REGISTERING RANGE IMAGES AND SEGMENTING 3D OBJECT MODELS

by

Olcay Sertel

Submitted to the Institute of Graduate Studies in

Science and Engineering in partial fulfillment of

the requirements for the degree of

Master of Science

in

Computer Engineering

Yeditepe University

2006

REGISTERING RANGE IMAGES AND SEGMENTING 3D OBJECT MODELS

APPROVED BY:

Assist. Prof. Dr. Cem Ünsalan (Thesis Supervisor)	[illegible]
Assist. Prof. Dr. Ender Özcan	[illegible]
Assist. Prof. Dr. Esin Onbaşıoğlu	[illegible]

DATE OF APPROVAL: 13.07.2006

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my family for always being supportive of my endeavors. My mother, father, brother and my fiance who have untiringly shown me guidance, patience and love. Without them, I never would have been able to get here.

I would like to express my deep gratitude both personally and professionally, to my advisor, Cem Ünsalan, without whose guide and encouragement the completion of this study would be impossible. I will remember every word of his valuable advices through all my future studies.

I would like to thank CSE faculty and research assistants for making my two years at Yeditepe University a great experience.

Finally, I would like to thank all my friends and loved ones outside of school for their motivation.

ABSTRACT

REGISTERING RANGE IMAGES AND SEGMENTING 3D OBJECT MODELS

We propose a crude range image registration algorithm based on edge detection in spherical coordinates. We used the edge features obtained by our edge detection method to register successive range images instead of using the whole point information. Different from previous edge detection methods, we first obtain a function representation in spherical coordinates. This representation allows detecting smooth edges on the object surface easily by a zero crossing edge detector. We use the well known ICP algorithm on these edges to register patches in a crude manner. Then, we apply the ICP to whole point set and obtain the final registration. The dual operation performed extremely fast compared to directly registering the points sets without any initial estimation. Our method also produced lower registration errors compared to the conventional ICP algorithm. One other important byproduct of our method is that, we can obtain the edges of the 3D object while registering it. These edge points may be of use in 3D object recognition and classification.

We also provide a free-form 3D object segmentation algorithm based on the same edge detection algorithm in spherical coordinates. Physically meaningful segments of a 3D object may be useful in various computer graphics applications. In order to segment free-form objects, we introduced a novel method in this study.

ÖZET

DERİNLİK İMGELERİNİN EŞLEŞTİRİLMESİ VE ÜÇ BOYUTLU MODELLERİN BÖLÜTLENMESİ

Bu çalışmada, küresel koordinatlara dayalı kenar çıkarma algoritması kullanılarak derinlik imgelerinin eşleştirilmesi gerçekleştirilmiştir. Derinlik imgelerini oluşturan tüm noktaları kullanmak yerine, önerdiğimiz kenar çikarma algoritmasından elde edilen kenar bilgileri kullanılmıştır. Daha önce önerilen yöntemlerden farklı olarak, kenar çıkarma aşamasında küresel koordinat gösterimi kullanılmıştır. Bu gösterim keskin olmayan kenarların da kolayca çıkarılmasını sağlamaktadır. Kenar bilgilerinin eşleştirilmesinde özyineli en yakın nokta (ICP-Iterative Closest Point) algoritması kullanılmıştır. Daha sonra, eşleştirmeyi iyileştirmek için aynı algoritma tüm noktalar üzerinde uygulanmıştır. Bu iki aşamalı eşleştirme işlemi, herhangi bir ilklendirmeye gerek kalmaksızın çok hızlı bir şekilde sonuç vermektedir. Ayrıca, yöntemimiz doğrudan uygulanan ICP algoritması ile kıyaslandığında, daha düşük hatalı sonuçlar üretebilmektedir. Bir diğer avantajımız da, eşleştirme işlemi sonunda elimizde ayrıca eşleştirilmiş kenar bilgisinin olmasıdır. Bu bilgi nesne tanıma ve ya sınıflandırma gibi işlemlerde kullanılabilir.

Üç boyutlu nesnelerin fiziksel anlamı olan bölütleri birçok bigisayar grafiği uygulamasında kullanılmaktadır. Bu çalışmada ayrıca aynı kenar çıkarma algoritması kullanılarak üç boyutlu nesne modellerinin bölütlenmesi gerçekleştirilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

$d(\overrightarrow{r_1}, \overrightarrow{r_2})$	Euclidian distance between two points
$D(\overrightarrow{p}, X)$	The distance between a point and model set
$f(\mathbf{R}, \overline{T})$	Registration error function
$F(\theta, \phi)$	Laplacian of Gaussian function
$lc(v)$	left child od node v
N_p	Number of points in the point set
N_x	Number of points in the model set
p_i	ith point in the point set
P	Point set
$rc(v)$	right child od node v
$R(t)$	parametric ray
$R(\theta, \phi)$	Radius function in spherical coordinates
$\mathbf{R}$	Rotation Matrix
$\overline{T}$	Translation vector
$T(u, v)$	parametric triangle where (u, v) are barycentric coordinates
X	Model set
x_i	ith point in the model set
θ	pan angle
ϕ	tilt angle
σ	scale parameter for the LOG function

CAD	Computer Aided Design
ICP	Iterative Closest Point
LEGION	Locally Excitatory Globally Inhibitory Oscillator Network
LOG	Laplacian of Gaussian
MIR	Mutual Inlier Ratio
SOFM	Self-organizing Feature Map
RHT	Randomized Hough Transform

1. INTRODUCTION

Advances in modern range scanning technologies and integration methods allow us to obtain detailed 3D model representations of real world objects without requiring humans to manually produce these models using laborious and error-prone CAD-based approaches. These 3D models are widely being used in many areas such as medical imaging (3D visualization of organs for diagnostic issues), virtual reality (reconstruction of cultural heritage sites, virtual museum applications), computer graphics (3D models to be used in animations), reverse engineering (analyzing deformations of manufactured products), and computer vision (object or face recognition).

Intensity images do not supply enough information about the surface geometry. Range images encode the position of a surface directly. Therefore, the shape can be acquired precisely. Actually, range images are a special class of digital images. Each pixel of a range image expresses the distance between a known reference frame and a visible point in the scene. Therefore, a range image reproduces the 3D structure of a scene. Range images are $m \times n$ grid of distances (range points) that describe a surface which are also referred to as depth images, depth maps, xyz maps, surface profiles and 2.5D images.

There are various 3D surface acquisition methods, basically classified into two main categories as passive and active. Some examples to passive methods are stereopsis, 3D information from silhouettes, and shape from shading. These methods are called passive, because they do not need an external light source to be projected on the object. However, in active methods, an external light source such as a line laser or a structured light pattern is projected to the object or scene to obtain the surface information.

In the structured light method, a predefined light pattern (generally binary or colored light stripes with different widths) is projected to the object. A one shot picture is taken by a camera. By analyzing the changes in the light pattern, the depth information can be obtained as illustrated in Fig. 1.1.

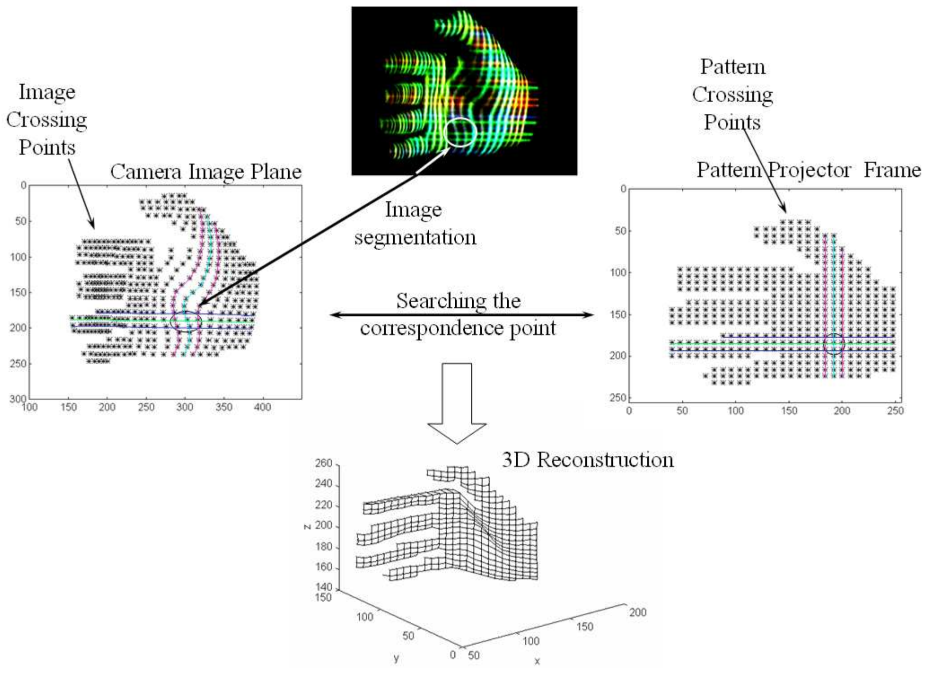


Figure 1.1. Range scanning using structured light

In laser range scanning, a single line laser is projected on the object. The process is repeated by changing the laser position and images are captured for every laser stripe position. We obtain the the depth information by analyzing the change in every laser stripe as shown in Fig. 1.2.

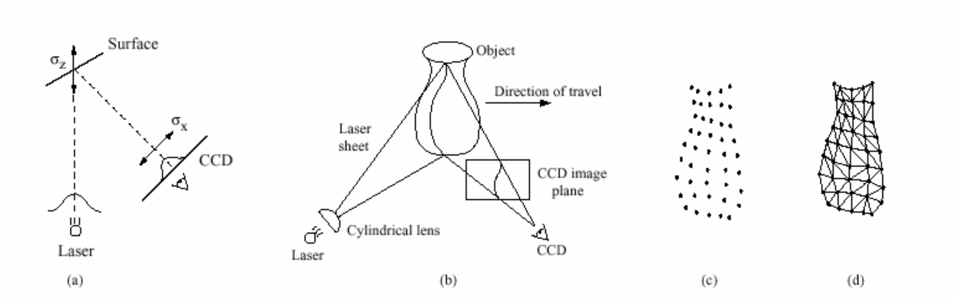


Figure 1.2. Range scanning using laser

As can be see in Figs. 1.1 and 1.2, we can only obtain the depth information of the object from one point of view at a time. In order to obtain the overall 3D model of the object, we need several range images from different viewpoints.

In Fig. 1.3 an example range image sequence of a toy bird object is given. The whole sequence for the bird object consists of 18 successive range images (with the object rotated around a common axis in 20 degree steps). Each range image is composed of 200 × 200 points aligned as a grid in horizontal and vertical directions. The depth information is encoded as the grayscale values in these images (white color representing closer depth values). The point set for the two images is provided in Fig. 1.4.

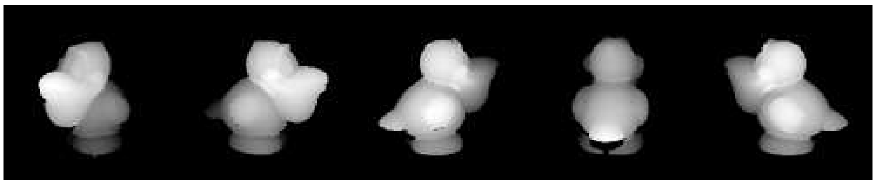


Figure 1.3. Sample range scans of the bird object from different viewpoints

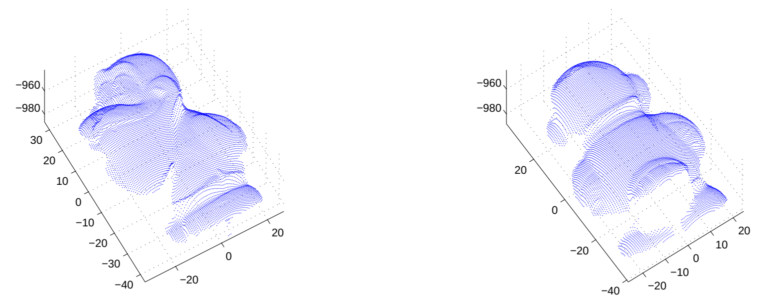


Figure 1.4. 3D plot of range images in the first and fourth range images of the bird object

These range images should be aligned in a common frame coordinate. Aligning multiple range images to a common coordinate system is known as registration. The goal of registration is to find a set of rigid transformations to align multiple views of the object. After registration, the 3D model of the object is obtained.

In this thesis, we aim to provide a more robust and accurate range image registration algorithm. We obtain the final 3D representation of the object from successive range images. We use edge features of the range images to provide a more efficient solution to rage image registration problem. In Fig. 1.5 first row, the final 3D representation of the bird object is given. Besides, our method also provides the final 3D edge representation of the object which can be used for further processing such as classification and recognition as shown in the second row of Fig. 1.5.

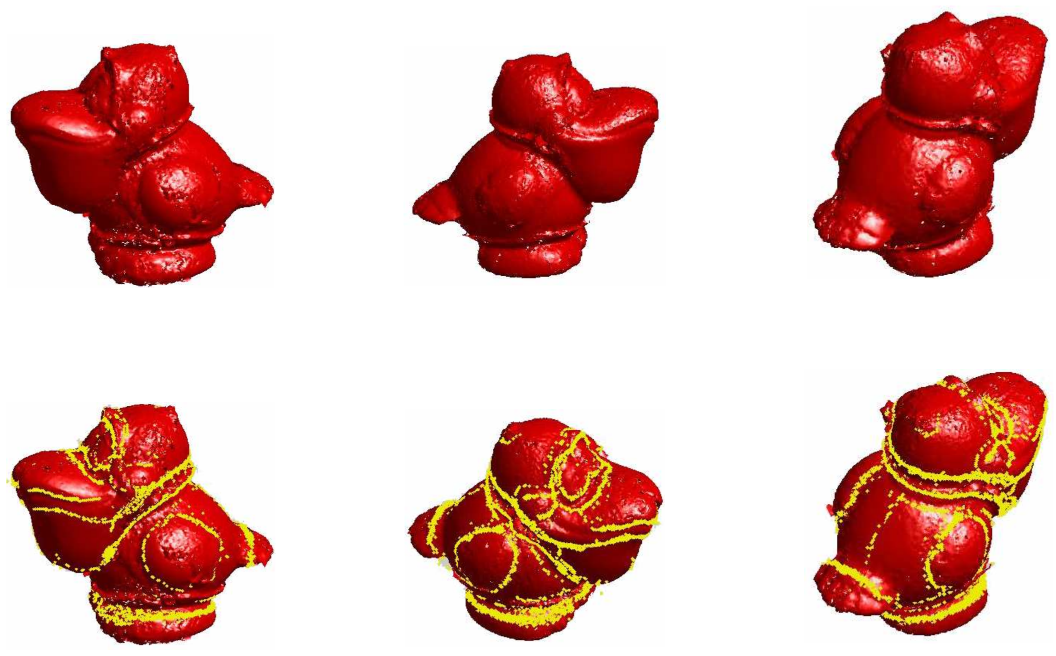


Figure 1.5. Final 3D representation of the bird object and its edges

2. LITERATURE REVIEW

This thesis consists of two main parts. The first one is range image registration. The second one is 3D object segmentation. Therefore, we divided the literature review chapter into two main sections accordingly.

2.1. Literature Review on Range Image Registration

There are various methods to register multiple range images. This section provides a thorough review on range image registration.

2.1.1. Fine Registration

The current state-of-art algorithm in fine registration is the *Iterative Closest Point* (ICP) algorithm proposed by Besl and McKay [1]. Most registration algorithms are based on ICP. In our study, we will also benefit from the ICP algorithm. Since we will consider the ICP algorithm in detail in Chapter 3, we postpone its explanation. Rusinkiewicz and Levoy [2] categorized and summarized variants of the ICP algorithm. Rodrigues *et al.* [3] also presented a survey on major registration algorithms. Let's focus on some registration algorithms next.

Arun *et al.* [4] presented a registration technique which assumes a prior knowledge of a set of matched point pairs, followed by a least squares minimization algorithm based on singular value decomposition. Horn [5] proposed a similar approach based on unit quaternions which is generalized for any representation of geometric data in the original ICP.

In order to decrease the registration error, Zhang [6] proposed a statistical model for the classification of outliers that lie on the non-overlapping region. He hypothesized that the distribution of the distances of the closest point pairs has a Gaussian distribution. Based on this assumption, Zhang used a heuristic to determine the threshold distance to detect the outlier point pairs.

Levine and Blais [7] proposed a stochastic optimization technique using calibration parameters. Using these calibration parameters, correspondences can be easily calculated by projecting pixels from one view into other plane of view. They used simulated annealing instead of least squares minimization to minimize the registration error and to avoid local minimums.

Turk and Levoy [8] used the triangle representation of range images. They introduced a confidence metric based on the surface normals which leads to a weighted-least squares optimization. They have rejected mesh pairs that lie on boundaries and integrated meshes after registration by re-triangulating them. This decreases the registration error.

Rodrigues and Liu [9] proposed a new representation of rigid body transformations based on geometric properties of reflected correspondence vectors with a comparative study. They put forward a novel representation of rigid body transformations based on the constraints about the distance, angle, and projection measurements. These constraints are derived from the geometric properties of reflected correspondence vectors. They aimed a more accurate and robust registration of two overlapping range images.

Okatani and Deguchi [10] dealt with measurement error properties. They proposed a method to correct the position of each point using the variance and extent of the error distribution. They selected the best transformation of all possible transformations by evaluating the correction.

Liu and Wei [11] proposed a method based on two structural constraints: proximity and closeness to be applied at correspondence searching stage to increase accuracy and efficiency. If the ICP algorithm converges to a local minimum, the structural constraints can help it traverse the local minimum through rejecting false matches and re-estimating the registration parameters based on the refined point matches.

Xiao *et al.* [12] presented a new registration method that uses both regional surface properties and the shape rigidity constraints of the objects. The algorithm does not require any feature extraction stage. However, it needs the statistical properties of the vertices on the object to obtain the initial candidate correspondences. This way, they were able to reject outliers.

Recently, Pottmann *et al.* [13] proposed an alternative to the ICP algorithm. Their method relies on instantaneous kinematics and the geometry of the squared distance function of a surface. Unfortunately, if the model shape is given as a point cloud, obtaining the local quadratic approximants is not straightforward. Therefore, the alternative proposed by Pottmann *et al.* is not so useful.

Campbell and Flynn [14] surveyed fine registration methods based on the ICP algorithm. They presented a comparative study on several popular techniques with a concentration on outlier classification. More detail on registration problem can be found there.

2.1.2. Crude Registration

Different from the fine registration algorithms described above, a large amount of research has focused on the crude registration of range images. The objective of crude registration is to find an approximate transformation to align both data sets. The crude registration is computationally more efficient than the fine registration. Also, fine registration methods require a good initial estimation (for the rotation and translation steps). This is not needed for the crude registration methods. In order to benefit both methods, most researchers used crude registration as an initial step. Then they used fine registration over the crude registration's results.

Lucchese *et al.* [15] proposed a frequency domain technique as a pre-alignment tool for fine registration. The rotation parameters are estimated through convenient representations and projections of the Fourier transform magnitudes. The translational displacement is recovered by means of a standard phase correlation technique after compensating one of the two views for rotation.

Krsek *et al.* [16] exploited the curvilinear features by calculating differential invariant parameters of the surface and used these features in their crude registration algorithms. Similarly, Sharp *et al.* [17] used Euclidian invariants for crude registration.

Sablatnig and Kampel [18] proposed a prealignment algorithm for registering the front and backview of rotationally symmetric objects. Their method uses the axis of rotation of fragments to bring two range images into alignment. They used their method for the classification system for archaeological fragments based on their profile.

Wyngaerd and Van Gool [19] proposed a method which uses bitangent curve pairs as landmarks on the surface and invariant signatures are used for matching curve pairs on different patches. This can also be taken as a crude registration step for fine alignment.

Sappa *et al.* [20] proposed a method using the edge features of range images to crudely register them. They obtain edge features using the scan line approximation algorithm. The scan line approximation algorithm simply splits range image into scan lines through vertical, horizontal and diagonal directions where every scan line represents a 2D planar curve [21]. A quadratic approximation function is first determined for a whole scan line based on the midpoint and the two endpoints. Then, whenever the largest error between the approximation function and the scan line is greater than a preselected threshold, the scan line is split into two parts at that location. The splitting algorithm proceeds recursively until the approximation error does not exceed the threshold value. The algorithm produced acceptable results, however the threshold determination step which is a crucial step in edge detection is highly depends on the data. Therefore, crude registration based on the scan line algorithm also depends on the data.

2.1.3. Simultaneous Registration

Some other researches have generalized classic pairwise point set registration task to simultaneous registration of multiple views to achieve a global optimal registration for all views. Some related work on the simultaneous registration is as follows.

Bergevin *et al.* [22] presented an algorithm that reduces the level of the registration errors between all pairs in a set of range views. Their method leads to a well-balanced network of views. This leads to an equally balanced registration errors.

Shum *et al.* [23] proposed an integral approach to reconstruct statistically optimal object models. They achieved this by simultaneously aggregating all the data from multiple views into a weighted least squares formulation. To reliably encode local curvature (taking local connectivity into account), they globally resample data using salient feature points. However, there is a tradeoff between accuracy and robustness in global resampling.

Pulli [24] introduced a multiview registration technique that uses pairwise alignments. He uses the constraints that the multiview step enforces while evenly diffusing the pairwise registration errors. The global alignment step attempts to balance registration errors across all views. This technique can also be used to handle large data sets (such as large number of scans or very dense scans).

Williams and Bennamoun [25] proposed a technique that requires the computation of a constant matrix to encode the point correspondence information. This matrix is used by an iterative algorithm to compute the optimal rotation in registration. They compute the translation through the solution of a linear equation system.

Castellani *et al.* [26] proposed a global registration method for the underwater scene construction and used data coming from an acoustic range sensor. They introduced a new outlier rejection ruled which is an improvement of Zhang's method.

Masuda [27] introduced the method of signed distance fields for simultaneous shape integration and registration. The method enables performing registration, integration, and outlier rejection together by matching signed distance fields. Unfortunately, the method requires large computational resources.

2.1.4. Registration using Evolutionary Algorithms

Besides the iterative error minimization approaches cited above, some researchers used evolutionary algorithms to solve the registration problem. Since evolutionary algorithms require efficient representations and fast evaluation functions, researchers evolved the three parameter rotation and translation vectors. Some related registration methods using evolutionary algorithms is as follows.

Robertson and Fisher [28] introduced a parallel genetic algorithm (GA) for registration. Their method is based on accelerating the process time by calculating the error of every possible solution at a separate computer node. Similarly, Chow *et al.* [29] proposed a novel genetic algorithm with a new fitness function and genetic operators (such as adaptive mutation). Silva *et al.* [30] proposed a hybrid genetic algorithm for the registration of range images and introduced a new evaluation metric based on surface interpenetration measure. They also presented a simultaneous registration method with genetic algorithms in [31].

2.2. Literature Review on 3D Object Segmentation

In computer vision, segmentation is an inevitable process for further high level analysis such as object recognition. The goal of 3D object segmentation is to partition the object based on the simplest possible 3-D surface primitives i.e. smooth surface regions and surface discontinuities. Most surface segmentation techniques can be classified as either edge-based or region-based depending on whether they emphasize the detection of surface discontinuities or the detection of smooth surface regions respectively. We review the related literature on both methods below.

2.2.1. Region Based Segmentation

The central idea behind region-based range image segmentation techniques is to estimate the surface curvature at each range pixel and cluster range pixels with homogeneous surface curvature properties to form smooth surface regions. There are two broad categories of region-based segmentation techniques: region-growing techniques and feature vector clustering techniques. In region growing techniques, an analytical surface is fitted in a local neighborhood surrounding a range pixel followed by spatial grouping of homogeneous pixels [32]. Although range image segmentation by region growing is popular, it needs a good criterion for merging and splitting adjacent regions. It is possible to oversegment or undersegment surfaces using this method.

Besl and Jain [32] designed an algorithm that starts from coarse segmentation initially created by using surface curvature signs. The algorithm refines segmentation by an iterative region growing that is based on the surface fitting errors.

Chang and Park [33] proposed a segmentation algorithm based on fusion of range and intensity images using robust trimmed methods. Objects are represented by a number of local planar surfaces in range images, and the parametric space for surface representation is constructed with the surface parameters estimated pixel-by-pixel by the least trimmed squares method. A final edge map is obtained that is constructed using the likelihood functions based on the edge information obtained from range and intensity images.

Köster and Spann [34] proposed an unsupervised region growing method based on a two-level hierarchical image structure. At first, the lower primitive components are extracted applying an estimation technique of least-median-of-squares. Then the extracted primitive components are iteratively merged into high-level primitive regions from the mutual inlier ratio (MIR). The ratio is obtained using robust regression techniques.

Wang *et al.* [35] proposed an algorithm for estimation-based range image segmentation. Aiming at surface-primitive extraction from range data, they focus on the reliability of the primitive representation in the process of region estimation. They introduced an optimal description of surface primitives, by which the uncertainty of a region estimate is explicitly represented with a covariance matrix. Then, the reliability of an estimate is interpreted in terms of "measure of uncertainty". The segmentation approach follows the region-growing scheme, in which the regions are estimated in an iterative way. The proposed algorithm focused on the "reliability" of the extracted surface primitive representation.

Segmentation techniques based on feature vector clustering assign a feature vector (based on surface curvature properties) to each range image pixel. The range image segmentation problem, in this case, can be treated as feature vector quantization. Feature vector quantization is a process of partitioning an n-dimensional feature vector space into M regions. Clustering techniques treat feature vectors simply as patterns in a high dimensional space. Thus, the resulted clusters are not guaranteed to be spatially connected unless feature vectors include positional information. Image segmentation by region growing guarantees that each segment is connected in the image where image segmentation by clustering does not. The other problem with clustering techniques is that the number of regions required by the clustering algorithm is not known in advance.

Fan *et al.* [36] use local surface curvature properties to identify significant boundaries present in the image. They use scale space tracking to detect features without loss of localization. The scale space tracking requires convolving the whole image with Gaussian masks having different values of variance, which is computationally intensive.

One of the limitations of classification-based approaches is that the number of regions must be given a priori which, generally is not available. Koh *et al.* [37] tried to address this issue by using a hierarchical self-organizing network for range image segmentation. At each level, a self-organizing feature map (SOFM) is used to segment range images into a given number of regions. An abstract tree is constructed to represent the output of the hierarchical SOFM network. The final segmentation is obtained by searching through the abstract tree, which is sequential and similar to a split-and-merge method. Thus the solution suffers from the disadvantages of region-based algorithms. In addition, the problem of prior specification of number of regions is not entirely solved because the number of regions for each level still needs to be specified.

Liu and Wang [38] presented a locally excitatory globally inhibitory oscillator network (LEGION). The feature detection associated with each oscillator estimates the surface normal and the curvature at its corresponding pixel location, while the segmentation process is the emergent behavior of the oscillator network. The main strong point of this method is that the segmentation is achieved mainly by local computation. As a consequence, a priori knowledge about the number of regions and accurate surface models is not required. The main weak points are high complexity to solve the differential equations required by the LEGION system and a high number of parameters that should be initialized.

Srikantiah *et al.* [39] introduced a method for obtaining reliable surface descriptions at multiple scales from the range data. They used mean and gaussian curvatures to segment the surface into regions of four saliency classes, each based on curvature consistency. One of the most attractive features of the approach is that it extracts surface segments sequentially in the order of their saliency by finding the largest segments first in a given consistency class, and by working through the consistency categories in the order of their inherent saliency.

2.2.2. Edge Based Segmentation

The basic idea behind edge-based range image segmentation techniques is to detect and classify range image pixels that signify surface discontinuities. One of the primary drawbacks of edge-based segmentation techniques is the inevitable fragmentation of the edges. If the edges are fragmented or discontinuous, they must be linked using a heuristic technique.

Wani and Batchelor [40] sliced the 3D image to create equidepth contours (EDC's) and obtain three types of critical points such as fold, semistep, and boundary edges. A subset of edge pixels is extracted using these critical edges. Edges are grown through these pixels through the application of morphological masks. The extraction of edge pixels under the guidance of sparse edge pixels and the adjustable edge mask constraints makes the method suitable for noisy images. The proposed method can also be split into parallel subtasks to achieve better performance by parallel computing.

Another approach to edge detection is residual analysis. Al-Hujazi and Sood [41] considered the absolute difference (residue) between the input image and its smoothed version, which possesses maxima at the locations of jump and crease edges. Edge detection is done by locating such maxima.

Jiang *et al.* [42] presented an algorithm for fast segmentation of range images into both planar and curved surface patches. Their approach makes use of high level features (curve segments) as segmentation primitives instead of individual pixels. This scan line grouping technique significantly reduces the amount of data the segmentation process is faced with.

Bellon *et al.* [43] presented a methodology to perform edge detection in range images in order to provide a reliable and meaningful edge map, to guide and improve range image segmentation by clustering techniques. They considered the problem of generating correct topological information of objects from edge detection to enhance range image segmentation by clustering algorithms.

Jiang [44] proposed an adaptive grouping algorithm to solve the contour closure problem that is the key to edge-based complete image segmentation. He compared several region-based segmentation methods within a well organized comparison framework. However, his edge grouping algorithm has a fundamental weakness in dealing with smooth contours.

Min and Bowyer [45] proposed a new approach to range image segmentation of planar and curved surface scenes. They chose the range segmentation algorithm developed by Jiang and Bunke as the baseline algorithm. They analyzed the types of errors made by the algorithm, proposed design modifications to decrease the error rate, and experimentally verified the results. By applying their approach, they designed an improved algorithm that is less sensitive to the edge extraction results.

Ding *et al.* [46] presented a novel range image segmentation algorithm based on randomized Hough transform (RHT). The algorithm finds planar regions by utilizing RHT and has the advantage of insensitivity to noise. Sappa [47] presented an efficient technique for extracting closed contours from range images' edge points in his study. The edge points are assumed to be given as input to his algorithm.

There are also the so called hybrid techniques that use both region and edge information to guide the segmentation process. Among the hybrid techniques, Yokoya and Levine [48] combined a region segmentation based on HK-sign maps with jump and roof edge maps to obtain a final segmentation. Ghosal and Mehrotra [49] proposed a method in which, the initial segmentation obtained by using region-based technique is refined based on the detected edge maps to produce the final segmentation.

Hoover *et al.* [50] compared four of the state-of the- art range image segmentation algorithms. They based the comparison systematically relative to a ground truth (labeled by an expert). One of the major conclusion of this analysis is that range image segmentation is still an unsolved problem even for simple scenes containing only polyhedral objects.

3. THE ICP ALGORITHM

Iterative Closest Point (ICP) is an algorithm introduced to register the two point clouds. This registration is extensively used to reconstruct the final 3D models of real objects from their range images. The algorithm is very simple and commonly used in many application areas such as medical 3D visualization, reverse engineering, and object recognition. It iteratively estimates the Euclidean transformation (translation and rotation) between the two point clouds. The algorithm requires three inputs as: the two point clouds (raw scans) to be registered, the initial estimation of the transformation, and a criteria for stopping the iteration. The output of the algorithm is the refined transformation.

Assume that we are trying to register the range image P (being the data set) to the point set X (being the model set) as in Fig. 3.1.

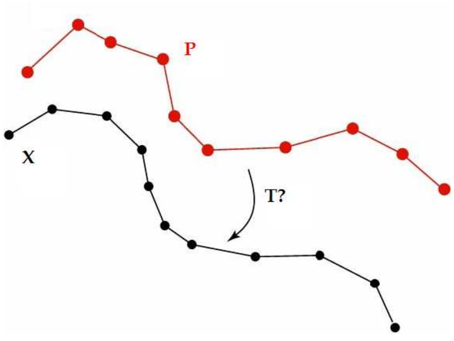


Figure 3.1. Registration of two point sets

The goal of registration is to find the 3D rotation matrix $\mathbf{R}$ and the translation vector $\overline{T}$ that minimizes the error:

$$f(\mathbf{R}, \overline{T}) = \frac{1}{N_P} \sum_{i=1}^{N_P} \|\overline{x}_i - \mathbf{R} \cdot \overline{p}_i - \overline{T}\| \tag{3.1}$$

where $\overline{p}_i$ is the *ith* point in the range image to be registered, $\overline{x}_i$ is the *ith* point in the reference image to which P is registering. For a given point $\overline{p}_i$, $\overline{x}_i$ is the closest point in data set X to the point $\overline{p}_i$. So $\overline{p}_i$ and $\overline{x}_i$ are called the corresponding point pairs. N_p is the number of points in the data set P.

The closest point is calculated using the Euclidian distance between two points $\overrightarrow{r_1} = (x_1, y_1, z_1)$ and $\overrightarrow{r_2} = (x_2, y_2, z_2)$ as:

$$d(\overrightarrow{r_1}, \overrightarrow{r_2}) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{3.2}$$

Let X be the model set with N_x points denoted by $\overrightarrow{x_i}$: $X = \{\overrightarrow{x_i}\}$ for $i = \{1, ..., N_x\}$. The distance between a point $\overrightarrow{p}$ and the model set X is:

$$D(\overrightarrow{p}, X) = \min_{i \in 1,\ldots,N_x} d(\overrightarrow{p}, \overrightarrow{x_i}) \tag{3.3}$$

For every point p_i in the data set P, the corresponding point from the model set X is computed using Eq. 3.3.

The ICP algorithm iteratively minimizes the error function. It supports many geometric primitives such as points, line segments, parametric curves, implicit curves, triangles, parametric surfaces, and implicit surfaces as mentioned in [1]. The algorithm performs iteratively as follows:

Algorithm 1 The ICP algorithm

```
Begin
$\mathbf{R}$=initial rotation matrix
$\overline{T}$=initial translation vector
$\epsilon$=registration error
repeat
    for $i = 1$ to $N_P$ do
        $x_i$=$D(\overrightarrow{p_i}, X)$
    end for
    compute $\mathbf{R}$ and $\overline{T}$ so that $f(\mathbf{R}, \overline{T})$ is minimized
    transform the data set $P$
    calculate the registration error $\epsilon$
until $\epsilon < threshold$
return $\mathbf{R}, \overline{T}$
End.
```

Figure 3.2. ICP algorithm

This iterative process is guaranteed to converge to a local minimum for any starting value of P when it is a subset of X. However, there are some limitations of the ICP algorithm when used in range image registration. First, range images are not subsets of each other. Instead, they partially overlap with each other depending on the viewpoint. Therefore, the algorithm requires the detection of outliers that comes through the non-overlapping regions. Second,the algorithm requires a good initial estimation close to the global minimum in order to avoid any local minimum.

The basic algorithm has been previously extended in a number of ways: correspondence between a point and a tangent plane to overcome the lack of an exact correspondence between the two sets [51]; robustifying the algorithm to the influence of outliers and features lacking correspondences [52, 6]; using a weighted least-square error metric [53]; matching between features using a metric trading off distance and feature similarity (based local shape invariants) [54]. All of these approaches assume a rigid Euclidean transformation between the corresponding features.

The ICP algorithm has an average complexity of $Q(N^2)$, where N is the number of points in the range image. One needs to compute the corresponding point pairs in every iteration. This increases the complexity and time consumption of the algorithm. To overcome the computational burden in the corresponding point search, we used a Kd-tree representation.

3.1. Kd-Tree Representation to Speed up the ICP Algorithm

A k-d tree is a data structure often used in fields such as computer vision, pattern recognition, spatial databases, and astronomy. A k-d tree is based on the well-known binary search tree [55]. Given a point set that lies in a k-dimensional space, the general purpose of a k-d tree is to recursively partition the space into cells such that each cell contains at most a certain number of input data points. Each interior node of the tree represents a hyperplane chosen to be orthogonal to one of the k axes. Each interior node has a left and a right subtree. These subtrees contain points located on either side of the hyperplane. Each leaf of the tree stores a non null data set, usually a single point.

Bentley and Friedman [56] improved the original k-d tree to create more efficient data structures. But for simplicity, we used the original k-d tree implementation to speed up the ICP algorithm.

In the 2-dimensional case each point has two values: its (x, y) coordinate. Therefore we first split on the x coordinate, next on the y coordinate to built up the 2d-tree representation. More precisely, the process is as follows. At the root, we split the set P with a vertical line l into two subsets of roughly equal size as in Fig 3.3. The splitting line is stored at the root. P_{left}, (the subset of points to the left or on the splitting line) is stored in the left subtree. P_{right}, (the subset to the right of the splitting line), is stored in the right subtree.

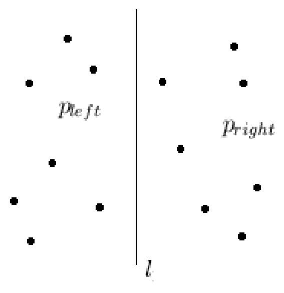
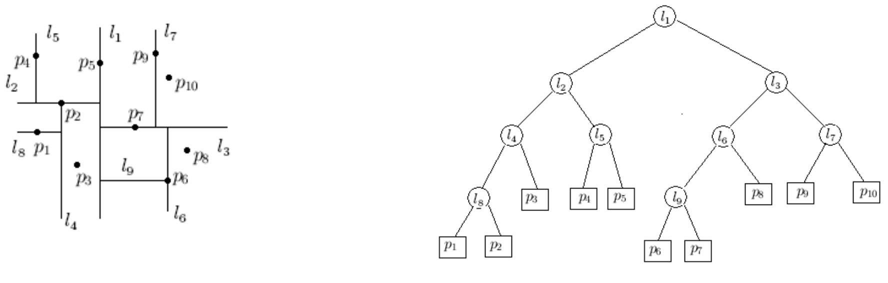


Figure 3.3. Kd-tree partitioning

At the left child of the root, we split P_{left} into two subsets with a horizontal line. The points below or on it are stored in the left subtree of the left child, and the points above it are stored in the right subtree. The left child itself stores the splitting line. Similarly, the set P_{right} is split with a horizontal line into two subsets, which are stored in the left and right subtree of the right child. At the grandchildren of the root, we split again with a vertical line. In general, we split with a vertical line at nodes whose depth is even, and we split with a horizontal line at nodes whose depth is odd. Fig. 3.4 illustrates how the splitting is performed and what the corresponding binary tree looks like.



(a) 2-d tree space partition (b) 2-d tree

Figure 3.4. Kd-tree representation

A tree like this is called a *kd-tree*. Originally, the name stood for k-dimensional tree. The tree we describe above would be a 2d-tree. The two algorithms to construct a kd-tree and search in the kd-tree with a recursive procedure is as follows:

Algorithm 2 $BUILDKDTREE(P, depth)$

```
$P$= A set of points
$depth$=the current depth
Begin
**if** $P$ contains only one point **then**
    **return** a leaf storing this point
**else**
    **if** $depth$ is even **then**
        Split $P$ into two subsets with a vertical line $l$ through the median $x$ coordinate of the points in $P$.
        Let $P_1$ be the set of points to the left of $l$ or on $l$, and let $P_2$ be the set of points to the right of $l$.
    **else**
        Split $P$ into two subsets with a horizontal line $l$ through the median $y$ coordinate of the points in $P$.
        Let $P_1$ be the set of points below $l$ or on $l$, and let $P_2$ be the set of points above $l$.
    **end if**
**end if**
$V_{left} \leftarrow BUILDKDTREE(P_1, depth + 1)$
$V_{right} \leftarrow BUILDKDTREE(P_2, depth + 1)$
Create a node $v$ storing $l$, make $v_{left}$ the left child of $v$, and make $v_{right}$ the right child of $v$.
**return** $v$
End.
```

Figure 3.5. Kdtree construction algorithm

Algorithm 3 $SEARCHKDTREE(v, r)$

```
$v$=The root of (a subtree of) a kd-tree, and a range $R$
$r$=All points at leaves below $v$ that lie in the range
Begin
if $v$ is a leaf then
   if $v$ is in region $r$ then
      return $v$
   else
      return NULL
   end if
else
   if $region(lc(v))$ is fully contained in $r$ then
      return $(lc(v))$
   else
      if $region(lc(v))$ intersects $r$ then
         $SEARCHKDTREE(lc(v), r)$
      end if
      if $region(rc(v))$ is fully contained in $r$ then
         return $(rc(v))$
      else
         if $region(rc(v))$ intersects $r$ then
            $SEARCHKDTREE(rc(v), r)$
         end if
      end if
   end if
end if
End.
```

Figure 3.6. Kdtree search algorithm

We partition the model point set into a 3-d tree and search the corresponding point pair (for each point) in the data set. We use the kd-tree representation in all our registration test given in the following chapter.

4. RANGE IMAGE REGISTRATION USING EDGE INFORMATION

In this chapter, we propose our range image registration method based on edge information. Our difference is the way we obtain the edge points from the range image set. In order to explain our range image registration method, we start with explaining our edge detection procedure. Then, we focus on applying ICP on the edge points obtained from different patches (to be registered).

4.1. Edge Detection on Range Images

Most of the commercial range scanners provide the point set of the 3D object in cartesian coordinates. Cartesian coordinates is not a good choice for detecting smooth edges in range images. We will provide an example to show this problem. Therefore, our edge detection method starts with a change the coordinate system.

4.1.1. Why Do We Need a Change in the Coordinate System?

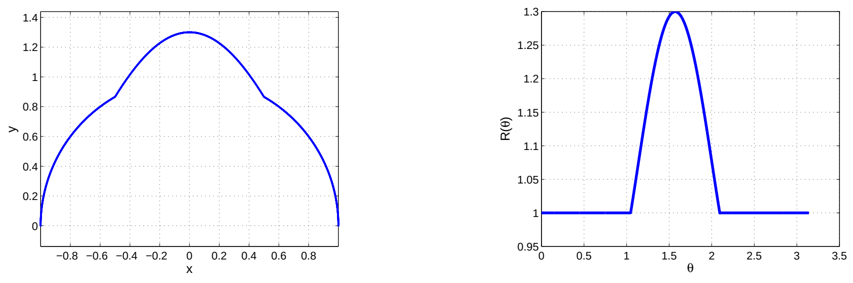
Researchers have focused on cartesian coordinate representations for detecting edges on 3D surfaces. Unfortunately, applying edge detection in cartesian coordinates do not provide acceptable results as shown in Figs. 4.3 and 4.4. The main reason for this poor performance is that, most edge detectors are designed to detect step edges in gray-scale images (it is assumed that, these step edges correspond to boundaries of objects in the image) [57]. In range images (3D surfaces), we do not have clear step edges corresponding to the actual edges of the object. For most objects, we have smooth transitions not resembling a step edge. Therefore, applying edge detection on these surfaces do not provide good results.

To overcome this problem, we hypothesize that representing the same object surface in spherical coordinates increases the detection rate of the object edges. Therefore, applying edge detection on this new representation provides improved results. As we detect edges on the spherical representation, we can obtain the cartesian coordinates of the edges and project them back to the actual 3D surface to obtain the edge points on the surface.

Let's start with a simple example to test our hypothesis. We assume a slice of a generic 3D object (at $z = 0$) for demonstration purposes. We can represent the point set at this slice by a parametric space curve $c(t) = (x(t), y(t), 0)$ as:

$$c(t) = \begin{cases} (\cos(t), \sin(t), 0) & t \in [0, \pi/3) \bigcup [2\pi/3, \pi) \\ (\cos(t), \sin(t) + 0.3sin(3t - \pi), 0) & t \in [\pi/3, 2\pi/3) \end{cases} \tag{4.1}$$

This space curve is plotted in Fig. 4.1 (a). As can be seen, the curve is composed of two parts. However, applying edge detection directly on this representation will not give good results, since we do not have step edge like transition between those curve parts.



(a) $c(t)$ in cartesian coordinates (b) $r(\theta)$ in spherical coordinates

Figure 4.1. A simple example emphasizing the effect of changing the coordinate system on detecting edges

We can obtain the spherical coordinate representation of $c(t)$ by defining $\theta = t$ and $r(\theta) = \sqrt{x^2(\theta) + y^2(\theta)}$. Now, our curve becomes:

$$r(\theta) = \begin{cases} 1 & \theta \in [0, \pi/3) \bigcup [2\pi/3, \pi) \\ 1 + 0.6sin(\theta)sin(3\theta - \pi) + 0.09sin^2(3\theta - \pi) & \theta \in [\pi/3, 2\pi/3) \end{cases} \tag{4.2}$$

As we plot $r(\theta)$ in Fig. 4.1 (b), we observe that the change in the curve characteristics is more emphasized similar to a step edge. This edge can easily be detected by an edge detector. Now, we can explore our hypothesis further for range images.

4.1.2. A Function Representation in Spherical Coordinates for Range Images

In practical applications, we use either a laser range sensor or a structured light scanner to obtain the range image of an object. Both systems provide a depth map for each coordinate position as $z = f(x, y)$. Our aim is to represent the same point set in spherical coordinates. Since we have a function representation in cartesian coordinates, by selecting a suitable center point, (x_c, y_c, z_c), we can obtain the corresponding function representation $R(\theta, \phi)$, in terms of pan (θ) and tilt (ϕ) angles as:

$$R(\theta, \phi) = \sqrt{(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2} \tag{4.3}$$

where

$$(\theta, \phi) = \left(\arctan\left(\frac{y - y_c}{x - x_c}\right), \arctan\left(\frac{\sqrt{(x - x_c)^2 + (y - y_c)^2}}{z - z_c}\right) \right) \tag{4.4}$$

This conversion may not be applicable for all range images in general. However, for modeling applications, in which there is one object in the scene to be scanned, this conversion is valid. It can easily be seen from 4.3 and 4.4 that selecting different center points will yield different representations in spherical coordinates which can affect our edge detection process.

In order to determine an appropriate center point for a particular patch, we use an adaptive center point selection procedure as:

$$x_c = \frac{x_{max} - x_{min}}{2} + x_{min} \tag{4.5}$$

$$y_c = \frac{y_{max} - y_{min}}{2} + y_{min} \tag{4.6}$$

$$z_c = \frac{|min(x_{min}, y_{min})|}{\tan(\gamma)} \tag{4.7}$$

where γ is the scale parameter used to adjust the viewing angle (in radians) for the patch. When $\gamma = \pi$, we assume that we have scanned a semi-sphere like patch. After experimenting on our object set, we observe that $\gamma = 1.31$ yields acceptable results. This value directly depends on the resolution of the point set at hand.

4.1.3. Edge Detection on the $R(\theta, \phi)$ Function

As we apply the cartesian to spherical coordinate transformation and obtain the $R(\theta, \phi)$ function, we have similar step like changes corresponding to physically meaningful segments on the actual 3D object. In order to detect these step like changes, we tested different edge detectors on $R(\theta, \phi)$ functions. Based on the quality of the final segmentations obtained on the 3D object, we picked Marr and Hildreth's [58] zero crossing edge detector. Zero crossing edge detector is based on filtering each $R(\theta, \phi)$ by the LoG filter:

$$F(\theta, \phi) = \frac{1}{\pi\sigma^4}\left(\frac{\theta^2 + \phi^2}{2\sigma^2} - 1\right)\exp\left(-\frac{\theta^2 + \phi^2}{2\sigma^2}\right) \qquad (4.8)$$

where σ is the scale (smoothing) parameter of the filter. This scale parameter can be adjusted to detect edges in different resolutions, such that a high σ value will lead to rough edges. Similarly, a low σ value will lead to detailed edges. To label edge locations from the LoG filter response, we extract zero crossings with high gradient magnitude. 3D plot of the Log filter is shown in Fig. 4.2.

Our edge detection method has some desirable characteristics. If the object is rotated around its center of mass, the corresponding $R(\theta, \phi)$ function will only translate. Therefore, the new edges obtained will be definitely same as in the original representation. We provide edge detection results for (one of the) single view of the bird object in Fig. 4.3 and the Buddha object in Fig 4.4in both cartesian and spherical coordinate representations.

As can be seen in in Figs. 4.3 and 4.4, edges detected from the spherical representation are more informative than the edges obtained from cartesian coordinate based initial representation. If we look more closely, the neck of the bird is detected both in cartesian and spherical representations. However, smooth transitions such as the eye-

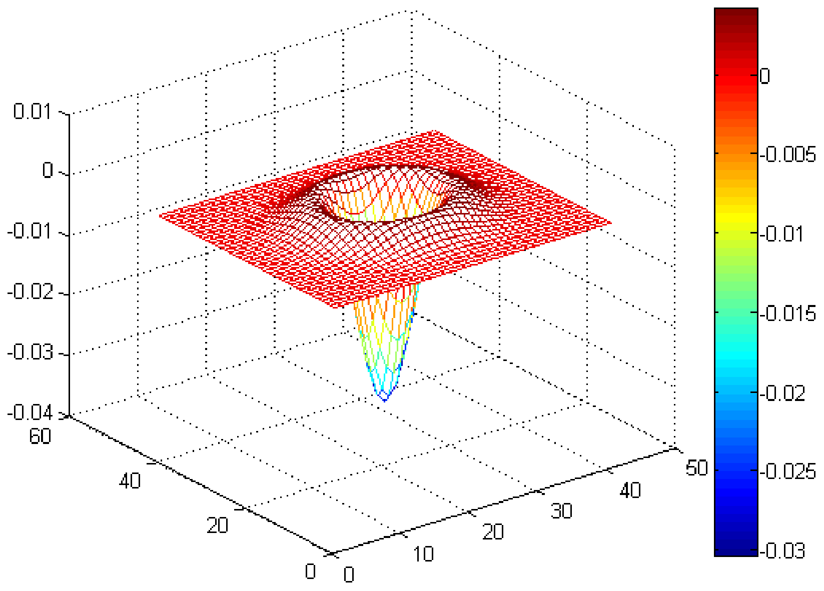
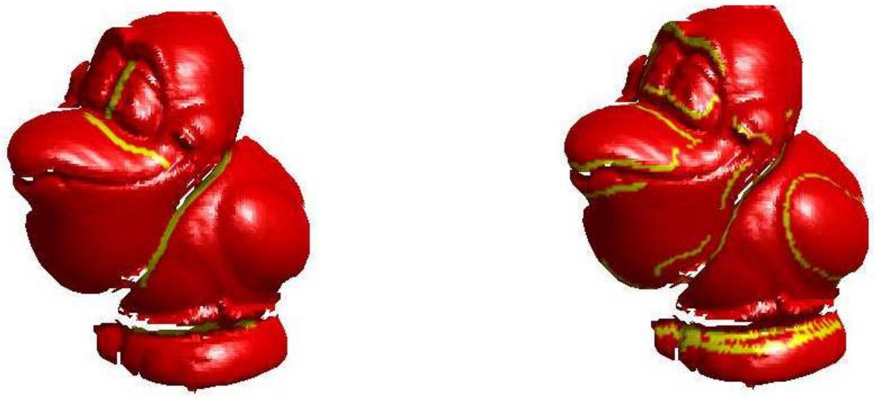
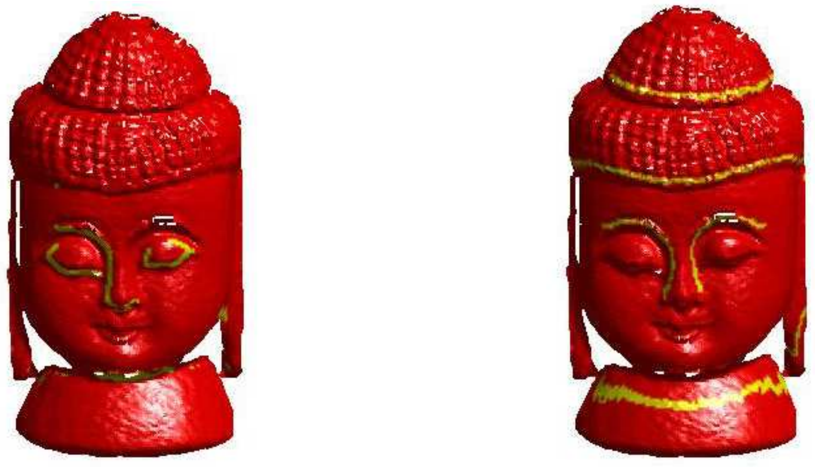


Figure 4.2. Laplacian of Gaussian (LOG) filter



(a) Edges detected in cartesian coordinates (b) Edges detected in spherical coordinates
Figure 4.3. Edge detection results for a patch of bird



(a) Edges detected in cartesian coordinates (b) Edges detected in spherical coordinates

Figure 4.4. Edge detection results for a patch of buddha

lids, wings, and the mouth of the bird is only detected in spherical coordinates. Again in the Buddha example, eyelids, nose, and the hat part is only detected in spherical coordinates. These more representative edges will be of great use in the registration step.

4.2. Model Registration using the ICP Algorithm

We use the ICP algorithm in two modes. In the first mode, we apply ICP on the edge points obtained by our method. This mode is fairly fast and corresponds to a *crude registration*. Then, we apply ICP again to the whole crudely registered data set to obtain the final *fine registration*. Applying this two mode registration procedure decreases the time needed for registration. It also leads to lower registration error compared to applying ICP alone from the beginning.

In both modes of the ICP, we implemented a *kd-tree* search algorithm to speed-up the computation time. Instead of using a linearly searching the corresponding point pairs at every iteration, we use the *kd-tree* representation of the model point set.

5. REGISTRATION RESULTS

We used range image sequences from 10 different objects from Ohio State University, Signal Analysis and Machine Perception Laboratory's database. The range images are obtained by a Minolta Vivid high performance laser ranging system. The dataset is briefly summarized in Table 5.1.

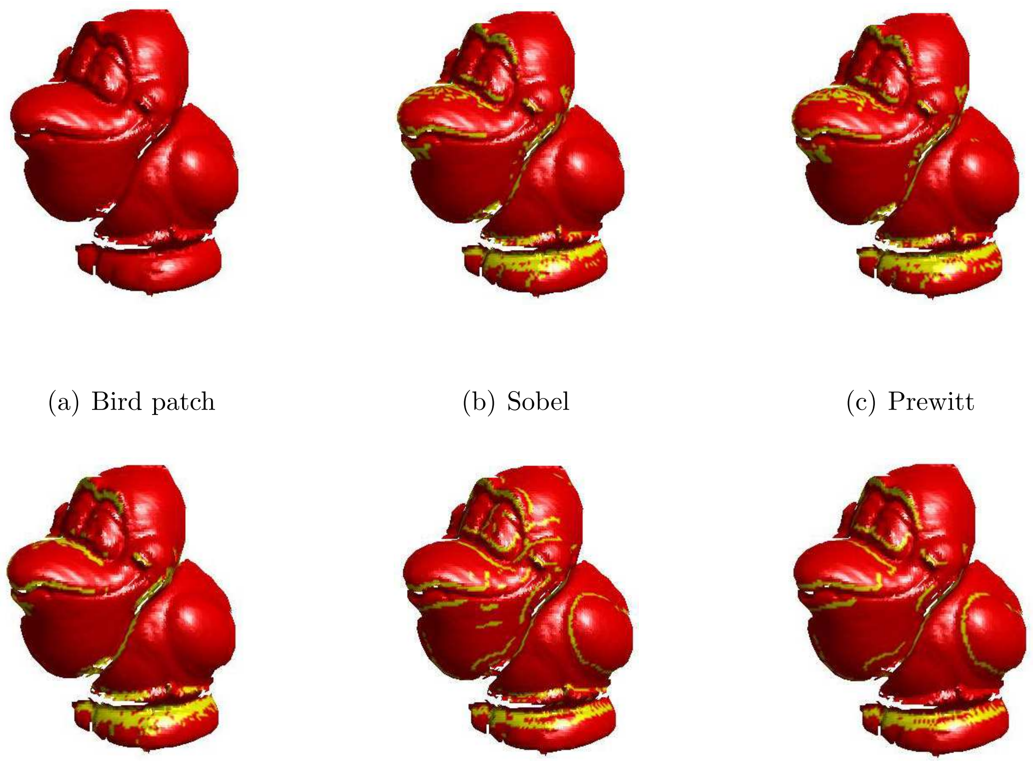
Table 5.1. Range images used

Object	Number of patches	Average number of points	Average number of edge points
angel	18	12413	1236
bird	18	9022	706
blue dino	36	12169	1018
Buddha	18	14051	857
bunny	18	6171	642
doughboy	18	5804	475
duck	18	14028	868
frog	18	10228	758
lobster	18	9925	768
red dino	10	7744	591

In this chapter, we first consider what happens if we use a different edge detection algorithm. Then, we compare our method with the conventional ICP algorithm in crude alignment step in terms of the iteration time and registration error. Based on our crude alignment with edge detection, we further apply fine registration and provide the results.

5.1. The Effect of the Edge Detector

We first compare our LoG edge detector with the Sobel, Prewitt, Roberts, and Canny edge detectors. Details on these edge detectors can be found on any computer vision book such as [57]. We used the Matlab's predefined *edge* function with the parameters adjusted to obtain best results. We tested these edge detectors on two sample object patches as the bird and the red dino. We provide the edge detection results for these patches in Figs. 5.1 and 5.2 respectively.



(a) Bird patch (b) Sobel (c) Prewitt

(d) Roberts (e) Canny (f) LoG

Figure 5.1. Different edge detector results for the first patch of the bird object

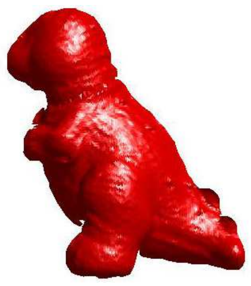
The desirable edge detection results should have two major properties. First, the edge detector should detect boundaries of meaningful parts such as the wing, eyes, eyelids, mouth, and the neck of the bird. Second, there should be no extra detected parts from the bird object corresponding to no physical part. In the light of these constraints, let's look at our edge detection results given in Fig. 5.1. All of the edge detectors were able to detect the neck which is the most salient transition. However, Sobel, Prewitt, and Roberts edge detectors could not detect the wing. These edge detectors also introduced spurious edge points at the bottom of the patch which is undesirable. Compared to the three previous edge detectors, the Canny edge detector was able to detect most desired edge points. Unfortunately, there are also many spurious points as the result of this edge detector. Finally, if we consider the LoG edge detector results, we observe that it is able to detect most of the actual edge points without any spurious pixels labeled as edges. Since these edge points will be used in the registration step, the existing spurious pixels after the Canny edge detection method may cause problems.

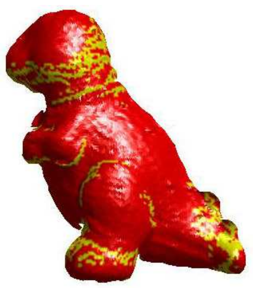
We also compare the existing edge detectors on the red dino object patch given in Fig. 5.2. For this object patch, the most salient edges are the neck, leg, foot, arm, and the spine on the tail. As can be seen in the figure, the Sobel, Prewitt, and Roberts edge detectors produced extra edge points in the foot and spine. These edge detectors also can not completely link the edge in the leg. The Canny edge detector is able to detect the most salient edges, but it also produces spurious points around the ridge. The LOG edge detector is able to detect all salient edges without producing any spurious pixels as in the previous example. Therefore, we can claim that the LoG is the suitable edge detector for our range image set.

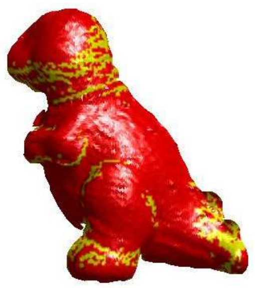
5.2. Crude Registration Results

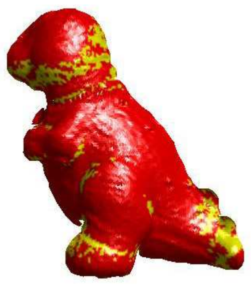
In our crude registration stage, we only used the edge points obtained from our edge detection algorithm explained in the previous chapter. We used the ICP algorithm to register the successive edge points. The rotation and translation parameters obtained from the crude registration is applied to the whole points set. Since we use considerably less number of points in the crude alignment step, there is no need to an initial estimation of rotation and translation parameters. Also the use of edge features avoids the ICP algorithm to be caught to local minima.

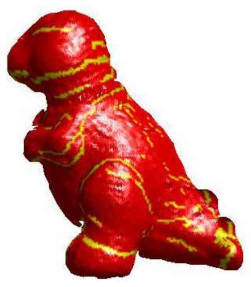
We provide the crude registration result (both edge points and the whole patch) of the bird, red dino and Buddha object patches in Figs. 5.3, 5.4 and 5.5 respectively. The crude registration step using edge points works fairly well. The edge features in the eyelids and the mouth registered perfectly after the crude registration step. As shown in Fig. 5.3 (b), applying the rotation and translation obtained from crude registration


(a) Reddino patch


(b) Sobel


(c) Prewitt


(d) Roberts


(e) Canny

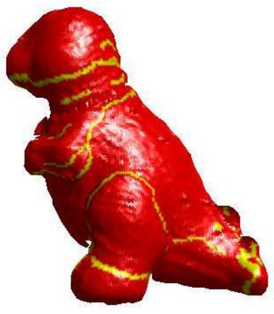
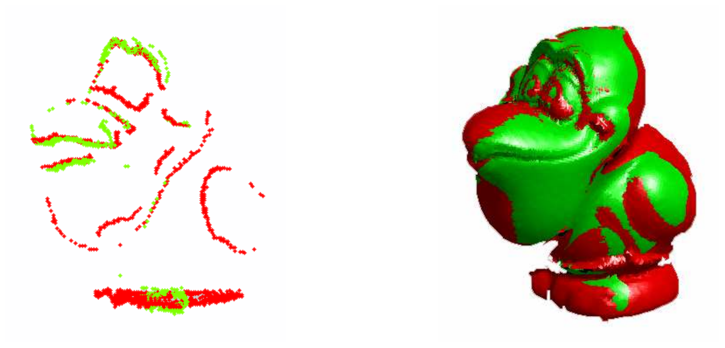

(f) LoG

Figure 5.2. Different edge detector results for the first patch of the red dino object

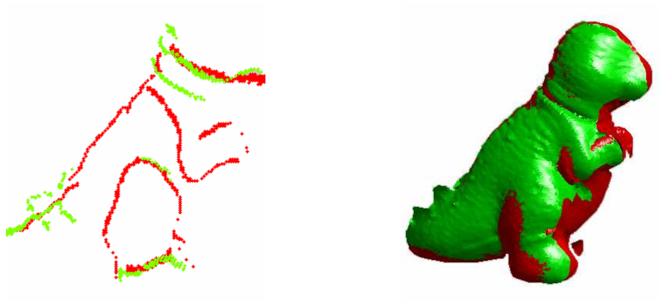
step to the whole points produces quite good results.



(a) Edges after registration (b) Patches after registration

Figure 5.3. Crude registration of the first and second bird patches

For the red dino object, although some edge features in either patch does not corresponds any other edge points, the crude registration step produced perfect results. As can be seen in Fig. 5.4, some edge features on the spine and the neck detected only on one patch depending on the viewpoint. However, edge features detected on the foot and the head part were enough for the registration.

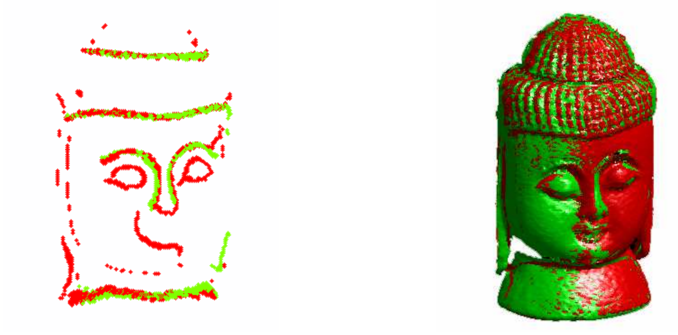


(a) Edges after registration (b) Patches after registration

Figure 5.4. Crude registration of the sixth and seventh red dino patches

Again, the edges in the eye, nose and the hat parts are registered perfectly for the Buddha as shown in Fig. 5.5 (a).

Although it may be perceived from the figures above that there is no need for a fine registration step, little differences leads to undesirable representations in the final



(a) Edges after registration (b) Patches after registration

Figure 5.5. Crude registration of the first and second Buddha patches

3D representation. Therefore, we should register our range images further using the crude registration as the starting point.

5.3. Fine Registration Results

After the crude registration step, we apply the ICP algorithm using the whole point set for fine registration. Since the crude registration brought the patches very close to the real solution, the fine registration step iterates only for few steps.

This section provides the final fine registration of both point and edge representations of the bird, red dino, and angel objects. Each registered patch is represented using a different color to emphasize the registration. We also provide the final 3D representation in one color. We obtained the final representation by applying each transformation obtained from crude and fine alignments of each pair-wise patch successively. At the end of the registration process, differently from the other registration algorithms, we also have the 3D edges of the object as a whole.

Final registration result of bird patches is shown in Fig. 5.6. Both the points and the edges are registered perfectly. When we have a look at the second line figures, we can see that there is not any distortions. Even the rough parts like the neck, mouth, eyes, wings and the tail parts are clear. On the third line, we presented the final registered edges of the bird object. To be more meaningful, the edge points are plotted with the final 3D representation of of the object in yellow color. We can see that all salient features like the neck, wings, mouth, tail, eyes and eyelids are extracted to represent the whole edges of the object. However, there some unlinked edge points around the edge features which can be handled by 3D morphological process. Again the edge features can also be dilated by 3D morphological operations for further processing like recognition as a future work.

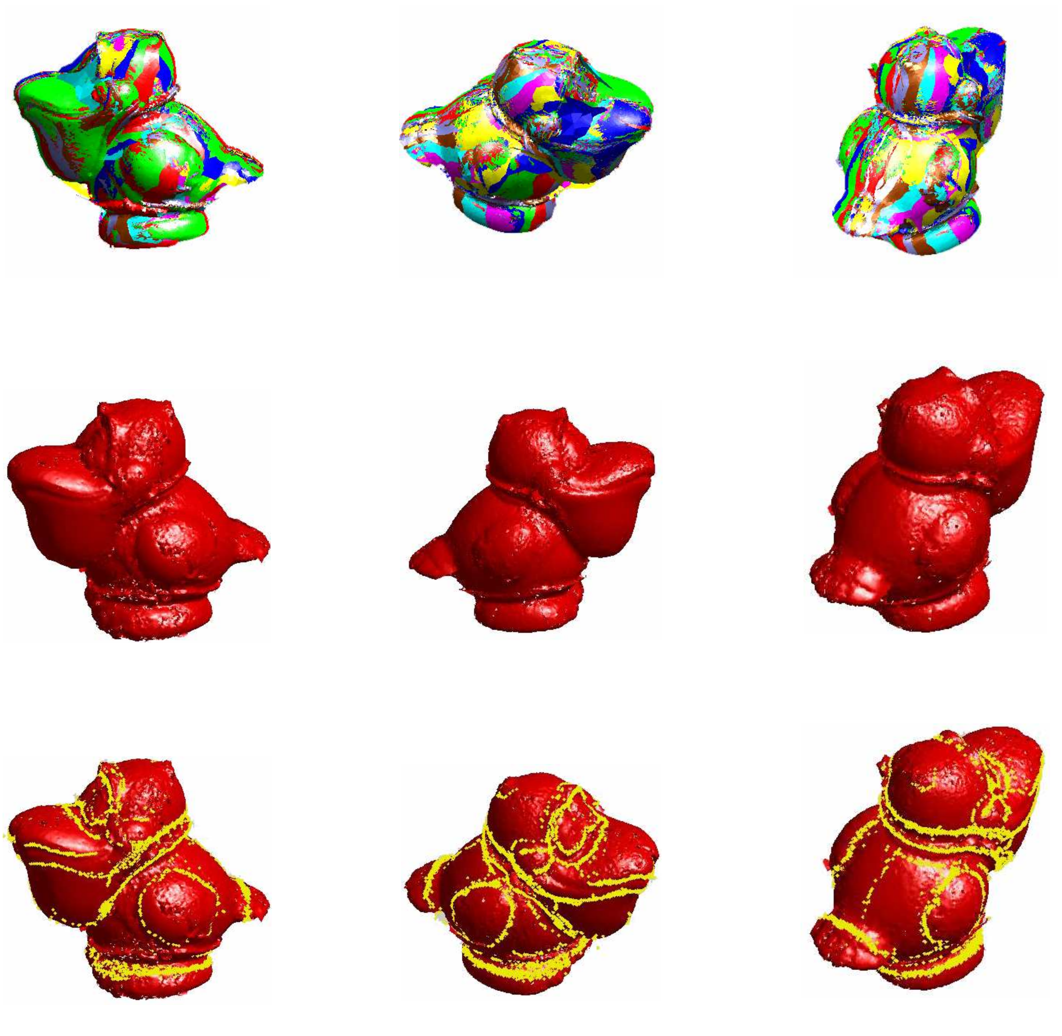


Figure 5.6. Final registration results of the bird object

Similarly, we provide the registration results for the red dino object in Fig. 5.7. Although there are minor problems because of the triangulation in the spines at the tail part, our method worked fine for the red dino object. If we have a look at the final edge representations, all valuable information is extracted. The left and the right arms, both legs and even the foot, the neck, the spines are clearly seen.

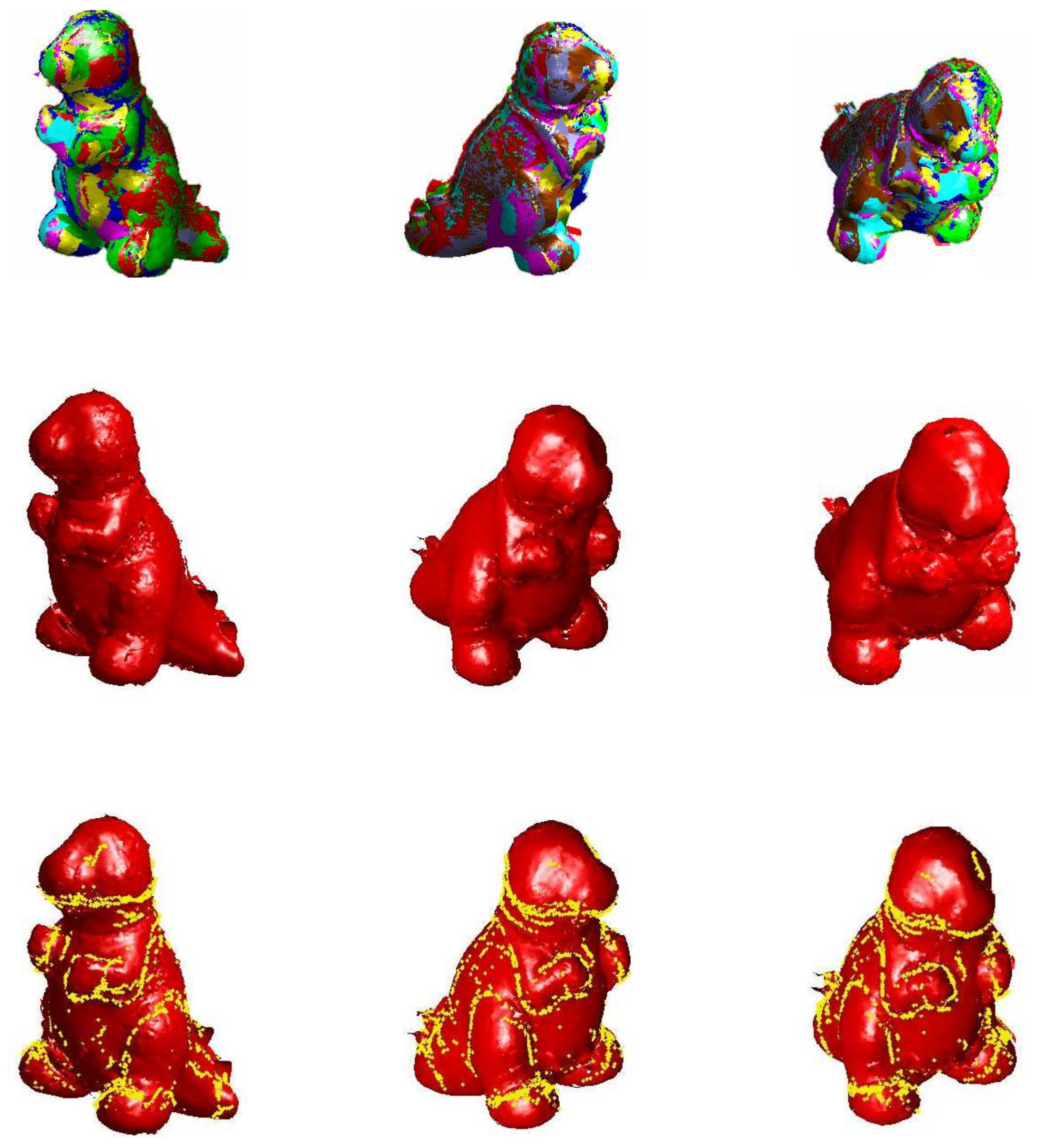


Figure 5.7. Final registration results of the red dino object

The fine registration results for the angel object is given in Fig. 5.8. Even the patterns inside wings of the angel and the hair parts are clearly seen. At first sight, one can think that there are too many spurious edge information. However, because of those patterns especially in the wings, we have a rough surface which leads more edge features. If we look closer, we can see the salient transitions are thicker like the arms, neck, eyelids.

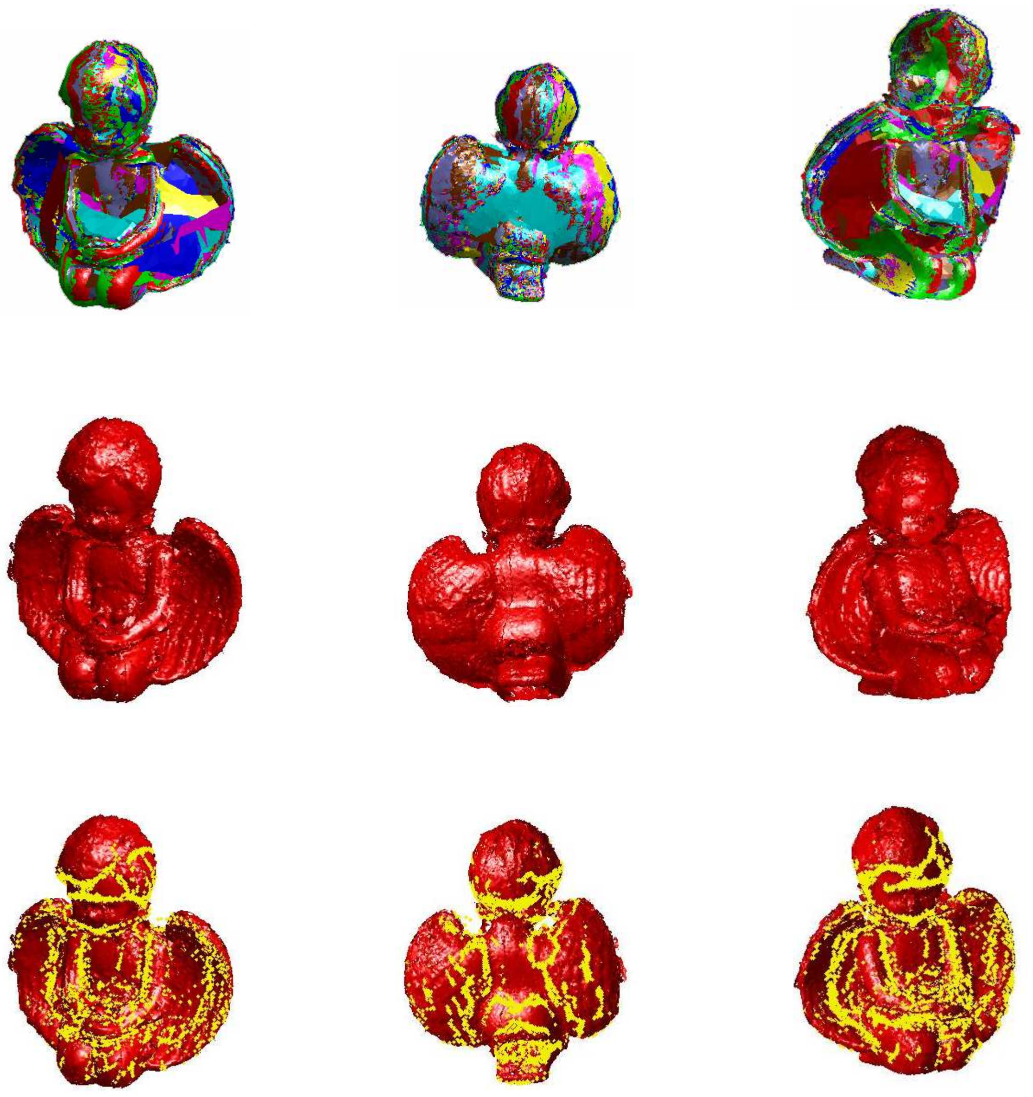


Figure 5.8. Final registration results of the angel object

5.4. Comparison with ICP in terms of Timings

We provide the comparison of our method with the conventional ICP algorithm and provide the results below. While performing registration tests, we used a PC with an AMD Athlon CPU with 3500 MHz. clock speed, with 2 GB RAM. In the tables below, the first column corresponds to the number of patches being registered. The second column (labeled as *ICP alone*) corresponds to constructing the model (from all patches) using ICP alone. The third and fourth columns correspond to crude and fine registration steps of our method, (labeled as *Crude registration* and *Fine registration* respectively). The timings in the third column include the edge detection and coordinate conversion steps. The fifth column indicates to the total time needed for our method for registration (labeled as *Crude + Fine reg.*). The last column corresponds to the gain if we switch from ICP alone to our two mode registration method.

In Table 5.2 we provide the pair-wise registration times of the red dino patches. As can be seen we have an average gain of 4.75 times faster computation. Actually, the time savings here are related to the acquisition viewpoint of the successive patches. More overlapping region between each successive scan leads more corresponding edge features which increases the speed up factor of our algorithm. As seen, our crude registration step needs at most 1.35 sec. compared to 138.06 for ICP alone since we are using only the edge points. This is huge time savings. Unfortunately, we perform the fine registration which requires extra processing. Nevertheless, our algorithm performed better compared with the conventional ICP algorithm in terms of computation time.

In Table 5.3 we provide the the registration times of different bird patches. Our method again provides 4.67 times faster execution time compared to ICP algorithm on average. However, in the registration of patches 11-12 and 12-13 our method performs worse than ICP. The reason for this is the low overlapping between the edge features of those patches. We can see that our fine registration step took more than the ICP alone although we applied crude registration. The reason for this is our method reaches better final registration error compared to ICP which requires a few more iterations.

Table 5.2. Comparison of the CPU timings (in sec.) for the registration of the red dino object

Patches	ICP alone	Crude registration	Fine registration	Crude and Fine registration	Gain
1-2	79.80	0.77	3.50	4.27	18.70
2-3	138.06	1.25	39.52	40.77	3.39
3-4	99.03	1.35	19.59	20.94	4.73
4-5	79.80	0.83	3.20	4.03	19.80
5-6	75.33	0.94	14.27	15.20	4.95
6-7	119.71	0.59	5.55	6.14	19.50
7-8	108.99	0.55	32.38	32.92	3.31
8-9	56.53	0.72	17.66	18.38	3.08
9-10	105.17	0.53	38.27	38.80	2.71
Average	**95.82**	**0.84**	**19.32**	**20.16**	**4.75**

We will compare the registration errors in the next section.

Finally, we compare the total iteration times to register all the pathes of ten objects in Table. 5.4 in terms of CPU timings (in sec.). Different from the previous tables in this section, the times presented in this table is the sum of the times to register all patches of a particular object. The numbers in parenthesis int he first column represents the number of total scans of that object. On the average we have a gain of **4.82** over ten objects. If we can tolerate crude alignment for any application, our gain becomes **126.37**.

5.5. Comparison with ICP in terms of Registration Error

Here, we compare our algorithm with the ICP in terms of registration error. We provide alignment errors for four pair of patches from three different objects (bird, red dino and angel) in Fig. 5.9, Fig. 5.10 and Fig. 5.11, respectively.

Table 5.3. Comparison of the CPU timings (in sec.) for the registration of the bird object

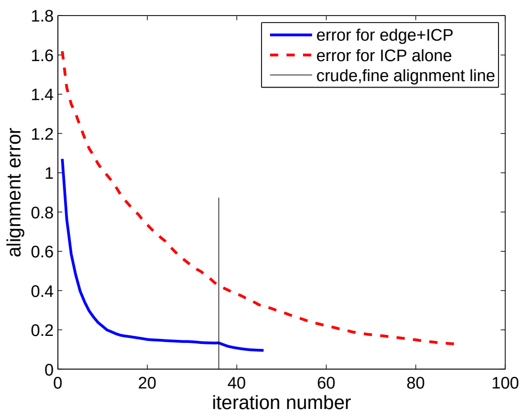
Patches	ICP alone	Crude registration	Fine registration	Crude and Fine registration	Gain
1-2	40.85	0.75	8.20	8.95	4.56
2-3	36.64	0.27	6.33	6.59	5.56
3-4	25.55	0.30	6.09	6.39	4.00
4-5	20.75	0.27	15.55	15.81	1.31
5-6	44.56	1.19	2.09	3.28	13.58
6-7	65.83	1.05	6.06	7.11	9.26
7-8	105.02	0.84	7.89	8.74	12.02
8-9	113.55	0.70	7.30	8.00	14.19
9-10	91.49	0.67	24.66	25.33	3.61
10-11	44.56	1.29	2.56	3.85	11.57
11-12	43.69	0.47	50.83	51.30	0.85
12-13	17.98	0.61	21.09	21.70	0.83
13-14	67.50	0.27	46.06	46.33	1.46
14-15	130.22	0.84	14.64	15.49	8.41
15-16	118.19	0.81	10.34	11.16	10.59
16-17	108.14	0.75	6.16	6.91	15.65
17-18	111.44	0.80	5.97	6.77	16.47
Average	**69.76**	**0.70**	**14.23**	**14.92**	**4.67**

Table 5.4. Comparison of the CPU timings (in sec.) over nine objects

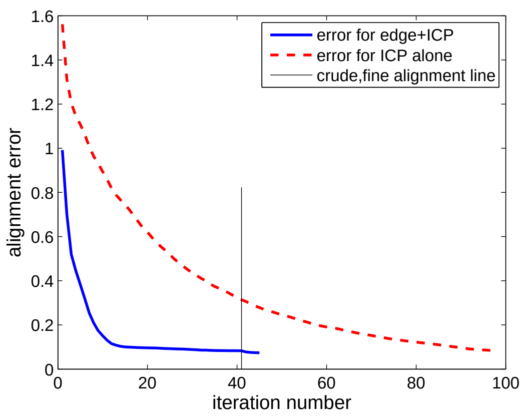
Object	ICP alone	Crude registration	Fine registration	Crude and Fine registration	Gain
red dino (10)	862.42	7.46	174.22	181.68	4.75
bird (18)	1185.94	11.77	241.37	253.14	4.68
frog (18)	2106.51	17.22	242.55	259.77	8.11
duck (18)	3292.84	27.27	646.33	673.61	4.89
angel (18)	2298.81	30.59	910.02	940.61	2.44
blue dino (36)	4780.04	26.31	1045.12	1071.43	4.46
bunny (18)	735.51	9.44	161.50	170.95	4.30
doughboy (18)	1483.77	6.77	233.38	240.15	6.18
lobster (18)	2964.70	19.18	586.66	605.85	4.89
Buddha (18)	2990.26	23.97	693.40	717.37	3.47
Average	**2220.08**	**18.00**	**493.46**	**511.45**	**4.82**

In all these figures, we provide the alignment error of the ICP algorithm wrt. iteration number (in red dashed lines). We also provide the alignment errors of our crude (using only edge points) and fine alignment (using all points after crude alignment) method in solid lines. We label the iteration step where we switch from crude to fine alignment by a vertical crude, fine alignment line in these figures.

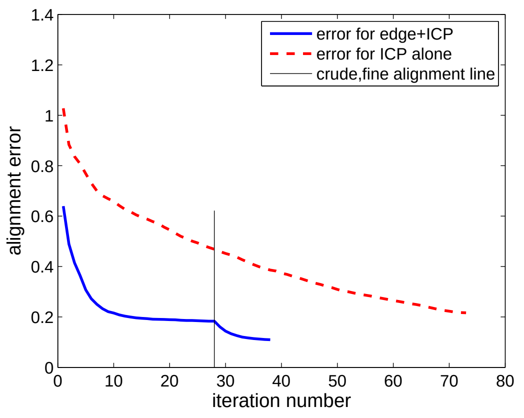
As can be seen, in our alignment tests the ICP algorithm has an exponentially decreasing alignment error. Our crude alignment method performs similarly on all experiments while converging in fewer iterations. It can be seen that our crude alignment reaches almost the same final error value. In order to have better visual results, we need to apply the fine registration for a few more steps.



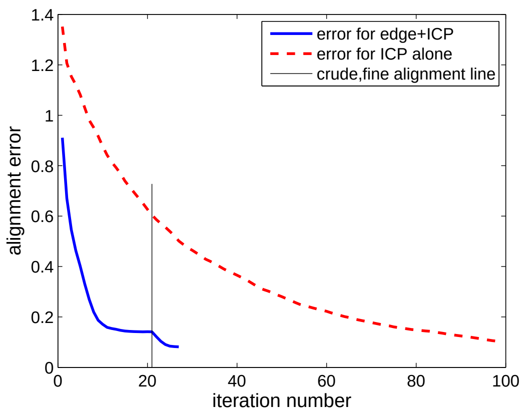
(a) The 1. and 2. bird patches



(b) The 5. and 6. bird patches

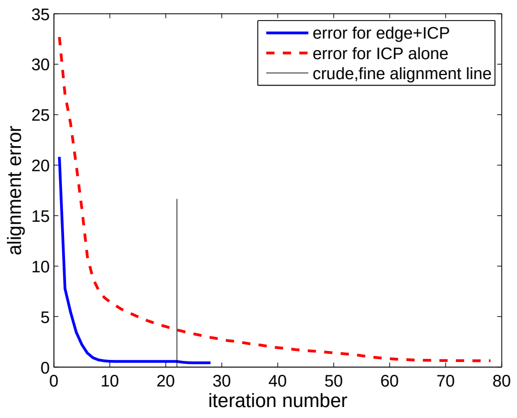


(c) The 15. and 16. bird patches

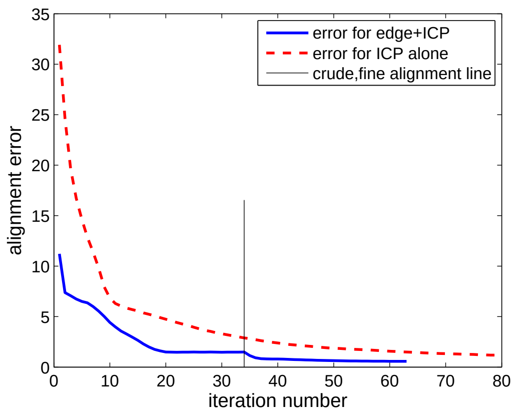


(d) The 17. and 18. bird patches

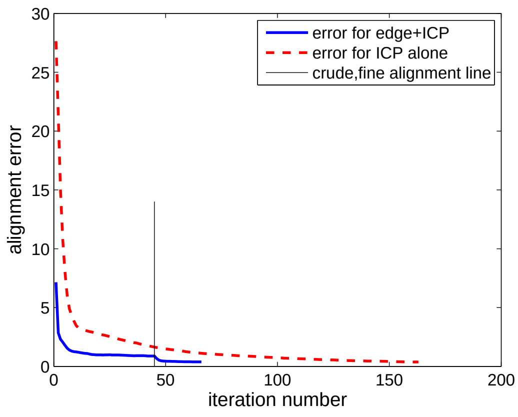
Figure 5.9. Comparison of registration errors on four pairs of bird patches.



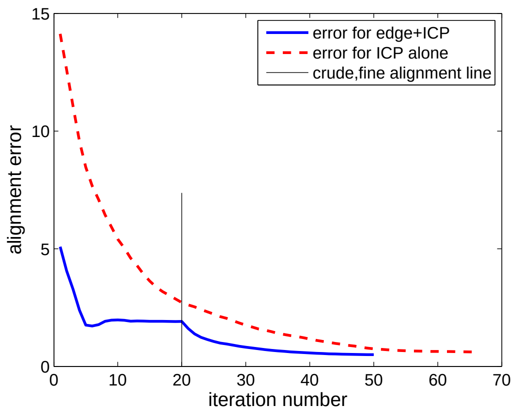
(a) The 1. and 2. red dino patches



(b) The 5. and 6. red dino patches

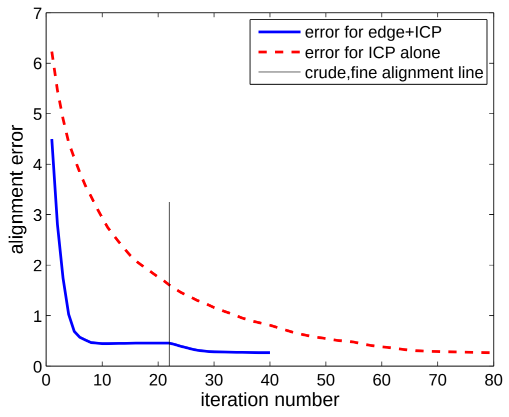


(c) The 6. and 7. red dino patches

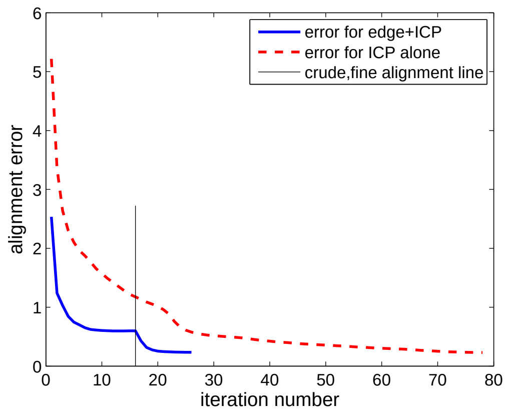


(d) The 8. and 9. red dino patches

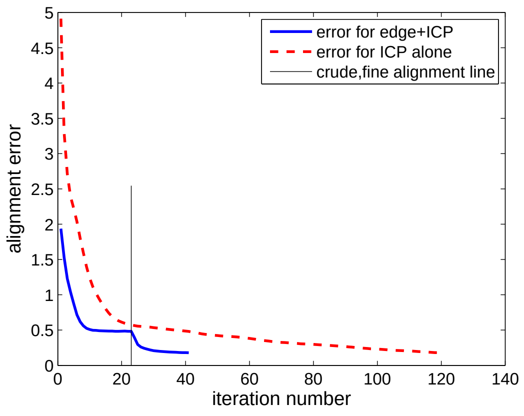
Figure 5.10. Comparison of registration errors on four pairs of red dino patches.



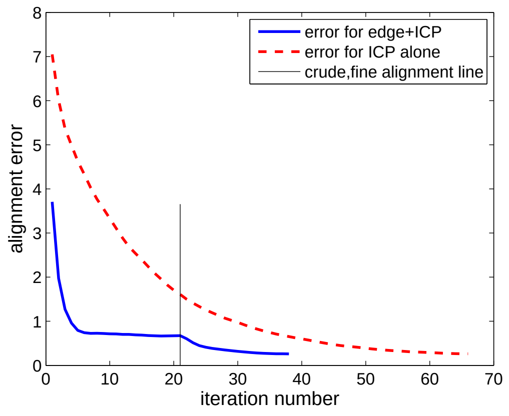
(a) The 3. and 4. angel patches



(b) The 8. and 9. angel patches



(c) The 11. and 12. angel patches



(d) The 16. and 17. angel patches

Figure 5.11. Comparison of registration errors on four pairs of angel patches.

6. SEGMENTING 3D OBJECT MODELS

Segmenting 3D object surfaces is required in many high level computer vision and graphics applications. In computer vision, recognizing and estimating poses of 3D objects heavily depend on segmentation results. Similarly, physically meaningful segments of a 3D object may be useful in various computer graphics applications. Therefore, there are many segmentation algorithms in the literature. Unfortunately, most of these algorithms can not perform reliably on free-form objects. In order to segment free-form objects, we introduce a novel method in this chapter.

Different from previous segmentation methods, we first obtain a function representation of the object surface in spherical coordinates as we did in Chapter 4. There, we were able to obtain the edge information from separate patches. There, we were able to represent those patches as functions in spherical coordinates. Unfortunately, we can not represent general 3D free form object models as functions in spherical coordinates. First, we provide a way to decompose a 3D object model as separate functions using ray tracing. Then, we apply our previous edge detection method on each function separately. Let's first start with explaining the ray tracing algorithm we used.

6.1. Ray Tracing

We use the ray tracer introduced by Moller and Trumbore [59]. This algorithm checks whether a ray intersects a triangle or not. The algorithm translates the origin of the ray and then changes the base of that vector which yields a vector $(t\ u\ v)^T$ where t is the distance to the plane in which the triangle lies and (u, v) represents the coordinates inside the triangle. One advantage of this method is that the plane equation need not be computed on the fly nor be stored which can amount to significant memory savings for triangle meshes.

A ray $R(t)$ with origin O and normalized direction D is defined as

$$R(t) = O + tD \tag{6.1}$$

and a triangle is defined by three vertices V_0, V_1 and V_2. A point $T(u,v)$ on a triangle is given by

$$T(u,v) = (1-u-v)V_0 + uV_1 + vV_2 \tag{6.2}$$

where (u,v) are the barycentric coordinates, which must fulfill $u \geq 0$, $v \geq 0$ and $u+v \leq 1$. Computing the intersection between the ray, $R(t)$, and the triangle, $T(u,v)$, is equivalent to $R(t) = T(u,v)$, which yields:

$$O + tD = (1-u-v)V_0 + uV_1 + vV2 \tag{6.3}$$

Rearranging the terms gives:

$$\begin{bmatrix} -D, & V_1 - V_0, & V_2 - V_0 \end{bmatrix} \begin{bmatrix} t \\ u \\ v \end{bmatrix} = O - V_0 \tag{6.4}$$

This means the barycentric coordinates (u, v) and the distance, t, from the ray origin to the intersection point can be found by solving the linear system of equations above. The above can be thought of geometrically as translating the triangle to the origin, and transforming it to a unit triangle in $y\&z$ with the ray direction aligned with x, as illustrated in Fig. 6.1 (where $M = [-D, V_1 - V_0, V_2 - V_0]$ is the matrix in Eqn. 6.4).

Denoting $\overrightarrow{E_1} = V_1 - V_0$, $\overrightarrow{E_2} = V_2 - V_0$ and $\overrightarrow{E_3} = O - V_0$, the solution to Eqn. 6.4 is obtained by using Cramer's rule:

$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{|-D\ E_1\ E_2|} \begin{bmatrix} |E_3\ E_1\ E_2| \\ |-D\ T\ E_2| \\ |-D\ E_1\ T| \end{bmatrix} \tag{6.5}$$

From linear algebra it is known that $|A\ B\ C| = -(A \times C) \cdot B = -(C \times B) \cdot A$. Eq. 6.5 could therefore be rewritten as

$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{(D \times E_2) \cdot E_1} \begin{bmatrix} (T \times E_1) \cdot E_2 \\ (D \times E_2) \cdot E_3 \\ (T \times E_1) \cdot D \end{bmatrix} = \frac{1}{W \cdot E_1} \begin{bmatrix} Q \cdot E_2 \\ W \cdot E_3 \\ Q \cdot D \end{bmatrix} \tag{6.6}$$

where $W = (D \times E_2)$ and $Q = T \times E_1$. In the implementation these factors are reused to speed up the computations.

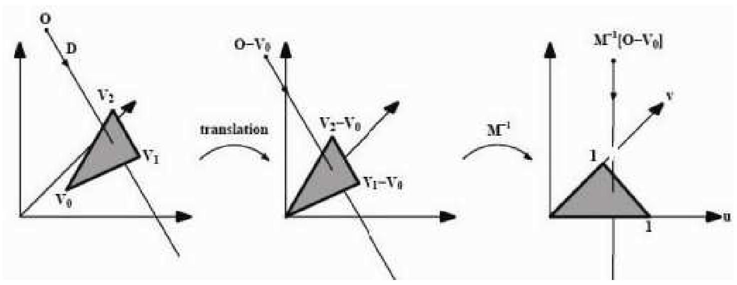


Figure 6.1. Translation and change of base of the ray origin

The algorithm in Fig. 6.2 explains the decomposition process for a given object to have spherical functions where N_p is the number of points, N_t is the number of triangles, v_i is the i_{th} point and t_j is the j_{th} triangle. The decomposition process goes on iteratively until all parts are star shaped.

Algorithm 4 Decomposition

```
Begin
$x_c = 0,\ y_c = 0,\ z_c = 0$
for $i = 1$ to $N_p$ do
  $x_c = x_c + x_i$
  $y_c = y_c + y_i$
  $z_c = z_c + z_i$
end for
$x_c = x_c \div N_p,\ y_c = y_c \div N_p,\ z_c = z_c \div N_p$
for $i = 1$ to $N_p$ do
  $x_i = x_i - x_c$: $y_i = y_i - y_c$; $z_i = z_i - z_c$;
end for
for $i = 1$ to $N_p$ do
  $\overrightarrow{v_i} = [x_i\ y_i\ z_i]$
  for $j = 1$ to $N_t$ do
    if $(\overrightarrow{v_i}$ intersects $t_j) \wedge (i \notin t_j)$ then
      $R(\theta, \phi) = \sqrt{x_i^2 + y_i^2 + z_i^2}$
    else
      $R_2(\theta, \phi) = \sqrt{x_i^2 + y_i^2 + z_i^2}$
    end if
  end for
end for
End.
```

Figure 6.2. Decomposition algorithm to obtain spherical functions

6.2. Decomposing the 3D Object Model to Separate Functions

In order to segment the 3D object model, we first obtain its representation in spherical coordinates. As we have the 3D object model in spherical coordinates, for a given pan and tilt angle there may be more than one radius value. Therefore, we may not be able to represent them in a single $R(\theta, \phi)$ function, since it violates the definition of being a function (having more than one value for any index). In order to avoid this problem, we use Moller and Trumbore's ray tracing method to obtain separate function representations for the 3D object model. Our 3D object model must be represented in terms of triangular patches. Fortunately, most of the model representations support this representation.

Let's explain our decomposition method in detail. We start with obtaining the center of mass of the object. From this center, we swipe all pan and tilt angles and apply ray tracing to obtain multiple intersections on the surface. More specifically, we test for ray triangle intersection and obtain the closest point set on the object. For each point, the ray is the line that comes from the center of mass through the point. We test if this ray is intersecting any triangle except the ones including that point. If it is, then we represent the further point in a different $R(\theta, \phi)$ function. We apply this procedure iteratively, until we obtain a full decomposition such that each decomposition corresponds to a separate function in spherical coordinates. Therefore, we obtain multiple $R(\theta, \phi)$ representations for the same object.

We test this decomposition operation on the crocodile object given in Fig. 7.3. In this 3D object model, we only have multiple radius values around the tail part for a given pan and tilt angles.

As we apply our decomposition method to this model, we obtain two function representations. The first function corresponds to the body of the crocodile, and the second corresponds to the tail. We provide the $R(\theta, \phi)$ functions for these two parts in Fig. 6.4.

As we obtain separate function representations from the 3D object model, we apply the edge detection method (explained in detail in Chapter 4) on each function separately. Then, we project the edge points obtained from each function to the original object to obtain the final segmentation of the 3D model. In the next chapter, we provide various examples using this segmentation method.

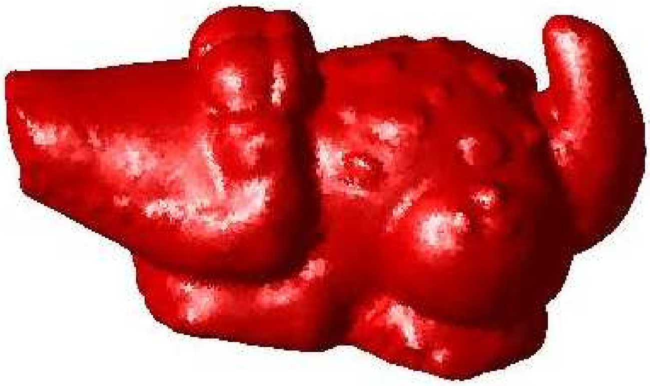
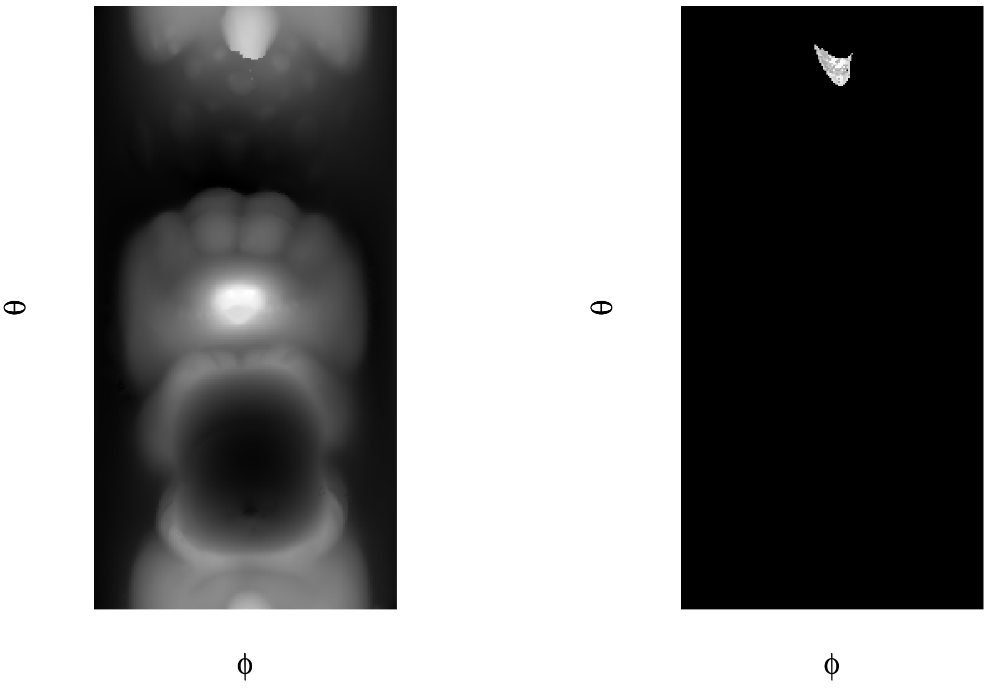


Figure 6.3. The crocodile object



(a) The first $R(\theta, \phi)$ function (b) The second $R(\theta, \phi)$ function

Figure 6.4. Decomposing the crocodile object to two $R(\theta, \phi)$ functions

7. SEGMENTATION RESULTS

We test our segmentation method on eight different free-form objects with two different filter scale parameters, $\sigma = 1.8$ (to obtain a detailed segmentation) and $\sigma = 4$ (to obtain a general segmentation). We provide three different views of the segmentation results for each object in these figures. Segmentation results for the apple object is provided in Fig. 7.1. Our segmentation method was able to segment the stem and body of the apple for σ scale 4.0, also the concave hole regions at the bottom and top parts are segmented. However, for the σ scale 1.8, our segmentation method produced some over segmented areas especially at the top hole part around the stem. Nevertheless, even those extra segmented regions have similar curvature properties at first sight.

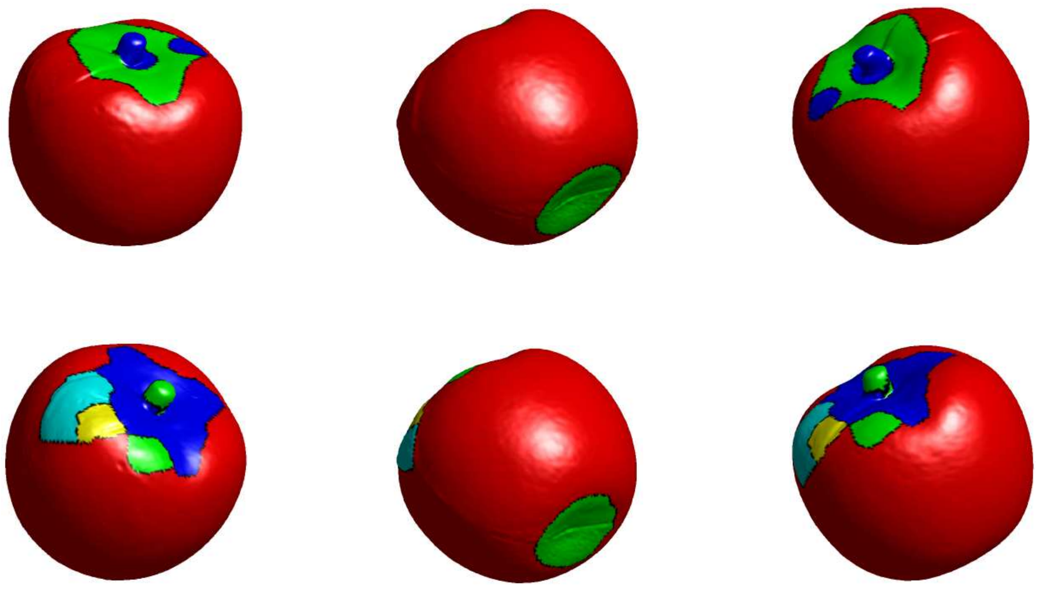


Figure 7.1. General segments of the apple object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

For the cow object, as we hypothesized, our method segmented the physically meaningful parts of the cow object like the legs, head and the body as provided in Fig. 7.2. While for σ scale 4.0, only the most salient parts are segmented such as head and legs, for σ scale 1.8, we can segment out the less salient parts such as ears and mouth.

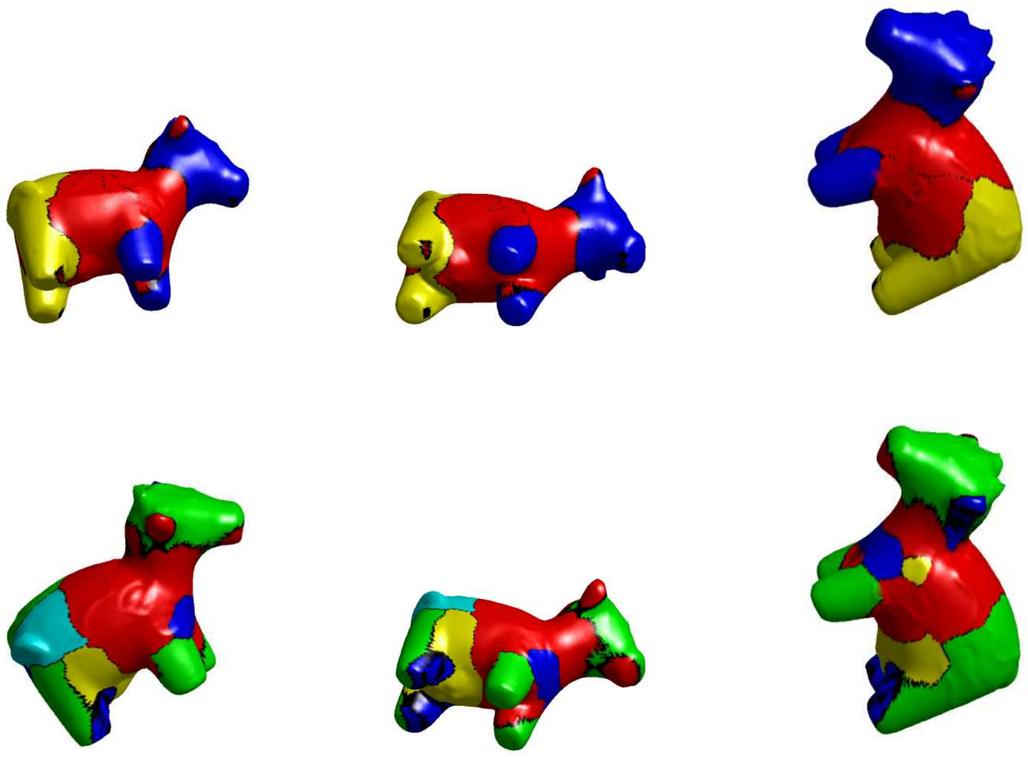


Figure 7.2. General segments of the cow object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

Our method produced similar results for the crocodile object like in the cow object as provided in Fig. 7.3. For higher σ values, more salient parts are segmented such as tail, legs and the eyes of the crocodile object. When the σ value is lower, our method even segmented the smoother parts such as the spots over the body of the crocodile and both eyes from each other.

For the dumbbell object, it is clear that the object has 3 different segments corresponding to physically meaningful parts such as both spherical parts on each side and the stick connecting these parts together. When we look at our segmentation results for this object in Fig. 7.4, we can see that our method segmented perfectly those parts for the σ value 4.0. However, for σ value 1.8, our method produced some extra segmented regions.

As can be seen from Figs. **??**, 7.5, 7.6 and 7.7 our segmentation method works fairly well on eight different free-form 3D objects having fairly diverse characteristics.

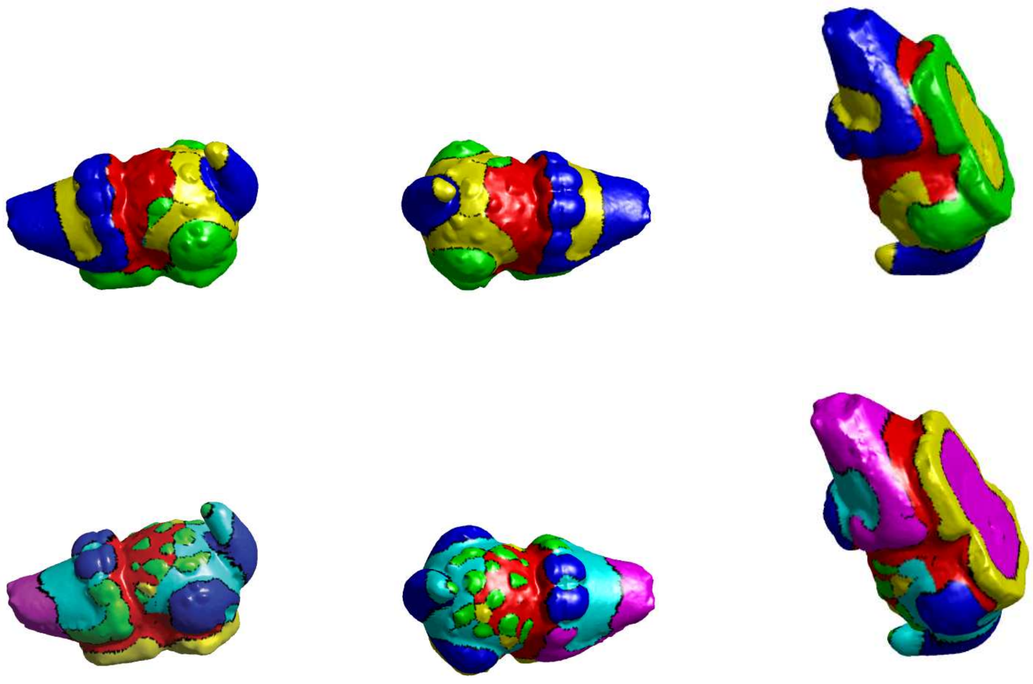


Figure 7.3. General segments of the crocodile object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

Almost all the segments labelled on the actual objects correspond to different object parts. In general, there are minor edge detection problems due to the scale of the filter for $\sigma = 1.8$. At this scale, obtained edges become weaker, hence there may be missing links on some objects. However, this problem can be fixed by a post processing step.

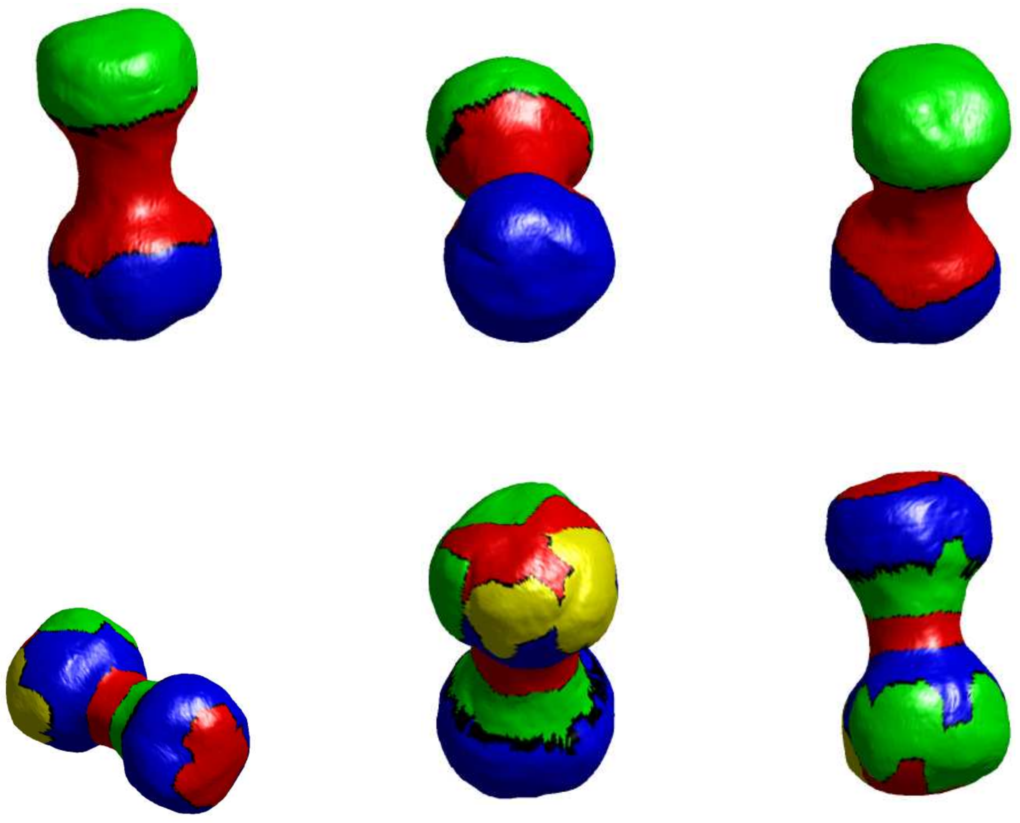


Figure 7.4. General segments of the dumbbell object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

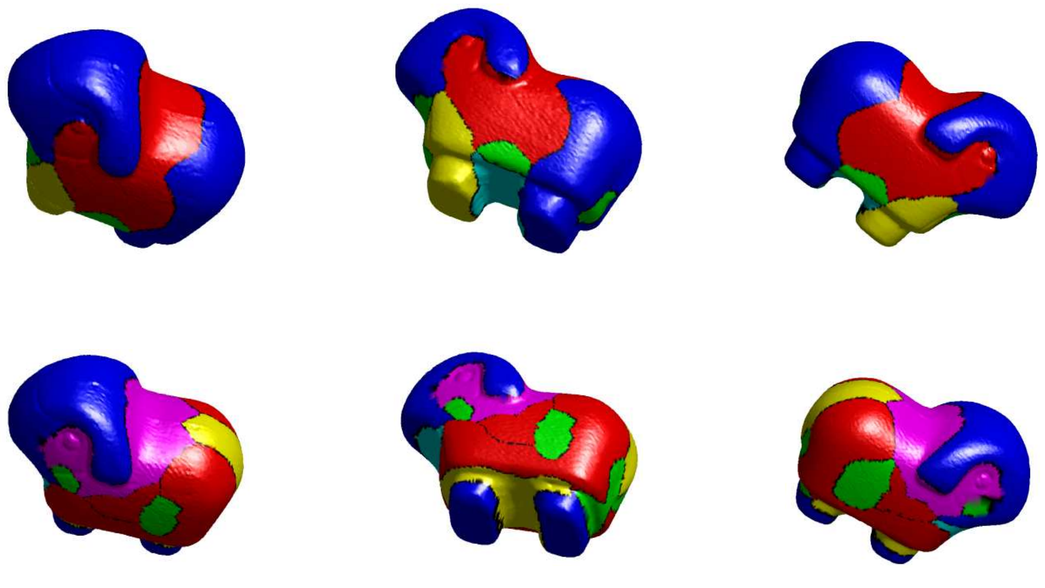


Figure 7.5. General segments of the lamb object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

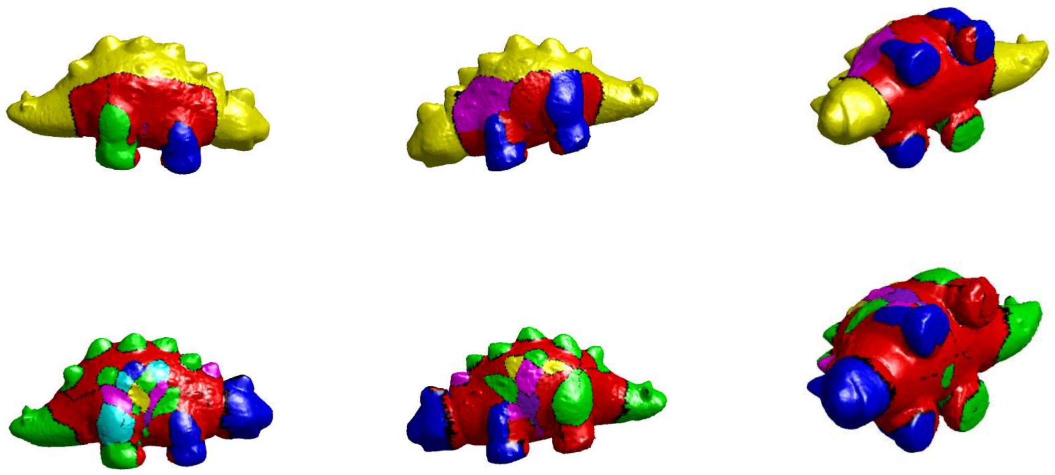


Figure 7.6. General segments of the orange dino object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

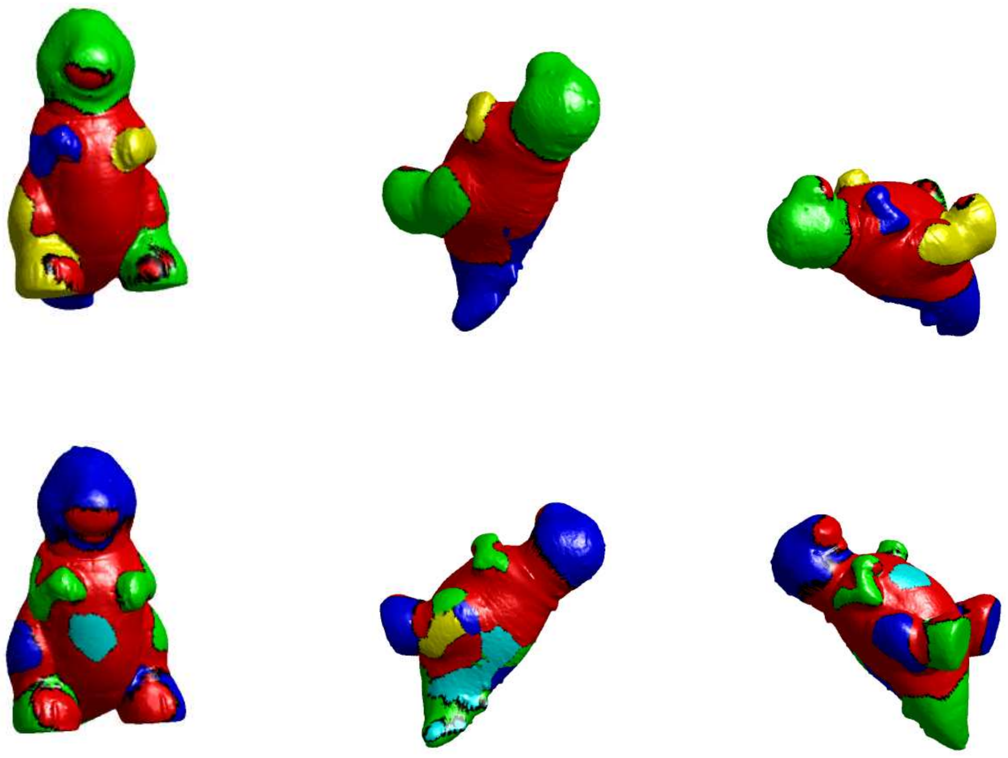


Figure 7.7. General segments of the red dino object with $\sigma = 4.0$ and $\sigma = 1.8$ respectively

8. CONCLUSIONS

In this thesis, we propose a robust and accurate range image registration algorithm and a novel edge detection method for range images. Our edge detection method is based on spherical coordinate conversion. Before applying edge detection, we apply a coordinate conversion (from cartesian to spherical coordinates). This conversion allows us to detect smooth edges easily. Therefore, we can detect edges from free-form objects. Instead of using the whole points of successive range images in the registration step, we use the edge features obtained by our edge detection algorithm. While this crude registration step provides extremely fast processing, for more precise results we need to apply ICP algorithm to the whole point set. Even in this case, our proposed method is much more efficient and accurate compared to conventional ICP algorithm as we provide in the previous chapters. Our method solved the initial estimation problem of ICP algorithm which provides automatically registering range images. Another contribution of our method is we avoid to be caught to any local minimum solutions since we use informative edge features corresponding to physically meaningful parts of the objects. Besides, at the end we also have the edge information of the registered 3D object to be used for classification or recognition.

We compare our method with the conventional ICP algorithm in terms of computational complexity and final registration error. The results show that our method is more efficient and more accurate. Registration process is five times than the conventional ICP algorithm. Also final registration errors is less which means we have better visual quality in the final 3D representation of the objects.

In this thesis, we also propose a 3D object segmentation method. Segmenting 3D objects is required for many high level processing such as classification and recognition. We use the same edge detection algorithm based on spherical coordinate conversion for segmentation. We also propose a ray tracing method to decompose any 3D object to spherical functions. We provide our segmentation results on eight different free-form 3D objects having fairly diverse characteristics.

REFERENCES

1. Besl, P. J. and D. N. McKay, "*A Medhod for Registration of 3-D Shapes*", *IEEE Trans. on PAMI*, Vol. 14, pp. 239–256, 1992.

2. Rusinkiewicz, S. and M. Levoy, "*Efficient Variants of the ICP Algorithm*", *Proc. Third Intl Conf. 3-D Digital Imaging and Modeling*, pp. 145–152, 2001.

3. Rodrigues, M., R. Fisher and Y. Liu, "*On the Representation of Rigid Body Transformations for Accurate Registration of Free-Form Shapes*", *Computer Vision and Image Understanding*, Vol. 87, pp. 1–7, 2002.

4. Arun, K. S., T. S. Huang and S. D. Blostein, "*Least Squares Fitting of Two 3-D Point Sets*", *Computer Vision and Image Understanding*, Vol. 81, pp. 117–142, 2001.

5. Horn, B. K. P., "*Closed Form Solution of Absolute Orientation Using Unit Quaternions*", *JOSA A.*, Vol. 4, pp. 629–642, 1987.

6. Zhang, Z., "*Iterative Point Matching for Registration of Free-form Curves and Surfaces,*", *Int. Jour. Computer Vision*, Vol. 13, pp. 119–152, 1994.

7. Blais, G. and M. D. Levine, "*Registering Multiview Range Data to Create 3D Computer Objects*", *IEEE Trans. on PAMI*, Vol. 17, pp. 820–824, 1995.

8. Turk, G. and M. Levoy, "*Zippered polygon meshes from range images*", *Proceedings on SIGGRAPH*, pp. 311–318, 1994.

9. Rodrigues, M. A. and Y. Liu, "*On the Representation of Rigid Body Transformations for Accurate Registration of Free-Form Shapes*", *Robotics and Autonomous Systems*, Vol. 39, pp. 37–52, 2002.

10. Okatani, I. S. and K. Deguchi, "*A Method for Fine Registration of Multiple View*

Range Images Considering the Measurement Error Properties", *Computer Vision and Image Understanding*, Vol. 87, pp. 66–77, 2002.

11. Liu, B. and B. Wei, "*Developing structural constraints for accurate registration of overlapping range images*", *Robotics and Autonomous Systems*, Vol. 47, pp. 11–30, 2004.

12. Xiao, G., S. H. Ong and G. W. C. Foong, "*Efficient Partial-Surface Registration for 3D Objects*", *Computer Vision and Image Understanding*, Vol. 98, pp. 271–294, 2005.

13. Pottmann, H., S. Leopoldseder and M. Hofer, "*Registration without ICP*", *Computer Vision and Image Understanding*, Vol. 95, pp. 54–71, 2004.

14. Dalley, G. and P. J. Flynn, "*Pair-Wise Range Image Registration: A Study in Outlier Classification*", *Computer Vision and Image Understanding*, Vol. 87, pp. 104–115, 2002.

15. Lucchese, L., G. Doretto and G. M. Cortelazzo, "*A Frequency Domain Technique for Range Data Registration*", *IEEE Trans. on PAMI*, Vol. 24, pp. 1468–1484, 2002.

16. Krsek, P., T. Pajdla and V. Hlavac, "*Differential Invariants as the Base of Triangulated Surface Registration*", *Computer Vision and Image Understanding*, Vol. 87, pp. 27–38, 2002.

17. Sharp, G. C., S. W. Lee and D. K. Wehe, "*ICP Registration Using Invariant Features*", *IEEE Trans. on PAMI*, Vol. 24, pp. 90–102, 2002.

18. Sablatnig, R. and M. Kampel, "*Model-Based Registration of Front- and Backviews of Rotationally Symmetric Objects*", *Computer Vision and Image Understanding*, Vol. 87, pp. 90–103, 2002.

19. Wyngaerd, J. V. and L. V. Gool, "*Automatic Crude Patch Registration: To-*

ward Automatic 3D Model Building", *Computer Vision and Image Understanding*, Vol. 87, pp. 8–26, 2002.

20. Sappa, A. D., A. R. Specht and M. Devy, "*Range Image Registration by using an Edge-Based Representation*", *Proc. Int. Symp. Intelligent Robotic Systems*, pp. 167–176, 2001.

21. Jiang, X. and H. Bunke, "*Edge Detection in Range Images Based on Scan Line Approximation*", *Computer Vision and Image Understanding*, Vol. 73, pp. 183–199, 1999.

22. Bergevin, R., M. Soucy, H. Gagnon and D. Laurendeau, "*Towards a General Multi-View Registration Technique*", *IEEE Trans. on PAMI*, Vol. 18, pp. 540–547, 1996.

23. Shum, H. Y., M. Hebert, K. Ikeuchi and R. Reddy, "*An Integral Approach to Free-Form Object Modeling*", *IEEE Trans. on PAMI*, Vol. 19, pp. 1366–1370, 1997.

24. Pulli, K., "*Multiview registration for large data sets*", *In Proc. 2nd Int. Conf. on 3D Digital Imaging and Modeling*, pp. 160–168, 1999.

25. Williams, J. and M. Bennamoun, "*Simultaneous Registration of Multiple Corresponding Point Sets*", *IEEE Trans. on PAMI*, Vol. PAMI-9, pp. 698–700, 1987.

26. Castellani, U., A. Fusiello and V. Murino, "*Registration of Multiple Acoustic Range Views for Underwater Scene Reconstruction*", *Computer Vision and Image Understanding*, Vol. 87, pp. 78–89, 2002.

27. Masuda, T., "*Registration and Integration of Multiple Range Images by Matching Signed Distance Fields for Object Shape Modeling*", *Computer Vision and Image Understanding*, Vol. 87, pp. 51–65, 2002.

28. Robertson, C. and R. B. Fisher, "*Parallel Evolutionary Registration of Range Data*", *Computer Vision and Image Understanding*, Vol. 87, pp. 39–50, 2002.

29. Chow, C. K., H. T. Tsui and T. Lee, "*Surface registration using a dynamic genetic algorithm*", *Pattern Recognition*, Vol. 37, pp. 105–117, 2004.

30. Luciano, S., O. R. P. Bellon and K. L. Boyer, "*Precision Range Image Registration Using a Robust Surface Interpenetration Measure and Enhanced Genetic Algorithms*", *IEEE Trans. on PAMI*, Vol. 27, pp. 762–776, 2005.

31. Silva, L., O. R. P. Bellon and K. L. Boyer, "*Multiview range image registration using the surface interpenetration measure*", *Image and Vision Computing*, pp. 1–12, 2006.

32. Besl, P. J. and C. J. Ramesh, "*Segmentation Through Variable-Order Surface Fitting*", *IEEE Trans. on PAMI*, Vol. 10, pp. 167–192, 1988.

33. Chang, I. S. and R. H. Park, "*Segmentation Based on Fusion of Range and Intensity Images Using Robust Trimmed Methods*", *Pattern Recognition*, Vol. 34, pp. 1951–1962, 2001.

34. Koster, K. and M. Spann, "*MIR: An Approach to Robust ClusteringApplication to Range Image Segmentation*", *IEEE Trans. on PAMI*, Vol. 22, pp. 430–444, 2000.

35. Wanga, G., Z. Houkesb, G. Jia, B. Zhenga and X. Lia, "*An Estimation-based Approach for Range Image Segmentation: On the Reliability of Primitive Extraction*", *Pattern Recognition*, Vol. 36, pp. 157–169, 2003.

36. Fan, T. J., G. Medioni and R. Nevatia, "*Segmented Descriptions of 3-D Surfaces*", *IEEE Journal of Robotics and Automotion*, Vol. RA-3, pp. 527–538, 1987.

37. Koh, J., M. Suk and S. M. Bhandarkar, "*A Multi-layer Kohonen's Self-Organizing Feature Map For Range Image Segmentation*", *Proc. Intl. Conf. on Neural Networks*, pp. 1270–1275, 1993.

38. Liu, X. and D. L. Wang, "*Range Image Segmentation Using a Relaxation Oscillator Network*", *IEEE Trans. Neural Networks*, Vol. 10, pp. 564–573, 1999.

39. Boyer, K. L., R. Srikantiah and P. J. Flynn, "*Saliency Sequential Surface Organization for Free-Form Object Recognition*", *Computer Vision and Image Understanding*, Vol. 88, pp. 152–188, 2002.

40. Wani, M. A. and B. G. Batchelor, "*Edge-Region-Based Segmentation of Range Images*", *IEEE Trans. on PAMI*, Vol. 16, pp. 314–319, 1994.

41. Al-Hujazi, E. and A. Sood, "*Range Image Segmentation with Applications to Robot Bin-Picking Using Vacuum Gripper*", *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 20, pp. 1313–1325, 1990.

42. Jianga, X., H. Bunke and U. Meierb, "*High-level Feature Based Range Image Segmentation*", *Image and Vision Computing*, Vol. 18, pp. 817–822, 2000.

43. Bellon, O. R. P., A. I. Direne and L. Silva, "*Edge Detection to Guide Range Image Segmentation by Clustering Techniques*", *IEEE Int. Conf. on Image Processing*, 1999.

44. Jiang, X., "*An Adaptive Contour Closure Algorithm and Its Experimental Evaluation*", *IEEE Trans. on PAMI*, Vol. 22, pp. 1252–1265, 2000.

45. Min, J. and K. W. Bowyer, "*Improved Range Image Segmentation by Analyzing Surface Fit Patterns*", *Computer Vision and Image Understanding*, Vol. 97, pp. 242–258, 2004.

46. Ding, Y., X. Ping, M. Hu and D. Wang, "*Range Image Segmentation Based on Randomized Hough Transform*", *Pattern Recognition Letters*, Vol. 26, pp. 2033–2041, 2005.

47. Sappa, A. D., "*Unsupervised Contour Closure Algorithm for Range Image Edge-Based Segmentation*", *IEEE Trans. on Image Processing*, Vol. 15, pp. 377–384, 2006.

48. Yokoya, N. and M. D. Levine, "*Range Image Segmentation Based on Differential*

Geometry: A Hybrid Approach", *IEEE Trans. on PAMI*, Vol. 11, pp. 643–649, 1989.

49. Ghosal, S. and R. Mehrotra, "*Segmentation of Range Images: An Orthogonal Moment-B ased Integrated Approach*", *IEEE Trans. on Robotics and Automotion*, Vol. 9, pp. 385–399, 1993.

50. Hoover, A., G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. B. Bunke, D. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon and R. B. Fisher, "*An Explerimental Comparison of Range Image Segmentation AIgorithms*", *IEEE Trans. on PAMI*, Vol. 18, pp. 673–689, 1996.

51. Chen, Y. and G. Medioni, "*Object Modeling by Registration of Multiple Range Images*", *Proceedings of the IEEE Conference on Robotics and Automotion*, 1991.

52. Masuda, T. and N. Yokoya, "*A Robust Method for Registration and Segmentation of Multiple Range Images*", *Computer Vision and Image Understanding*, Vol. 61, pp. 295–307, 1995.

53. Dorai, C., J. Weng and A. K. Jain, "*Optimal Registration of Object Views Using Range Data*", *IEEE Trans. on PAMI*, Vol. 19, pp. 1131–1138, 1997.

54. Sharp, G. C., S. W. Lee and D. K. Wehe, "*Invariant Features and the Registration of Rigid Bodies*", *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 932–937, 1999.

55. de Berg, M., M. van Kreveld, M. Overmars and O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, Springer-Verlag, 2. edn., 2000.

56. Bentley, L. J. and J. H. Friedman, "*Data Structures for Range Searching*", *Computing Surveys*, Vol. 11, pp. 397–409, 1979.

57. Sonka, M., V. Hlavac and R. Boyle, *Image Processing, Analysis and Machine Vision*, PWS Publications, 2. edn., 1999.

58. Marr, D. and E. C. Hildreth, "*Theory of Edge Detection*", *Proceedings of the Royal Society of London. Series B, Biological Sciences*, Vol. B-207, pp. 187–217, 1980.

59. Moller, T. and B. Trumbore, "*Fast, minimum storage ray-triangle intersection*", *Journal of Graphics Tools*, Vol. 2, pp. 21–28, 1997.