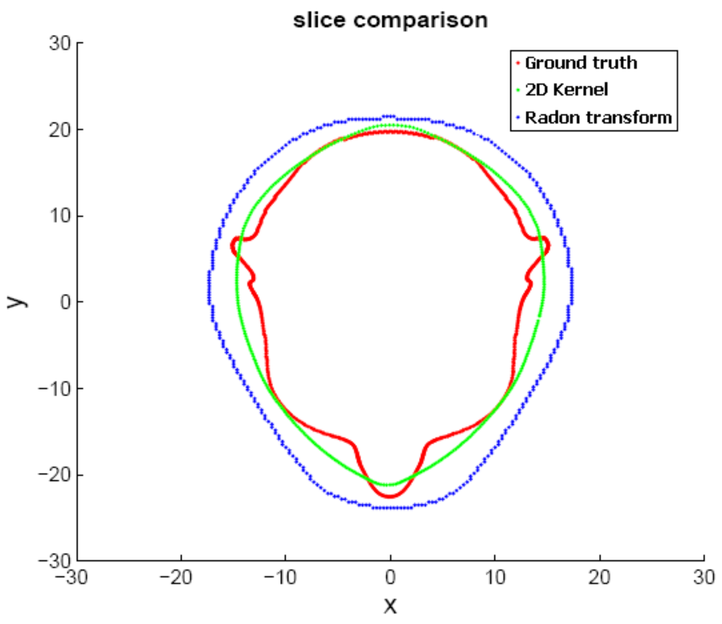


Figure 4.17: Slice reconstruction of a head (using 36 views).

contour of a head estimated by the 2D kernel density estimate (green) and the Inverse Radon Transform (blue) in a slice of a 3D head compared to the ground truth (red). Both estimates are convex approximation to the reference and the Meanshift reconstruction is closer to the ground truth. Figure 4.18 presents several 3D objects and their reconstructions using the 2D kernel density estimate with the Meanshift from 36 camera views of the horizontal camera setting (the 3D reconstruction is computed by stacking all 2D slices along the Z-axis).

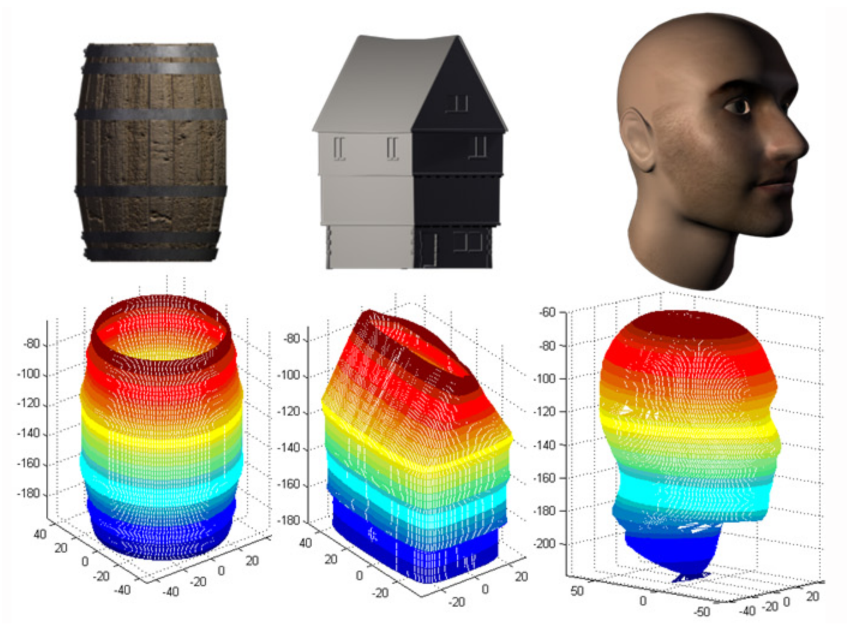


Figure 4.18: Ground-truth models (top row) and its reconstruction results using 2D kernel density estimates with 36 horizontal cameras(bottom row).

For the 3D kernel density estimate presented in Section 4.2, each object is reconstructed using two different camera settings of the horizontally and spherically located. Initial 3D points (x, y, z) are randomly scattered around an object of interest and then converged toward the contour of the 3D object using the Meanshift. The resultant 3D reconstruction is shown in Figure 4.19.

The extensions of 2D and 3D kernel density estimates to include colour information presented in Section 4.3 are computed using the same settings as previous 2D and 3D kernel density estimates. The estimated colour only contains rg chromaticity information, and therefore artificial intensity information is added to visualise the 3D reconstructions shown in Figure 4.20. Further 3D reconstruction results can be found in Appendix B.

4.5.3 Quantitative Evaluation

For the purpose of validation, these reconstructions must be compared with those computed using the inverse Radon transform. The 3D surface of each object in the database is estimated from the silhouette images using the methods proposed

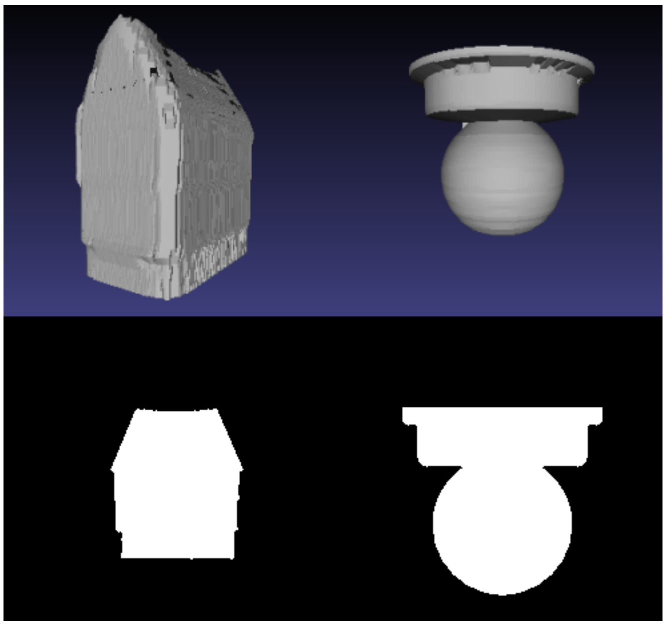


Figure 4.19: 3D kernel density estimate results from silhouettes of 36 horizontal cameras (left column) and 38 spherical cameras (right column). One of the silhouettes used for the reconstructions is illustrated in the bottom row.

here and the inverse Radon transform (computed by the function *iradon* in Matlab and associated with the canny edge detector for finding the maxima), as described in Pintavirooj and Sangworasil (58). The filtered back-projection algorithm in Kak and Slaney (34) is used for *iradon* function (cubic interpolation and ramp filter used in the back projection) to operate the inverse Radon transform. The original mesh object is used as the ground truth.

Radial distance is used to assess the quality of the reconstruction between the reconstructed 3D point clouds and their ground truth. Figure 4.21 shows these distances computed for several sets of cameras (4 - 36) of cameras, which are both horizontally and spherically located. The plot (in Figure 4.21) shows the average radial distance computed over 10 objects. It should be noted that for each camera (abscissa in Figure 4.21), the experiment is repeated several times on each object by randomly selecting equally spaced camera views. This is done to remove any effect caused by some particular camera views being more informative than others. As can be seen in Figure 4.21, the reconstruction of the

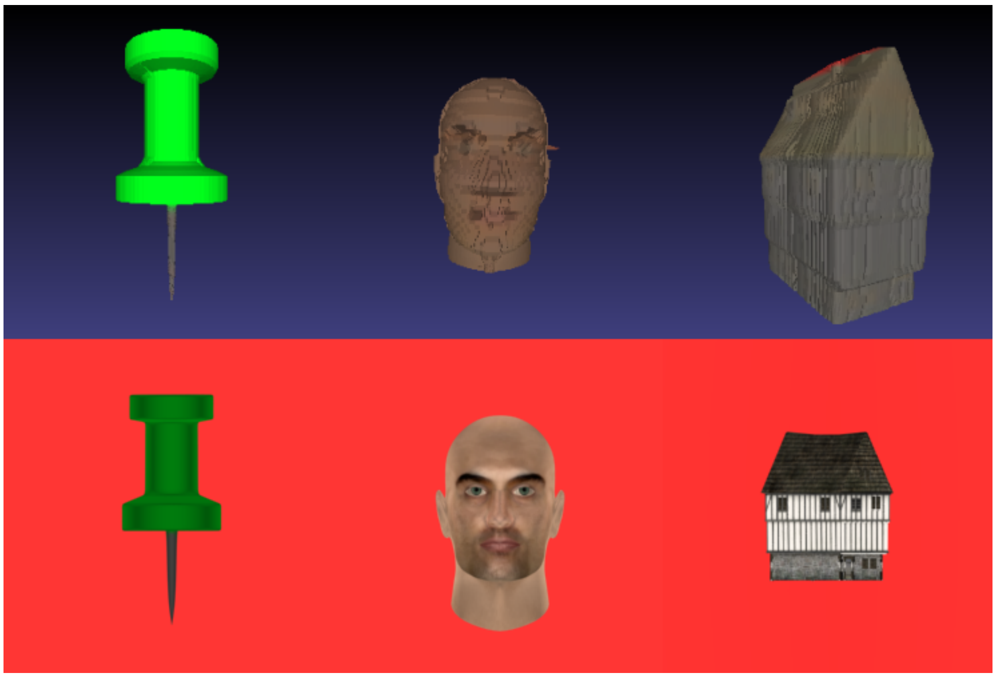


Figure 4.20: The results of the colour extensions from left to right using 2D KDE using (silhouettes+colour) and (36 horizontal cameras), 3D KDE using (silhouettes+colour) and (36 horizontal cameras) and 3D KDE using (silhouettes+colour) and (38 spherical cameras).

proposed kernel density estimates outperforms the inverse Radon transform-based one (the distance to the reference is smaller) and as expected for those methods, the distance decreases up to a point as more camera views are available. All graphs have a similar pattern as the number of cameras changes. However, the methods presented here are more accurate, and when the distance is close to nearly 0.8, the addition of more cameras does not improve the accuracy. This indicates the estimate calculated in this reconstruction is very close to the true visual hull (that is a convex approximation to the true shape). Reconstructions using our proposed methods are slightly better when silhouettes only are used without adding any colour information. This is because the colour initialisation is not entirely accurate, even if the results are very close. Details of how colour information can help this modelling process will be presented in Chapter 5.

More experiments are carried out using noisy images to test noise effects

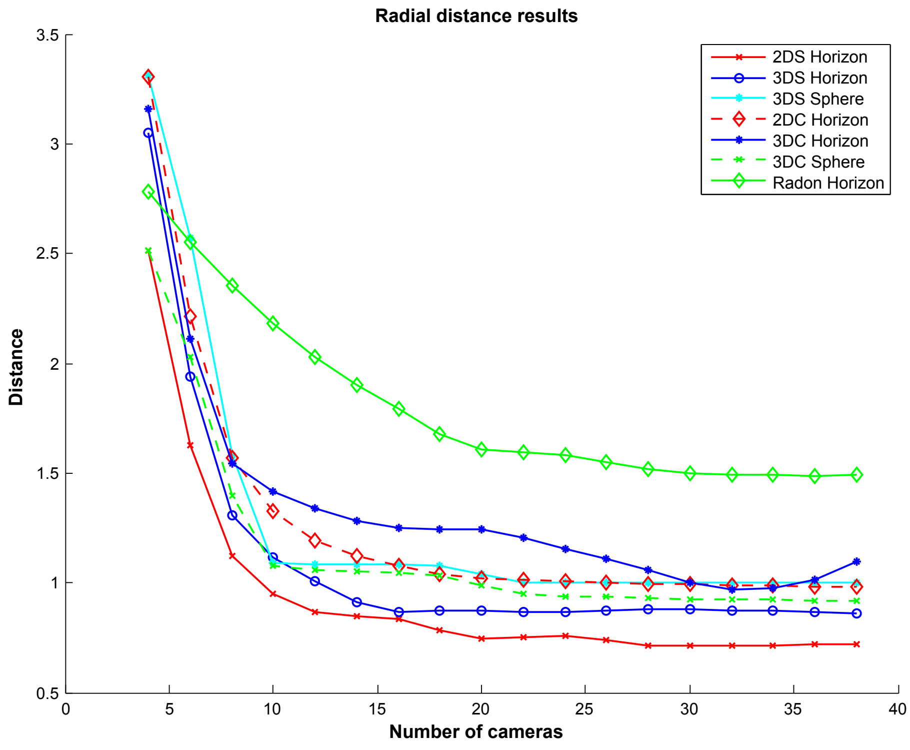


Figure 4.21: Radial distance plots using all objects (2DS: 2D KDE using silhouettes, 3DS: 3D KDE using silhouettes, 2DC: 2D KDE using (silhouettes+colour), 3DC: 3D KDE using (silhouettes+colour), Radon: the inverse Radon transform-based approach using silhouettes, Horizon: a horizontal camera setting and Sphere: a spherical camera setting).

in terms of the robustness of the estimation method (3D kernel density estimate). Four objects were used (barrel, hydrant, sculpture and head) and variable amounts of ‘salt and pepper’ noise were added to the silhouette images. 1, 5, 10, 15 and 20% of the noise was added in advance of the reconstruction, and 20% noise is illustrated in Figure 4.23. The results show that while the performance decreases with increasing noise, the method that uses the 3D kernel density estimate still manages to produce accurate reconstructions. This indicates that the method proposed in this chapter is robust to noise (see Figures 4.22).

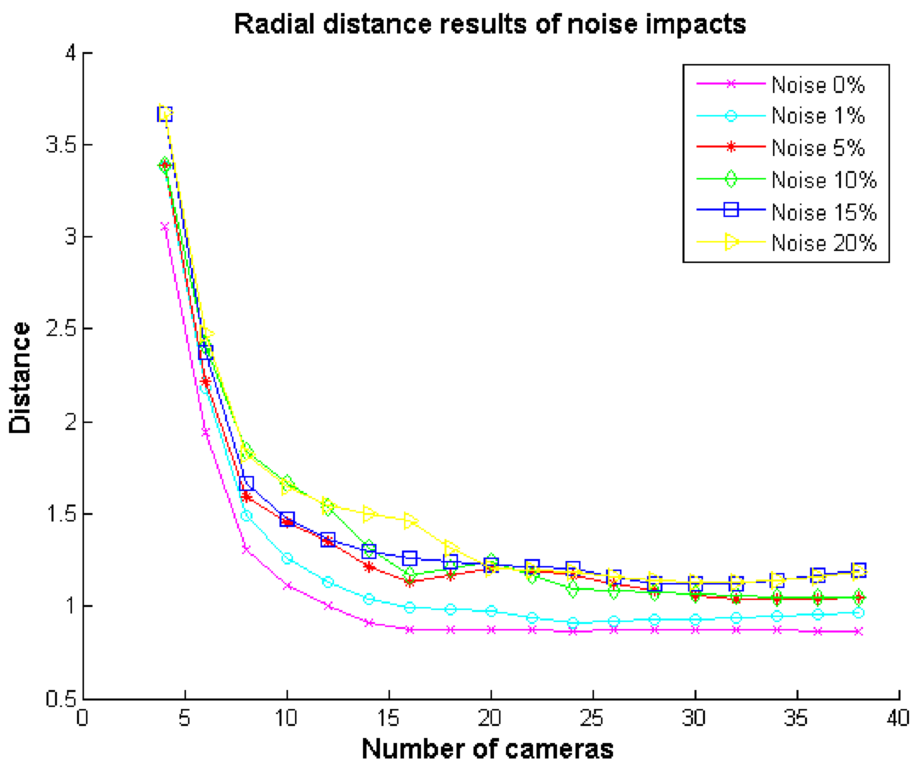


Figure 4.22: Radial distance plot from noisy silhouettes using Meanshift with a 3D kernel density estimate.

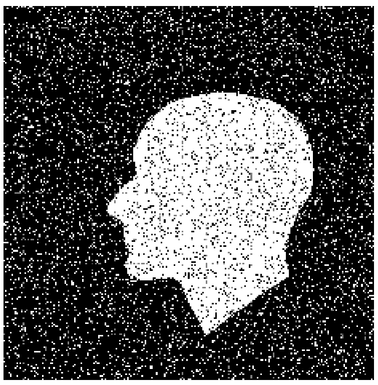


Figure 4.23: 20% 'salt and pepper' noise.

4.6 Conclusion

This chapter has presented four modellings to estimate 3D shapes from the images of multiple views by using 2D silhouette images and colour information. This approach is based on a new kernel density estimate of the density function of the spatial position with colour information and its corresponding mean shift algorithm for finding its maxima.

Experimental results show that these new statistical approaches are more accurate than the Radon transform-based approach where the inverse transform is computed using the filtered back-projection algorithm. The experimental results also indicate that the methods are robust to noise, as they still produces solid reconstructions even if the silhouettes are contaminated with up to 20% 'salt and pepper' noise.

For the purposes of testing, it is assumed that the cameras were orthographic leading to a modelling by kernel density estimates that can be optimised using the Meanshift. The framework presented in this chapter has recently been extended to the use of pin-hole cameras. In this case, the Newton-Raphson algorithm is used (64). One important benefit of this framework is that the resulting iterative algorithms are suitable for parallel architectures (see Section 2.2.3) and have low memory requirements.

The modellings introduced in this chapter do not take into account any prior information about the object in view. Indeed inference is performed using only the observations available from the cameras, and our modellings can be considered to be likelihoods. In the case of using only silhouette information, concave regions cannot be recovered with our kernel-based modelling. The following chapter will go on to investigate the benefit of adding a prior information regarding the object in view, to further improve the reconstruction process.

Chapter 5

Prior for Kernel Density Estimates

Chapter 4 introduced several kernel density estimates (KDEs) to reconstruct a photo-realistic 3D model using silhouettes and colour information recorded from a multi-view system. These modellings can be assumed to be likelihood functions since no prior information about the object in view is taken into consideration. Relying on observation alone can produce poor results when the recorded images are of low resolution, or when only a small number of views are available (such as when observations are scarce and/or low quality). Furthermore, when using only silhouettes, the object concavities are not recovered. Hence, in order to improve the reconstruction, we extend the kernel density estimates from Chapter 4 to take into account a prior for the shape.

Prior information for shape or texture has been proposed with, for instance, 3D morphable models that contain 3D surface shape or texture or both (5; 16; 17; 22; 51). Most recent methods of 3D face reconstruction (10; 52; 82) using morphable models analyse high quality face images where 2D feature points representing face components (eyes, nose, mouth, face contour, etc.) can be well extracted to facilitate the process of matching with the morphable template. As a consequence of this fitting process, these methods are not well applied to lower quality images directly (see Appendix F).

In section 5.1 we describe a novel shape description in 2D slices that will be used as prior information for accurately reconstructing 3D rigid objects. The prior is created by adaptively updating a k-nearest neighbour-based method. A posterior function is then proposed by combining the prior with the likelihoods defined in Chapter 4 (see Section 5.1.3). Inference from this posterior is performed using the mean shift algorithm and a Gaussian shape stack to avoid local solutions. The results from the k-nearest neighbour method are compared to the Principal Components Analysis (PCA)-based method (described in section 5.2). The results of multiple experiments in 2D and 3D reconstructions are presented in Section 5.3 and summarised in Section 5.4.

5.1 A Prior for Shapes

A descriptor for 2D shapes is presented in Section 5.1.1. In Section 5.1.2, comparison between shapes is then performed using the absolute distance (L1 norm) between descriptors and, we propose to reconstruct any new shape as a linear combination of the K-nearest neighbours (as defined with the chosen metric). The prior is modelled using this descriptor, and the adaptive k-nearest neighbour algorithm is introduced for reconstruction (Section 5.1.3). The prior is combined with the likelihood, and the posterior is optimised using the mean shift algorithm (see section 5.1.4). To avoid local solutions, a Gaussian stack (Section 5.1.5) was introduced.

5.1.1 Shape Descriptor

This section introduces a shape descriptor along with the metric used compare shapes. In 2D, a shape consists of a sequence of points that define a connected contour, as shown in Figure 5.1. To describe the shape, the points are chosen uniformly along the contour in an anti-clockwise direction, as illustrated in Figure 5.2. The shape descriptor presented here contains not only the list of 2D points $\{(x_i, y_i)\}$, but also the angles $\{\alpha_i\}$ calculated at each location with its two nearest neighbour points (see Figure 5.3).

The shape descriptor of each point $\mathbf{x}_i = (x_i, y_i)$ has α_i computed by a function

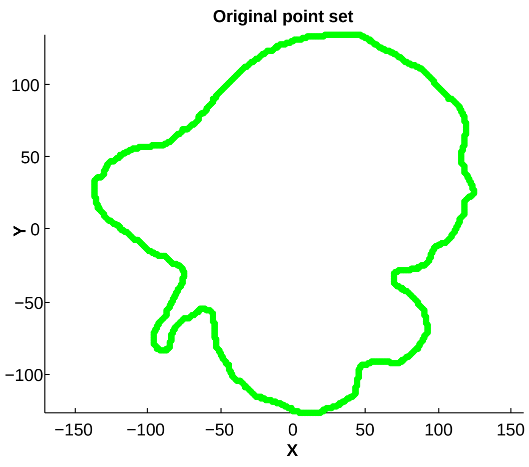


Figure 5.1: Original point set from ALOI database (29).

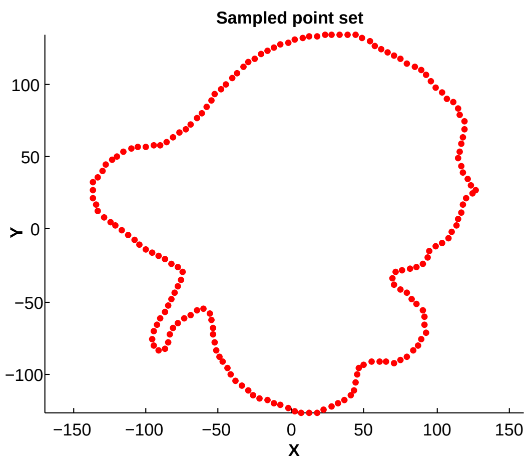


Figure 5.2: Sampled point set ($M = 180$ points).

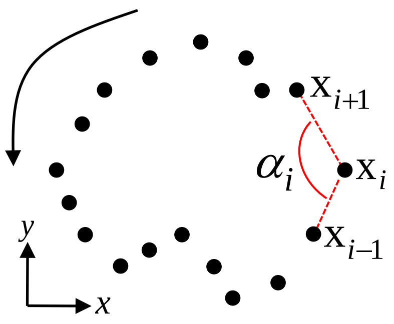


Figure 5.3: 2D points and angles for the shape descriptor.

f as follow:

$$f(\mathbf{X}) = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_M \end{bmatrix} \quad \text{with} \quad \mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \vdots \\ \mathbf{x}_M \end{bmatrix} \tag{5.1}$$

where M is the number of points sampled to describe the shape. $\mathbf{X}$ and $f(\mathbf{X})$ are not invariant to rotation i.e. choosing a different starting point $\mathbf{x}_1 = (x_1, y_1)$ on the shape will lead to other vectors $\mathbf{X}'$ and $f(\mathbf{X}')$ that will be cyclic permutation of $\mathbf{X}$ and $f(\mathbf{X})$. Note that the representation $f(\mathbf{X})$ is however invariant to scale changes on $\mathbf{X}$.

5.1.2 Shape Prior Approximation based on K-nearest neighbours Method

Having a new shape $\mathbf{X}$ and a training database of N shape exemplars $\{\mathbf{X}_1^e, \mathbf{X}_2^e, \cdots, \mathbf{X}_N^e\}$, the next step is to find the best K exemplars to define a prior for $\mathbf{X}$. Note that exemplar shapes are normalised by subtracting their respective mean of point coordinates, and by dividing by their respective variance. This is a standard pre-processing step before a representation invariant to translation and scale in the exemplar set.

We define the following distance metric between two shapes $\mathbf{X}$ and $\mathbf{Y}$ both represented by M points:

$$d(\mathbf{X}, \mathbf{Y}) = \sum_{i=0}^{M} \frac{|\alpha_i^{\mathbf{X}} - \alpha_i^{\mathbf{Y}}|}{\tau_s} \tag{5.2}$$

where τ_s is positive scalar for the normalisation. This metric is a weighted absolute distance between $f(\mathbf{X})$ and $f(\mathbf{Y})$. We define our prior by selecting the K exemplars of the training database that will be at the shortest distance of a shape $\mathbf{X}$. To be insensitive to rotation, we also consider all cyclic permutation of the exemplars. For instance, considering the first exemplar available in the training database noted $\mathbf{X}_1^e$ we find its cyclic permutation m (noted $\mathbf{X}_1^{e(m)}$) having the minimum distance $d(\mathbf{X}, \mathbf{X}_1^{e(m)})$ defined by:

$$\hat{m}_1 = \arg\min_m \{d(\mathbf{X}, \mathbf{X}_1^{e(m)})\} \tag{5.3}$$

Having computed all best distances between $\mathbf{X}$ and the N exemplars:

$$\{d(\mathbf{X}, \mathbf{X}_1^{e(\hat{m}_1)}), d(\mathbf{X}, \mathbf{X}_2^{e(\hat{m}_2)}), \cdots, d(\mathbf{X}, \mathbf{X}_N^{e(\hat{m}_N)})\} \tag{5.4}$$

The K exemplars with the shortest distances are then selected. Assuming the indexes $\{i_1, i_2, \cdots, i_K\}$ indicates the K nearest neighbours, then $\mathbf{X}$ can be reconstructed efficiently by $\hat{\mathbf{X}}$ as follows:

$$\hat{\mathbf{X}} = \sum_{k=1}^{K} \omega_k \, \mathbf{X}_{i_k}^{e(\hat{m}_{i_k})}, \tag{5.5}$$

where $\{\omega_k\}_{k=1,\cdots,K}$ are the weights and $\{\mathbf{X}_{i_k}^{e(\hat{m}_{i_k})}\}_{k=1,2,\cdots,K}$ are the selected K-nearest neighbours. The weight is calculated as follows:

$$\omega_k = \frac{1}{(K-1)}\left(1 - \frac{d_k}{d_{sum}}\right). \tag{5.6}$$

where $d_{sum} = \sum_{k=1}^{K} d(\mathbf{X}, \mathbf{X}_{i_k}^{e(\hat{m}_{i_k})})$ and $d_k = d(\mathbf{X}, \mathbf{X}_{i_k}^{e(\hat{m}_{i_k})})$. Note that by definition, the weights sum to 1, $\sum_{k=1}^{K} \omega_k = 1$. $\hat{\mathbf{X}}$ will be used as a prior to update a current estimate of the observed shape $\mathbf{X}$.

Figure 5.4 shows an example of reconstruction: the test image (corresponding to the angle 75°) is reconstructed using the two nearest exemplars ($K = 2$) at angles 60° and 90° (see the database used in Figure 5.25).

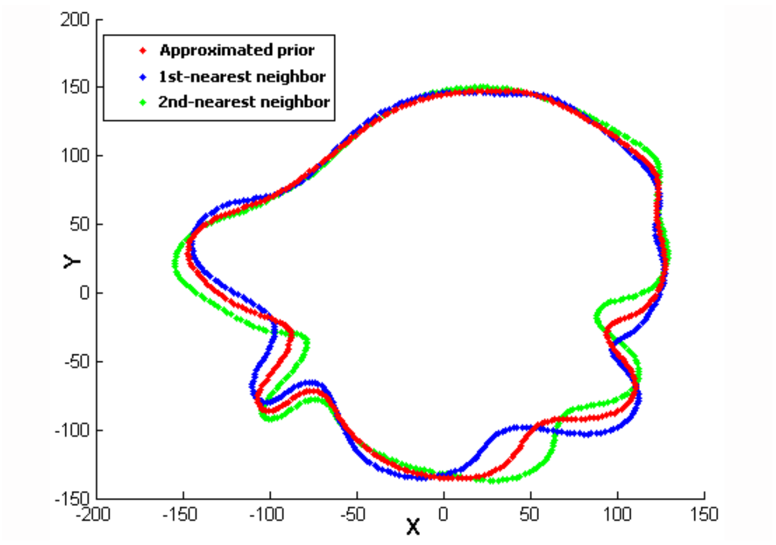


Figure 5.4: An example of the prior approximation using k-nearest neighbours ($K = 2$). The nearest neighbours illustrated by blue and green colours are selected by the distance metric (equation 5.2). Distances are calculated between an input test shape (75°) and a set of exemplars as shown in Figure 5.25. The exemplars 60° and 90° (from the training set $\mathcal{S}_{prior} = [1°, 30°, 60°, 90°]$ are chosen in this example and the prior (red colour) is approximated by the weighted summation in equation 5.5.

5.1.3 Shape Prior Modelling

Having a current guess of the shape noted $\mathbf{X}^{(t)}$, which can be estimated using one of the likelihood estimates $\hat{p}_{lik}$ defined with kernel mixtures in Chapter 4, we can compute a reconstruction $\hat{\mathbf{X}}^{(t)}$ using the training database of exemplars $\mathcal{S}_{prior}$. We model a prior using $\hat{\mathbf{X}}^{(t)}$ to allow for the estimation of a refined shape noted $\mathbf{X}^{(t+1)} = \left[\mathbf{x}_1^{(t+1)}, \mathbf{x}_2^{(t+1)}, \cdots, \mathbf{x}_M^{(t+1)}\right]$. Each of the M points is updated individually. Consider the first point $\mathbf{x}_1^{(t+1)}$. From Chapter 4, the likelihood $\hat{p}_{lik}$ is modelled using the kernel density estimates, and the prior for $\mathbf{x}_1^{(t+1)}$ is modelled given both the points $\{\mathbf{x}_2^{(t)}, \mathbf{x}_3^{(t)}, \cdots, \mathbf{x}_M^{(t)}\}$ and the reconstruction points $\{\hat{\mathbf{x}}_1^{(t)}, \hat{\mathbf{x}}_2^{(t)}, \cdots, \hat{\mathbf{x}}_M^{(t)}\}$. Similarly to the likelihood, the prior is modelled using a kernel density estimate. The reconstruction $\hat{\mathbf{X}}^{(t)}$ is converted into angles $\{\theta_m\}$ such that θ_m corresponds to the slope of the line defined by $(\hat{\mathbf{x}}_1^{(t)}, \hat{\mathbf{x}}_m^{(t)})$ (i.e. θ_m is the orientation of the unitary vector $\vec{n}_m$ such that $\vec{n}_m^T(\hat{\mathbf{x}}_1^{(t)} - \hat{\mathbf{x}}_m^{(t)}) = 0$). We assume that $(\mathbf{x}_1^{(t+1)}, \mathbf{x}_m^{(t)})$ follows the same equation:

$$\vec{n}_m^T \mathbf{x}_1^{(t+1)} = \vec{n}_m^T \mathbf{x}_m^{(t)} \tag{5.7}$$

with $\vec{n}_m = (\cos(\theta_m), \sin(\theta_m))^T$, and the point $\mathbf{x}_m^{(t)}$ is used to define the constant necessary to define the line.

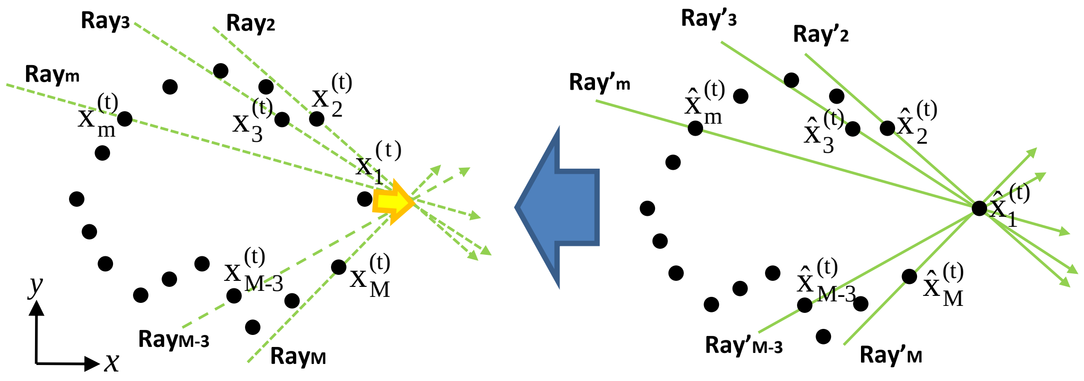


Figure 5.5: Illustration for $\hat{p}_{prior}(\mathbf{x}_1^{t+1})$. Left: current shape observation $\mathbf{X}^{(t)}$, right: current reconstruction $\hat{\mathbf{X}}^{(t)}$ for $\mathbf{X}^{(t)}$. The yellow arrow in the left means the direction of high prior probability.

Therefore given $\hat{\mathbf{X}}^{(t)}$ and $\{\mathbf{x}_m^{(t)}\}_{m=2,\cdots,M}$ the probability of $\mathbf{x}_1^{(t+1)}$ is modelled

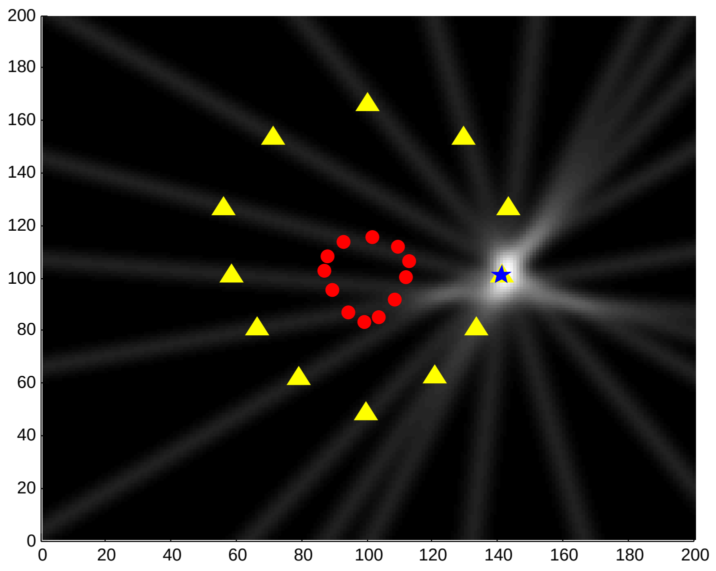


Figure 5.6: Kernel density estimate for the prior of $\mathbf{x}_i$ (blue and star marker) in $\mathbf{X}$ (yellow and triangle marker) generated by equation (5.10) with $h_p = 3$ and $M = 12$. The approximated shape prior $\hat{\mathbf{X}}$ is plotted by red and circle markers.

as follows:

$$\hat{p}_{prior}(\mathbf{x}_1^{(t+1)}|\hat{\mathbf{X}}^{(t)}, \{\mathbf{x}_m^{(t)}\}_{m=2,\cdots,M}) \propto \frac{1}{\sqrt{2\pi}(M-1)h_p} \sum_{m=2}^{M} \exp\left(-\frac{(\vec{n}_m^T(\mathbf{x}_1^{(t+1)} - \mathbf{x}_m^{(t)}))^2}{2h_p^2}\right), \tag{5.8}$$

where h_p is the bandwidth of the Gaussian kernel. The concept of $\hat{p}_{prior}(\mathbf{x}_1^{(t+1)})$ is illustrated in Figure 5.5. $\mathbf{x}_1^{(t+1)}$ is then updated by taking the max of the posterior:

$$\hat{p}_{post}(\mathbf{x}_1^{(t+1)}) \propto \arg\max \left\{\hat{p}_{lik}(\mathbf{x}_1^{(t+1)}) \times \hat{p}_{prior}(\mathbf{x}_1^{(t+1)})\right\} \tag{5.9}$$

The mean shift algorithm is used to maximise the posterior p_{post} (see Section 5.1.4). The prior for $\mathbf{x}_i^{(t+1)}$ is generalised by:

$$\hat{p}_{prior}(\mathbf{x}_i^{(t+1)}|\hat{\mathbf{X}}^{(t)}, \{\mathbf{x}_m^{(t)}\}_{m=1,\cdots,M,m\neq i}) \propto \frac{1}{\sqrt{2\pi}(M-1)h_p} \sum_{m=1,m\neq i}^{M} \exp\left(-\frac{(\vec{n}_m^T(\mathbf{x}_i^{(t+1)} - \mathbf{x}_m^{(t)}))^2}{2h_p^2}\right). \tag{5.10}$$

We use only slopes from the reconstruction and therefore this method is invariant to scale difference between the observed shape $\mathbf{X}^{(t)}$ and the normalised reconstruction $\hat{\mathbf{X}}^{(t)}$. Also, the method is invariant to translation because we use a point, $\mathbf{x}_m^{(t)}$ from the current shape observation to define the line equation. Figure 5.6 shows the $\hat{p}_{prior}(\mathbf{x}_i)$ generated by equation (5.10).

Remark. A simpler prior centred on the intersection of the $(M-1)$ rays could have been modelled using a simple Gaussian distribution. However the chosen KDE is more robust to the noisy data in $\mathbf{X}^{(t)}$.

5.1.4 Mean Shift Iteration

The posterior is optimised using a mean shift algorithm to find its maxima representing the estimate $\mathbf{X}^{(t+1)}$. The mean shift algorithm is performed by differentiating the posterior and equalling the result to zero. Detailed equations with respect to the mean shift iteration are described in Appendix C.1 and C.2.

Overall system flow is summarised in Figure 5.7. From initial point set $\mathbf{X}^{(0)}$, the iteration is repeated until all points in $\mathbf{X}^{(t)}$ have converged. If there is no difference between $\mathbf{X}^{(t)}$ and $\mathbf{X}^{(t+1)}$, the iteration is stopped. The prior $\hat{\mathbf{X}}^{(t)}$ to estimate $\mathbf{X}^{(t+1)}$ is approximated by the k-nearest method using a current observation $\mathbf{X}^{(t)}$ (see Section 5.1.2). Through the iteration across time (t) the prior is adaptively updated. Then $\mathbf{X}^{(t+1)}$ is estimated by maximising the posterior.

One of the important advantages of the method presented here is that the prior is adaptively updated. It enables more accurate and intelligent estimation of a shape of interest by reconstructing it onto its nearest neighbours instead of using a fixed basis of functions (e.g. as computed by PCA).

Our proposed method requires the setting of five bandwidths, which are for the likelihood (h_s, h_r, h_g), the priors (h_p) and a Gaussian convolution (h_e) to smooth the shape curvatures of the training database. A minimum bandwidth of h_s, h_r, h_g and h_p was used to achieve the greatest accuracy possible, and hence those were set $h_s = h_r = h_g = h_p = 1$ reflecting the uncertainty on the pixel resolution. These bandwidths can be relaxed if the images are not of great quality. The following section will now explain Gaussian convolution h_e.

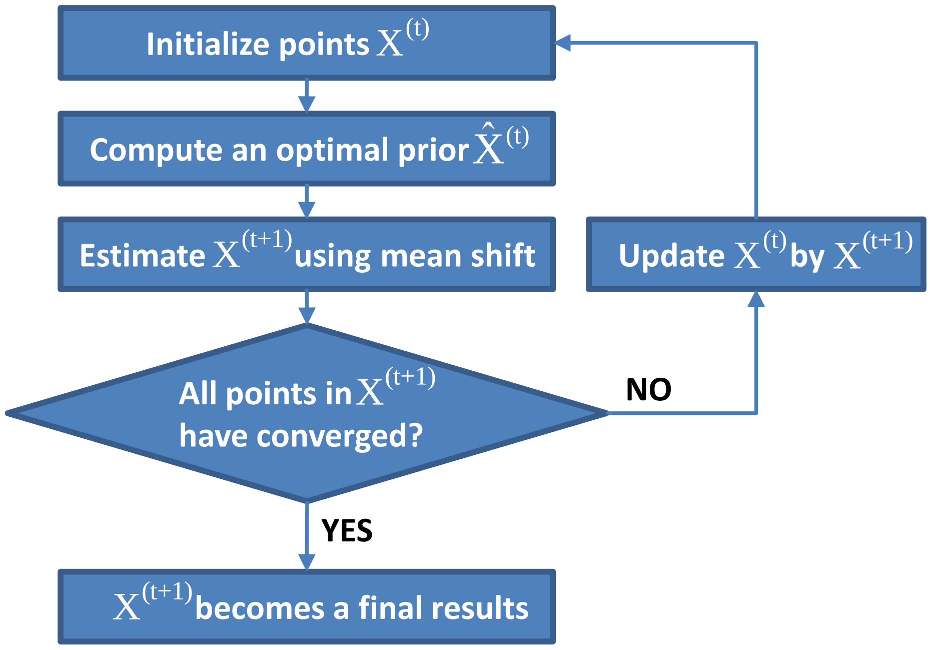

Initialize points $X^{(t)}$
Compute an optimal prior $\hat{X}^{(t)}$
Estimate $X^{(t+1)}$ using mean shift
Update $X^{(t)}$ by $X^{(t+1)}$
All points in $X^{(t+1)}$ have converged?
NO
YES
$X^{(t+1)}$ becomes a final results

Figure 5.7: System flow for refining a shape estimate using a prior.

5.1.5 Gaussian Shape Stack

The shape of the initial point set $\mathbf{X}^{(0)}$, which can be the result of the estimation using only likelihoods, as proposed in Chapter 4, is quite different from the exemplars in the training database. In order to converge iteratively towards a satisfactory solution, even if the starting guess $\mathbf{X}^{(0)}$ is far from it, care must be taken in the prior approximation process, as already outlined in section 5.1.2. In particular, the selection of the K nearest neighbours may not be the best initial basis of reconstruction functions. Indeed, finding the best cyclic permutation could create a mismatch. To avoid this problem, a Gaussian shape stack, whose concept is introduced in Lefebvre and Hoppe (41), is constructed. The Gaussian stack is constructed by smoothing the shapes in the prior sets using increasing bandwidths (noted $h_e^{(t)}$) without downsampling the shapes as it is usually done in Gaussian pyramids. This stack is computed using the convolution with a Gaussian (with bandwidth $h_e^{(t)}$) on all exemplars $\{\mathbf{X}_i^e\}_{i=1,\cdots,N}$ in the training database from large to small bandwidth as a smoothing factor, then the best cyclic permutations $\{\mathbf{X}_i^{e(\hat{m}_i)}\}_{i=1,\cdots,N}$ are chosen. This procedure allows the achievement of a coarse-to-fine strategy, as shown in Figure 5.8. The reconstruction at time t,

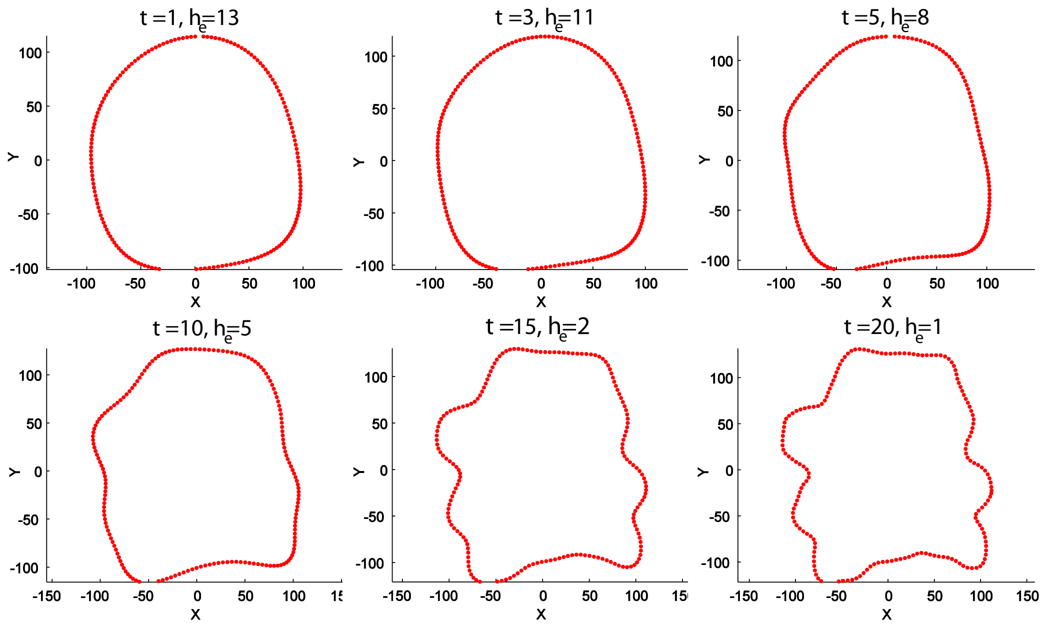


Figure 5.8: The variations of one of the exemplars $\mathbf{X}^{e(t)}$ in ALOI training database along the iteration t, smoothed using $h_e^{(t)}$.

$\hat{\mathbf{X}}^{(t)}$, that is approximated from the selected exemplars in the training database, is then iteratively refined to get more accurate shape estimation results. The bandwidth $h_e^{(t)}$ decreases at each iteration of the mean shift algorithm as follows:

$$h_e^{(t)} = \alpha^t \; h_{max} \text{ until } h_e = h_{min} \text{ with } \alpha = 0.9 \tag{5.11}$$

where, $h_{max} = 13$, $h_{min} = 1$, h_{max} is selected experimentally to be as similar as possible to the optimal initial point set. Figure 5.9 shows the shape estimation results and the approximated shape prior across time t.

5.1.6 Considering Colour Information

Consider for a moment the shape descriptor in Section 5.1.1. This descriptor can be extended to take colour information into account. For instance, a 2D head slice and its sampled points containing colour information are illustrated in Figure 5.10 and 5.11. The shape descriptor of each point $\mathbf{x}_i = (x_i, y_i, r_i, g_i)$ is

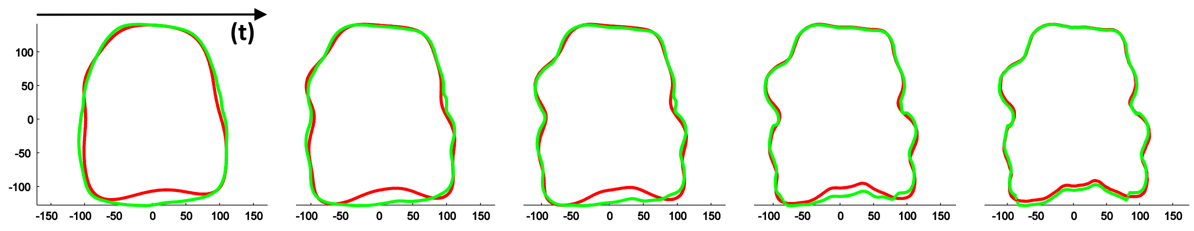
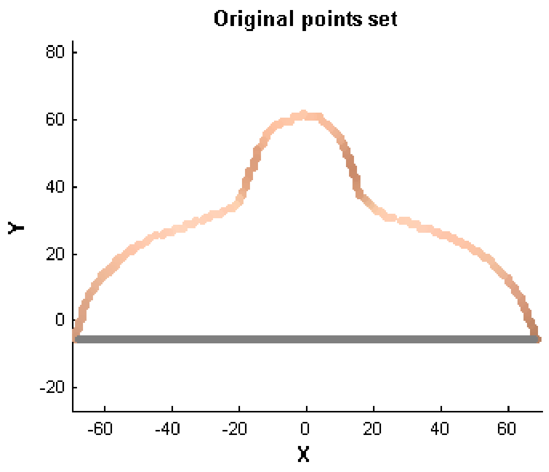


Figure 5.9: Shape estimation results along the iteration t (Green: $\mathbf{X}^{(t)}$, Red: $\hat{\mathbf{X}}^{(t)}$ approximated by the k-nearest neighbours method ($K = 2$) for which the exemplars are smoothed using $h_e^{(t)}$).



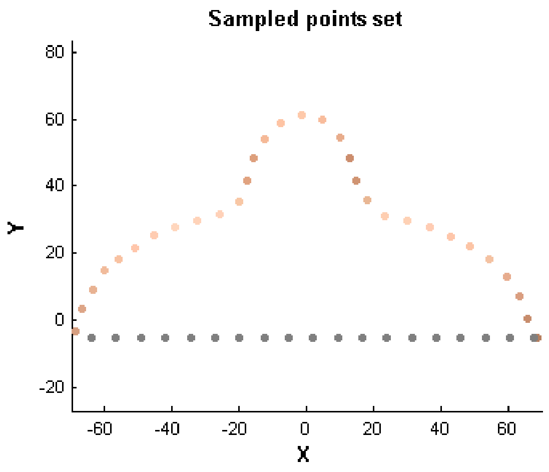


Figure 5.10: 2D head slice from 3D Basel face database (57)

Figure 5.11: Sampled 2D head slice with colour information $\mathbf{X}^e$

(α_i, r_i, g_i), which constitutes its angle and colour information (chromaticity -red and -green) interpolated by two nearest points. This is computed by a function f as follow:

$$f(\mathbf{X}) = \begin{bmatrix} \alpha_1 \\ r_1 \\ g_1 \\ \vdots \\ \alpha_M \\ r_M \\ g_M \end{bmatrix} \quad \text{with} \quad \mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_M \end{bmatrix} \tag{5.12}$$

where, M is the number of sampled points to describe the shape. Then, the distance metric in equation 5.2 can be replaced by:

$$d(\mathbf{X}, \mathbf{Y}) = \sum_{i=0}^{M} \frac{|\alpha_i^{\mathbf{X}} - \alpha_i^{\mathbf{Y}}|}{\tau_s} + \frac{|r_i^{\mathbf{X}} - r_i^{\mathbf{Y}}| + |g_i^{\mathbf{X}} - g_i^{\mathbf{Y}}|}{\tau_c}, \tag{5.13}$$

where (τ_s, τ_c) are normalisation factors. Using the same reconstruction method as that shown in equation 5.5, the reconstruction $\hat{\mathbf{X}}$ is computed with its colour information. When using likelihood functions that include colour information, the colours of the initial points are assigned using the colour of the reconstruction $\hat{\mathbf{X}}^{(0)}$ (e.g. see Figure 5.12). The Gaussian shape stack is also constructed with

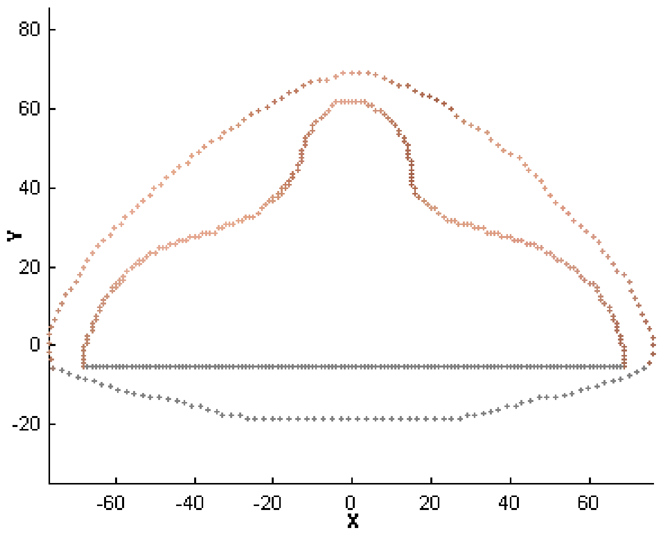


Figure 5.12: Colour assignment to the reconstruction $\hat{\mathbf{X}}^{(0)}$ compared to the ground-truth.

colour information as illustrated in Figure 5.13.

5.2 PCA-based Method

We compare our method based on the k-nearest neighbours algorithm with the Prinicipal Component Analysis (PCA)-based method. The PCA-based representation has been widely used to model shapes such as 3D faces called 3D eigenfaces (33; 85) and Active Appearance Model or Active Shape Model (16; 17; 22; 51).

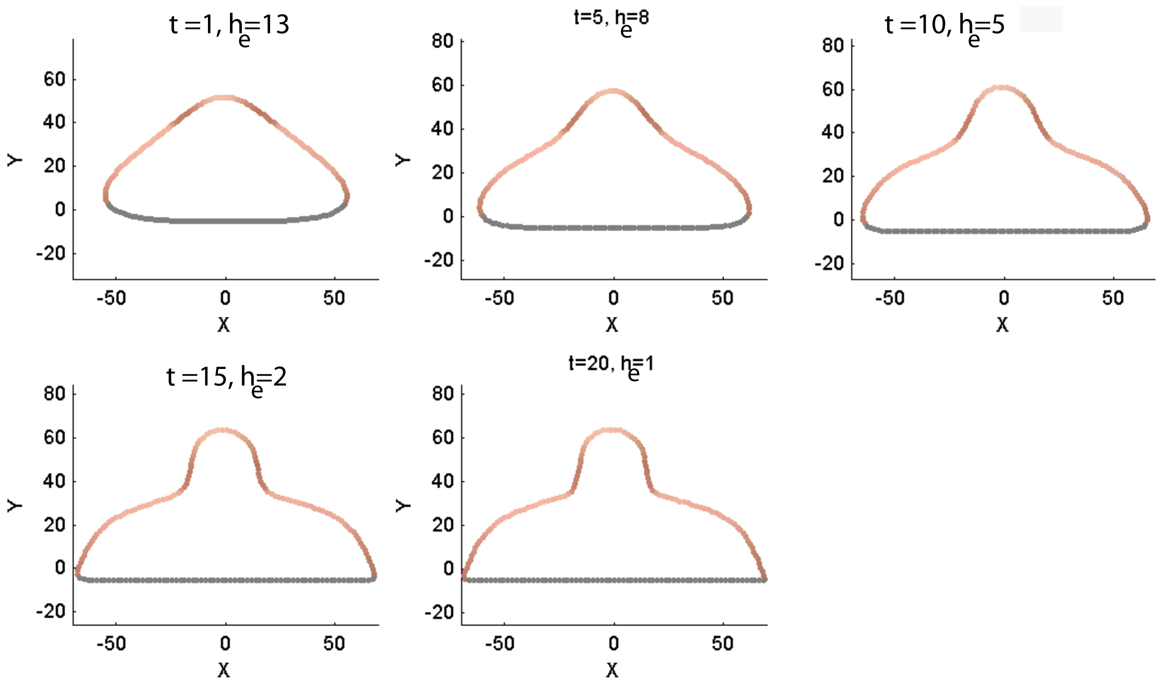


Figure 5.13: The variations of one of the exemplars ($\mathbf{X}^{e(t)}$) in 3D Basel face database (57) along the iteration t smoothed using $h_e^{(t)}$.

PCA enables approximation of the shape by a linear combination of eigenvectors calculated from the covariance matrix computed with the exemplars $\{\mathbf{X}_1^{e(\hat{m}_1)}, \mathbf{X}_2^{e(\hat{m}_2)}, \cdots, \mathbf{X}_N^{e(\hat{m}_N)}\}$. The eigenvectors can be regarded as a set of features characterising the variation between the shapes. The feature vector of the shape consists of the sequence of points and its colour information $\{\mathbf{X}_i^{e(\hat{m}_i)}\}_{i=1,2,\cdots,N}$ to represent the contour of an object of interest. The spatial coordinate (x, y) and chromaticity values (r, g) are used to create the feature vector, and therefore the dimension of the feature vector is $4 \times M$, where M is the number of points. In the training database, the 3D heads are translated to the origin and scaled for normalisation, then the hair and ears are removed to concentrate on only different frontal face shapes as shown in Figure 5.14.

Let the selected slices of the 3D heads be represented by $\{\mathbf{X}_i^{e(\hat{m}_i)}\}_{i=1,\cdots,N}$, where N is the number of the 3D heads in the training database and $\{\mathbf{X}_i^{e(\hat{m}_i)}\}$ is the best cyclic permutation of the slice $\mathbf{X}_i^e$. The mean shape, ψ, is calculated by:

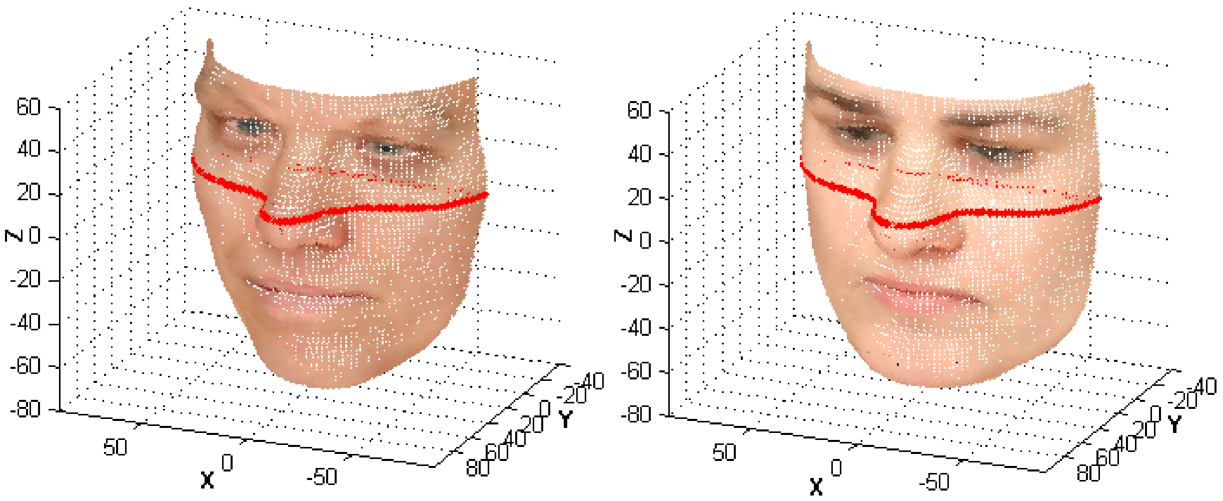


Figure 5.14: Examples of training database from the 3D Basel face model (57). (Red: a selected 2D slice)

$$\psi = \frac{1}{N}\sum_{i=1}^{N} X_i^{e(\hat{m}_i)} \tag{5.14}$$

Figure 5.15 shows the mean shape and colour (to visualise colour, full *RGB* is used instead of chromaticity values). $M = 180$ points are sampled from the slice contour to compute the mean shape. Then, the covariance matrix is computed by:

$$\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N}(X_i^{e(\hat{m}_i)} - \psi)(X_i^{e(\hat{m}_i)} - \psi)^T = \mathbf{A}\mathbf{A}^T \tag{5.15}$$

where $\mathbf{A} = [X_1^{e(\hat{m}_1)} - \psi, X_2^{e(\hat{m}_2)} - \psi, \cdots, X_N^{e(\hat{m}_N)} - \psi]$. The dimension of the covariance matrix is $4M \times 4M$. The first K eigenvectors associated with the K biggest eigenvalues are computed with singular value decomposition in Turk and Pentland (78). Let $\nu_i, i = 1, \cdots, K$ be the eigenvectors of $\mathbf{C}$. Assuming that the contour shapes are representative of the feature space, the resulting eigenvectors are sufficient to represent the subspace so that an arbitrary contour $\mathbf{X}$ can be represented by a linear combination of the eigenvectors (a weighted sum of the eigenvectors),

$$\hat{\mathbf{X}} \approx \psi + \sum_{i=1}^{K} \omega_i \; \nu_i \tag{5.16}$$

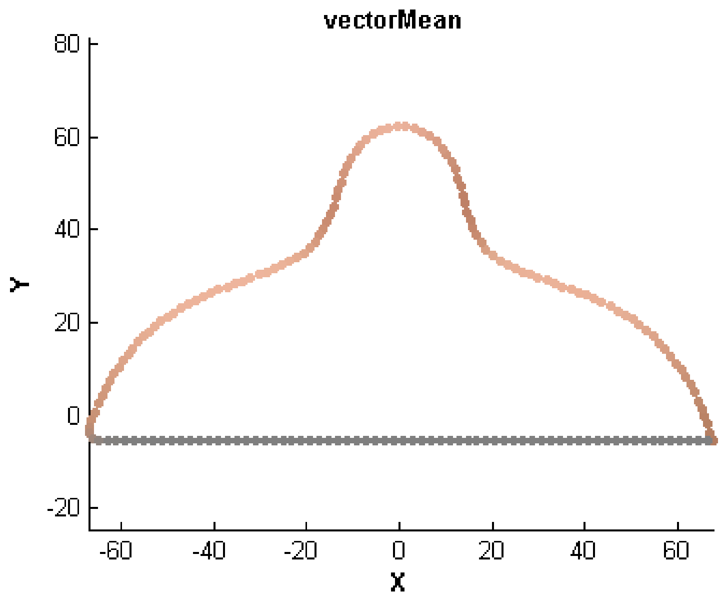


Figure 5.15: The mean shape of 35 head slices (from $N = 35$ individuals) and $M = 180$ points are used.

where ω_i is the weight of the eigenvector ν_i computed by:

$$\omega_i = (\mathbf{X} - \psi)^T \nu_i \tag{5.17}$$

The reconstruction $\hat{\mathbf{X}}$ is used to model the shape prior p_{prior} as presented in Section 5.1.3 as an alternative to our k-nearest neighbours algorithm. The disadvantage of the PCA-based method is that the normalisation step to remove the effects of translation and scaling is required for the current observation $\mathbf{X}$ to become as similar as possible to the training database $\{\mathbf{X}_i^{e(\hat{m}_i)}\}_{i=1,2,\cdots,N}$. In practice, accurate normalisation and alignment are very important to create the feature vectors for PCA, and this is not an easy problem to solve because accurate detection of the eyes, nose and mouth are required for a good alignment between faces. Note that our proposed method based on the k-nearest neighbours method and our shape descriptor does not require these normalisation steps between the observation and training database.

5.3 Experiments

Two sets of experiments were carried out to evaluate the proposed method:

- Section 5.3.1 presents reconstructions of faces in the 3D space. Note that the 3D faces are decomposed into a stack of horizontal 2D slices, and each slice has its own 2D prior.
- Section 5.3.2 presents additional results using an image database of objects captured on a turning table. The (colour) contours of the object are used in the experiment, and the observations (colour silhouettes) are computed by orthogonal projection of the coloured contours. This experiment is equivalent to the process carried out for one slice in the first experiment. This second database has more variability in the exemplars used.

5.3.1 Faces

Section 5.3.1.1 presents the database used in the experiment. Section 5.3.1.2 compares both approaches based on KNN and PCA. Section 5.3.1.4 explains how the full 3D face is reconstructed while inference has been performed independently on each 2D slice.

5.3.1.1 Database

We used the 3D Basel face model (57) which is composed of 200 registered faces, acquired with a structured light scanner shown in Figure 5.16. In addition, synthetic faces can be generated from random model coefficients as proposed by Paysan et al. (57). For our experiments, a total of 44 heads are created from the 3D Basel face model and translated to the origin. 9 heads are used for the test set $\mathcal{S}_{test}$ shown in Figure D.1 and 35 heads ($N = 35$) are for the training set $\mathcal{S}_{prior}$ of both k-nearest neighbours and PCA-based methods illustrated in Figure D.2.

To create silhouettes in multiple views, the 3D heads are projected in several directions using an orthographic projection. The silhouettes have foreground and background pixels as well as colour information on the foreground. This operation corresponds to computing the Radon transform from several directions (i.e. camera views) and thresholding these projections to produce binary silhouettes.

Figure 5.16: Examples in the Basel face models (57).

5.3.1.2 PCA vs KNN (2D)

The experiments described here investigate how well a prior and colour information may help to estimate concave regions, compared to methods which use only the likelihood. In order to do this, the kernel density estimates of the likelihood in Chapter 4 are first tested. The posteriors, which are formulated from the likelihoods and the approximated priors of k-nearest neighbours (KNN) and PCA-based methods, are then assessed. Remember that several likelihoods have been proposed and these can be divided into two categories: the ones using only silhouettes (shapes) (Section 4.1.2) and, the ones using both silhouettes (shapes) and colour information (Section 4.3.1). Combining these likelihoods with the two priors (PCA and KNN), the posteriors can be classified into four categories (two types of likelihoods combined with two types of priors). We defined the following methods below:

- likelihood using only silhouettes noted $\mathcal{L}_{shape}$,
- likelihood using silhouettes and colour noted $\mathcal{L}_{shape+colour}$,
- posterior with the likelihood using silhouettes and a shape prior approximated by the KNN method ($K = 23$), noted:

$$\mathcal{P}\mathrm{ost}_{shape,knns} \propto \mathcal{L}_{shape} \times \mathcal{P}\mathrm{rior}_{knns}$$

- posterior with the likelihood using silhouettes and colour, and a shape prior approximated by the KNN method ($K = 23$)

$$\mathcal{P}\mathrm{ost}_{shape+colour,knns} \propto \mathcal{L}_{shape+colour} \times \mathcal{P}\mathrm{rior}_{knns}$$

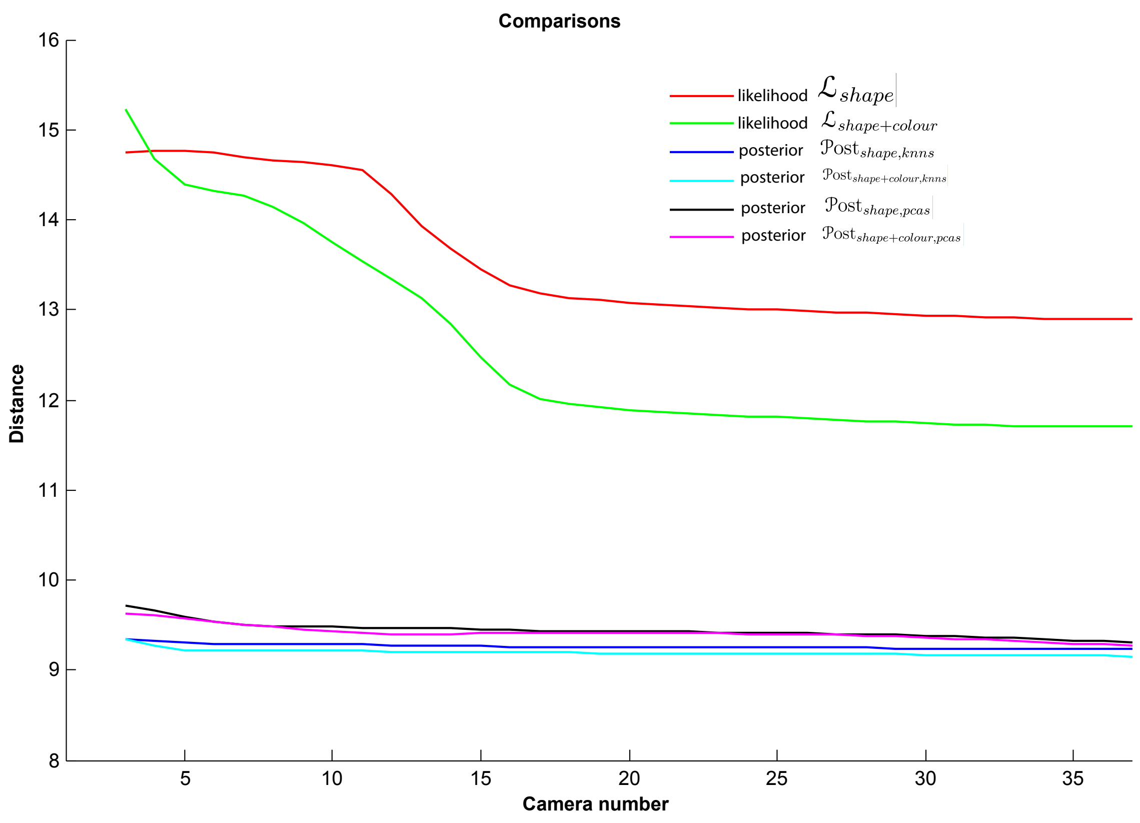


Figure 5.17: Comparison of the reconstructions. The distance between the estimates and the ground truth is computed with respect to the number of cameras.

- posterior with the likelihood using silhouettes with the shape prior approximated by the PCA-based method ($K = 3$)

$$\mathcal{P}ost_{shape,pcas} \propto \mathcal{L}_{shape} \times \mathcal{P}rior_{pcas}$$

- posterior with the likelihood using silhouette and colour with a shape prior approximated by the PCA-based method ($K = 3$)

$$\mathcal{P}ost_{shape+colour,pcas} \propto \mathcal{L}_{shape+colour} \times \mathcal{P}rior_{pcas}$$

Figure 5.17 shows a comparison of these methods. The shape distances (see equation 5.2) between the ground-truth and the estimated shapes in the 2D slices are computed. The average distance shown in Figure 5.17. The experiment is repeated for different numbers of cameras. From Figure 5.17, we can see that:

- The methods that use the posteriors are superior to only using the likelihoods. This result supports that the proposed priors help to produce a good reconstruction of the shape and improve the accuracy of the likelihoods even when few cameras are used.

- Comparing the results of the likelihoods $\mathcal{L}_{shape}$ and $\mathcal{L}_{shape+colour}$, colour information also helps to improve the overall accuracy.

- The posteriors using PCA have a slightly better result than the one using KNN. In addition, PCA is using $K = 3$ eigenfaces for the reconstruction whereas KNN uses $K = 23$ nearest neighbours. The selection of K is explored in more detail in Section 5.3.1.3.

The 3D faces have been split into 70 horizontal slices and the horizontal slice representation is described in more detail in Section 5.3.1.4. The results of the reconstruction of the 45^{th} slice is shown in Figure 5.18. Some of the exemplars used to model the prior are shown in Figure D.5. The likelihood using only silhouettes leads to a convex reconstruction, but the likelihood using silhouettes and its colour information can reconstruct a part of the concave regions (see the second row of Figure 5.18). In the third and fourth row of Figure 5.18, posterior functions are able to restore nearly perfect concavities. More results for the slices 37^{th} and 53^{th} are shown in Appendix D.2.

5.3.1.3 Selection of K

The two posteriors based on the KNN prior are analysed with respect to the number of neighbours K in Figure 5.19. For comparison, the distance for the two posteriors using PCA with $K = 3$ is also represented (see horizontal lines in Figure 5.19). The distance gradually decreases while more K neighbours are used indicating that the reconstruction gets closer to the ground truth. When $K \geq 23$, the KNN posteriors have better performance than the PCA-based one for which three eigenvectors are used. The eigenvalues of the PCA for the posterior using

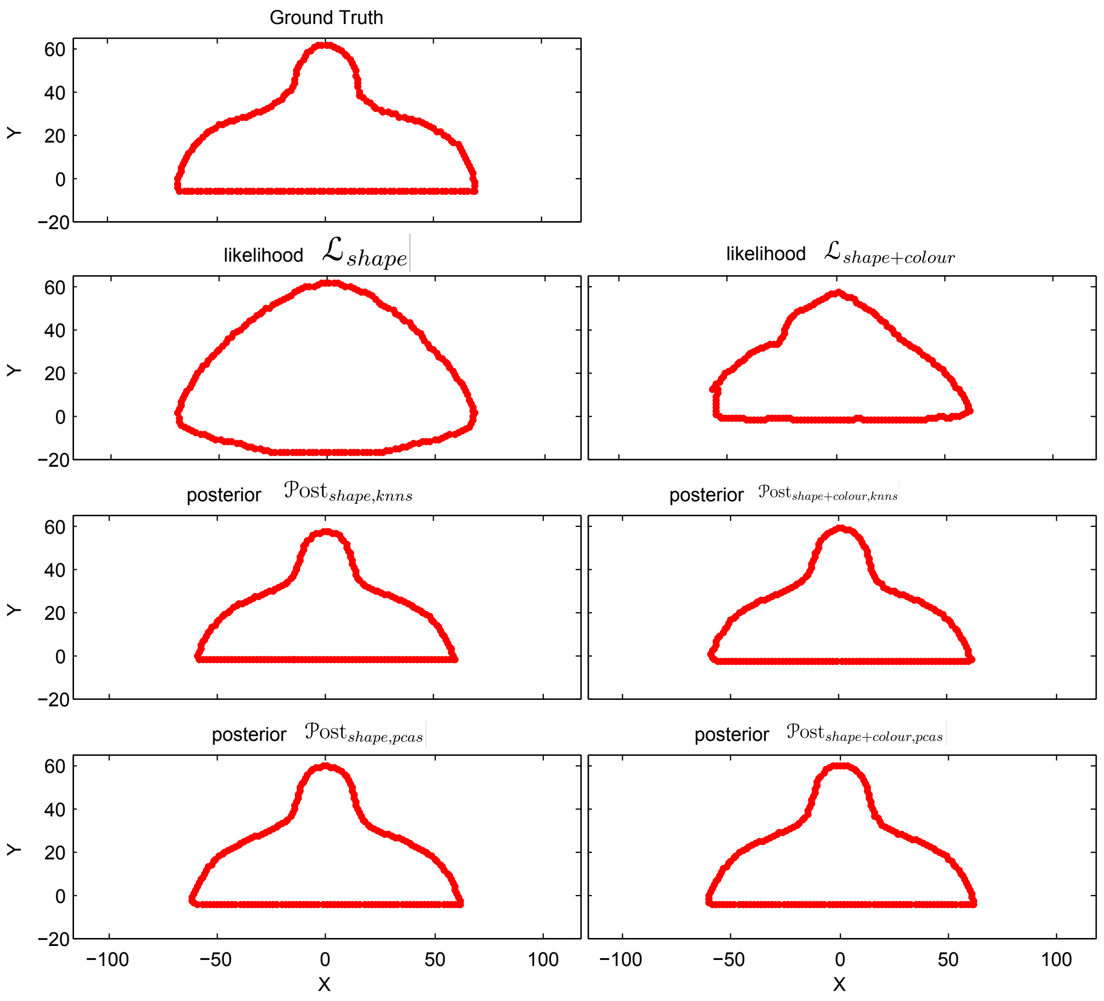


Figure 5.18: Reconstruction of the 45th slice using 36 camera views.

shape and colour information are shown in Figure 5.20 and 5.21. In Figure 5.21, the top three eigenvectors have a variance of nearly 90%.

5.3.1.4 3D Reconstruction

Complete 3D faces are reconstructed using the modellings listed in Section 5.3.1.2. A 3D reconstruction is computed by stacking the estimated horizontal 2D slices along the Z-axis. Here 36 camera views are used and faces have been split into 70 horizontal slices. The shape descriptor is redefined as follows for both KNN

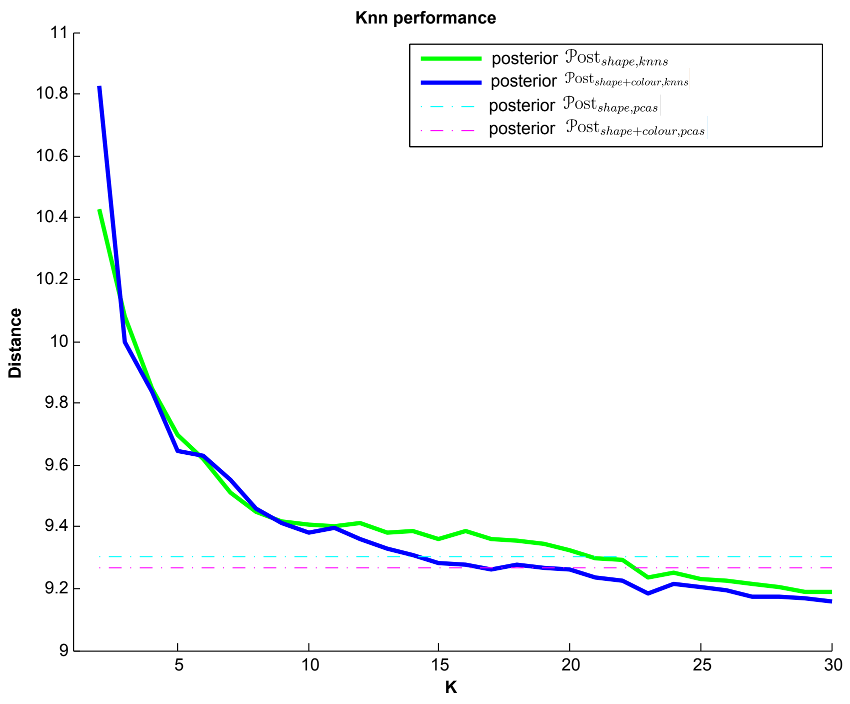


Figure 5.19: KNN performance: distance to the ground truth with respect to the number K of neighbours.

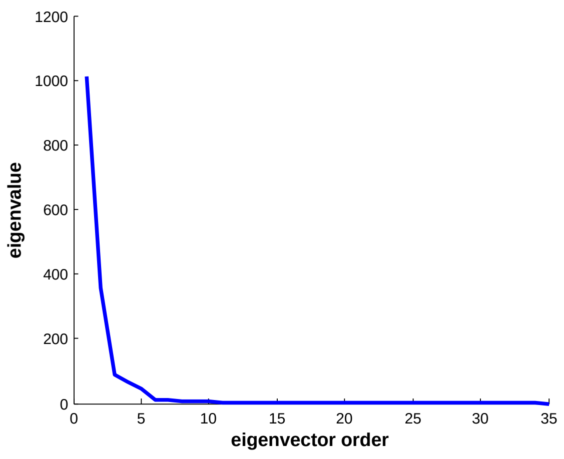


Figure 5.20: Eigenvalues of the PCA in descending order.

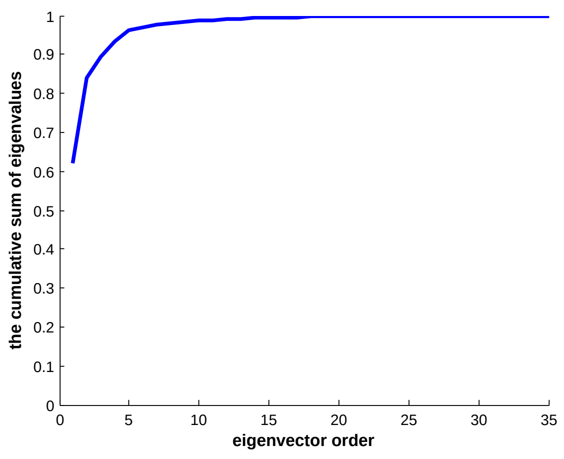


Figure 5.21: The corresponding eigenvalue spectrum.

and PCA priors:

$$\mathbf{X} = [\mathbf{S}_1; \mathbf{S}_2; \cdots ; \mathbf{S}_s] \tag{5.18}$$

where $\mathbf{S}_i$ is the point set in the 2D i^{th} slice and s is the total number of the 2D slices (here $s = 70$). The order of the 2D slices to create the feature vector is from top to bottom and the sequence of the points for the slice is in an anti-clockwise direction illustrated in Figure 5.22. All the 3D faces in both the test

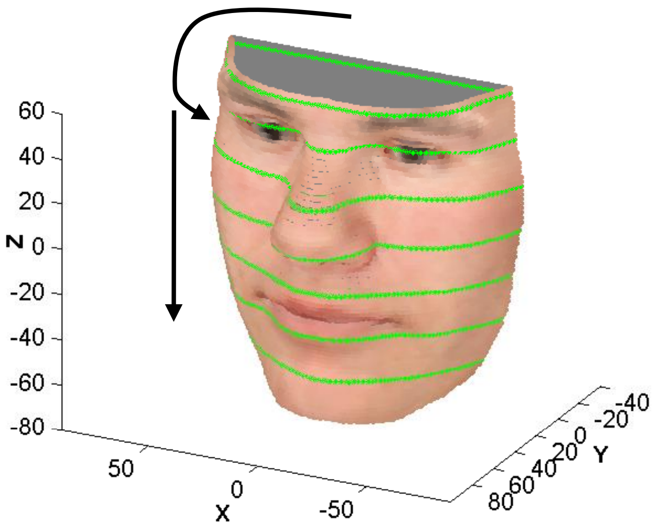


Figure 5.22: 3D head with oriented slices.

set $\mathcal{S}_{test}$ (9 heads) and the training set $\mathcal{S}_{prior}$ (35 heads) (see Figures D.1 and D.2) have this same representation (i.e. having $s = 70$ horizontal slices Figure D.3). To estimate the prior for the 3D faces, the distance metric in equation 5.2 is used with the descriptors as defined in equation 5.18. Then the reconstruction is computed using equation 5.5 (hence the reconstruction for the 3D face consists of 70 horizontal slices altogether).

Note that in this experiment only the face is considered; the back of the head is discarded. Hence all 3D heads have a flat surface at the back side whose colour is uniformly grey. It makes the rotational alignment (permutation) of the shape descriptor relatively easy in this case. However, Section 5.3.2 presents additional results using a different database of more complicated 2D shapes where the rotational alignment is not anymore trivial.

Figure 5.23 presents some 3D reconstruction results with their shape error surfaces. All these 3D face reconstruction results are computed using $K = 23$ for KNN and $K = 3$ eigenvectors for the PCA. The shape error surface is computed using the Euclidean distance between the point clouds of the ground-truth and the reconstruction surface. The average height of the 3D heads in the database is 150mm, and the error ranges from 0mm to 20mm in the error plots. Note that most of the error surface is blue (low error) indicating that the reconstruction is very accurate. However the low cheek region is not as well reconstructed as other parts of the head. This may be due to the limitations caused by the number of 3D heads used in the training database (i.e. the test face variability is not well represented in the training set) and by the linear approximations (i.e. number of components K used in KNN and PCA) in the prior. If more 3D heads are available in the training database which are more similar to the input 3D head, a better reconstruction results may be produced.

Figure 5.24 shows the estimated colour textures on the 3D reconstructed surface results. The texture error surface is also shown. Note that since only chrominance information is estimated in our algorithm, we use the intensities of the approximated prior to convert the chromaticity -red and -green into the RGB colour space for visualisation purposes only. Then the RGB colour differences between the ground truth and the reconstruction results are calculated to visualise the error plots. The maximum difference is 1 in the error surfaces. In Figure 5.24, the colour textures are well estimated globally and well matched with the estimated shapes. However it is more difficult for the methods to estimate colour information in some tiny regions with more complicated shapes and colour patterns such as a part of the mouth and the upper eye. To deal with the problem, more initial points in the Meanshift algorithm for the inference of the colour surface would be required and this would result in greater computation time.

Further results from the experiments are provided in Appendix D.3. The mesh reconstruction is operated using the Rapidform-XOR software (62). The methods proposed here are successfully applied to the 3D reconstruction problem in terms of overall 3D face shapes and textures. It is worth paying particular attention to the fact that concave parts and their colours can be well reconstructed using the proposed methods.

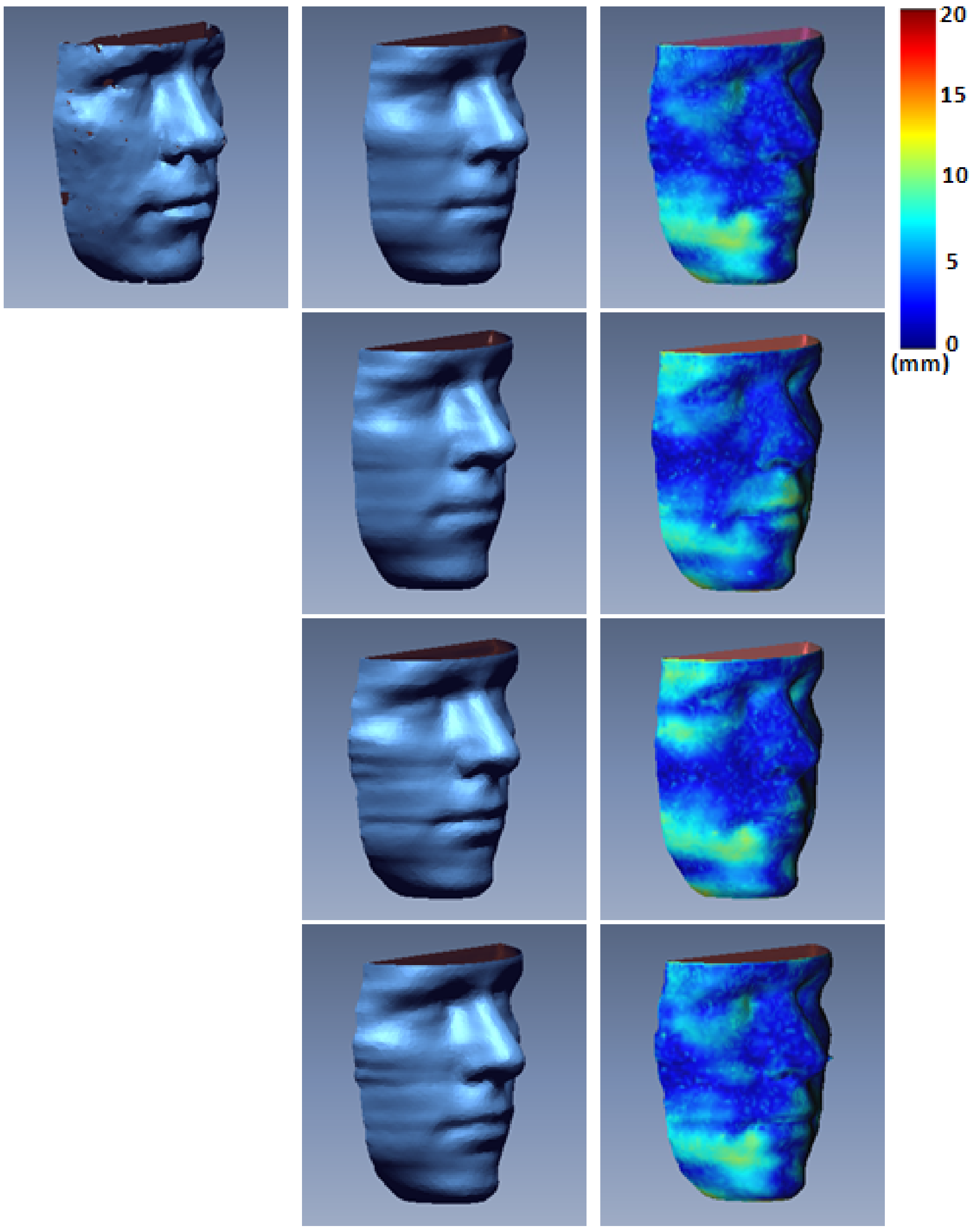


Figure 5.23: 3D Face reconstruction: ground truth (left column), the estimates (middle) and the distance between the estimates and the ground truth in mm (right column). From top to bottom, reconstructions using $\mathcal{P}\text{ost}_{shape,knns}$, $\mathcal{P}\text{ost}_{shape+colour,knns}$, $\mathcal{P}\text{ost}_{shape,pcas}$ and $\mathcal{P}\text{ost}_{shape+colour,pcas}$.

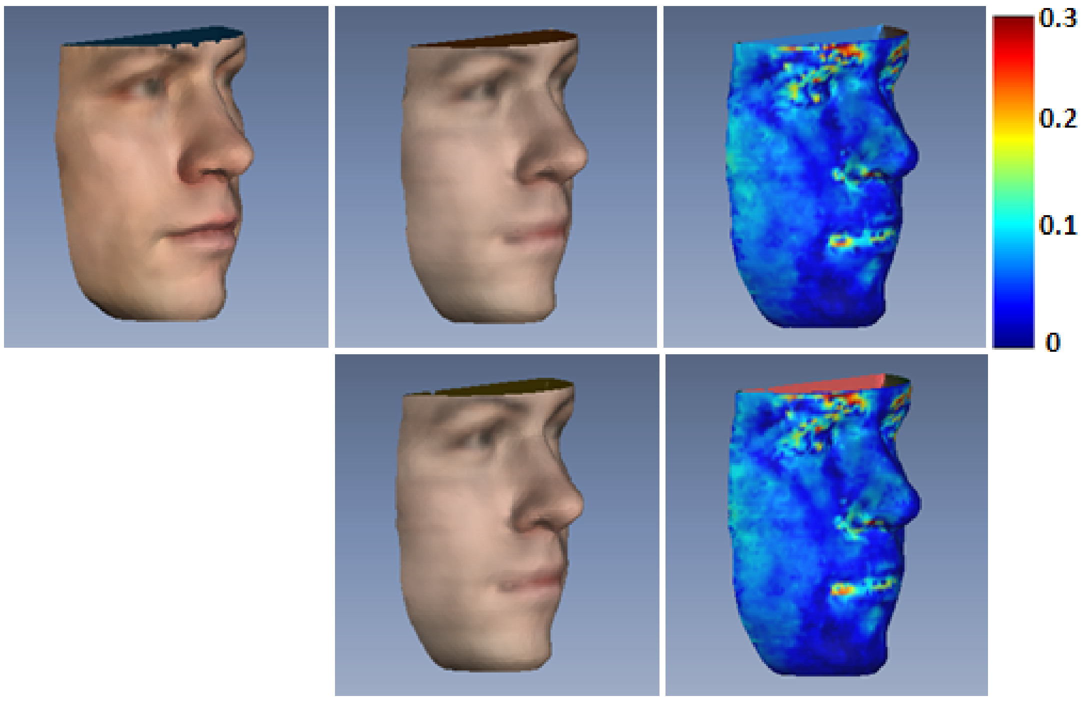


Figure 5.24: 3D Face reconstruction with texture mapping (middle column) and its texture error surfaces (right column). Ground-truth, (top-left), reconstruction with $\mathcal{P}\text{ost}_{shape+colour,knns}$ with its distance to the ground truth (top row) and reconstruction with $\mathcal{P}\text{ost}_{shape+colour,pcas}$ with its distance to the ground truth (bottom row).

5.3.2 Other Experiments with ALOI

More complicated 2D shapes are tested in this section using the ALOI database (29). The database is presented in Section 5.3.2.1 and the posteriors computed with KNN is compared with the likelihoods (Section 5.3.2.2).

5.3.2.1 Database used

The ALOI database (29) is a colour image collection (1000 small objects) with different settings such as illuminations or viewpoints. A rotating table is used to record 72 aspects of the objects in the plane at 5 degree resolution. Six objects are selected and each object class has seven images of different viewing angles $[0°, 15°, 30°, 45°, 60°, 75°, 90°]$ illustrated in Figure 5.25 and Appendix E. The contours of the objects in the images have been extracted using the Canny edge detector. All images are translated to the origin. The seven images are

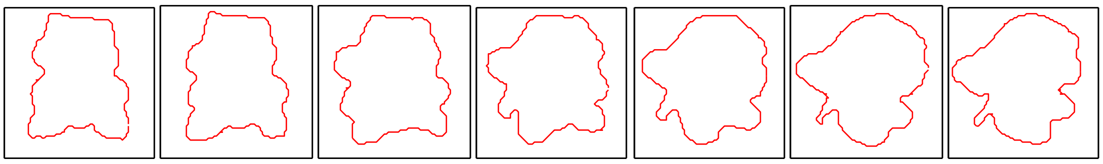


Figure 5.25: One object used in the ALOI database. From left to right viewing angles of the exemplars are $[0°, 15°, 30°, \cdots, 90°]$. The three images at angles $[15°, 45°, 75°]$ are in the test set $\mathcal{S}_{test}$, whereas the other 4 images at angles $[0°, 30°, 60°, 90°]$ are for the training set $\mathcal{S}_{prior}$. Five other objects are in this database and presented in appendix E.

divided into two classes: test database (three images $[15°, 45°, 75°]$) and training database (four images $[0°, 30°, 60°, 90°]$). To create the silhouettes (i.e. observations for the likelihood), the 2D images in the test database are back-projected using orthographic projection. Note that the silhouettes correspond now to 1D binary signals. The training database $\mathcal{S}_{prior}$ is used to approximate the prior for the shape in the k-nearest neighbours method. The total number of exemplars in the training set $\mathcal{S}_{prior}$ is 24 ($N = 6 \times 4 = 24$). The exemplar $\mathbf{X}^e$ is sampled in $M = 360$ points to represent the contour in the training database. $K = 2$ is used for the k-nearest neighbours method.

5.3.2.2 Reconstruction with KNN

The following methods are compared here:

- likelihood using only silhouettes noted $\mathcal{L}_{shape}$,
- posterior with the likelihood using silhouettes and a shape prior approximated by the KNN method ($K = 2$), noted:

$$\mathcal{P}\mathrm{ost}_{shape,knns} \propto \mathcal{L}_{shape} \times \mathcal{P}\mathrm{rior}_{knns}$$

- posterior with the likelihood using silhouettes and colour, and a shape prior approximated by the KNN method ($K = 2$)

$$\mathcal{P}\mathrm{ost}_{shape+colour,knns} \propto \mathcal{L}_{shape+colour} \times \mathcal{P}\mathrm{rior}_{knns}$$

using this database containing six objects chosen from ALOI (29).

Two metrics (the Euclidean distance and shape distance in equation 5.3) are used to measure the distance between the ground-truth and the estimated shapes. Figure 5.26 shows the mean Euclidian distance and standard error computed using the objects in $\mathcal{S}_{test}$ with respect to the number of projections (i.e. camera views) available. Firstly, the proposed prior allows the recovery of concave regions, thereby reducing the distance between the ground truth and the estimated shape. By comparison, the distance for $\mathcal{L}_{shape}$ can only decrease up to a point where the convex visual hull is recovered. The standard errors for the posterior $\mathcal{P}\mathrm{ost}_{shape,knns}$ is quite large when very few cameras are used. Indeed, if the shape can not be well discriminated from different viewing angles using only the silhouettes information, then it is hard to choose the optimal exemplars (the K neighbours) to compute the prior. Also, there are sometimes problems in finding the best cyclic permutation of an exemplar $\mathbf{X}^{e(\hat{m})}$ which is misleading when creating the prior for the shape. However we note that the performance of the posterior $\mathcal{P}\mathrm{ost}_{shape,knns}$ is better than the likelihood $\mathcal{L}_{shape}$ when more than 7 cameras are used. The posterior $\mathcal{P}\mathrm{ost}_{shape+colour,knns}$, which includes colour information, is overall the best method, regardless of the number of cameras used. Figure 5.27 shows the shape distance (equation 5.3) for the two posteriors. It should be

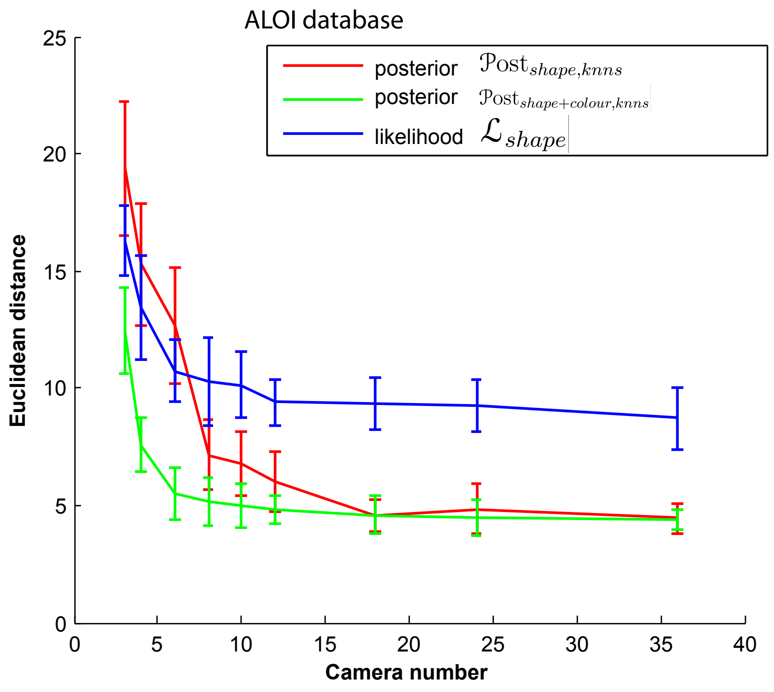


Figure 5.26: Euclidean distance plot with standard error w.r.t. the number of camera views: $\mathcal{L}_{shape}$ (blue) , $\mathcal{P}\text{ost}_{shape,knns}$ (red), $\mathcal{P}\text{ost}_{shape+colour,knns}$ (green).

noted that the result of the posterior using colour is more stable compared to that which does not, particularly when less than 10 cameras are used. Even in the case of a small number of camera views, $\mathcal{P}\text{ost}_{shape+colour,knns}$ consistently achieves accurate results. In the first experiment, no big differences were observed in performances with the 3D reconstruction (as shown in Figure 5.17), whether colour information was used or not. However, in this experiment, the results on more complex objects showed how much colour information can help to obtain consistently accurate results.

The 2D estimates are shown in Figure 5.28. Concavities are well recovered using the posteriors compared to the likelihood. The top left result of the posterior $\mathcal{P}\text{ost}_{shape,knns}$ (see Figure 5.28(a)) illustrates a special case where the algorithm leads to a non optimal solution. The estimated shapes depend on which exemplars are selected for designing the prior. This case is difficult for choosing optimal exemplars because the silhouettes from different viewing angles are similar for both the ground truth and the solution found. Consequently, there are not enough

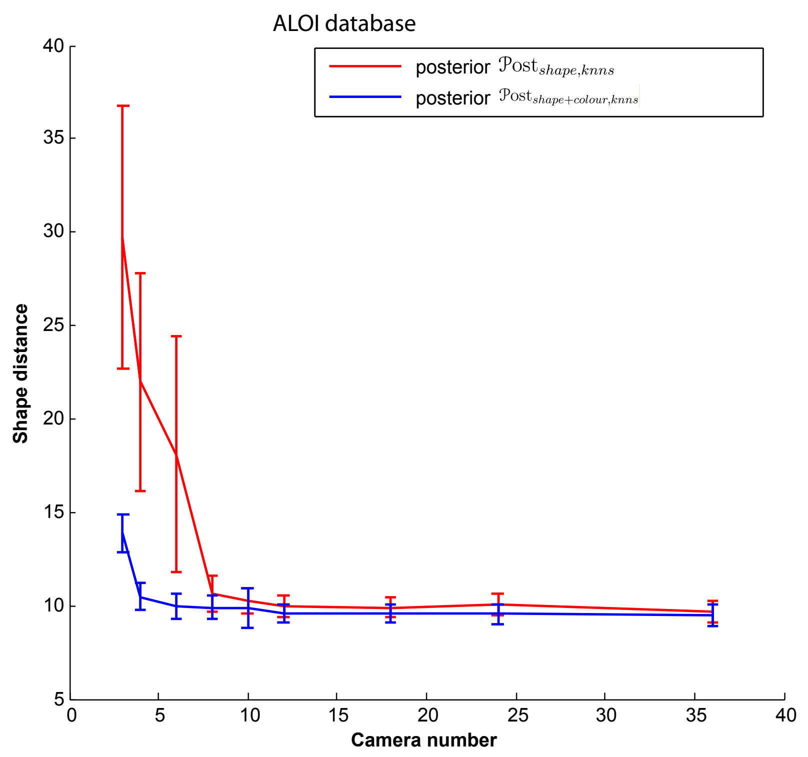


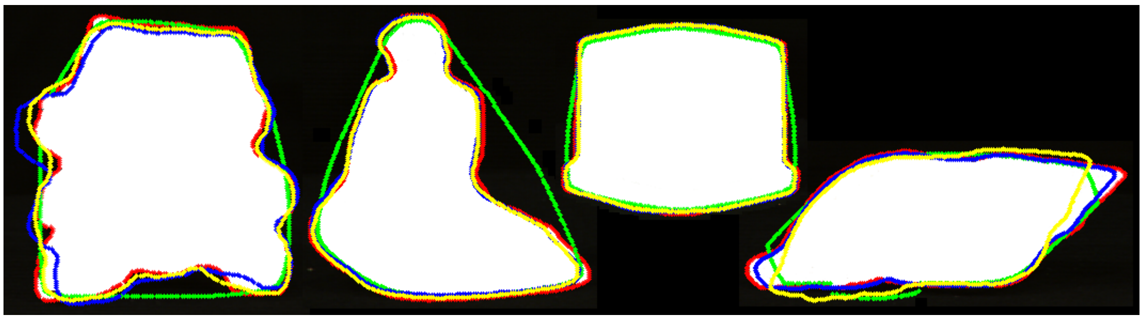
Figure 5.27: Shape distance with standard error w.r.t. the number of camera views: $\mathcal{P}\text{ost}_{shape,knns}$ (red) and $\mathcal{P}\text{ost}_{shape+colour,knns}$ (blue).

clues (in the observations) to guide the selection of the best exemplars.

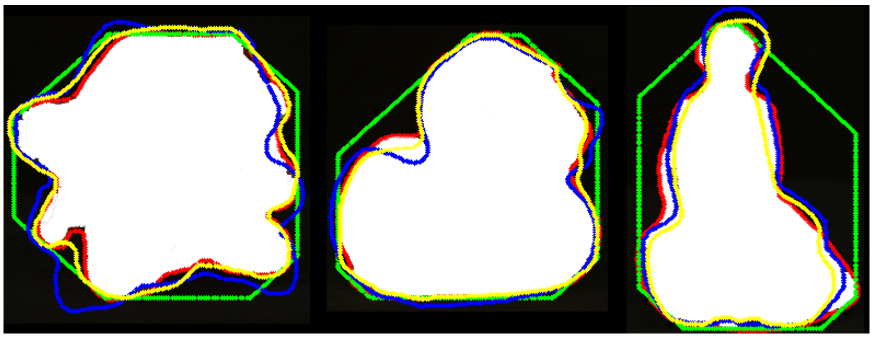
5.4 Conclusion

The likelihoods using silhouette information proposed in Chapter 4 were not able to recover concavities. This chapter proposed that the likelihoods can be complemented with a shape prior to improve the 3D reconstruction. Inference using the mean shift algorithm is performed with posterior kernel density estimates. The proposed prior is updated iteratively so that concavity information can be introduced progressively by using a Gaussian stack; in other words, the prior is updated with a coarse-to-fine strategy.

The proposed prior is also refined at each step by choosing the nearest neighbours of the current estimate (KNN approach). This strategy differs from standard PCA where the reconstruction is computed as a linear combination of fixed



(a) Results from 36 camera views



(b) Results from 3 camera views

Figure 5.28: ALOI results: Ground-truth (red), likelihood $\mathcal{L}_{shape}$ (green), posterior $\mathcal{P}\mathrm{ost}_{shape,knns}$ (blue) and posterior $\mathcal{P}\mathrm{ost}_{shape+colour,knns}$ (yellow).

pre-selected components. Our KNN-based approach on the contrary refines the selection of these components iteratively in our estimation process. The framework is extended so that colour information is also taken into account. Experimental results have shown how well the proposed posteriors are able to recover concave regions thanks to both prior and colour information even only if a small number of cameras are capturing the scene.

Our prior was designed for 2D slices, and the 3D reconstruction of a 3D surface is performed by considering several slices simultaneously. As a consequence, there is a reconstruction artifact in the 3D face results: discontinuous circular bands appear on the surface of the faces. However, simple post-processing filtering can easily remove these discontinuities, for instance by smoothing the surface along the vertical direction (orthogonal direction to the plane containing the slice).

Chapter 6

Conclusions and Future Work

This thesis focused on merging images recorded from multiple camera views to reconstruct an object's 3D colour surface. A variety of methods to estimate 3D models in multi-view systems were presented based on the *Visual Hull*, *Photo Hull* and inverse Radon transform. In contrast to these approaches, however, we proposed continuous modelling methods using binary silhouettes and colour information from multiple views. Several kernel density estimates for the likelihood were proposed to improve discrete formulations. Moreover, it was suggested that posteriors combining the likelihood with a prior for the shape can obtain better reconstructions, which recover the object concavities. In this chapter, we summarise the findings of this thesis, and discuss possible directions for future research in this area.

6.1 Achievements

Chapter 3 presented an experiment conducted to evaluate a method designed to recover an accurate 3D shape by merging estimated visual hulls reconstructed at different times, given motion information for a rigid object. The experiment illustrated the concept that multiple views captured a number of times can be merged efficiently to refine the 3D shape. This approach was found to be practical in application when a small number of cameras are in use, but capture the object of interest simultaneously. The visual hulls were estimated using 3D histograms:

a discrete representation that is memory demanding and computationally expensive when searching for maxima. In addition, visual hulls were estimated from the silhouettes of the objects that need to be segmented from the images. In Chapter 2, it was established that silhouette information is not necessary when inferring the photo hull which uses both pixel location and colour information to reconstruct a coloured surface.

Drawing on these findings, Chapter 4 proposed the following:

- **kernel density estimates of the visual hull, and**
- **kernel density estimates of the photo hull.**

These modellings give smooth cost functions that can be optimised by gradient ascent methods. For the sake of simplicity, volume is taken to be a stack of 2D surfaces (slice). 3D reconstruction is then simplified to 2D shape reconstruction in each slice (Section 4.1). Two kernel density estimates (for *Visual Hull* and *Photo Hull*) are modelled for each slice and the mean shift algorithm is proposed to recovered the contour of the volume in each slice.

The kernel is then extended for inference in the 2D slice, to be defined in the 3D space, eliminating the need to decompose the 3D space into slices (Section 4.2). In Section 4.3, we went on to propose that these 3D modelling methods can be extended to consider colour information. Section 4.4.1 describes a simulated annealing approach for the bandwidth used as a temperature to avoid local maxima in the kernel density estimates. Section 4.5 demonstrates empirical how well the proposed methods perform.

The kernel density estimates proposed in Chapter 4 take only the observations into account and do not include any prior information, and can thus be viewed as likelihoods. To improve this method's accuracy, **the kernel density estimates of the visual hull and photo hull are extended to take into account a prior** for the shape in Chapter 5. A posterior density function is then modelled by combining the likelihood with a prior. The prior is approximated by the k-nearest neighbours method. It is refined and updated to create a coarse-to-fine process implemented by a Gaussian shape stack. The results from the k-nearest neighbours are compared to the PCA-based method for comparison.

Both the likelihoods and the priors are modelled using kernel mixtures. Consequently **the resulting posteriors are also kernel density estimates that can be optimised using the mean shift algorithm.** Experimental results show that concavities can be well recovered when a prior is used, even if only a small number of cameras are available.

6.2 Future Work

There are five broad potential directions for future research.

1. Extension to pinhole camera: Our proposed kernel density estimates to reconstruct shapes of interest are based on an orthographic camera model. Recently, this limitation has been solved by Ruttle et al. (64). They have defined a kernel density estimate for the pin-hole camera, and optimisation is performed using a Newton-Raphson algorithm as an alternative to Meanshift.

2. Extension to 3D prior: To reconstruct a 3D shape of interest, our prior for the 3D shape consists of an accumulation of all 2D priors from the 2D slices. Therefore, a 2D slice representation of the 3D shape is required to apply the prior when estimating the 3D shape. An alternative approach would be for a 3D prior to infer the 3D shape reconstruction directly.

3. Motion estimation: The methods presented in this study assume that 3D motion information of an object of interest is available. In multi-view video streams, 3D motion estimation needs to be performed to restore the 3D shape. Therefore, implementation of 3D motion estimation for compensation of movements between frames is essential for 3D reconstruction using sequential images.

4. Non-rigid object (face): In this study, 3D faces are assumed to be a rigid object. Although this is true when considering multiple images recorded at the same time, the 3D face is deformable when viewed over a period of time and facial expressions appear. This problem can be overcome by compensating facial expression when sequential images from multi-view cameras

are merged together to estimate a 3D reconstruction, though further work is necessary before this can be achieved.

5. Uncertainty in temporal data: It is assumed in this study that the network of cameras used to capture the scene was well synchronised. In practice this may not be the case and the images may be recorded with a small temporal lag. One solution that may be explored in the future is to extend the modelling of the spatial random variable $\mathbf{x}$ to a spatio-temporal random variable $(\mathbf{x}, t)$ so that uncertainty about the time can be taken into account explicitly. The observations would then be the pixel positions in the silhouettes and their time of recording.

Appendix A

Database used in Chapter 3

This appendix shows the information used to estimate the 3D reconstructions in Chapter 3.

Figure A.1 and A.2 present the recorded images of the head as seen by four cameras in the HumanEva-II database (71) and the corresponding silhouettes that have been manually created. Figure A.3 and A.4 show the segmented heads that have been used for reconstructing 3D head results. Some example silhouettes used to reconstruct the dinosaur in the Middlebury database (67) are also shown in Figure A.5.

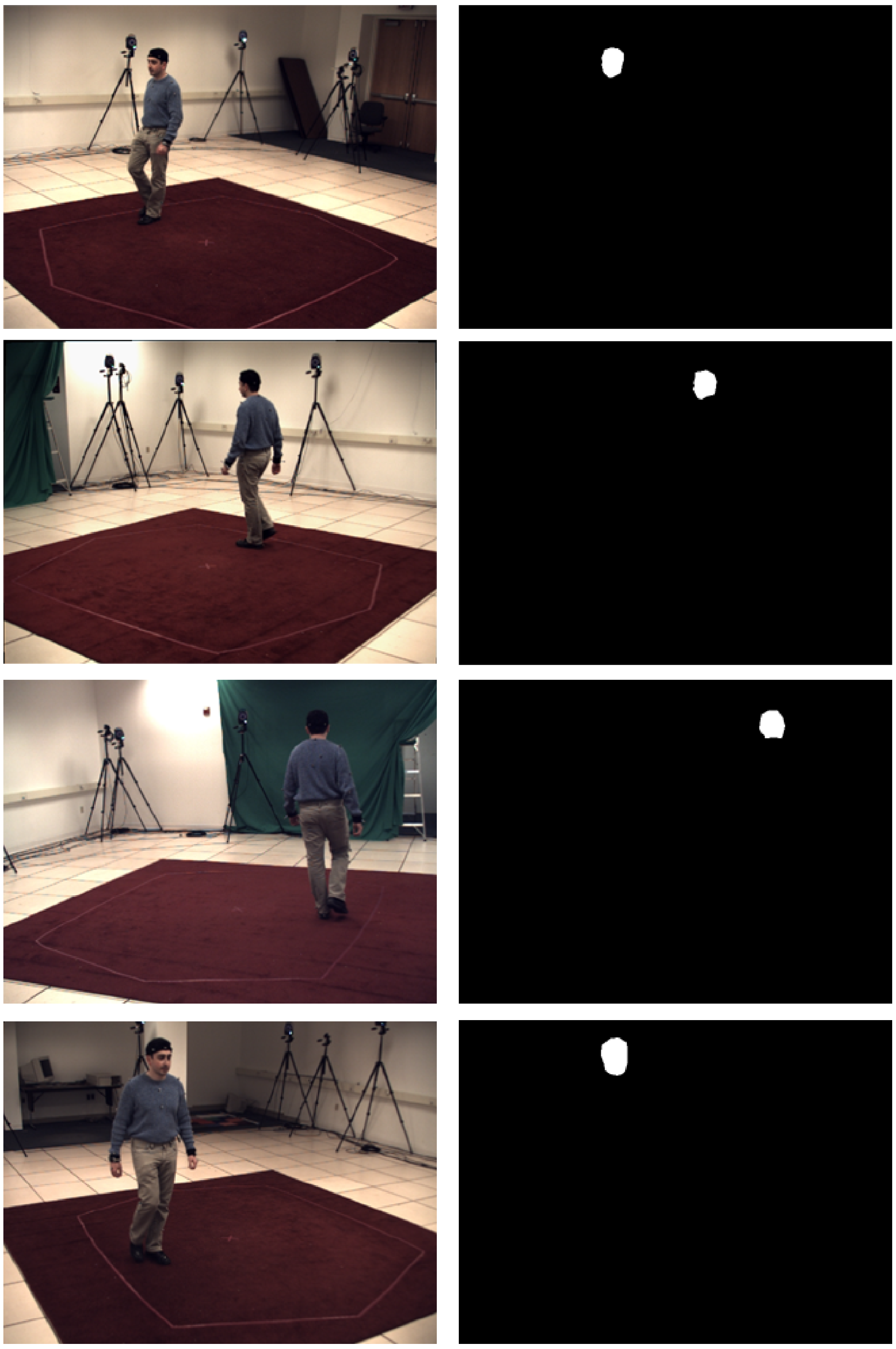


Figure A.1: Four views of a head in HumanEva-II datasets and its silhouettes (71).

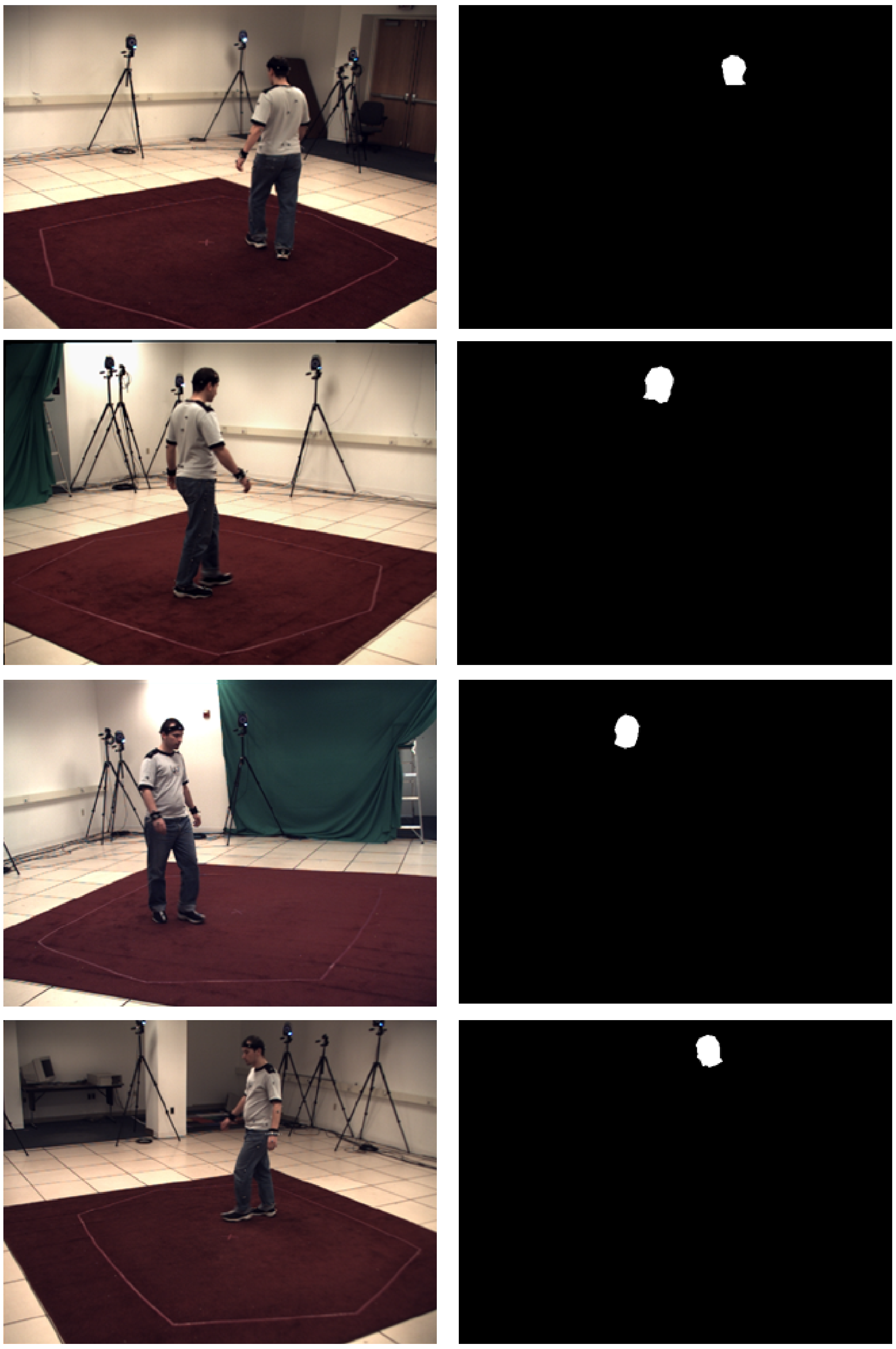


Figure A.2: Four views of a head in HumanEva-II datasets and its silhouettes (71).

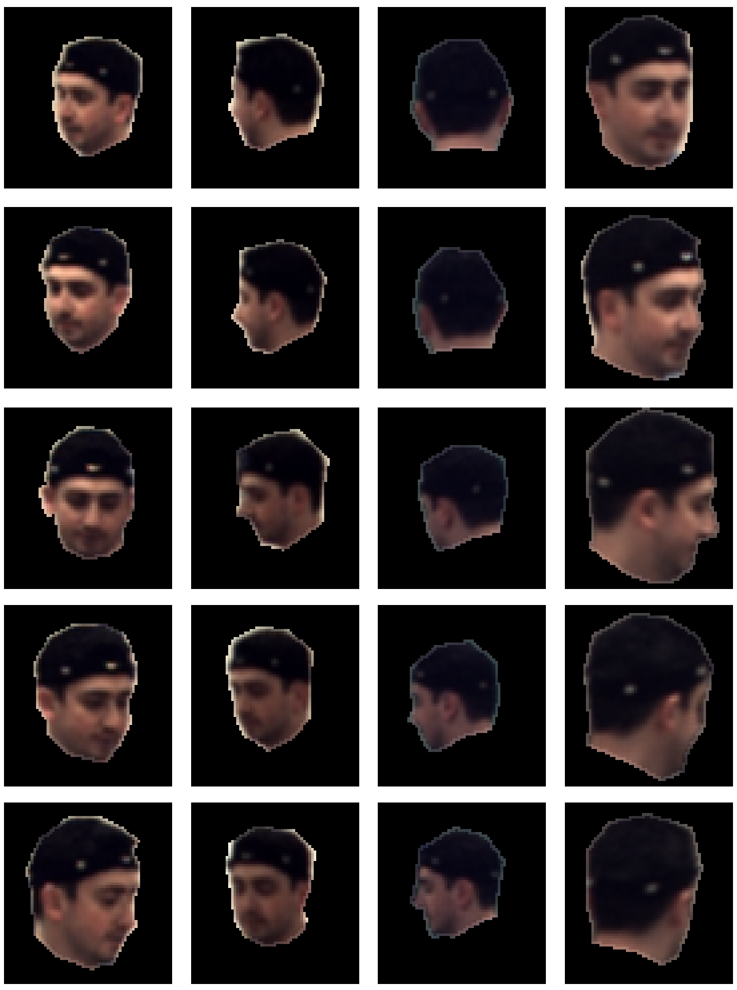


Figure A.3: Silhouette examples of a head (head images in each column are recorded by one of four cameras, and four images in each row is used to reconstruct 3D head at time (t)).

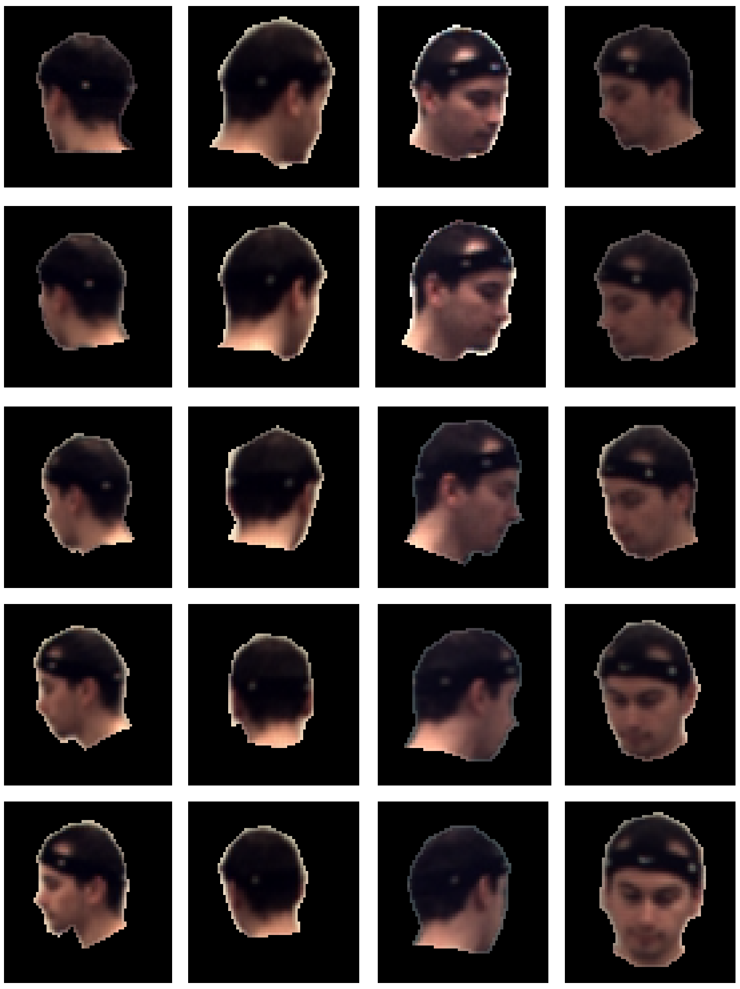


Figure A.4: Silhouette examples of a head (head images in each column are recorded by one of four cameras, and four images in each row is used to reconstruct 3D head at time (t)).

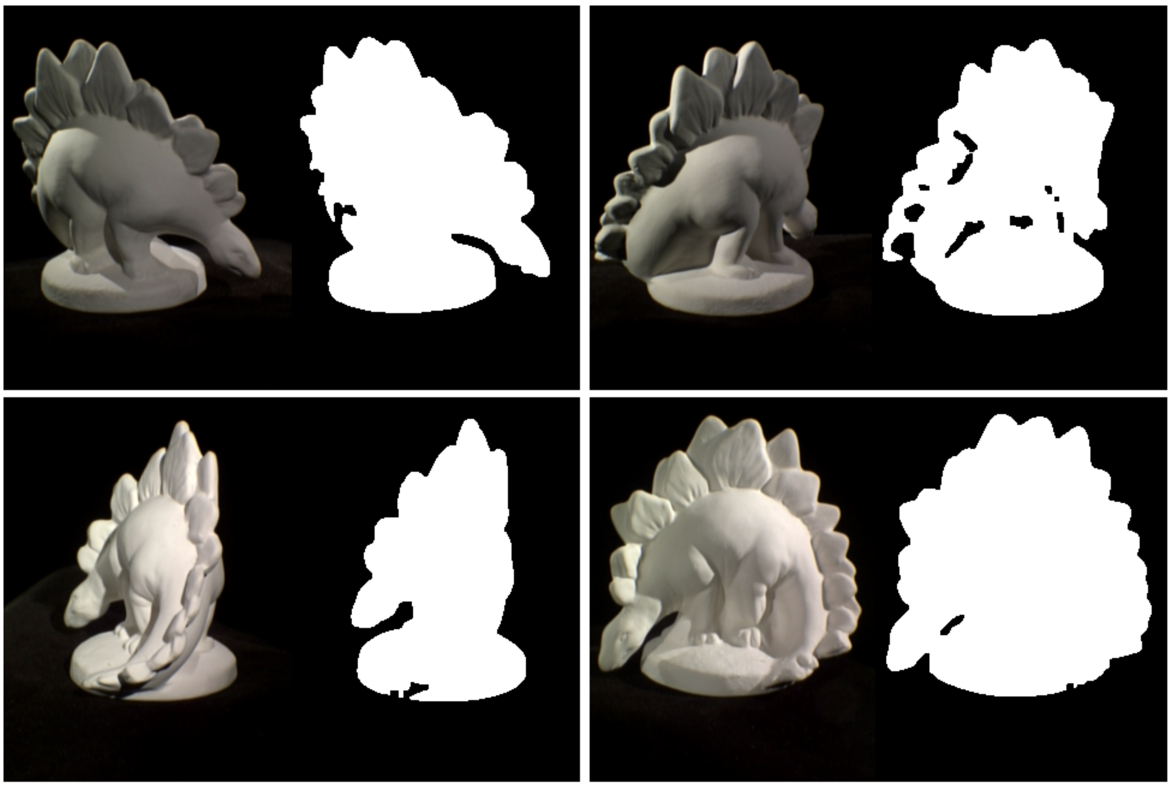


Figure A.5: Left images illustrate captured images from different views in the Middlebury database (67), and its silhouettes segmented by a background subtraction method are shown in the right. The silhouettes are not perfect due to shadows on the dinosaur.

Appendix B

Estimation Results in Chapter 4

Figure B.1 illustrates examples of the reconstructions of 2D KDE using silhouettes of 36 horizontal cameras.

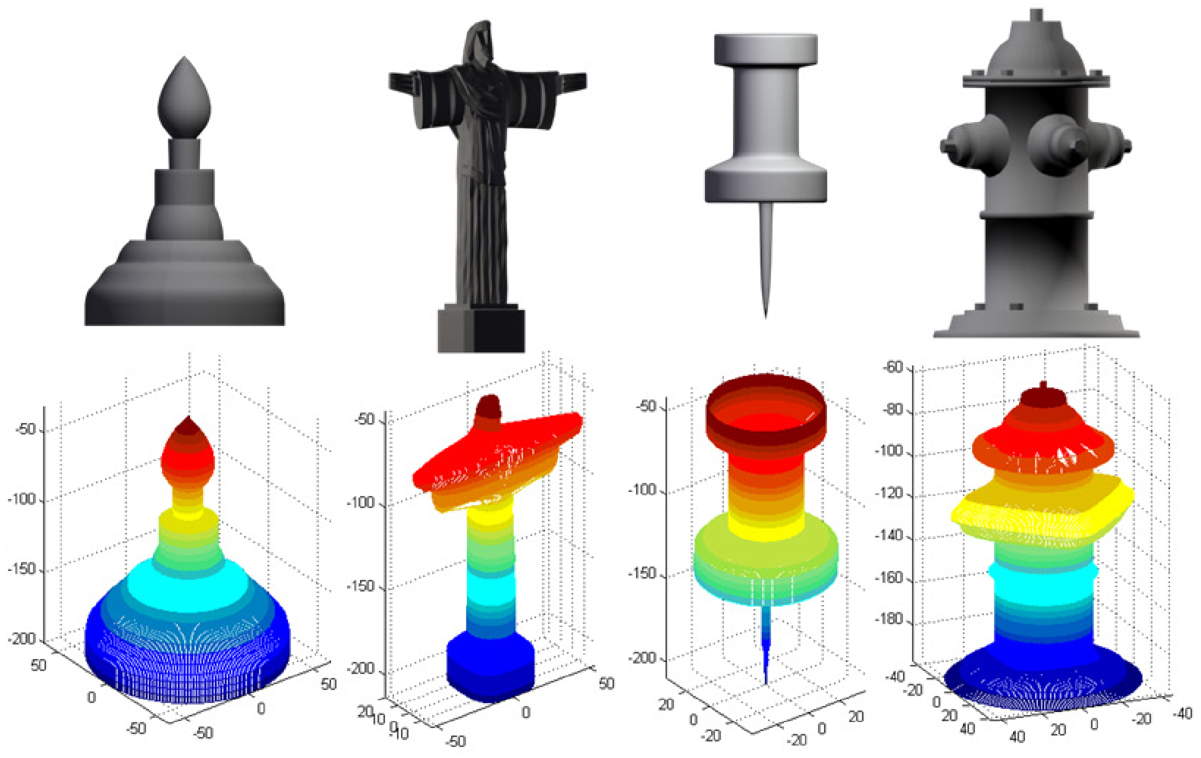


Figure B.1: Ground-truth models (top row) and its reconstruction results using 2D KDE with 36 horizontal cameras (bottom row).

The variations of 2D KDE results with changing the number of cameras are shown in Figure B.2.

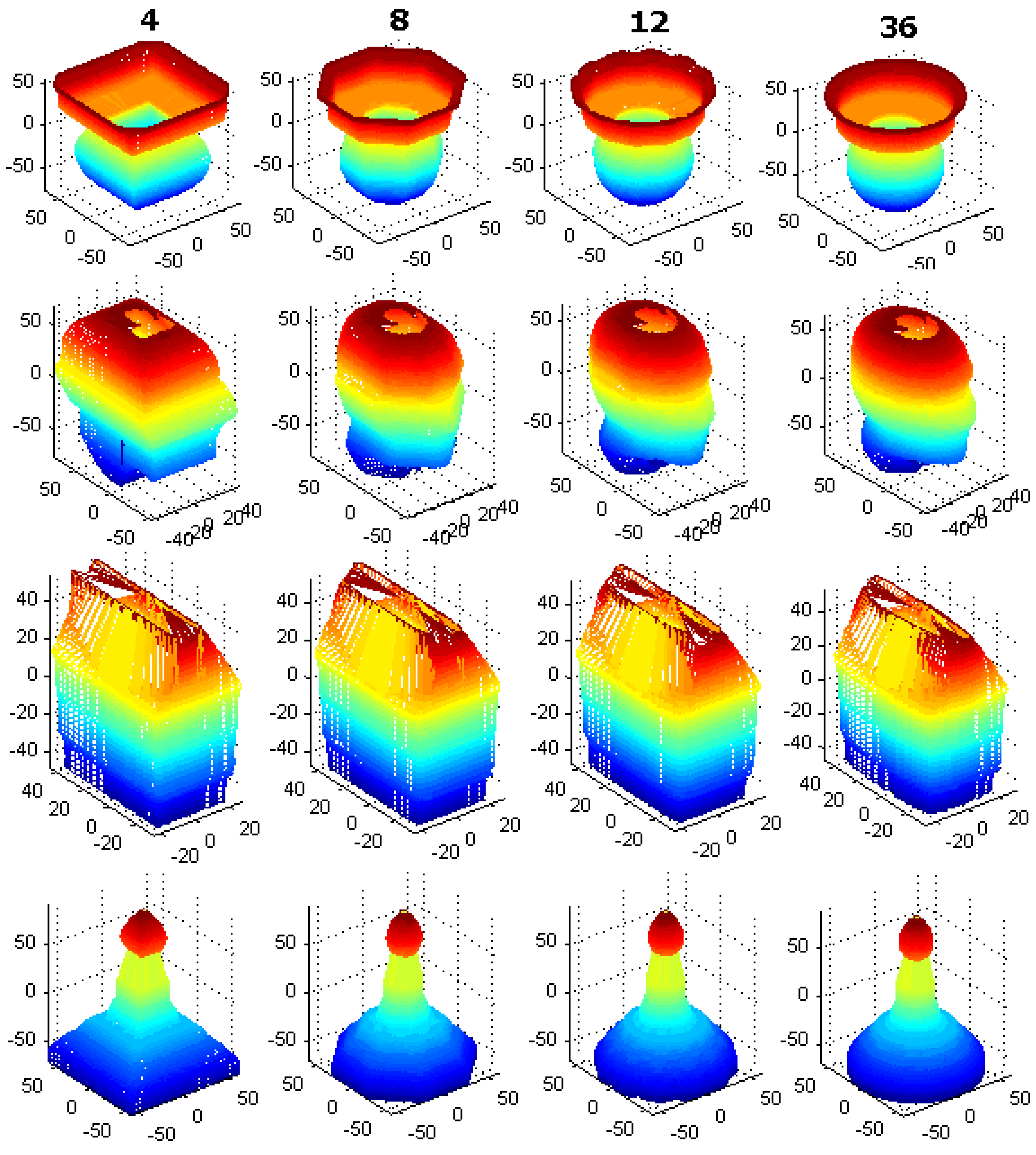


Figure B.2: 2D KDE results (lamp, head, house and bulb) in the horizontal camera settings. The results from left to right column are estimated using from 4 to 36 cameras.

The variations of 3D KDE results with changing the number of cameras are shown in Figure B.3.

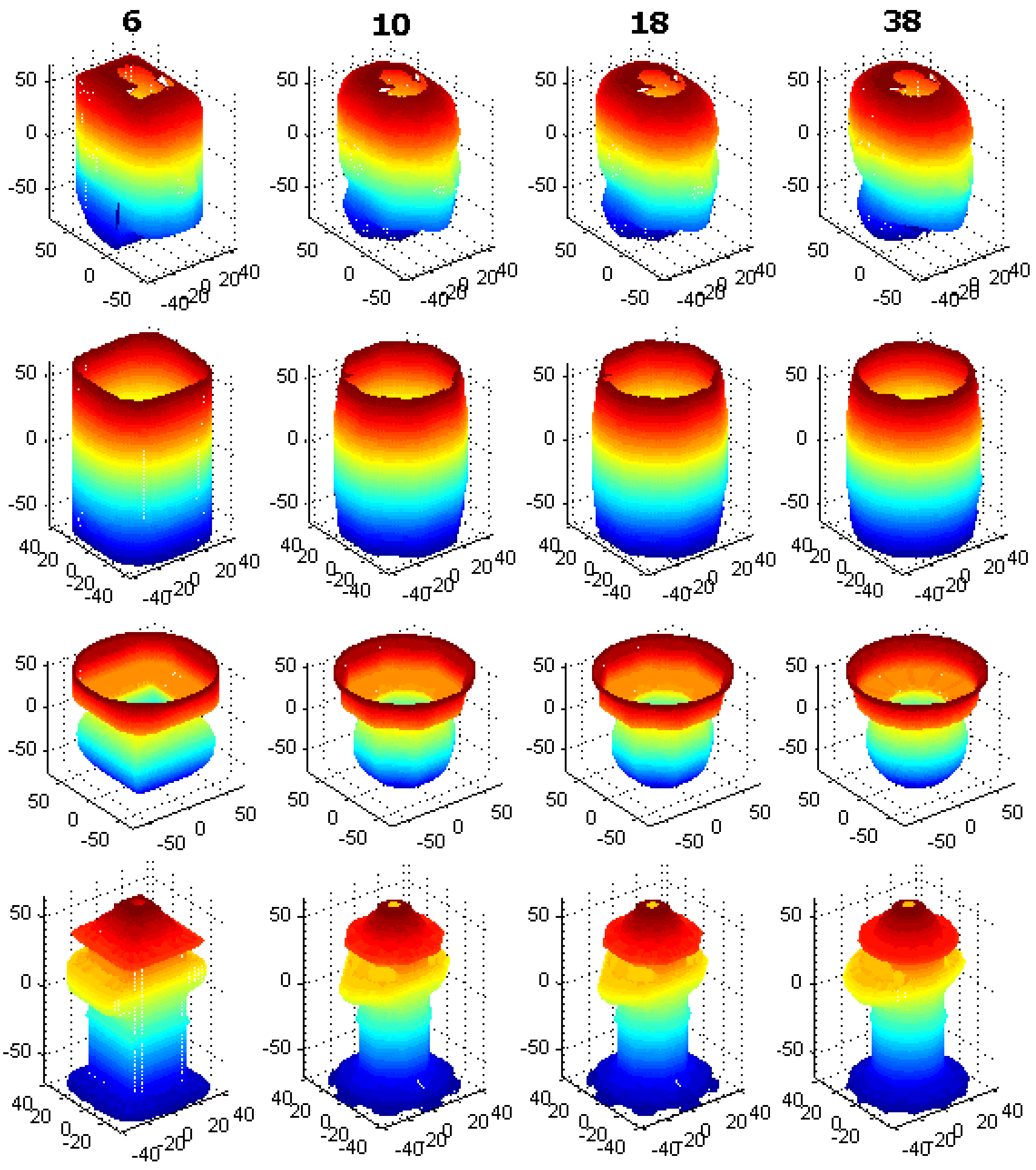


Figure B.3: 3D KDE results (head, barrel, lamp and hydrant) in the spherical camera settings. The results from left to right column are estimated using from 6 to 38 cameras.

Reconstructions from noise images are shown in Figure B.4.

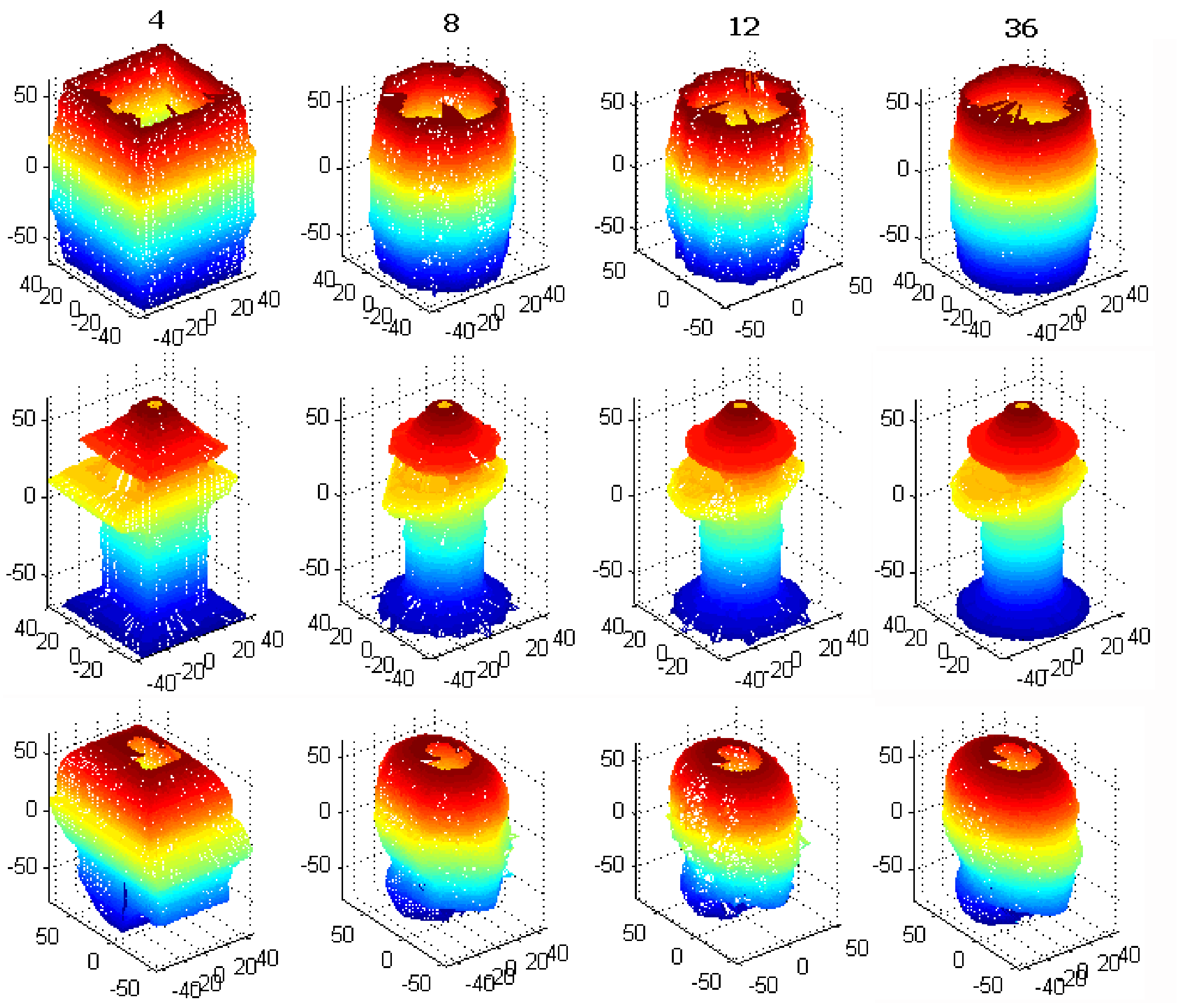


Figure B.4: 3D KDE results with 20% noise.

Appendix C

Derivation of Posteriors

C.1 Posterior using Only Shape

The likelihood of the point $\mathbf{x}_i = (x_i, y_i)^T$, $\{\mathbf{x}_i\}_{i=1,\cdots,M} \in \mathbf{X}$ is described in Section 4.1.2 as follows:

$$\hat{p}_{lik}(\mathbf{x}_i) \propto \frac{1}{N} \sum_{j=1}^{N} \frac{1}{\sqrt{2\pi} h_s} \exp\left(\frac{-D_j(\mathbf{x}_i)^2}{2h_s^2} \right), \tag{C.1}$$

where $D_j(\mathbf{x}_i) = \rho_j - x_i \cos\theta_j - y_i \sin\theta_j$, N is the number of foreground pixels and h_s is the bandwidths of the Gaussian kernels for the spatial domains.

The prior probability ($\hat{p}_{prior}$) of $\mathbf{x}_i$ is given using the approximated shape prior $\{\hat{\mathbf{x}}_m\}_{m=1,\cdots,M} \in \hat{\mathbf{X}}$ in Section 5.1.3:

$$\hat{p}_{prior}(\mathbf{x}_i) \propto \frac{1}{\sqrt{2\pi}(M-1)h_p} \sum_{m=1,m\neq i}^{M} \exp\left(-\frac{D_m(\mathbf{x}_i)^2}{2h_p^2} \right). \tag{C.2}$$

where h_p is the bandwidth of the Gaussian kernel, M is the number of points and

$$\begin{aligned} D_m(\mathbf{x}_i) &= \vec{n}_m^T(\mathbf{x}_i - \mathbf{x}_m) \\ &= \rho_m - x_i \cos\theta_m - y_i \sin\theta_m. \end{aligned} \tag{C.3}$$

θ_m is given by the angle of the line passing through the points $(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_m)$ and ρ_m is calculated by $\mathbf{x}_m$.

Finally, the resulting posterior is given by the likelihood and prior:

$$\hat{p}_{post}(\mathbf{x}_i) \propto \hat{p}_{lik}(\mathbf{x}_i) \times \hat{p}_{prior}(\mathbf{x}_i) \tag{C.4}$$

The posterior is optimised using a mean shift algorithm to find its maxima representing the final estimate $\mathbf{X}$. The mean shift algorithm is performed by differentiating the posterior and equating the result to zero as follows:

$$\mathbf{x}_i^{(n+1)} = \left(L(\mathbf{x}_i^{(n)})\right)^{-1} \cdot Q(\mathbf{x}_i^{(n)}). \tag{C.5}$$

where $\mathbf{x}_i^{(n)}$ is the current point and $\mathbf{x}_i^{(n+1)}$ is the next iteration step result of $\mathbf{x}_i^{(n)}$ which has higher probability than the probability of $\mathbf{x}_i^{(n)}$. $\mathbf{x}$ is defined as a vector concatenating $\mathbf{x} = (x, y)$ in one column vector. L is 2×2 matrix:

$$L = [L_{1,1}, L_{1,2}; L_{2,1}, L_{2,2}]$$

$$\begin{aligned}
L_{1,1} &= \sum_j \cos^2\theta_j A \sum_m B + \sum_m \cos^2\theta_m B \sum_j A \\
L_{1,2} &= \sum_j \sin\theta_j \cos\theta_j A \sum_m B + \sum_m \sin\theta_m \cos\theta_m B \sum_j A \\
L_{2,1} &= \sum_j \sin\theta_j \cos\theta_j A \sum_m B + \sum_m \sin\theta_m \cos\theta_m B \sum_j A \\
L_{2,2} &= \sum_j \sin^2\theta_j A \sum_m B + \sum_m \sin^2\theta_m B \sum_j A
\end{aligned} \tag{C.6}$$

Q is 2×1 matrix:

$$Q = \begin{bmatrix} \sum_j \rho_j \cos\theta_j A \sum_m B + \sum_m \rho_m \cos\theta_m B \sum_j A \\ \sum_j \rho_j \sin\theta_j A \sum_m B + \sum_m \rho_m \sin\theta_m B \sum_j A \end{bmatrix} \tag{C.7}$$

where $A = \exp\left(-\frac{D_j(\mathbf{x}_i^{(n)})^2}{2h_s^2}\right)$ and $B = \exp\left(-\frac{D_m(\mathbf{x}_i^{(n)})^2}{2h_p^2}\right)$.

C.2 Posterior using Shape and Colour

In Section 4.3.1, the likelihood of the point and colour $\mathbf{x}_i = (x_i, y_i, r_i, g_i)^T$, $\{\mathbf{x}_i\}_{i=1,\cdots,M} \in \mathbf{X}$ is formulated as:

$$\hat{p}_{lik}(\mathbf{x}_i) \propto \frac{1}{N} \sum_{j=1}^{N} \frac{1}{(2\pi)^{\frac{3}{2}} h_s h_r h_g} \exp\left(\frac{-D_j(x_i, y_i)^2}{2h_s^2}\right) \times \exp\left(\frac{-(r_j - r_i)^2}{2h_r^2}\right) \times \exp\left(\frac{-(g_j - g_i)^2}{2h_g^2}\right) \tag{C.8}$$

where $D_j(x,y) = \rho_j - x\cos\theta_j - y\sin\theta_j$, N is the number of foreground pixels, and h_s, h_r and h_g are the bandwidths of the Gaussian kernels for the spatial and colour domains.

The prior and the posterior formulation are same as Equation C.2 and C.4 respectively. The mean shift iteration to find maxima of the posterior is operated by:

$$\mathbf{x}_i^{(n+1)} = \left(L(\mathbf{x}_i^{(n)})\right)^{-1} \cdot Q(\mathbf{x}_i^{(n)}). \tag{C.9}$$

where $\mathbf{x}_i^{(n)}$ is the current point and $\mathbf{x}_i^{(n+1)}$ is the next iteration step result of $\mathbf{x}_i^{(n)}$ which has higher probability than the probability of $\mathbf{x}_i^{(n)}$. $\mathbf{x}$ is defined as a vector concatenating $(x,y,r,g)^T$ in one column vector. Equation C.9 is extracted from differentiating the posterior and equating the result to zero.

L is 4×4 matrix:

$$\begin{bmatrix} L_{1,1} & L_{1,2} & 0 & 0 \\ L_{2,1} & L_{2,2} & 0 & 0 \\ 0 & 0 & L_{3,3} & 0 \\ 0 & 0 & 0 & L_{4,4} \end{bmatrix} \tag{C.10}$$

$$\begin{aligned}
L_{1,1} &= \sum_j \cos^2\theta_j ACD \sum_m B + \sum_m \cos^2\theta_m B \sum_j ACD \\
L_{1,2} &= \sum_j \sin\theta_j \cos\theta_j ACD \sum_m B + \sum_m \sin\theta_m \cos\theta_m B \sum_j ACD \\
L_{2,1} &= \sum_j \sin\theta_j \cos\theta_j ACD \sum_m B + \sum_m \sin\theta_m \cos\theta_m B \sum_j ACD \\
L_{2,2} &= \sum_j \sin^2\theta_j ACD \sum_m B + \sum_m \sin^2\theta_m B \sum_j ACD \\
L_{3,3} &= \sum_j ACD \sum_m B \\
L_{4,4} &= \sum_j ACD \sum_m B
\end{aligned} \tag{C.11}$$

Q is 4×1 matrix:

$$Q = \begin{bmatrix} \sum_j \rho_j \cos\theta_j ACD \sum_m B + \sum_m \rho_m \cos\theta_m B \sum_j ACD \\ \sum_j \rho_j \sin\theta_j ACD \sum_m B + \sum_m \rho_m \sin\theta_m B \sum_j ACD \\ \sum_j r_j ACD \sum_m B \\ \sum_j g_j ACD \sum_m B \end{bmatrix} \tag{C.12}$$

where $A = \exp\left(-\frac{D_j(x_i^{(n)}, y_i^{(n)})^2}{2h_s^2}\right)$, $B = \exp\left(-\frac{D_m(x_i^{(n)}, y_i^{(n)})^2}{2h_p^2}\right)$, $C = \exp\left(-\frac{(r_j - r_i^{(n)})^2}{2h_r^2}\right)$ and $D = \exp\left(-\frac{(g_j - g_i^{(n)})^2}{2h_r^2}\right)$.

Appendix D

Face surface reconstruction

This appendix gives additional information to the experiments done in Chapter 5 using the face database.

D.1 3D Heads Database

Figure D.1 presents the 9 faces used for testing, and Figure D.2 shows 12 faces out of 35 that have been used for designing the priors.

The 3D faces are split in to 70 horizontal slices (Figure D.3). Figures D.4, D.5 and D.6 show the variations of the slices 37_{th}, 45_{th} and 53_{th} of the 3D faces respectively. Note that the meshing of the point cloud is performed using the Rapidform-XOR software (62).

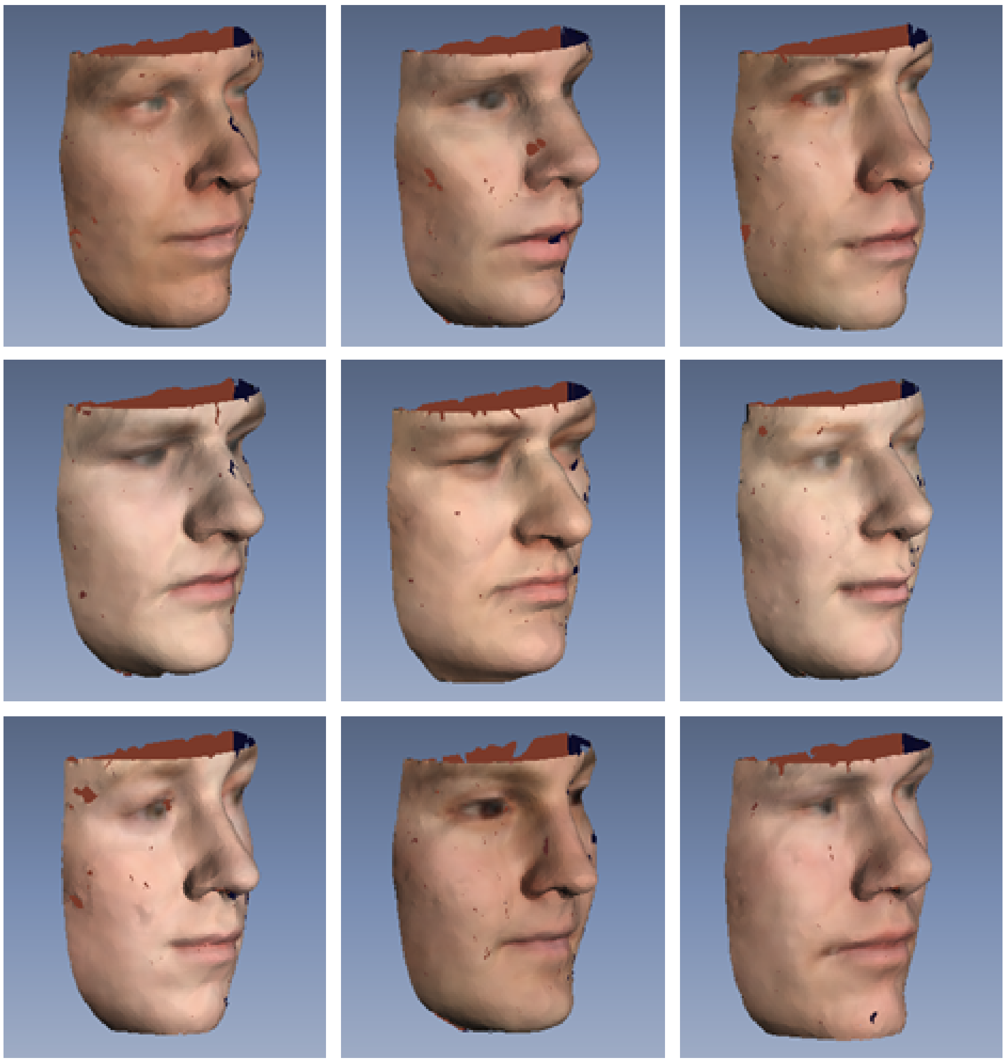


Figure D.1: 3D heads in the test set $\mathcal{S}_{test}$ (9 heads) generated from the 3D Basel face model (57).

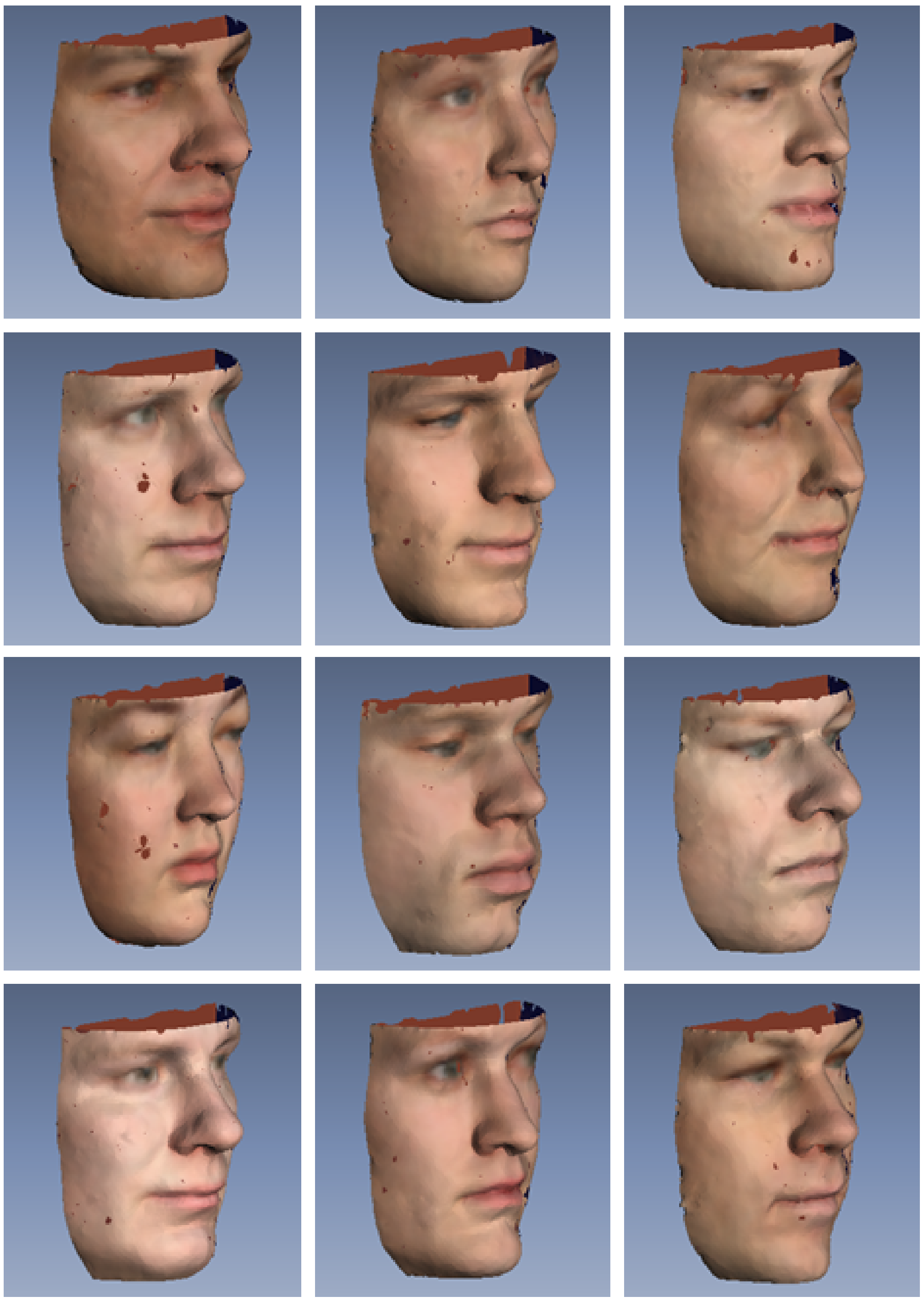


Figure D.2: Some examplar 3D heads from the training set $\mathcal{S}_{prior}$ (12 heads shown out of a total $N = 35$ heads) generated from the 3D Basel face model (57).

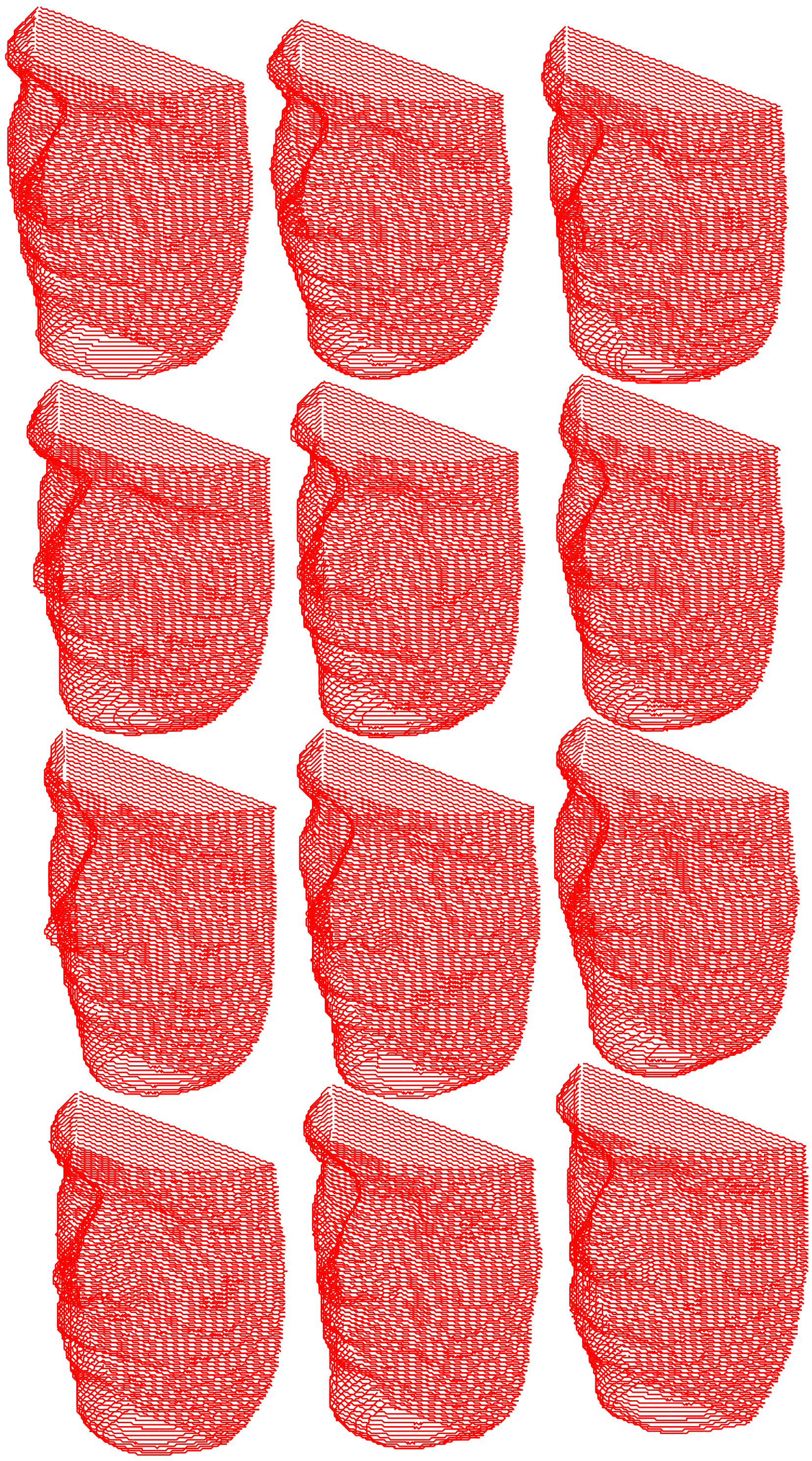


Figure D.3: 70 slices from the 3D Basel face model (57).

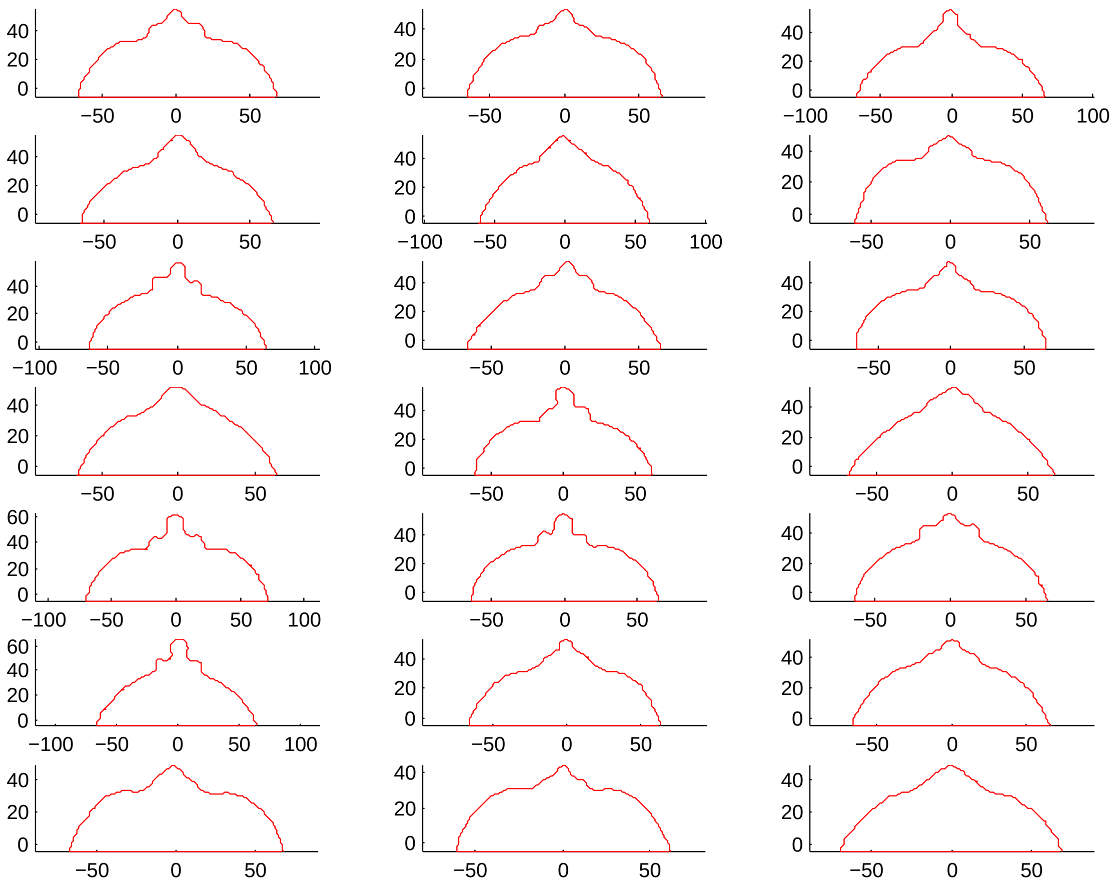


Figure D.4: The 37th horizontal slice variations of 21 different 3D heads amongst the 35 in $\mathcal{S}_{prior}$.

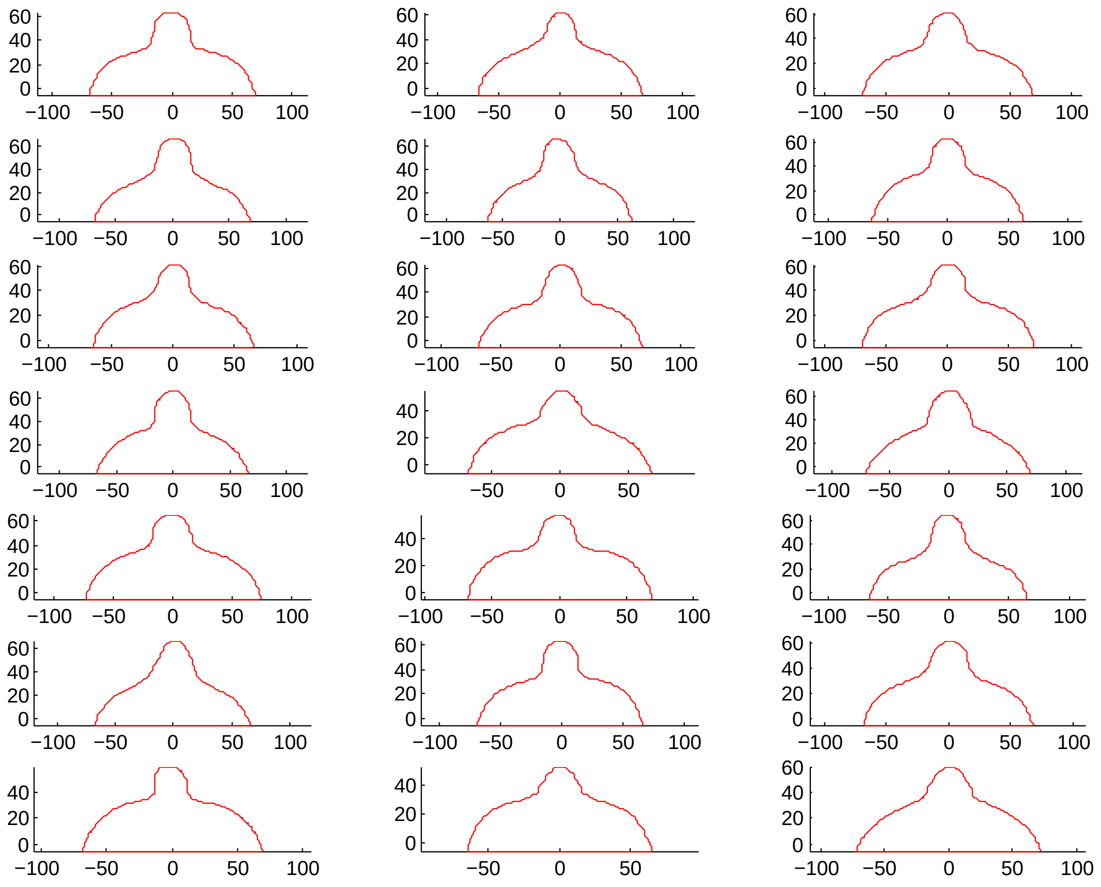


Figure D.5: The 45th horizontal slice variations of 21 different 3D heads amongst the 35 in $\mathcal{S}_{prior}$.

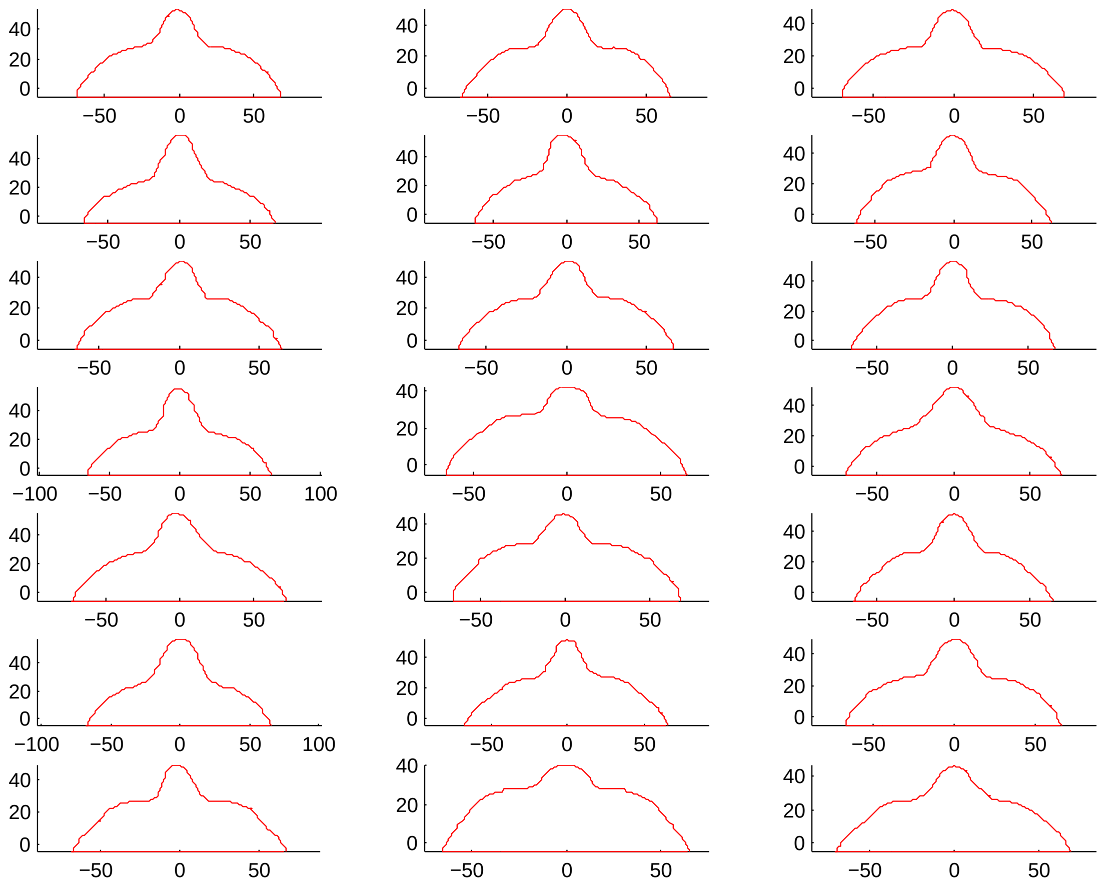


Figure D.6: The 53th horizontal slice variations of 21 different 3D heads amongst the 35 in $\mathcal{S}_{prior}$.

D.2 Experimental Results in 2D

Figures D.7 and D.8 present the additional results of slice reconstructions with the methods tested in section 5.3.1.2.

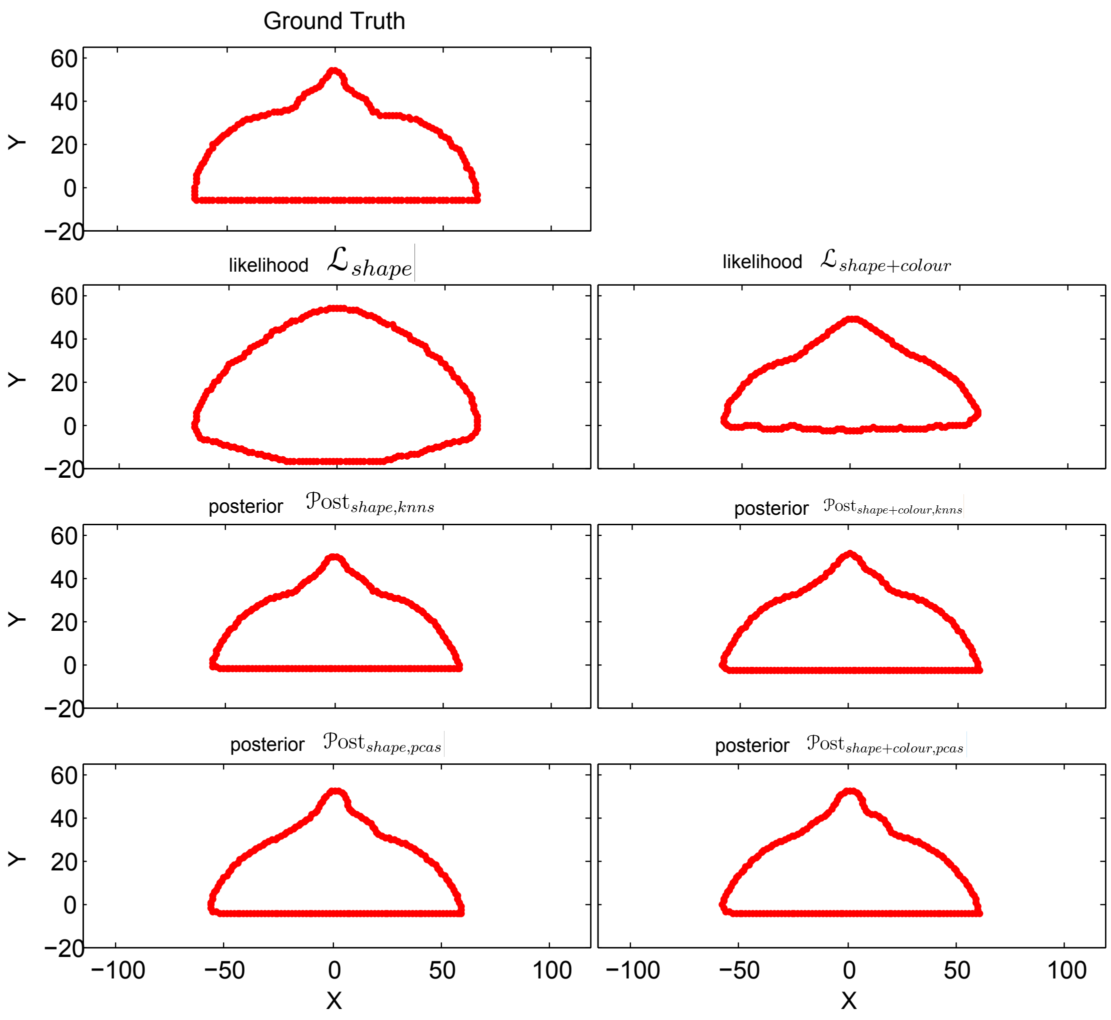


Figure D.7: Reconstruction of the 37th slice using 36 camera views.

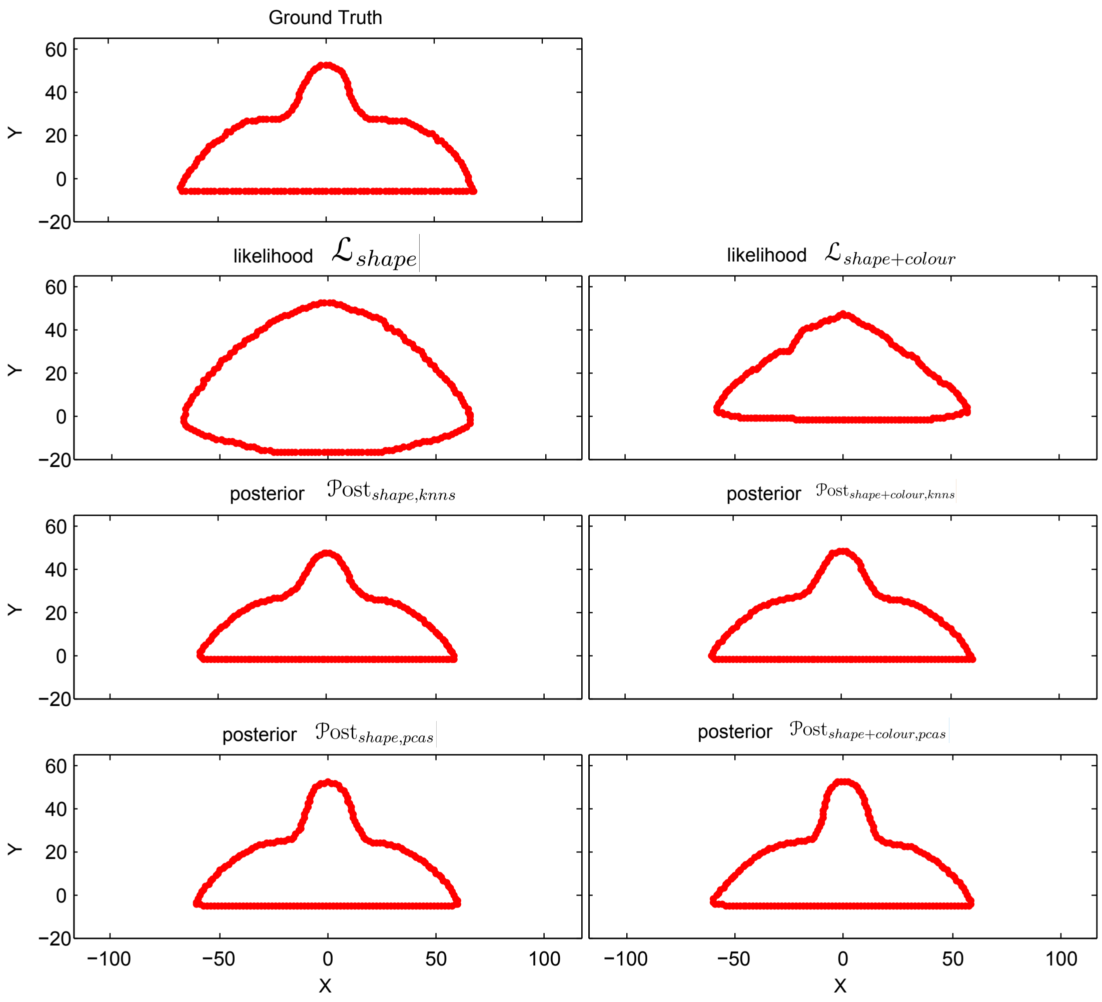


Figure D.8: Reconstruction of the 53th slice using 36 camera views.

D.3 Experimental Results in 3D

Figures D.9 and D.10 show additional 3D face surface reconstructions (see Section 5.3.1.4). Figures D.11 and D.12 shows the results when both the shape and the colour are estimated.

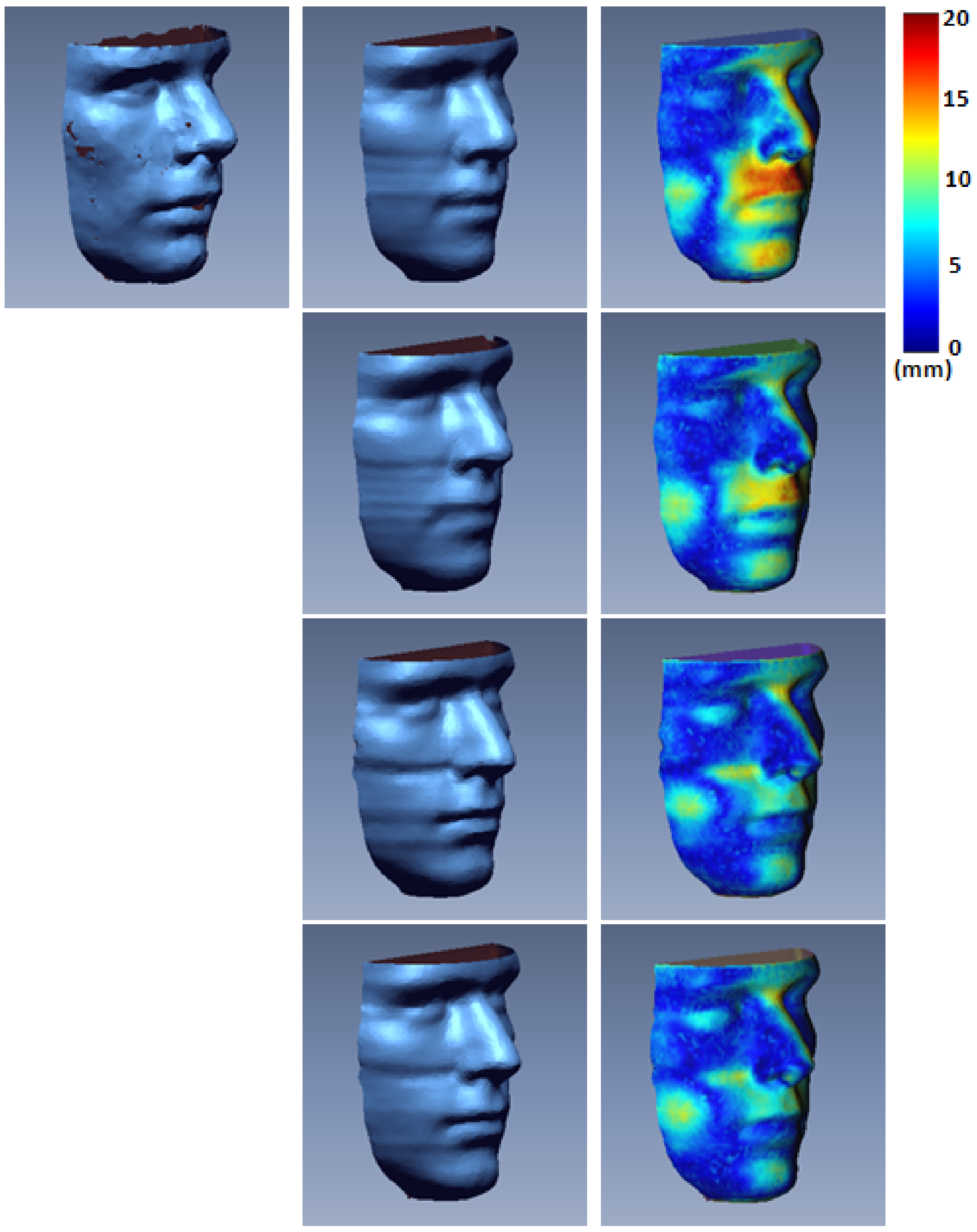


Figure D.9: 3D Face reconstruction: ground truth (left column), the estimates (middle) and the distance between the estimates and the ground truth in mm (right column). From top to bottom, reconstructions using $\mathcal{P}\mathrm{ost}_{shape,knns}$, $\mathcal{P}\mathrm{ost}_{shape+colour,knns}$, $\mathcal{P}\mathrm{ost}_{shape,pcas}$ and $\mathcal{P}\mathrm{ost}_{shape+colour,pcas}$.

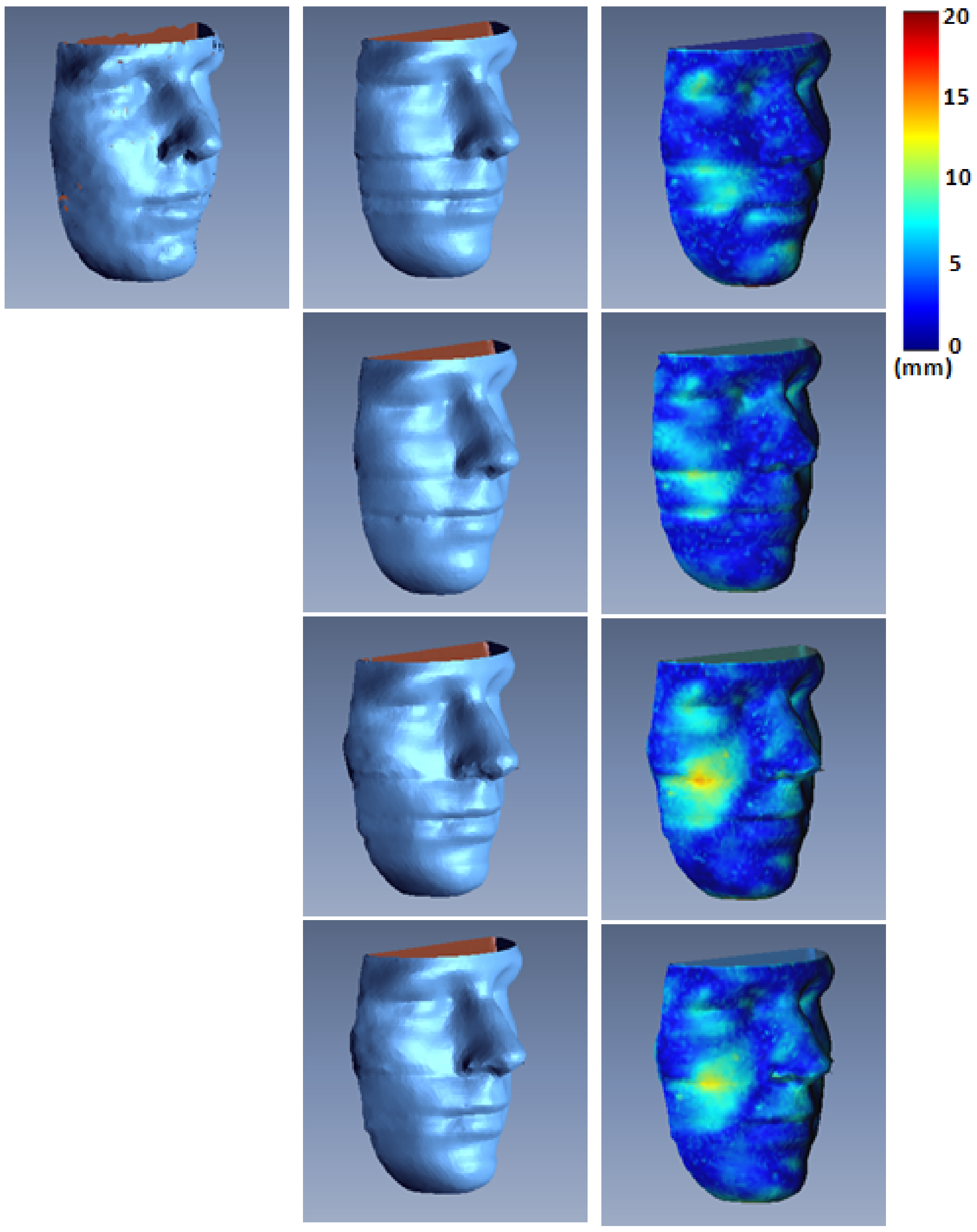


Figure D.10: 3D Face reconstruction: ground truth (left column), the estimates (middle) and the distance between the estimates and the ground truth in mm (right column). From top to bottom, reconstructions using $\mathcal{P}\text{ost}_{shape,knns}$, $\mathcal{P}\text{ost}_{shape+colour,knns}$, $\mathcal{P}\text{ost}_{shape,pcas}$ and $\mathcal{P}\text{ost}_{shape+colour,pcas}$.

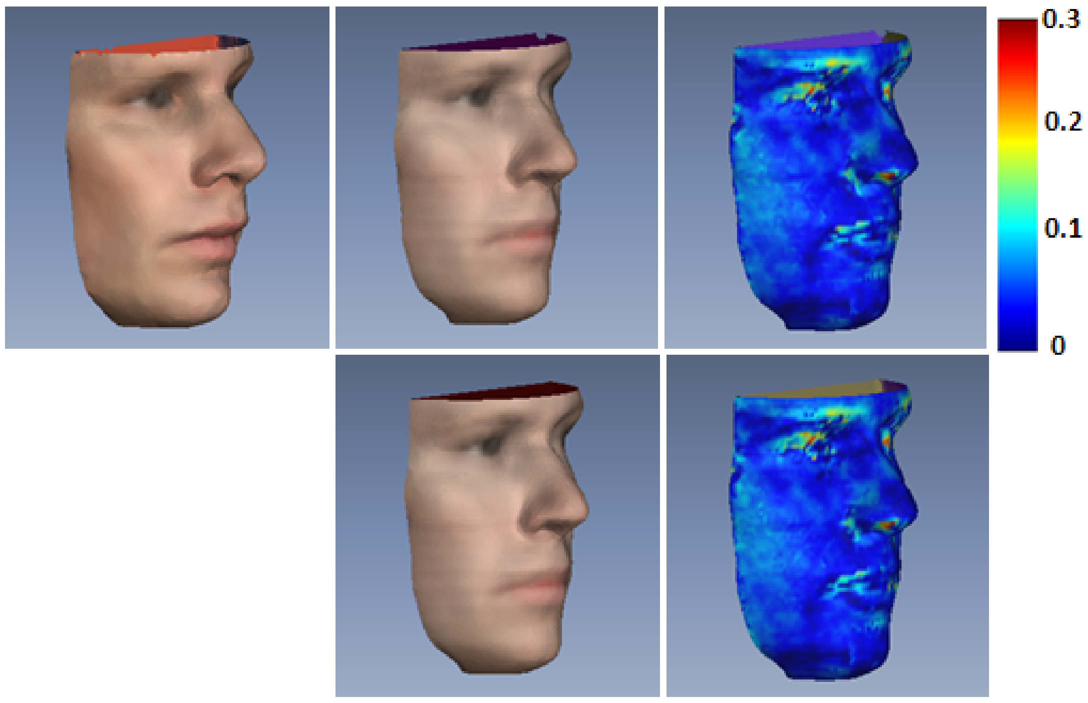


Figure D.11: 3D Face reconstruction with texture mapping (middle column) and its texture error surfaces (right column). Ground-truth, (top-left), reconstruction with $\mathcal{P}\mathrm{ost}_{shape+colour,knns}$ with its distance to the ground truth (top row) and reconstruction with $\mathcal{P}\mathrm{ost}_{shape+colour,pcas}$ with its distance to the ground truth (bottom row).

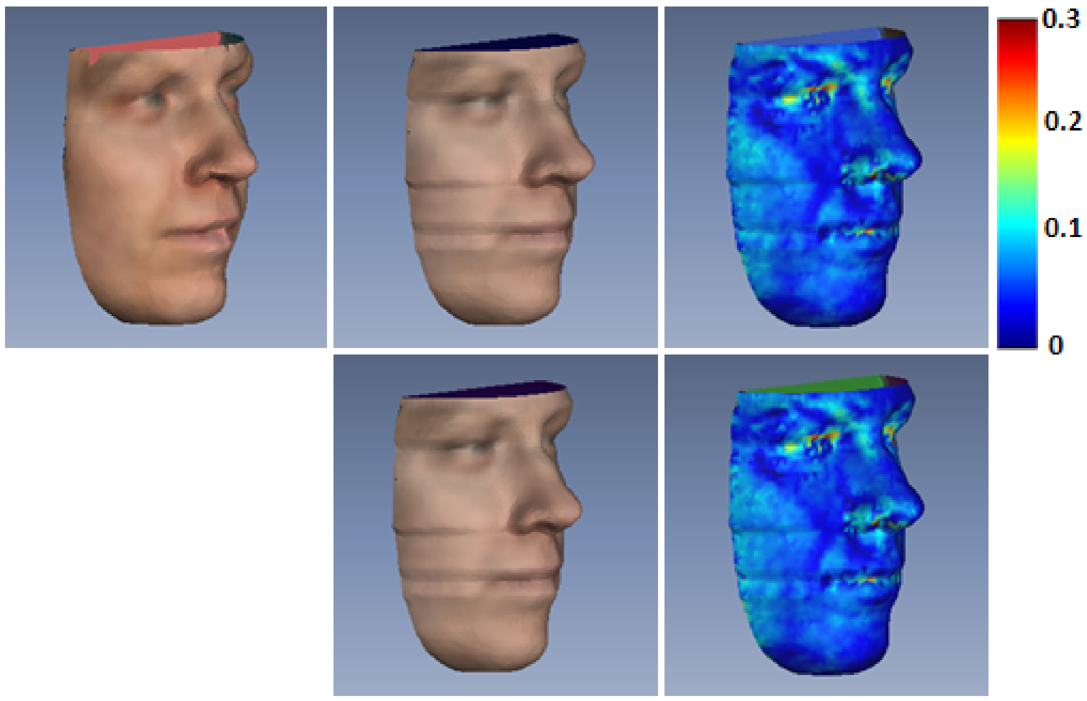


Figure D.12: 3D Face reconstruction with texture mapping (middle column) and its texture error surfaces (right column). Ground-truth, (top-left), reconstruction with $\mathcal{P}\mathrm{ost}_{shape+colour,knns}$ with its distance to the ground truth (top row) and reconstruction with $\mathcal{P}\mathrm{ost}_{shape+colour,pcas}$ with its distance to the ground truth (bottom row).

Appendix E

2D slices in ALOI database

In this appendix, we show the 2D shapes used for the experimental results using ALOI database (29) in Section 5.3.2. Six objects have been used for these experiments and each object has 7 images captured from different viewing angles $[0°, 15°, 30°, 45°, 60°, 75°, 90°]$. The test set $\mathcal{S}_{test}$ collects the three images at angles $[15°, 45°, 75°]$ for each object and the other 4 images $[0°, 30°, 60°, 90°]$ are for the training set $\mathcal{S}_{prior}$ (total number of exemplars $N = 6 \times 4 = 24$). The training set corresponds to the exemplars to approximate a prior. Figures E.1, E.2, E.3, E.4, E.5 and E.6 present five objects with their contours in $\mathcal{S}_{prior}$ and $\mathcal{S}_{test}$. The sixth object is presented in Figure 5.25.

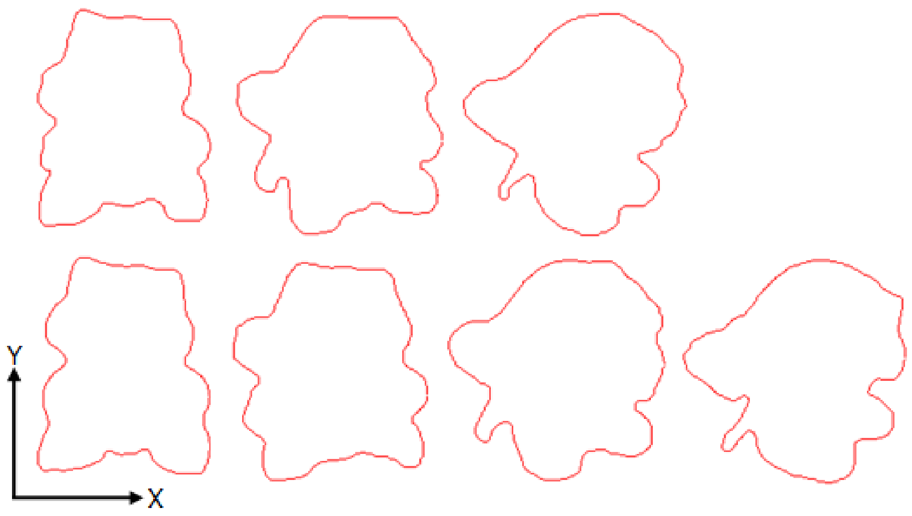


Figure E.1: Top row: 2D slices $[15°, 45°, 75°]$ in the test set $\mathcal{S}_{test}$, bottom row: 2D slices $[0°, 30°, 60°, 90°]$ in the training set $\mathcal{S}_{prior}$.

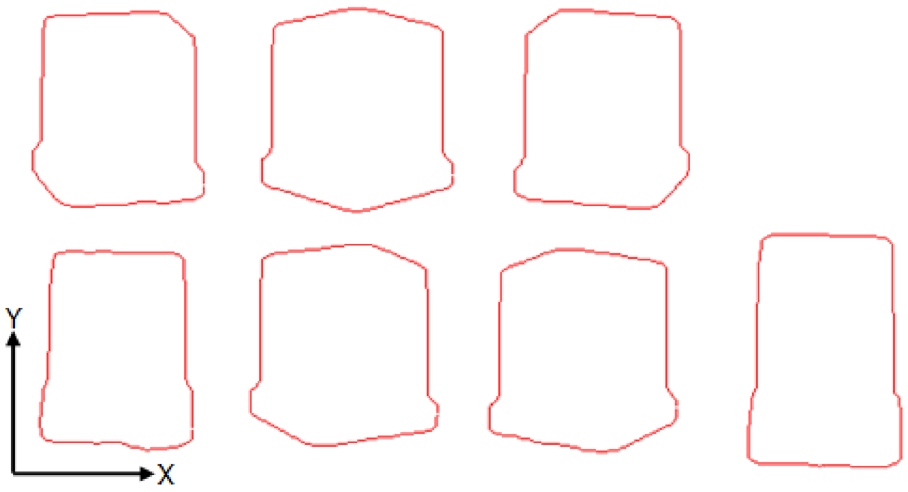


Figure E.2: Top row: 2D slices $[15°, 45°, 75°]$ in the test set $\mathcal{S}_{test}$, bottom row: 2D slices $[0°, 30°, 60°, 90°]$ in the training set $\mathcal{S}_{prior}$.

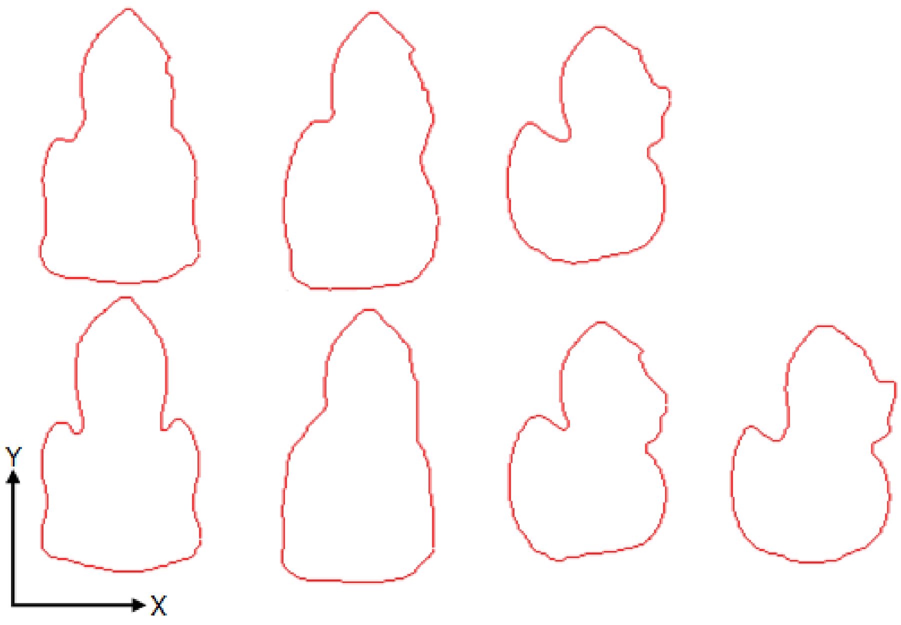


Figure E.3: Top row: 2D slices $[15°, 45°, 75°]$ in the test set $\mathcal{S}_{test}$, bottom row: 2D slices $[0°, 30°, 60°, 90°]$ in the training set $\mathcal{S}_{prior}$.

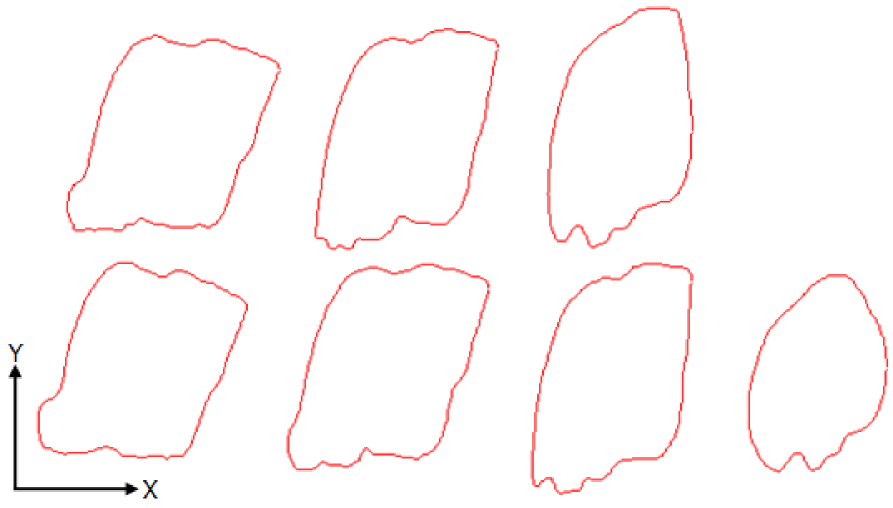


Figure E.4: Top row: 2D slices $[15°, 45°, 75°]$ in the test set $\mathcal{S}_{test}$, bottom row: 2D slices $[0°, 30°, 60°, 90°]$ in the training set $\mathcal{S}_{prior}$.

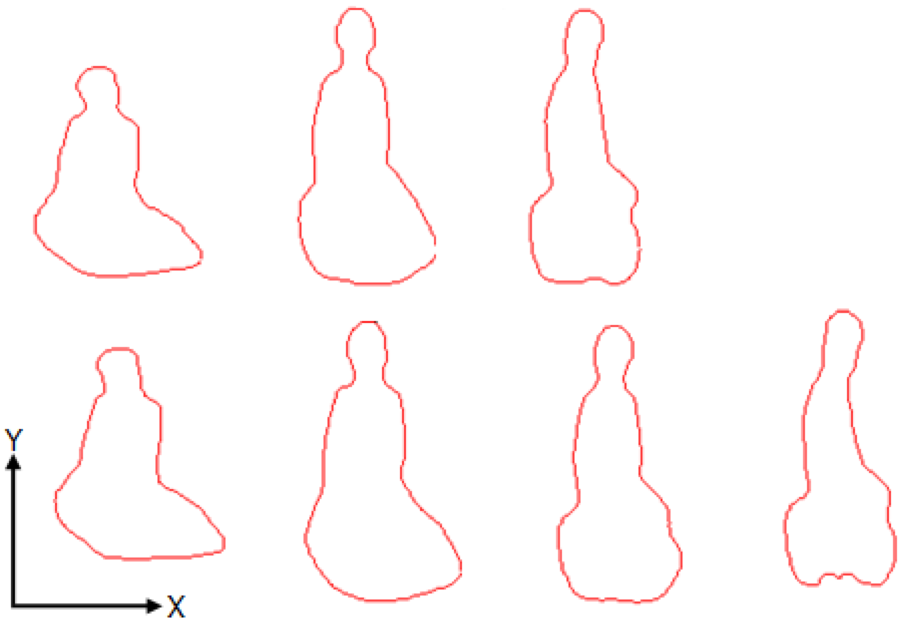


Figure E.5: Top row: 2D slices $[15°, 45°, 75°]$ in the test set $\mathcal{S}_{test}$, bottom row: 2D slices $[0°, 30°, 60°, 90°]$ in the training set $\mathcal{S}_{prior}$.

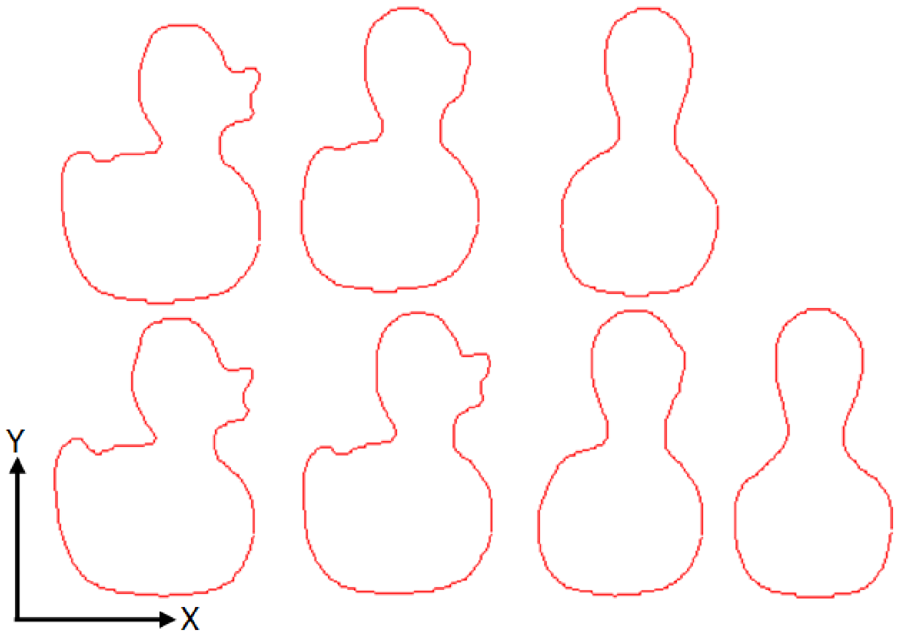


Figure E.6: Top row: 2D slices $[15°, 45°, 75°]$ in the test set $\mathcal{S}_{test}$, bottom row: 2D slices $[0°, 30°, 60°, 90°]$ in the training set $\mathcal{S}_{prior}$.

Appendix F

Face Components Detection

In this appendix[1] local descriptors are assessed for their ability to detect faces in images, and to label the individual components-thereof (eyes, nose and mouth). In the case of multiple camera systems such as video surveillance, faces can appear at a low resolution and be viewed from different angles. For the sake of simplicity, and to stay within the scope of this thesis, this section will focus solely on how to address the issue of low resolution images, and will assume that only views of the front of the face are of interest

F.1 Feature description

Finding correspondences between two images of the same scene or object is a part of many computer vision applications such as camera calibration, 3D reconstruction, and object recognition. In particular, object detection and recognition in cluttered scenes require local features that are unaffected by nearby clutter or partial occlusion. Therefore, the features have to be robust to noise, detection errors and, geometric and photometric variations. On the other hand, the features must be sufficiently distinctive to identify specific objects. In addition, the operating speed has to be considered.

Several interest point detectors have been proposed such as the Hessian detector, Harris detector (32), Hessian/Harris-Laplacian/Affine detector based on

[1]This research has been published in Kim and Dahyot (35).

affine normalisation around Harris and Hessian points (54), MSER (Maximally Stable Extremal Regions) detector (50), SIFT (Scale Invariant Feature Transform) detector (46), and SURF (Speeded Up Robust Features) detector (3) since the first corner detector, which uses the Moravec corner detection algorithm, was developed in the late 1970.

In general, the search for discrete image correspondences can be divided into three main steps: 1) detection, 2) description, 3) matching. Feature points are first selected using interest point detectors. Once the feature points have been detected, the neighbourhood of every feature point is represented by a feature vector as a descriptor. Finally, the descriptors are matched between different images. The process of matching is often based on the distance between vectors using the Mahalanobis or Euclidean distance. The dimension of the descriptor has a direct impact on the operating time, so the fewer the dimensions, the better.

In the next section, two representative methods, SIFT in Lowe (46) and SURF in Bay et al. (3), will be introduced. This is because, according to Mikolajczyk and Schmid (55), the SIFT descriptor has been shown to outperform others, and SURF is the most recently proposed descriptor, demonstrating a comparatively fast operating speed and a reasonably distinctive performance. The following section will now go on to describe each of these descriptors in further detail.

F.1.1 Scale Invariant Feature Transform (SIFT)

SIFT is invariant to image translation, scaling, and rotation, and partially invariant to illumination changes. The computation of SIFT is divided into four stages, scale-space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor.

The first stage involves the detection of interest points (called keypoints in the SIFT framework), which is accomplished by searching the scale space. The scale space is given by

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{F.1}$$

where $*$ is the convolution operation and $I(x, y)$ is an input image. $G(x, y, \sigma)$ is a variable-scale Gaussian, defined by

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{F.2}$$

DoG (Difference of Gaussians) is used for detecting interest points by taking extrema (maxima and minima) in the scale space. The DoG, $D(x,y,\sigma)$ is computed by the difference between two images, one with scale k times the other. $D(x,y,\sigma)$ is given by

$$D(x,y,\sigma) = L(x,y,k\sigma) - L(x,y,\sigma) \tag{F.3}$$

Figure F.1 illustrates the procedure of creating the scale space by the DoG. For each octave of scale space, the initial image is repeatedly convolved with Gaussian to produce the set of scale space images shown in the left of Figure F.1. Adjacent Gaussian images are subtracted to produce the DoG images in the right of Figure F.1 to create the scale space.

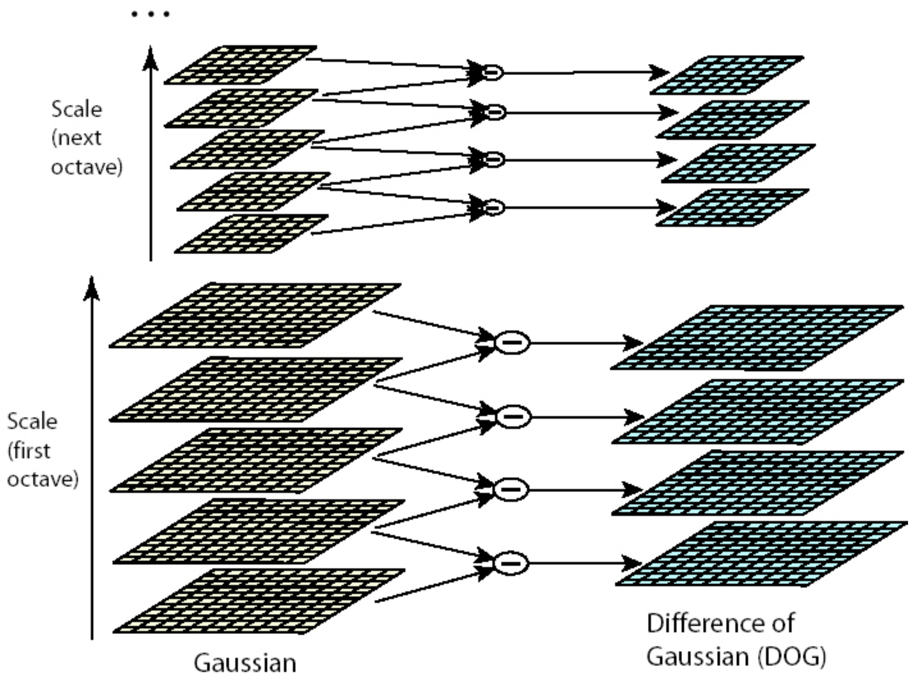


Figure F.1: The construction procedure of the scale space (46).

In Figure F.2, maxima and minima of the scale space are detected by comparing a pixel (marked with X) to its 26 neighbours in 3×3 regions at the current and adjacent scales (marked with circles). If the pixel is a local maximum or minimum, it is selected as a candidate keypoint.

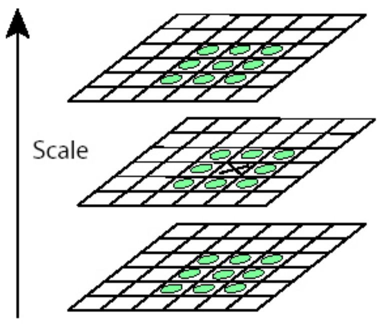


Figure F.2: Finding maxima and minima of the scale space (46).

The second stage is keypoint localization. Extrema detection in the scale space produces a number of candidate keypoints, some of which are unstable. The purpose of this stage, therefore, is attempt to eliminate points from the candidates which have low contrast or are poorly localized on an edge. This is achieved by some conditions based on a contrast threshold and the ratio of principal curvatures in the scale space.

The third stage is orientation assignment. At each keypoint, an orientation is selected by determining the peak of a histogram of local image gradient orientation. The gradient orientation histogram is calculated in the neighbourhood of the keypoint. The contribution of each neighbouring pixel is weighted by the gradient magnitude and a Gaussian window with a σ that is 1.5 times the scale of the keypoint. Peaks in the orientation histogram correspond to dominant directions of local gradients.

The final stage is keypoint description. The gradient information of the neighbour keypoints is used to obtain a descriptor. The descriptor is computed as a set of orientation histograms on 4×4 sub-patches. Each orientation histogram has 8 bins and forms a single feature vector. This leads to a SIFT feature vector with $4 \times 4 \times 8 = 128$ elements. Figure F.3 (left) illustrates that the gradient magnitudes and orientations are sampled around the keypoint location. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarising the contents over a 4×4 sub-region, as shown in Figure F.3 on the right, as the length of each arrow corresponding to the sum of the gradient magnitudes near the direction within the region.

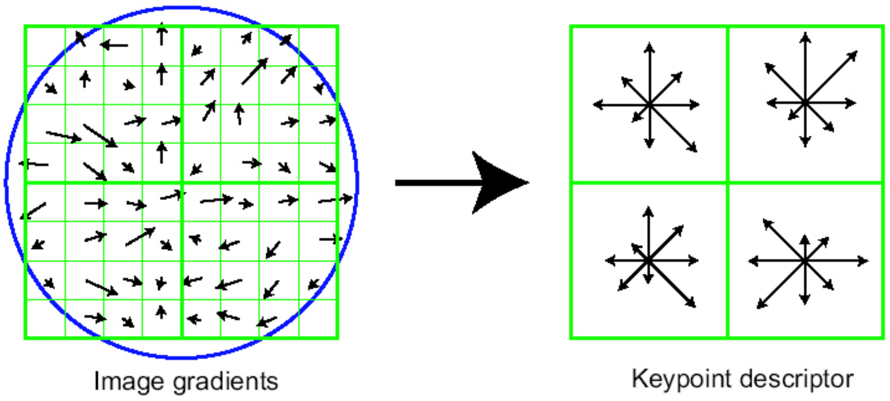


Figure F.3: SIFT keypoint description procedure (46).

F.1.2 Speeded Up Robust Features (SURF)

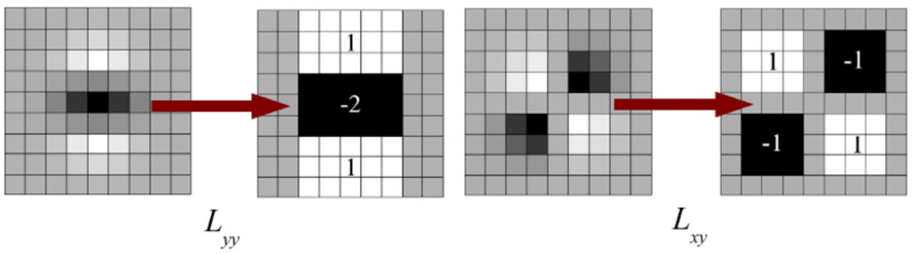


Figure F.4: Approximated second order derivatives with box filter (3).

One of the main advantages of SURF is to be able to compute distinctive descriptors quickly. In addition, the SURF descriptor is invariant to common image transformations including image rotation, scale changes, illumination changes, and small changes in viewpoint. This section provides a brief summary of its construction process which is broken into 2 steps: interest point localization and interest point description.

In the interest point localization step, an interest point detection method is based on the Hessian matrix. Given a point $X = (x, y)$ in an image I, the Hessian matrix $H(X,\sigma)$ at X and scale σ is defined as

$$H(X,\sigma) = \begin{bmatrix} L_{xx}(X,\sigma) & L_{xy}(X,\sigma) \\ L_{xy}(X,\sigma) & L_{yy}(X,\sigma) \end{bmatrix}, \tag{F.4}$$

where $L_{xx}(X,\sigma)$ is the convolution of the Gaussian second order derivative $\frac{\partial^2}{\partial x^2}g(\sigma)$ with the image I at point X, and similarly for $L_{xy}(X,\sigma)$ and $L_{yy}(X,\sigma)$. The convolution of the Gaussian second order derivative, L, is approximated by box filters (mean or average filter) shown in Figure F.4. Also, the calculation of the box filters is operated on integral images which allow fast operation. The location and scale of interest points are selected by relying on the determinant of the Hessian matrix. Interest points are localized in scale and image space by applying non-maximum suppression in a $3\times3\times3$ neighbourhood, using a similar approach to SIFT.

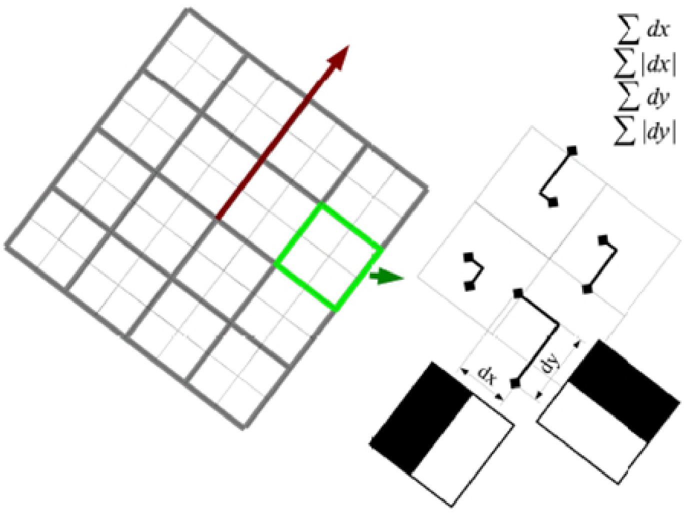


Figure F.5: SURF descriptor (3).

In the description step, a unique orientation of each descriptor is first assigned in order to achieve invariance to image rotations. The orientation is computed by accumulating Haar wavelet responses in a circular region around the detected interest points. The Haar wavelets can be quickly computed using integral images. When the dominant orientation is estimated, SURF descriptors are constructed by extracting square regions around the interest points. The windows are split up into 4×4 sub-regions shown in Figure F.5. The intensity pattern (first derivatives) of each sub-region is described by a vector containing 4 elements, $v = [\sum d_x, \sum d_y, \sum |d_x| \sum |d_y|]$. The length of the SURF descriptor is therefore $4 \times 4 \times 4 = 64$ dimensions.

F.2 Context

Automatic salient feature extraction is an essential task for the analysis of video streams in video surveillance systems. Of particular interest for the purpose of this study is the specific case of determining the features of faces (around 30×30 pixels) in a low resolution environment using multiple camera views, where the general face detectors proposed (79; 80; 81; 86) do not perform well. Recent developments in the field of face detection involve the use of local informative descriptors such as Haar wavelets in Viola and Jones (81), SIFT (Scale Invariant Feature Transform) in Lowe (46) and SURF (Speeded Up Robust Feature) in Bay et al. (3). The use of local descriptors versus global ones usually ensures the system a certain natural robustness to partial occlusion. Moreover, adequate normalisation of the descriptors allows them to be invariant to some geometrical transformations like rotation, scale changes or illumination. These are interesting properties for the detection of an object appearing at different scales or orientations in the images.

Once the sets of representative descriptors are available to train both the target object and its complement (non-object), object detection is performed by classifying new observations between those two classes. Boosting and SVMs are classifiers that have been applied to face detection and have provided comparable results. However, there are still several challenges to be dealt with in order to get a reliable face feature detector. Low resolution images, partial occlusion, variation in lighting conditions or head-pose changes are all difficulties to be overcome. As the environment becomes more complex, the process of reliable feature extraction becomes more important than the performance of classifiers. In particular, in such a complex environment, it is necessary to extract salient features which are able to steadily discriminate each different class (e.g. face and non-face).

The goal of this appendix is to robustly detect face components such as a nose, mouth, and eyes in those problems based on a local feature descriptor (SURF) with a SVMs classifier.

F.3 Methodology

In the varied and complex environment of images or video sequences, defining salient features able to steadily discriminate plays a major role in target object detection. The latest technologies of feature extraction, SIFT and SURF features, are assessed for classification of faces and non-faces. SURF is found to perform better in terms of discrimination performance between faces and non-faces, as illustrated in Figure F.6. This is because SURF features are more densely and consistently located around facial components, as shown in Figure F.7. SURF features are also faster to compute.

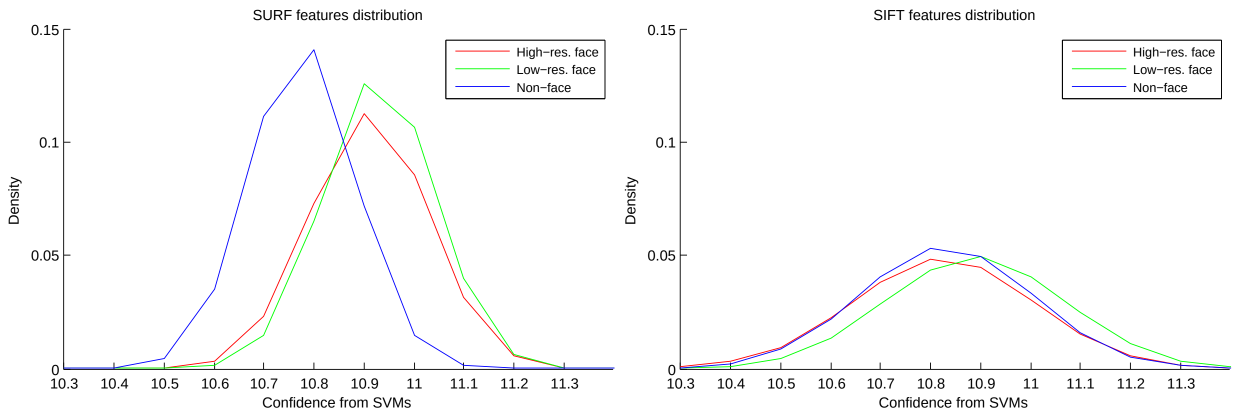


Figure F.6: Confidence distribution computed by SVMs.

For classification, the AdaBoost algorithm has two drawbacks: the length of training time required is long, and a large quantity of training images is required. On the other hand, SVMs have faster training times and also generalise well on smaller training sets.

In this chapter, a feature-based method is proposed not only to classify salient points between two classes (face and background(non-face)) but also to detect facial components. A SURF descriptor is used to generate informative feature

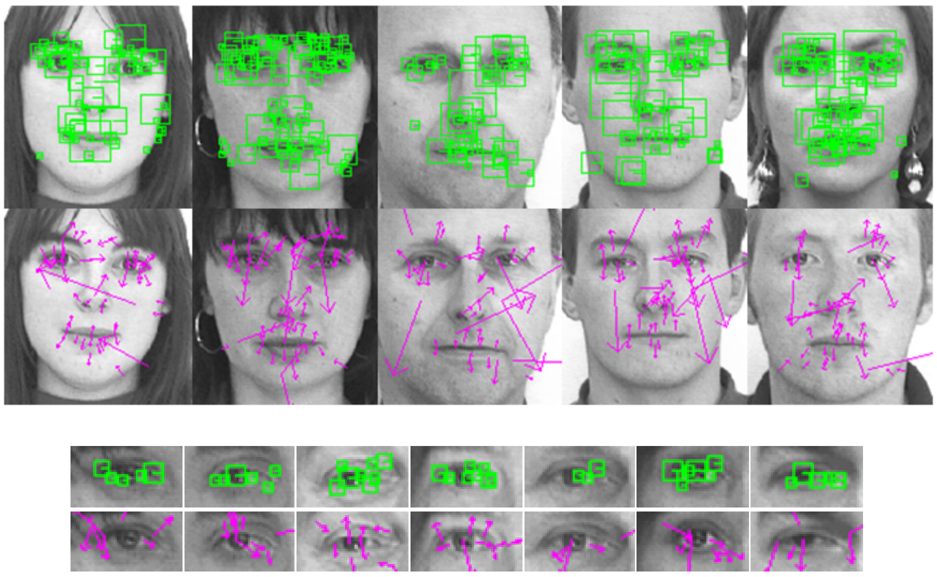


Figure F.7: SURF and SIFT features on faces and eyes (Top: SURF results, bottom: SIFT results).

vectors, and SVMs are used as classifiers. The proposed system consists of a two-level hierarchy of SVMs classifiers. On the first level, a single classifier checks whether feature vectors belong to facial images or not. On the second level, component labelling is operated using component classifiers for the eye, mouth, and nose. This approach is time-efficient since no additional window-scanning is needed. An outline of the proposed system is illustrated in Figure F.8.

F.4 Step1: Skin Region Segmentation

This approach begins with the segmentation of skin areas using the YCbCr (Luminance, chrominance-blue, chrominance-red) colour space. A luminance element largely depends on the variation of illumination, and therefore thresholds are only defined in relation to Cb and Cr, which are more robust to the variation of illumination. A result of the skin region segmentation is shown in Figure F.9. SURF descriptors are then computed on the skin region.

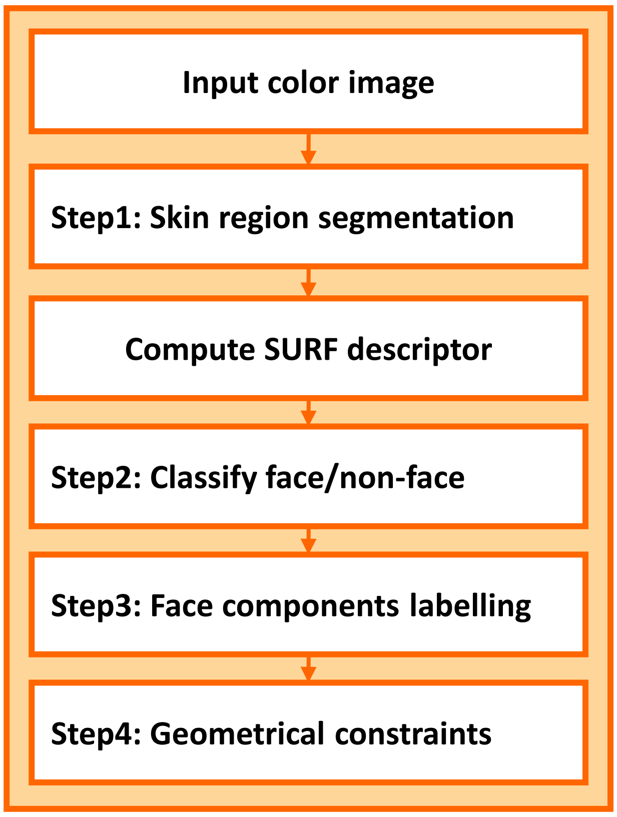

Input color image
Step1: Skin region segmentation
Compute SURF descriptor
Step2: Classify face/non-face
Step3: Face components labelling
Step4: Geometrical constraints

Figure F.8: Overview of the proposed system.

F.5 Step2: Classification Face/Non-face

The next step is for two layer SVMs classifiers to check the SURF descriptors. A linear SVM classifier of the first layer is trained using SURF descriptors as feature vectors from faces and non-faces(not containing faces) shown in Figure F.10. 251 points from faces and 340 from non-faces SURF descriptors are used to train SVMs for the first classifier. Figure F.11 shows the result of the first layer classifier: green points are classified as a face. Most of the facial features are located on the face region of the image, however some false alarms also appear on the window and the corners on the wall. This first result is promising, but shows that a further is needed to disqualify false alarms.

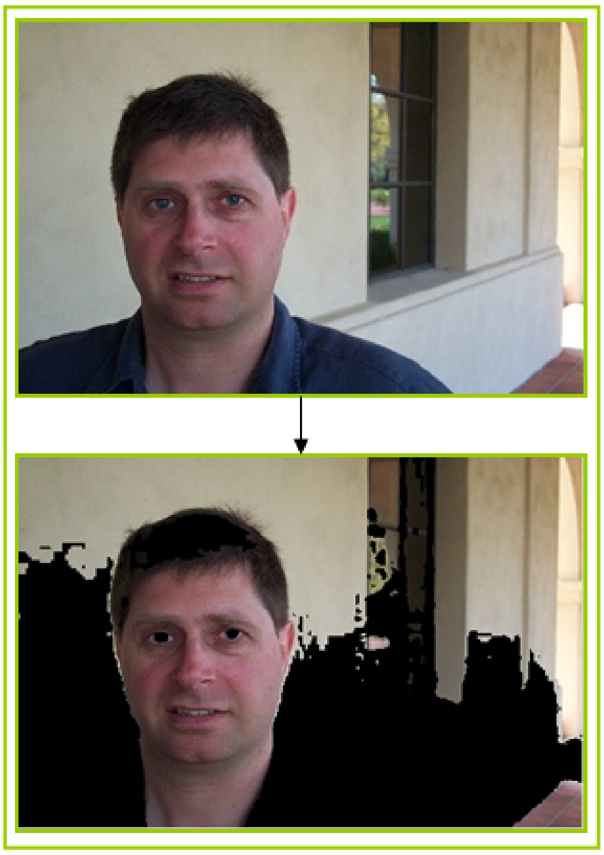


Figure F.9: Skin colour segmentation results.

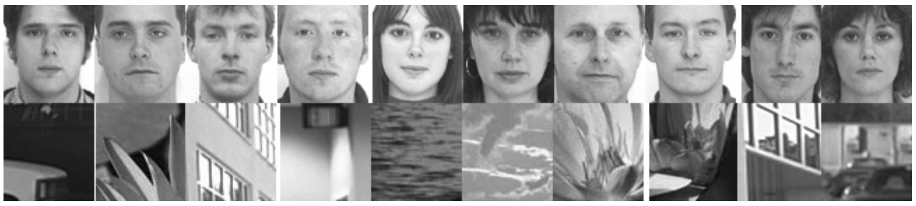


Figure F.10: Example images (faces and non-faces) in training database for SVMs.

F.6 Step3: Face Components Labelling

The second layer classifiers are to assign component labels to the descriptors refined by the first layer classifier. In other words, for each feature point classified as a face in the previous step, another label corresponding to the subclasses left

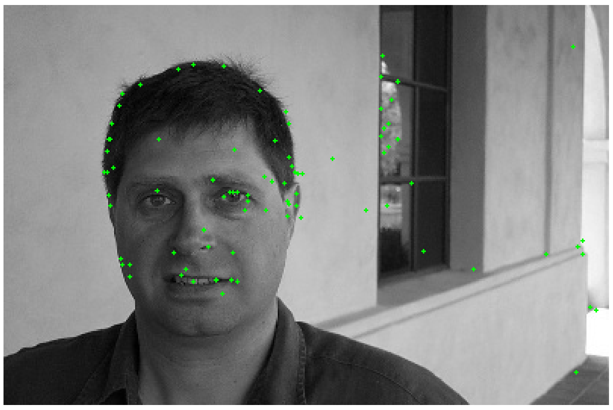


Figure F.11: The result of the first layer classifier.

eye, right eye, nose, or mouth, must be added. A linear SVM classifier for each of these subclasses is trained, resulting in the creation of four classifiers, illustrated in Figure F.12. For example, a left eye classifier is computed by training it to use the descriptors extracted from the left eye as positive examples, and the mouth and nose images as negative examples. The right eye subclass is excluded from the negative samples in the training since this is too similar to the target left eye subclass. Also, the mouth classifier is trained by all the other subclasses containing the nose, and right and left eyes, as negative examples. Training data images are manually cropped at high resolution 130×140 to have a maximum number of selected features: 100 from left eyes, 126 from right eyes, 149 from mouths and 61 from noses SURF descriptors are used for training.

F.7 Step4: Geometrical Constraints

This final step is to eliminate the wrongly classified descriptors and also to estimate the position and scale of the facial components using their label (subclass) and geometrical information. To use geometrical information, an eye pairing process is first performed owing to the fact that the most robust classification results are obtained from the left and right eye descriptors in facial components. Once eye features have been localized on the basis of the spatial distance between the two eyes, the coordinates of the other facial descriptors are rotated until a frontal view of the face is reached, where both eyes have the same y value. This

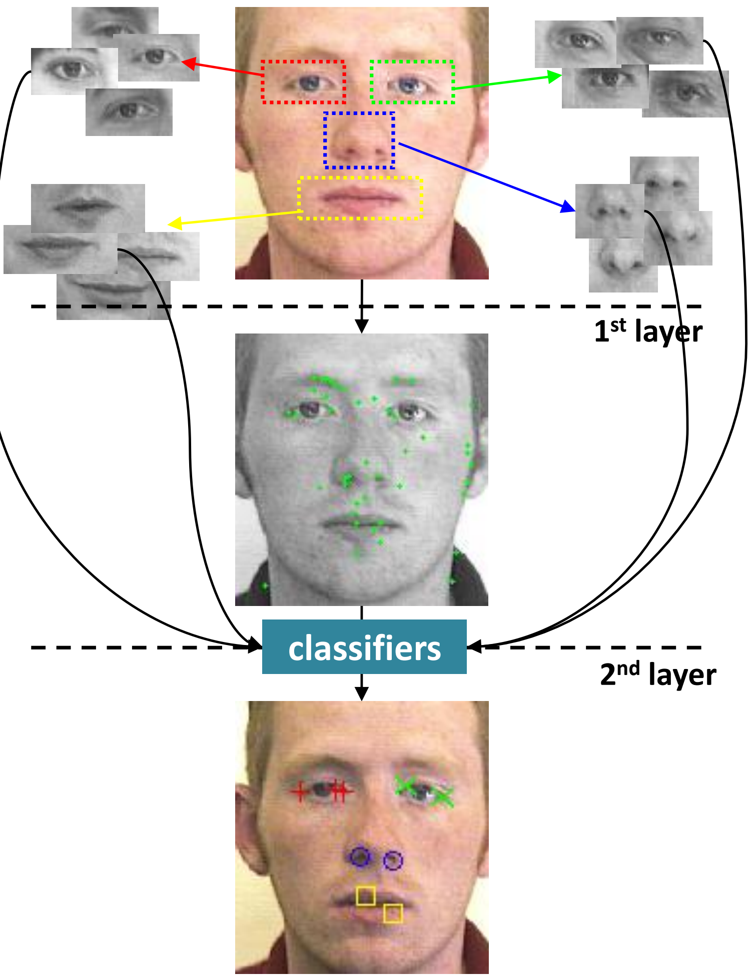


Figure F.12: The second layer classifier (Red colour: left eye, Green colour: right eye, Blue colour: nose, Yellow colour: mouth).

facilitates interpretation of each facial component using geometrical constraints. When the eye pairing procedure has finished, the position of the nose and mouth can be estimated through label information and geometrical constraints. This process also eliminates falsely classified features. For instance, in Figure F.13, the condition of the nose position is such that the descriptors of the labelled nose have to exist in the triangle region. The condition can eliminate some labelled nose descriptors.

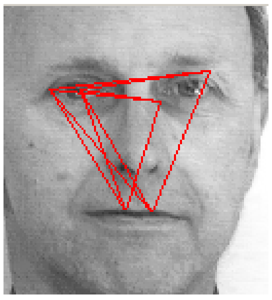


Figure F.13: Geometrical constraint for a nose position.

F.8 Experimental Results

The test database for faces consists of three subsets which contain high, lower and lowest resolution images (test database dose not contain training images.). The purpose of the subsets is to evaluate the performance with respect to the scaling factor. The high resolution 130×140 subset of the faces comprises of 100 face images randomly selected and cropped from the AR face database (49) and Caltech face database (84). The other subsets, the lower resolution 65×70 and the lowest resolution 43×46 subsets of the faces, are created by resizing high resolution images. The example and results are shown in Figure F.14.

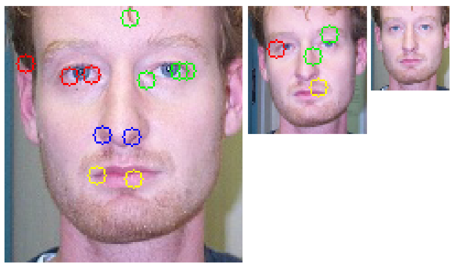


Figure F.14: Examples in face database (Left: a high resolution image, Centre: a lower resolution image, Right: a lowest resolution image). The green, red, blue and yellow circles correspond to the right eye, left eye, nose, and mouth, respectively.

To investigate the performance of the proposed approach, two sets of experiments were conducted, with the following aims:

1. To calculate detection rates using true positives and error rates using false positives of each classifier in the face dataset.
2. To calculate error rates using false positives of each classifier in the non-face dataset.

In the first experiment, true positives and false positives from the face dataset were counted according to whether a labelled descriptor existed in the correct region taken from the coordinate data file (ground-truth) or not. For example, if a detected descriptor labelled as a left eye is in the region of the left eye, the case is a true positive; if not, the case is a false positive. The detection rates are computed based on the true positives of the detected descriptors. The number of the false positives is divided by the total number of the detected descriptors to calculate the error rates.

In the second experiment, false positives from the non-faces are counted. All detected descriptors become false positives, and then the error rates are calculated by using the same method to the first experiment. The result is shown in Table F.1 and Figure F.15.

Table F.1: Detection accuracy (%) of the subclass classifiers.

DB\Classifier		Left eye	Right eye	Mouth	Nose	No.images	No. total descriptors
Face high res.	Detection rate	97	97	93	72	100	9095
	Error rate	6.4	8.2	4.7	0.57		
Face lower res.	Detection rate	88	93	56	28		2914
	Error rate	3.0	3.7	2.4	0.68		
Face lowest res.	Detection rate	43	49	5	1		1218
	Error rate	0.65	0.98	0	0.16		
nonFace	Error rate	6.7	6.7	6.7	0.87	242	18012

In the test database of high and lower resolutions, classification results were extremely accurate for the eyes and mouth, but not for the nose. However, at the lowest resolution, detection results were unreliable (below 50%) for all components . This method depends on the salient point descriptors. Although a SURF descriptor is invariant to scales, the system does not perform well at small faces. This is because, in small faces, there are a few features extracted from a

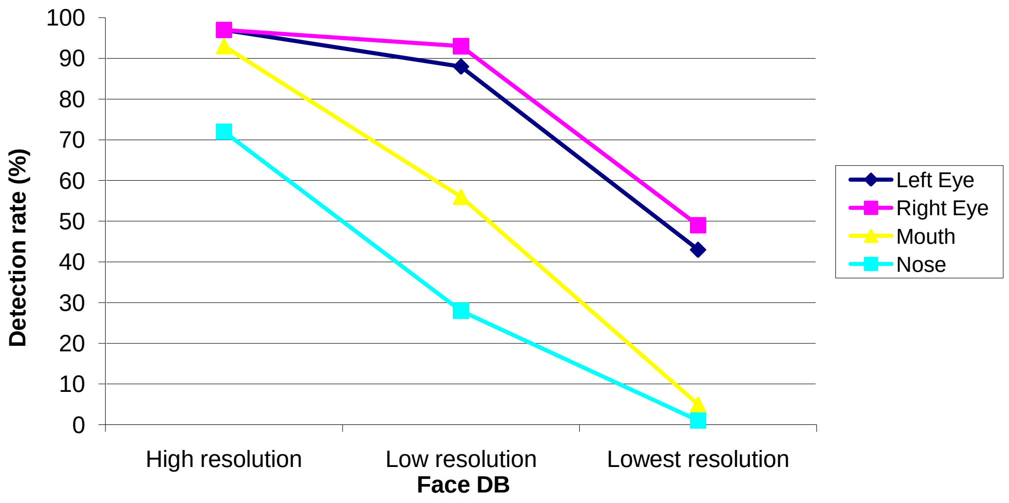
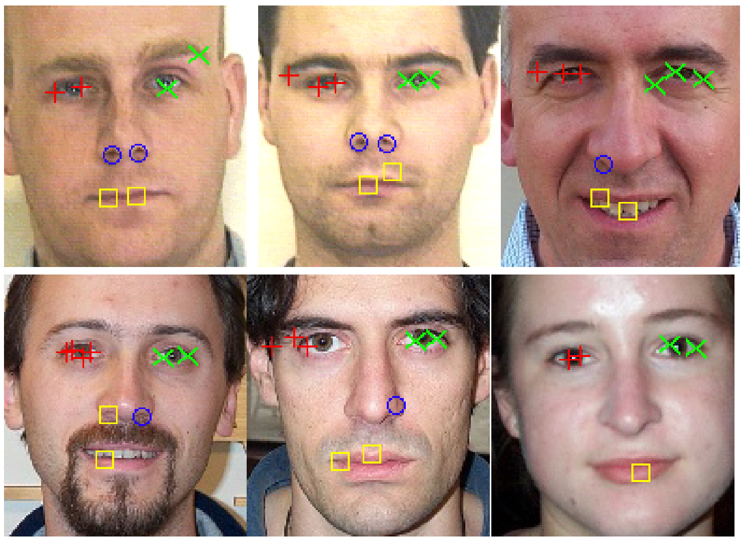


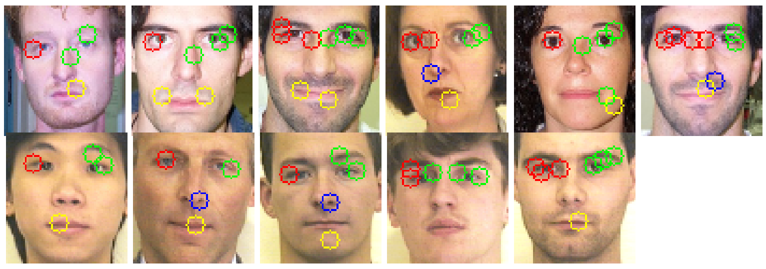
Figure F.15: Detection accuracy(%) of the subclass classifiers.

SURF descriptor, reducing the performance (see the rightmost column in Table F.1). For comparison, at the lowest resolution using 100 face images, the detection rate of the OpenCV face detector (81) is just 25%, even if the classifier of the face detector is trained on 20×20 face and non-face images. The OpenCV face detector is highly dependent on eye region features, calculated by Haar-wavelet function, rather than mouth and nose regions especially in the low resolution images. The mouth and nose are too small to extract at the lowest resolution, and therefore it may be ignored. In terms of eye detection performance, the method proposed here is better (eye detection rates of the proposed method are nearly 50% in Figure F.15). All processes (skin detection, first layer classifier, second layer classifiers and geometrical constraints) have been applied for these results.

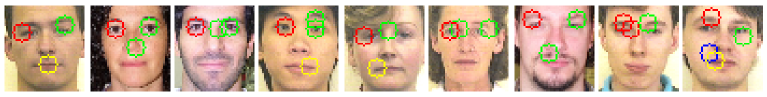
Further experiment results of different resolutions are shown in Figure F.16. Red , green , blue, and yellow colour correspond to left eye, right eye, nose, and mouth, respectively.



(a) High-res. results (130 × 140)



(b) Low-res. results(65 × 70)



(c) Lowest-res. results (43 × 46)

Figure F.16: Detection results.

F.9 Conclusion

This appendix has presented a two layer classifier to perform face component detection (using frontal views) and has shown how performance deteriorates when considering low resolution images. One main problem with the point descriptor such as a SURF and SIFT is that they are not well detected at low resolution images. This may also pose problems when matching points between stereo pairs or multiple views for 3D reconstruction in Furukawa and Ponce (27), Vu et al. (83) and Bradely et al. (8) which are the top rankers of the Middlebury challenge (66).

References

[1] Zeeshan Anwar and Frank Ferrie. Towards robust voxel-coloring: Handling camera calibration errors and partial emptiness of surface voxels. *International Conference on Pattern Recognition*, 1:98–102, 2006. 18

[2] Bruce G. Baumgart. Geometric modeling for computer vision. Technical report, Artificial Intelligence Laboratory, Stanford University, October 1974. 8, 12

[3] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *In Proceedings of the Ninth European Conference on Computer Vision*, 2006. xvi, 133, 136, 137, 138

[4] Daniel Berwick and Sangwook Lee. A chromaticity space for specularity, illumination color- and illumination pose-invariant 3d object recognition. *IEEE International Conference on Computer Vision*, pages 165–170, 1998. 51

[5] V. Blanz and T. Vetter. Face recognition based on fitting a 3d morphable model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1–12, 2003. 65

[6] J.Y. Bouguet. Camera calibration toolbox for matlab, 2008. 34

[7] Edmond Boyer and Jean-Sébastien Franco. A hybrid approach for computing visual hulls of complex objects. *In IEEE Conference on Computer Vision and Pattern Recognition*, 1:695–701, 2003. 12

[8] Derek Bradley, Tamy Boubekeur, and Wolfgang Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. ix, 18, 19, 149

[9] Gary Bradski and Adrian Kaehler. *Learning Opencv.* O'Reilly Media, Inc, 2008. 34

[10] Qin Cai, David Gallup, Cha Zhang, and Zhengyou Zhang. 3d deformable face tracking with a commodity depth camera. *In European Conference on Computer Vision*, 2010. 65

[11] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:790–799, 1995. 23

[12] German Cheung, Simon Baker, and Takeo Kanade. Visual hull alignment and refinement across time: A 3d reconstruction algorithm combining shape-from-silhouette with stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 695–701, 2003. 13

[13] R. Collins. Mean shift blob tracking through scale space. *IEEE conference on Computer Vision and Pattern Recognition*, 2:234–240, 2003. 23

[14] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002. ix, 23, 24

[15] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:564–575, 2003. x, 23, 24

[16] T. F. Cootes, G. J. Edwards, and C.J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 681–685, 2001. 65, 76

[17] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, January 1995. 65, 76

[18] Rozenn Dahyot. Mean-shift for statistical hough transform. Technical report, Technical report department of Statistics, Trinity College Dublin, 2009. 52

[19] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. *In Computer Vision and Pattern Recognition*, 2:126–133, 2000. 26

[20] Mark S. Drew, Graham D. Finlayson, and Steven D. Hordley. Recovery of chromaticity image free from shadows via illumination invariance. *IEEE International Conference on Computer Vision Workshop on Color and Photometric Methods in Computer Vision*, pages 32–39, 2003. 51

[21] Charles R. Dyer. Volumetric scene reconstruction from multiple views. *Foundation of Image Understanding, L.S. Davis, ed., Kluwer, Boston*, pages 469–489, 2001. 7

[22] G. J. Edwards, A. Lanitis, C. J. Taylor, and T. F. Cootes. Statistical models of face images - improving specificity. *In British Machine Vision Conference*, pages 765–774, 1996. 65, 76

[23] David Exner, Erich Bruns, Daniel Kurz, Anselm Grundhöfer, and Oliver Bimber. Fast and robust camshift tracking. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–16, 2010. 25

[24] Z. Fan, Y. Wu, and M. Yang. Multiple collaborative kernel tracking. *In IEEE Conference on Computer Vision and Pattern Recognition*, 2:502–509, 2005. 24

[25] J. S. Franco and E. Boyer. Efficient polyhedral modeling from silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 414–427, March 2008. 7, 11, 13

[26] Keinosuke Fukunaga and Larry D. Hostetler. The estimation of the gradient of a density function, applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–41, 1975. 23

[27] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010. ix, 18, 19, 20, 149

[28] B. Georgescu, I. Shimshoni, and P. Meer. Mean shift based clustering in high dimensions: a texture classification example. *IEEE International Conference on Computer Vision*, 2:456–463, 2003. 23

[29] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders. The amsterdam library of object images. *International Journal Computer Vision*, 61(1):103–112, 2005. xii, 67, 91, 92, 128

[30] Igor Guskov. Kernel-based template alignment. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 610–617, 2006. 25

[31] Gregory D. Hager, Maneesh Dewan, and Charles V. Stewart. Multiple kernel tracking with ssd. *In IEEE Conference on Computer Vision and Pattern Recognition*, pages 790–797, 2004. 24

[32] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, Manchester, UK, 1998. 132

[33] Takuma Iwasa, Takuya Shima, Masanori Sai, and Gang Xu. 3d eigenfaces for face modeling. *The 5th Asian Conference on Computer Vision*, pages 23–25, 2002. 76

[34] A. C. Kak and Malcolm Slaney. Principles of computerized tomographic imaging. *Society of Industrial and Applied Mathematics*, 2001. 60

[35] Donghoon Kim and Rozenn Dahyot. Face components detection using surf descriptor and svms. *In International Machine Vision and Image Processing Conference*, 2008. 132

[36] Donghoon Kim and Rozenn Dahyot. 3d head reconstruction using multicamera stream. *Irish Machine Vision and Image Processing Conference*, 2009. 26

[37] Donghoon Kim, Jonathan Ruttle, and Rozenn Dahyot. 3d shape estimation from silhouettes using mean-shift. *In IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010. 42

[38] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38:307–314, 1998. 17

[39] A. Laurentini. How far 3d shapes can be understood from 2d silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):188–195, 1995. 8

[40] Svetlana Lazebnik, Edmond Boyer, and Jean Ponce. On computing exact visual hulls of solids bounded by smooth surfaces. In *In IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 156–161, 2001. 12

[41] Sylvain Lefebvre and Hugues Hoppe. Parallel controllable texture synthesis. In *ACM SIGGRAPH*, 2005. 73

[42] P. Li and L. Xiao. Mean shift parallel tracking on gpu. In *In Iberian Conference on Pattern Recognition and Image Analysis*, pages 120–127, 2009. 25

[43] Tyng-Luh Liu and Hwann-Tzong Chen. Real-time tracking using trust-region methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:397–402, 2004. 25

[44] L. Liyuan, H. Weimin, Y.H.G. Irene, and T. Qi. Foreground object detection from videos containing complex background. In *In MULTIMEDIA : Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10, 2003. 27

[45] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21, 1987. 11

[46] D. Lowe. Distinctive image features from scale-invariant keypoints, cascade filtering approach. In *International Journal of Computer Vision*, pages 91–110, 2004. xvi, 133, 134, 135, 136, 138

[47] Donald Ludwig. The radon transform on euclidean space. *Communications on Pure and Applied Mathematics*, 19:49–81, 1966. 16

[48] W. N. Martin and J. K. Aggarwal. Volumetric description of objects from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):150–158, 1987. 8, 9

[49] A.M. Martinez and R. Benavente. The ar face database. Technical report, CVC Technical Report 24, 1998. 145

[50] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, pages 384–393, 2002. 133

[51] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135–164, 2004. 65, 76

[52] I. Matthews, J. Xiao, , and S. Baker. 2d versus 3d deformable face models: Representational power, construction, and real-time fitting. *International Journal of Computer Vision*, 75:93–113, 2007. 65

[53] Wojciech Matusik, Chris Buehler, and Leonard McMillan. Polyhedral visual hulls for real-time rendering. *In Eurographics Workshop on Rendering*, pages 115–126, 2001. ix, 12

[54] K. Mikolajczyk and C.Schmid. An affine invariant interest point detector. In *European Conference on Computer Vision*, pages 128–142, Copenhagen, Denmark, 2002. 133

[55] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 257–263, 2003. 133

[56] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962. 21

[57] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3d face model for pose and illumination invariant face recognition. *The 6th IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS) for Security, Safety and Monitoring in Smart Environments*, 2009. xiii, xv, 75, 77, 78, 80, 81, 115, 116, 117

[58] C. Pintavirooj and M. Sangworasil. 3d shape reconstruction based on radon transform with application in volume measurement. In *10th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2002. 16, 41, 55, 60

[59] M. Potmesil. Generating octree models of 3d objects from their silhouettes in a sequence of images. *Computer Vision, Graphics, and Image Processing*, 40:I-29, 1987. 8

[60] Wei Qu and Dan Schonfeld. Robust control-based object tracking. *IEEE Transactions on Image Processing*, 17:1721–1726, 2008. 24

[61] G. N. Ramachandran and A. V. Lakshminarayanan. Three-dimensional reconstruction from radiographs and electron micrographs: Application of convolutions instead of fourier transforms. *National Academy Science USA*, 68(9):22362240, 1971. 16

[62] Rapidform-XOR. http://www.rapidform.com, 2010. 87, 114

[63] Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *Annals of Mathematical Statistics*, 27(3):832–837, 1956. 21

[64] Jonathan Ruttle, Michael Manzke, and Rozenn Dahyot. Smooth kernel density estimate for multiple view reconstruction. In *The 7th European Conference for Visual Media Production*, 2010. 4, 41, 64, 98

[65] M. K. Saad and S. Mubarak. Tracking multiple occluding people by localizing on multiple scene planes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(3):505–519, March 2009. 26

[66] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. Multi-view stereo evaluation web page, 2006. 18, 149

[67] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *In IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 519–528, 2006. xiv, 18, 31, 100, 105

[68] Steven M. Seitz and Charles R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *In IEEE Conference on Computer Vision and Pattern Recognition*, pages 1067–1074, 1997. 17

[69] C. Shen, M. Brooks, and A. van den Hengel. Fast global kernel density mode seeking: Applications to localization and tracking. *IEEE Transactions on Image Processing*, 16, 2007. 52

[70] Chunhua Shen, Michael J. Brooks, and Anton van Hengel. Fast global kernel density mode seeking: Applications to localization and tracking. *IEEE Transactions on Image Processing*, 16(5):1457–1469, 2007. 23, 25

[71] L. Sigal and M. J. Black. Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion. Technical Report CS-06-08 Providence, Rhode Island 02912, Dept. of Computer Science, Brown University, 2006. ix, x, xiv, 2, 10, 31, 34, 36, 100, 101, 102

[72] Greg Slabaugh, Bruce Culbertson, Ron Schafer, and Tom Malzbender. A survey of methods for volumetric scene reconstruction from photographs. In *International Workshop on Volume Graphics*, 2001. 18

[73] Dan Snow, Paul Viola, and Ramin Zabih. Exact voxel occupancy with graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 345–352, 2000. 11

[74] Balaji Vasan Srinivasan, Qi Hu, and Ramani Duraiswami. Graphical processors for speeding up kernel machines. *Workshop on High Performance Analytics - Algorithms, Implementations, and Applications, Siam Conference on Data Mining*, 2010. 25

[75] P. Srivasan, P. Liang, and S. Hackwood. Computational geometric methods in volumetric intersections for 3d reconstruction. *Pattern Recognition*, 23(8):843–857, 1990. 8

[76] Steve Sullivan and Jean Ponce. Automatic model construction, pose estimation, and object recognition from photographs using triangular splines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1091–1096, 1998. 12

[77] Richard Szeliski. Rapid octree construction from image sequences. *CVGIP: Image Understanding*, 58(1):23–32, 1993. ix, 9, 10

[78] M. Turk and A.P. Pentland. Face recognition using eigenfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1991. 78

[79] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3d tracking using online and offlines information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1385–1391, 2004. 138

[80] R. Verma, C. Schmid, and K. Mikolajczyk. Face detection and tracking in a video by propagating detection probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1215–1228, 2003. 138

[81] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, HI, 2001. 138, 147

[82] Christian Vogler, Zhiguo Li, Atul Kanaujia, Siome Goldenstein, and Dimitris Metaxas. The best of bothworlds: Combining 3d deformable models with active shape models. In *IEEE International Conference on Computer Vision*, 2007. 65

[83] H. H. Vu, R. Keriven, P. Labatut, and J.P. Pons. Towards high-resolution large-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 149

[84] Markus Weber. Caltech face database; http://www.vision.caltech.edu/html-files/archive.html, 2005. 145

[85] Chenghua Xu, Yunhong Wang, Tieniu Tan, and Long Quan. A new attempt to face recognition using 3d eigenfaces. In *Asian Conference on Computer Vision*, volume 2, pages 884–889, 2004. 76

[86] M.H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002. 138

[87] Fangfang Zhou, Ying Zhao, and Kwan-Liu Ma. Parallel mean shift for interactive volume segmentation. *Machine learning in medical imaging, Lecture notes in Computer science*, pages 67–75, 2010. 25

[88] Karel Zimmermann, Tomas Svoboda, and Jiri Matas. Multiview 3d tracking with an incrementally constructed 3d model. In *The Third International Symposium on 3D Data Processing, Visualization, and Transmission*, 2006. ix, 2, 3, 26