공학박사학위논문

3D object recognition using scale-invariant features

규모 불변 특징을 이용한 3차원 물체 인식

2017년 8월

서울대학교 대학원

기계항공공학부

임 정 훈

ABSTRACT

3D object recognition using scale-invariant features

Jeonghun Lim
School of Mechanical and Aerospace Engineering
The Graduate School
Seoul National University

As 3D scanning technology has developed, it has become easier to acquire various 3D surface data; thus, there is a growing need for 3D data registration and recognition technology. In particular, techniques for finding the exact positions of 3D objects in a cluttered scene in which many parts of an object are occluded and multiple objects may be present is an important technology required by various fields such as industrial inspections, medical imaging, and games.

Many existing studies have used local descriptors with local surface patches, and most of these use a fixed support radius so they cannot cope perfectly when the model and scene are at different scales. In this paper, we propose a new object recognition algorithm that exceeds the performance of existing studies. The process of 3D object recognition in a cluttered scene is largely composed of three steps: feature selection, feature description, and matching.

In this study, we propose a perfectly scale-invariant feature selection algorithm by extending the 2D SIFT algorithm to a 3D mesh. The feature selection method proposed in this study can obtain highly repeatable feature points and support radii regardless of the scale. The selected features can effectively describe local information using the new shape descriptor proposed in this study. Unlike existing shape descriptors, it is possible to perform scale-invariant 3D object recognition and achieve a high recognition rate when combined with the feature point selection algorithm proposed in this study using the gradients of the scalar functions as defined on the 3D surface. We also reduced the searching space and lowered the false positive rate by suggesting a new RANSAC-based transformation hypothesis generation algorithm.

Our 3D object recognition algorithm achieves recognition rates of 99.5% and 97.8% when tested on U3OR and CFVD datasets, respectively, which exceeds the results of previous studies.

Keywords: 3D object recognition, scale-invariant feature, scale-invariant recognition, 3D feature descriptor, RANSAC matching

Student Number: 2010-23229

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1.
INTRODUCTION

1.1 Background

Over the past several decades, several studies have examined feature detection and object recognition in 2D image areas. However, along with the recent development of various scanning technologies, the demand for 3D models in film, medical, gaming, and industrial fields has increased rapidly, and research on feature detection, registration, and shape retrieval for 3D models is also increasing. In particular, 3D object recognition technology that detects the exact position and size of a model in a cluttered scene is one of the most challenging and actively studied topics. For example, after scanning the inside of a factory, you can build an automated system by locating specific parts accurately. In addition, a range-based SLAM system can use a 3D object recognition algorithm to achieve more robust robot localization. Finally, non-rigid fitting can be used to recognize body movements in next-generation human–computer interaction systems.

Many existing studies of 3D object recognition have used local feature-based methods because global feature-based methods are not suited to complex cluttered scenes in which only a part of the object in question appears, because the shape details cannot be grasped and only the overall shape is considered. A typical local feature-based object recognition algorithm consists of three steps: feature selection,

feature description, and surface matching.

The feature selection process is the most basic part of the overall process, and the recognition rate may vary depending on the feature detector's performance. The simplest feature selection methods are surface sparse sampling and mesh decimation [1, 2]. However, these methods have low repeatability and are unstable. In addition, many existing studies cannot be applied to applications that match two objects of different scales because they only search for features on a fixed scale. In this paper, we propose a complete scale-invariant feature selection algorithm that can even be matched when models and scenes have different and unknown scales. This study's feature selection method was derived from the 2D SIFT algorithm [3]. Many previous studies have been inspired by the 2D SIFT algorithm, but the scale invariance properties of the algorithm are not typically achieved in the process of extending the algorithm to 3D. However, this study makes a contribution in that it completely inherits the scale-invariant property in three dimensions. In addition, highly repeatable feature points and the support radius can be extracted regardless of the mesh scale, so it can be combined with many existing local surface descriptors.

We also propose a new feature descriptor using the gradient of scalar function as defined on the 3D surface. The proposed feature selection algorithm uses a scalar function defined on the 3D surface. Therefore, the most efficient way to describe the selected feature is to use the surrounding scalar function. In this respect, our proposed feature descriptor can achieve the best performance when combined with the proposed feature selection algorithm.

Another contribution is that it improves the existing Random Sample Consensus (RANSAC) algorithm in the matching phase, thereby reducing false positives. Only rigid transforms can be considered in existing fixed-scale applications; however, in situations where the scale differs between the model and the scene, the searching space becomes very large because rigid transforms cannot express accurate transformations. This paper proposes a RANSAC-based transform hypothesis generation algorithm that takes account of similarity transformations, thereby effectively reducing the searching space and increasing the recognition rate.

The 3D object recognition system used in this study follows the most general scheme [4-9]. We show the block diagram of the 3D object recognition system in Fig. 1.1. In this system, recognition tests were conducted using the UWA 3D Object Recognition Dataset (U3OR) [2] and Ca' Fascari Venezia Dataset (CFVD) [10].

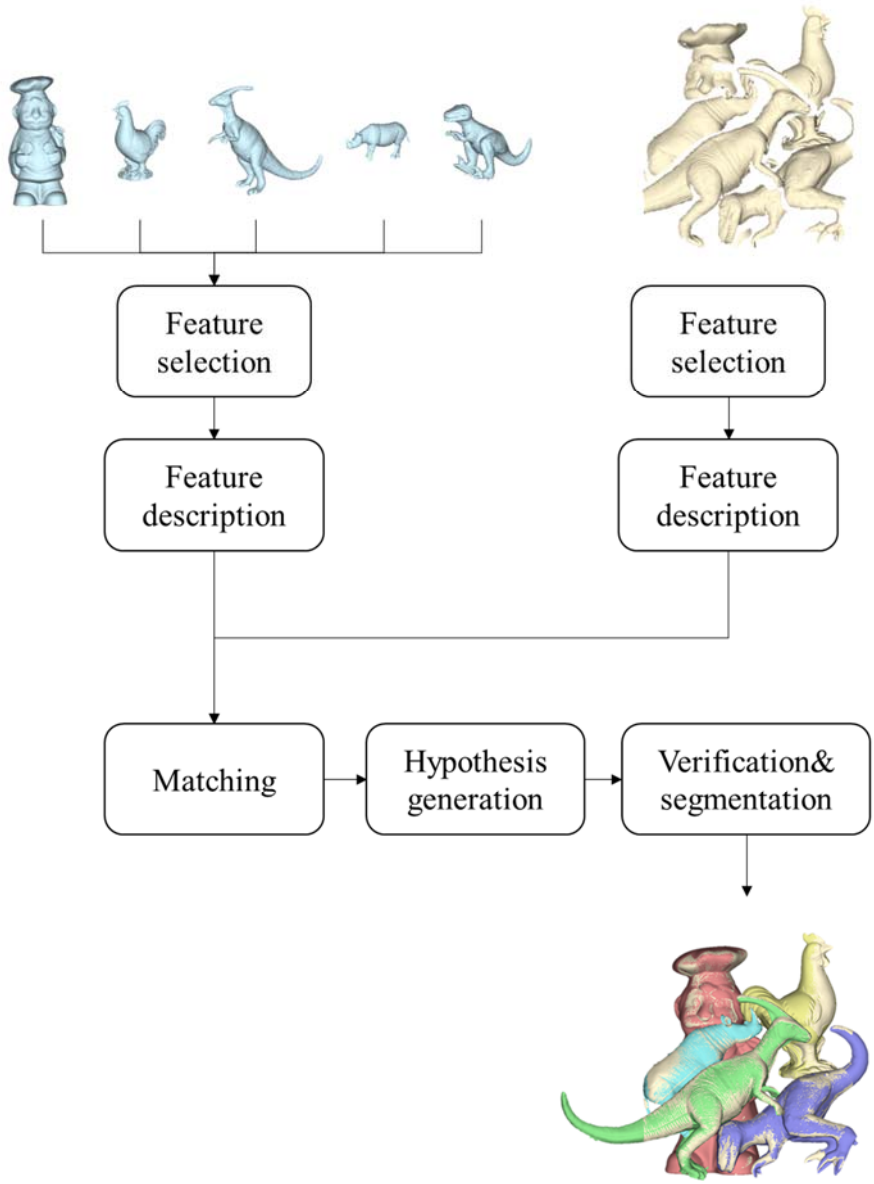


Fig. 1.1 Block diagram of 3D object recognition system.

CHAPTER 2.
RELATED WORKS

In this section, we review previous works on each step of feature selection, description, and matching in 3D feature recognition systems.

2.1 Feature selection

As mentioned above, the feature selection stage was the first and most basic part of the 3D object recognition system. The existing feature selection methods can be divided into two groups: the fixed-scale method and the adaptive-scale method.

2.1.1 Fixed-scale methods

Mokhtarian et al. [11] used surface curvature values to detect features. If the curvature value of an arbitrary mesh vertex was larger than that of neighboring vertices, the corresponding vertex was selected as a feature.

Yamany and Farag [12] introduced the concept of the simplex angle, which corresponds to the mean curvature of the surface. In this study, when the absolute value obtained by taking the sine function at the simplex angle of the vertex is larger than a certain threshold value, it is selected as the feature.

Gal and Cohen-Or [13] used the concept of the saliency grade, which was calculated as a linear combination of the curvature sum and the variance of neighboring vertices.

Likewise, if the saliency grade was larger than the threshold value that was defined by the user, the vertex is selected as the feature point. In addition, methods using surface variation measures other than the surface curvature have been studied.

Matei et al. [14] used the neighboring vertices to construct a covariance matrix and then calculated the smallest eigenvalue of the matrix to measure the surface variation. Zhong [15] used the ratio between the eigenvalues. Only vertices with $\lambda_1/\ \lambda_2$ and $\lambda_1/\ \lambda_3$ greater than a certain threshold were selected as features. Guo et al. [6] used a similar method; after decimating the given mesh, features were selected using $\lambda_1/\ \lambda_2$.

Sipiran and Bustos [16] proposed a "Harris 3D" detector that extended the Harris detector [17] used in a 2D image to a 3D mesh. The methods described thus far have only chosen features on a fixed scale. Therefore, their implementation is simple and fast, but there are disadvantages in that almost no features can be selected in the planar region and all selected features have a fixed support radius.

2.1.2 Adaptive-scale methods

Adaptive scale feature selection methods have also been studied. Most adaptive scale feature selection methods use Gaussian smoothing as in a 2D SIFT algorithm [3] to create a Difference of Gaussian (DoG) pyramid and detect features in the DoG scale space.

Castellani et al. [18] detected features by downsampling the mesh by the number of octaves and then creating a DoG pyramid for each octave.

Darom and Keller [19] used a density-invariant Gaussian filter, which is more robust to changes in mesh resolution. However, these methods used the 3D coordinates of the vertex directly when Gaussian smoothing was applied. Therefore, the geometry of the mesh changes, which breaks the characteristics of the DoG function that can detect stable features in a scale space. Studies have been conducted on smoothing the geometric attributes such as surface curvature instead of directly changing the mesh's geometry to avoid this phenomenon.

Bariya et al. [20], Novatnack and Nishino [21] used a mesh parameterization technique to obtain a 2D normal map image, and then detected features by applying a geodesic Gaussian kernel. This method is robust to mesh resolution changes, but it has the disadvantage of requiring a complicated parameterization technique.

Zaharescu et al. [22] proposed a MeshDoG algorithm that computes the DoG function for arbitrary scalar functions defined on the mesh. However, since the Gaussian kernel used at this time only considers the one-ring neighbor, it cannot be said that a proper DoG function is obtained. This paper proposes a feature selection method that takes advantage of the features of the stable DoG function in the scale space while minimizing the disadvantages of the methods mentioned above.

2.2 Feature description

After selecting the features on the surface, the geometry information of the local surface around the feature can be described in various ways. The existing feature description methods can be divided into signature-based methods and histogram-based methods depending on how they encode the surrounding geometry information.

2.2.1 Signature-based methods

Signature-based methods describe the local surface by encoding one or more geometric measurements that are individually calculated at vertices around a feature. Chua and Jarvis [23] proposed a descriptor called Point Signature. They first obtained a contour on a surface by intersecting a sphere with a support radius r around the feature point. They then calculated the plane that approximates the contour and projected each vertex of the contour onto the plane. The projected vertices could be expressed as the distance from the feature point and the rotation angle from the direction, which was defined as the unit vector from the feature point to the projected vertex that has the largest distance. The Point Signature was expressed by a discrete set of distance and angle values and was robust to noise but sensitive to varying the mesh resolution.

Malassiotis and Strintzis [24] proposed a Snapshot descriptor. They first constructed a Local Reference Frame (LRF) via an eigenvalue decomposition method, and then

they projected the local surface vertices onto the image plane of the virtual camera that was aligned to the LRF.

Castellani et al. [18] proposed a Hidden Markov Model (HMM) descriptor. They first built a clockwise spiral pathway around the feature point and extracted information such as the saliency level, curvature, and normal deviation along the pathway. Then, they used a discrete time Hidden Markov Model to encode this information. The HMM descriptor was robust to non-uniform sampling and varying mesh resolution.

do Nascimento et al. [25] proposed a Binary Robust Appearance and Normal (BRAND) descriptor; they extracted a local patch around the feature point from an RGB-D image and aligned that local patch with a dominant direction. The BRAND descriptor used the intensity variations and surface normal displacements.

2.2.2 Histogram-based method

Histogram-based methods describe the local neighborhood of a feature point by using histograms of geometric or topological measurements. In [7], existing algorithms was classified "spatial distribution histograms" and "geometric attribute histograms."

Spatial distribution histogram-based descriptors generate histograms using the coordinates of the surface vertices. They also use LRF or Local Reference Axis (LRA) to gain invariant rotation and translation properties.

Johnson and Hebert [1] proposed the Spin Image (SI) descriptor. The surface normal

of feature point was used as the LRA. The vertices on the local surface are represented by two parameters, the horizontal distance from the feature point and the vertical distance from the LRA. These two parameter spaces were discretized into a 2D array, and the SI descriptor was generated by accumulating the local surface vertices into the 2D array.

Frome et al. [26] proposed a 3D Shape Context (3DSC) descriptor. The support region was divided into several bins along the radial, azimuth, and elevation dimensions and the 3DSC descriptor was generated by counting the weighted number of vertices falling into each bin of the 3D grid. Unique Shape Context (USC) [27] is an extension of 3DSC that avoids gaining multiple descriptors at a feature point.

Guo et al. [6] proposed a Rotational Projection Statistics (RoPS) descriptor. First, they constructed a unique LRF and rotated the local surface around the three axes of the LRF. For each rotation, the vertices of the surface were projected onto the three planes of the LRF and three distribution matrixes were generated by dividing each plane into several regions and counting the number of vertices that fell into each. Each distribution matrix was encoded with five statistics, and the RoPS descriptor was generated by concatenating all of these statistics.

The Tri Spin Image (TriSI) [8, 28] is similar to the RoPS. After constructing the LRF, three SI were generated using three axes of the LRF. The TriSI was generated by concatenating the three SI.

Geometric attribute histogram-based descriptors generate histograms using

geometric attribute information such as the normal or curvature rather than coordinate information.

Chen and Bhanu [29, 30] proposed a Local Surface Patch (LSP) descriptor that is a 2D histogram formed by accumulating vertices along two dimensions. One dimension is the shape index [31] and the other is the cosine of the angle between the surface normals.

Point Feature Histogram (PFH) [32] is a multi-dimensional histogram over several features of vertex pairs in the support region. PFH is generated by accumulating vertices in specific bins along the four dimensions. Four features are calculated for each pair of vertices in the local surface using the Darboux frame, the normal vectors, and positions. Fast Point Feature Histogram (FPFH) [33], is an improvement on PFH in which features are generated for each local surface vertex by calculating the relationships between that vertex and its neighbors.

Salti et al. [34], Tombari et al. [35] proposed the Signature of Histogram of Orientations (SHOT) descriptor. LFR was constructed from local surface vertices and the support region was divided into several regions along the azimuth, elevation, and radial axes. The SHOT descriptor was generated by concatenating all of the local histograms computed in each region.

In this study, we proposed a new descriptor that uses the histogram generated from gradient of the scalar function as defined on a 3D surface. Many previous studies have considered feature selection and feature description methods separately, some studies have focused on feature selection methods, and some studies have only

focused on feature description methods. The feature descriptor proposed in this study improves the efficiency and performance of the 3D object recognition algorithm using the same scalar function as used in the feature selection method that is proposed in this study.

2.3 Surface matching

The surface matching of most existing studies can be divided into three stages of feature matching, hypothesis generation and verification. The Nearest Neighbor Distance Ratio (NNDR) is the most frequently used in feature matching [36] and the Iterative Closest Point (ICP) algorithm [37] is the most commonly used trend in the verification phase, which is the process of determining whether the hypotheses obtained through the hypothesis generation process are true. Unlike the other two, relatively different techniques have been proposed for the hypothesis generation stage. Rodolà et al. [10] used the Game Theory technique in which all correspondences obtained at the feature matching stage start competing in a non-cooperative game. As competition continues, only reliable correspondences survive and these are used to calculate transform hypotheses. Guo et al. [6, 8] used a pose clustering method that began with the assumption that similar hypothesis transformations will form a group in the transformation space near the ground truth. However, this method has the drawback that it can only be used when the scales of the model and the scene are the same. Taati and Greenspan [5] and Papazov et al. [38, 39] used a RANSAC-based method. Recognition in a cluttered scene has many

outliers. Therefore, the RANSAC algorithm has a suitably good outlier removal performance for this, and the algorithm is simple and easy to implement. The RANSAC algorithm first arbitrarily selects a minimum set of correspondences to calculate a rigid transformation that can align a model to a scene, and then computes the number of correspondences that match this transformation. Finally, the transformation that involves the largest number of inliers is considered the hypothesis. This paper proposes a new RANSAC-based algorithm that considers both rigid transformation and similarity transformation to cope with an unknown scale situation. In addition, in selecting the inliers, the angular difference between the vertex normal of the model and the scene feature is taken into consideration to obtain a more stable result.

CHAPTER 3.
Datasets

Before describing the 3D object recognition algorithm, this Chapter describes the two datasets that are used in this study. As mentioned earlier, this study uses the U3OR and the CFVD datasets.

3.1 U3OR dataset

The U3OR dataset was first introduced in [2] and has been the most widely used dataset ever since [5, 6, 8-10, 20, 40, 41]. This dataset consists of five models and 50 real scenes. Each scene was scanned using a Minolta Vivid910 scanner with four or five models randomly positioned. There were 217 objects included in the 50 scenes in total. Excluding the Rhino model, that number is 188. The reason the Rhino model may be excluded is that spin image-based methods have failed to recognize it in all 50 scenes because the model includes a large hole [6, 40]. Fig. 3.1 shows five models and nine of the 50 scenes.

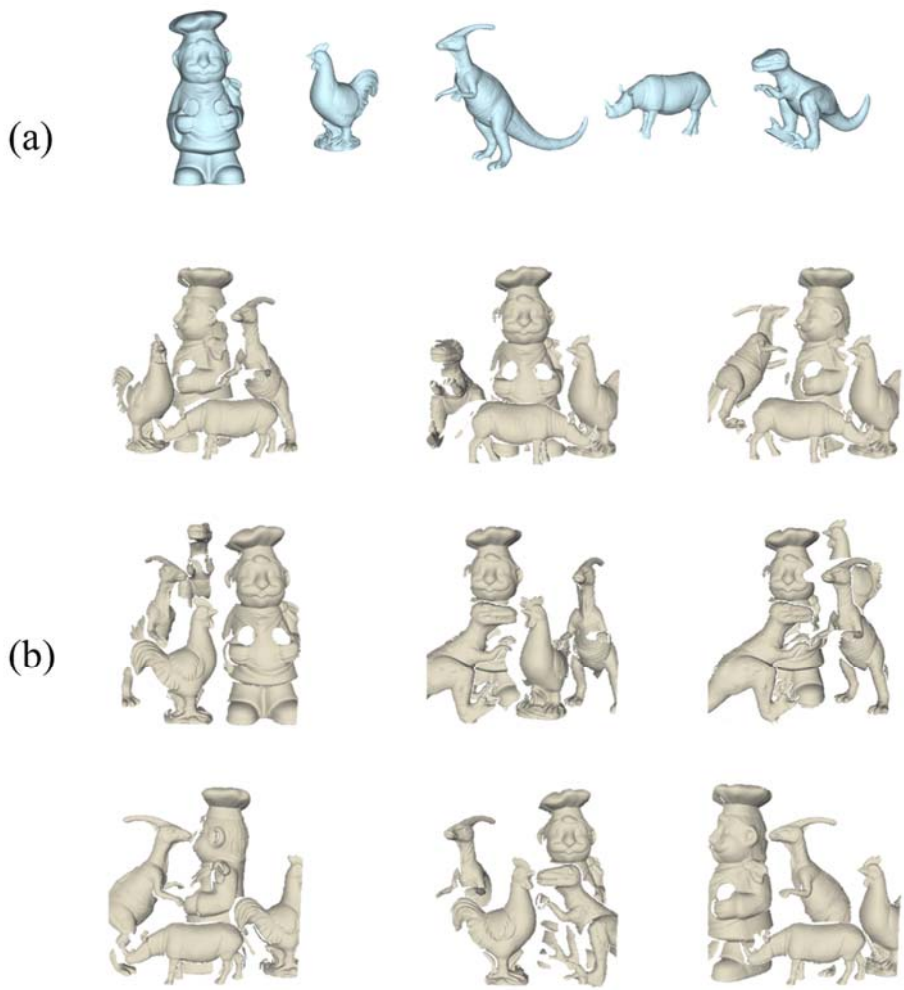


Fig. 3.1 (a) Five models (from the left Chef, Chicken, Para, Rhino and T-rex) and (b) nine of the 50 scenes in the U3OR dataset.

3.2 CFVD dataset

The CFVD dataset was first introduced in [10]; it contains 20 models and 150 scenes, each of which was captured using a virtual camera and contains three to five models. There are 497 objects in all scenes. The size of the dataset is the largest available 3D object recognition dataset. Existing studies [6, 8-10] have excluded two models in this dataset to produce additional clutter; we show nine of the 150 scenes in Fig. 3.2 and all 20 models in Fig. 3.3.

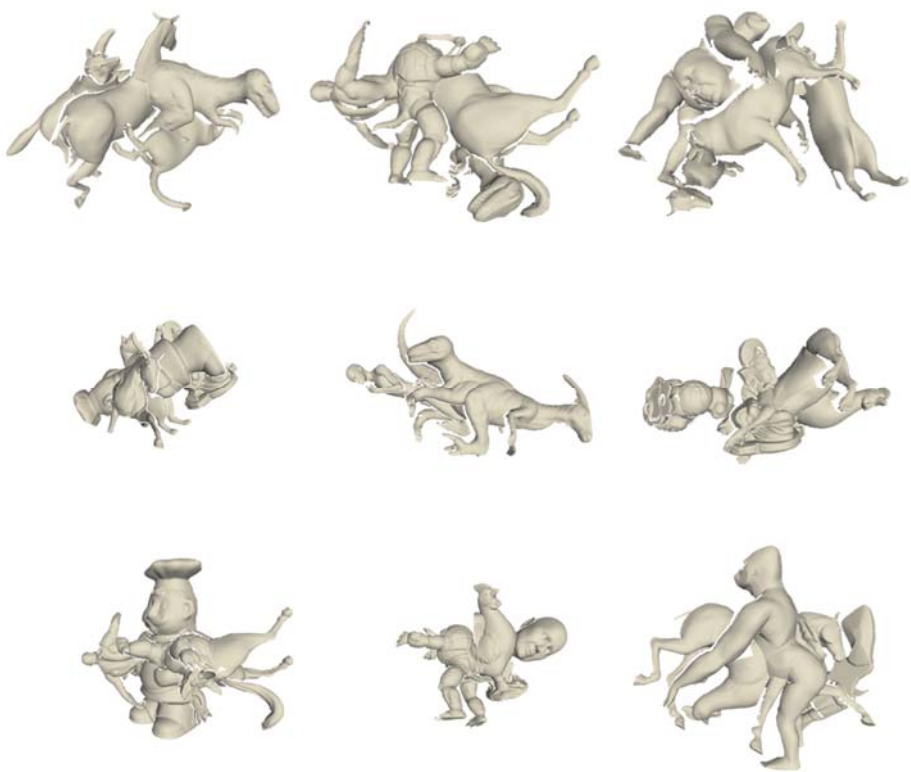


Fig. 3.2 Nine examples of scenes in the CFVD dataset.

Fig. 3.3 All models of the CFVD dataset. The final two models were excluded from the recognition test.

CHAPTER 4.
FEATURE SELECTION

4.1 Concepts

As described above, the feature selection method in this paper is based on the idea of the existing 2D SIFT algorithm [3]. As mentioned in [42], the most stable and invariant features of the scale space are the extrema of the normalized Laplacian of Gaussian (LoG) function. The most effective method of approximating the LoG function is to use the DoG function. We define $\mathbb{M}$ as the set of all polygonal meshes that can be defined in $\mathbb{R}^3$. We will consider a special uniformly sampled triangular mesh $M \in \mathbb{M}$ because the surface curvature can be obtained more reliably in a uniformly sampled mesh. A uniform mesh can be obtained from a non-uniform mesh using the algorithm proposed in [43]. The triangular mesh M can also be expressed as a graph, $M(V, E)$ where $V = \{v_i \in \mathbb{R}^3 | i = 1, \dots, n_V\}$ is the set of mesh vertices and $E = \{e_{ij} | i, j = 1, \dots, n_V, i < j\}$ is the set of mesh edges that connect two adjacent vertices. The average edge length of M is defined as e_{avg}.

We consider the scalar function, $f: \mathbb{M} \to \mathbb{R}$, to define the discrete convolution operator on a mesh. The convolution of the function f with a Gaussian kernel g is defined as

$$f * g(v_i, \sigma) = \frac{\sum_{v_j \in N_r(v_i)} g(\|v_j - v_i\|, \sigma) f(v_i)}{\sum_{v_j \in N_r(v_i)} g(\|v_j - v_i\|, \sigma)},$$

$$g(x,\sigma) = \frac{\exp(-\frac{x^2}{2\sigma^2})}{\sigma\sqrt{2\pi}}.$$

σ is the standard deviation of the Gaussian kernel g. $N_r(v_i)$ is the set of vertices whose distance from v_i is within r. As r increases, the number of features decreases. Instead, features take more time to acquire because this requires more computation. Therefore, it is important to choose an appropriate value for r. Fig. 4.1 shows the number of features according to the change of r; this decreases sharply until r is $3\sigma e_{avg}$, and the decrease is not large thereafter. This paper uses $r = 3\sigma e_{avg}$.

Only one-ring neighbors have been considered in many existing studies. If only one-ring neighbors are considered, it cannot be said that a proper Gaussian kernel has been applied. Additionally, stable and scale-invariant features cannot be obtained.

The DoG function $D(v_i,\sigma)$, which represents the difference between two Gaussian functions with a constant multiple filter scale, can be defined as

$$D(v_i,\sigma) = f * g(v_i,k\sigma) - f * g(v_i,\sigma).$$

In this study, the scalar function f was defined using the surface mean curvature.

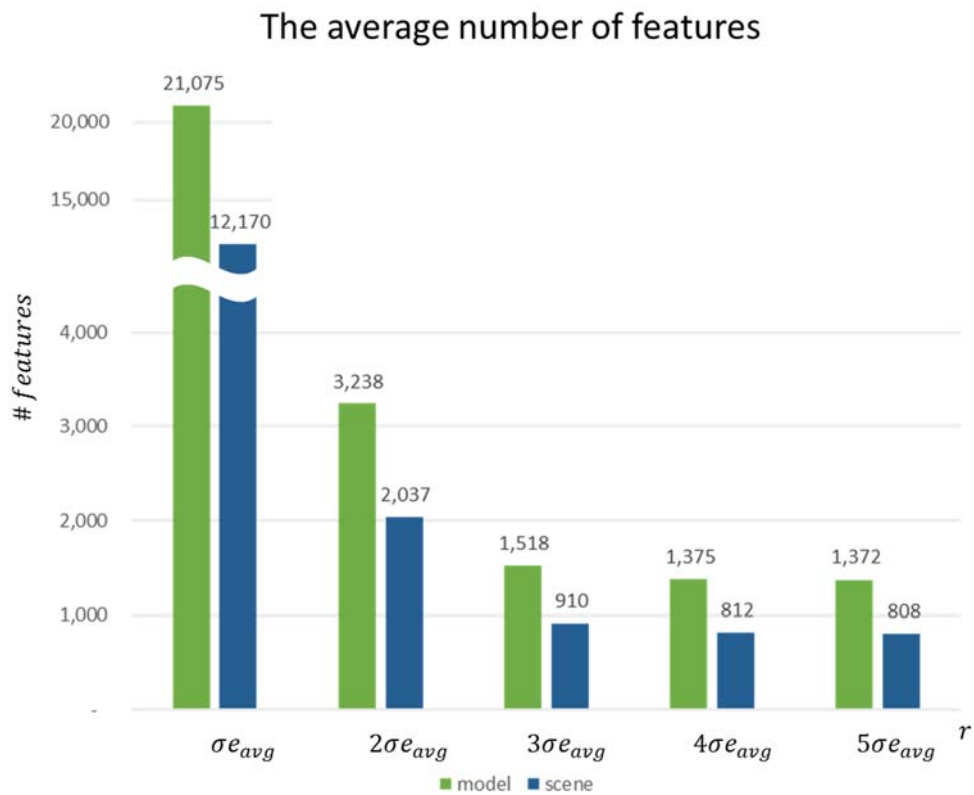


Fig. 4.1 The number of features according to r in the U3OR dataset.

4.2 Gaussian and DoG pyramid

The first step in selecting features is creating a Gaussian pyramid and a DoG pyramid. If the number of intervals for each octave is n_I, then $n_I + 3$ Gaussian functions should be created. The first Gaussian function f_1 is constructed by convolving the initial scalar function f_0 with a Gaussian kernel of initial sigma σ_0. In this paper, f_0 is the mean curvature of the triangular mesh M_0 at the start.

$$f_1 = f_0 * g(\sigma_0).$$

The second Gaussian function f_2 can be obtained by convolving the initial scalar function f_0 with the Gaussian kernel of $k\sigma_0$, but this calculation ($f_2 = f_0 * g(k\sigma_0)$) is inefficient, so we use the previous Gaussian function as follows.

$$g(k\sigma_0) = g(\sigma_0) * g\left(\sqrt{k^2 - 1}\ \sigma_0\right),$$

$$f_2 = f_0 * g(k\sigma_0) = f_0 * g(\sigma_0) * g\left(\sqrt{k^2 - 1}\ \sigma_0\right),$$

$$f_2 = f_1 * g(\sqrt{k^2 - 1}\ \sigma_0).$$

In this manner, an arbitrary i-th Gaussian function f_i can be calculated as follows; in this paper, $k = 2^{1/n_I}, \sigma_0 = \sqrt{2}e_{avg}$.

$$f_i = f_{i-1} * g(\sigma_i\ ),$$

$$\sigma_i = k^{i-2}\sqrt{k^2 - 1}\ \sigma_0, \qquad i = 1, \dots, n_I + 3.$$

Once all $N_I + 3$ Gaussian functions have been computed, a new mesh M_1 is created such that the average edge length of M_1 is twice that of M_0. This process corresponds to downsampling in the 2D SIFT algorithm; most previous studies

omitted the downsampling operation. If the number of octaves is n_O, then we need to create $n_O - 1$ downsampled meshes $M_1, M_2, \ldots, M_{n_O-1}$. The first Gaussian function f_1 of the second octave can be obtained by linearly sampling f_{n_I+1} from the previous octave. The reason for using f_{n_I+1} is that the σ of f_{n_I+1} is twice that of σ_0. Afterwards, the above steps are repeated for all octaves. Fig. 4.2 shows examples of downsampling.

After obtaining $(n_I + 3) \times n_O$ Gaussian functions in all octaves, we can obtain $(n_I + 2) \times n_O$ DoG functions by calculating the difference between pairs of adjacent Gaussian functions. Example Gaussian and DoG pyramids can be observed in Fig. 4.3.

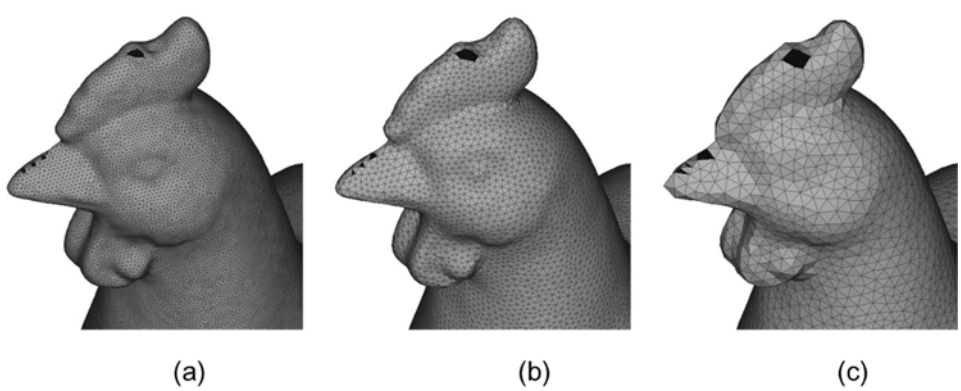


Fig. 4.2 Downsampling examples (a) M_0, (b) M_1, and (c) M_2.

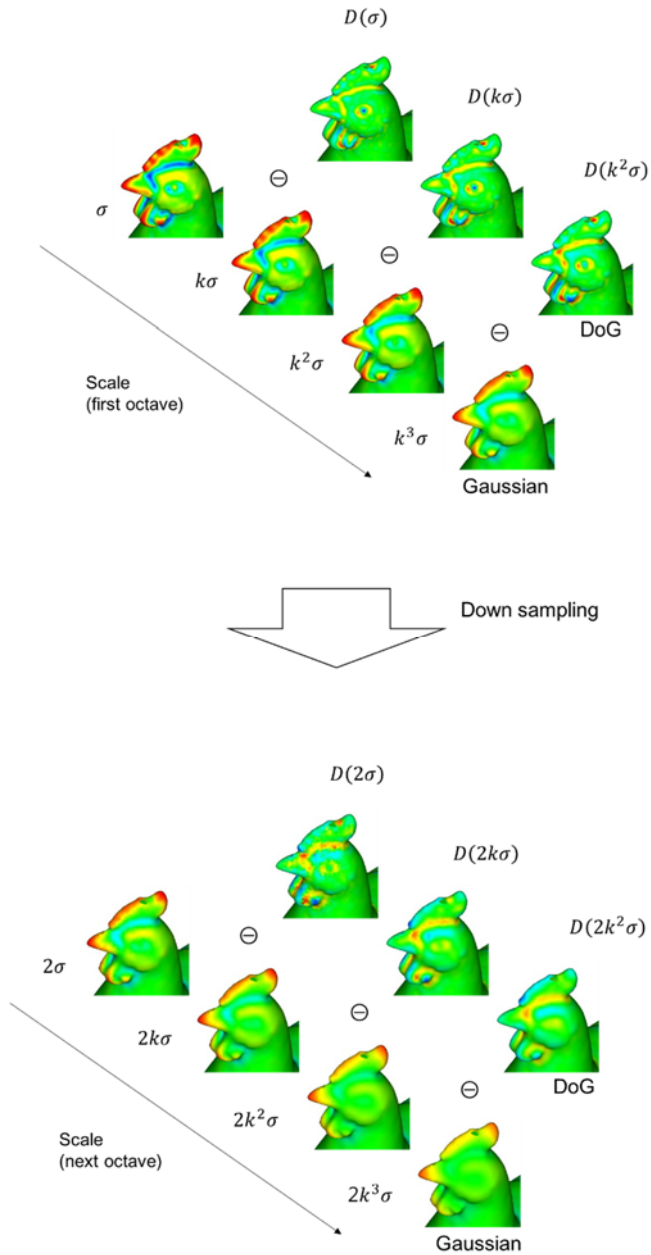


Fig. 4.3 An example of processing. The initial scalar function for each octave is repeatedly convolved with Gaussian kernels and adjacent Gaussian functions are subtracted to produce the DoG functions. This process is repeated with downsampled Gaussian functions in the next octave.

4.3 Local Extrema Detection

The next step is to extract the local extreme vertices from the DoG functions of each octave. Determining whether $D(v_i, \sigma)$ is extreme requires comparing the previous and next scales and the neighboring vertices of v_i on the current scale. If the number of neighboring vertices of v_i is n_{v_i}, we need to compare $3n_{v_i} + 2$ vertices in total (Fig. 4.4). All extracted local extrema are selected as features. If $D(v_i, \sigma)$ is selected as a feature, then the location of the feature will be the coordinate of v_i and the support radius will be determined by multiplying σ by a constant c. The larger the value of c, the greater the feature's descriptiveness, but the greater the sensitivity to occlusion and clutter. The effects of parameter c were investigated by conducting experiments using the datasets that were used in this study. Fig. 4.5 shows how the precision and recall changed when the value of c was varied from 2 to 10. The precision and recall values are calculated as follows:

$$precision = \frac{true\ \ positive}{total\ \ positive}, \qquad recall = \frac{true\ \ positive}{real\ \ positive}.$$

When the value of c was less than 4, both the precision and recall decrease sharply. Although the rate of change was relatively small when the value of c was smaller than 4, if the value was too large then the performance was rather deteriorated. Therefore, in all experiments in this study, we used 4 for the value of c. We show examples of the feature selection results in Fig. 4.6.

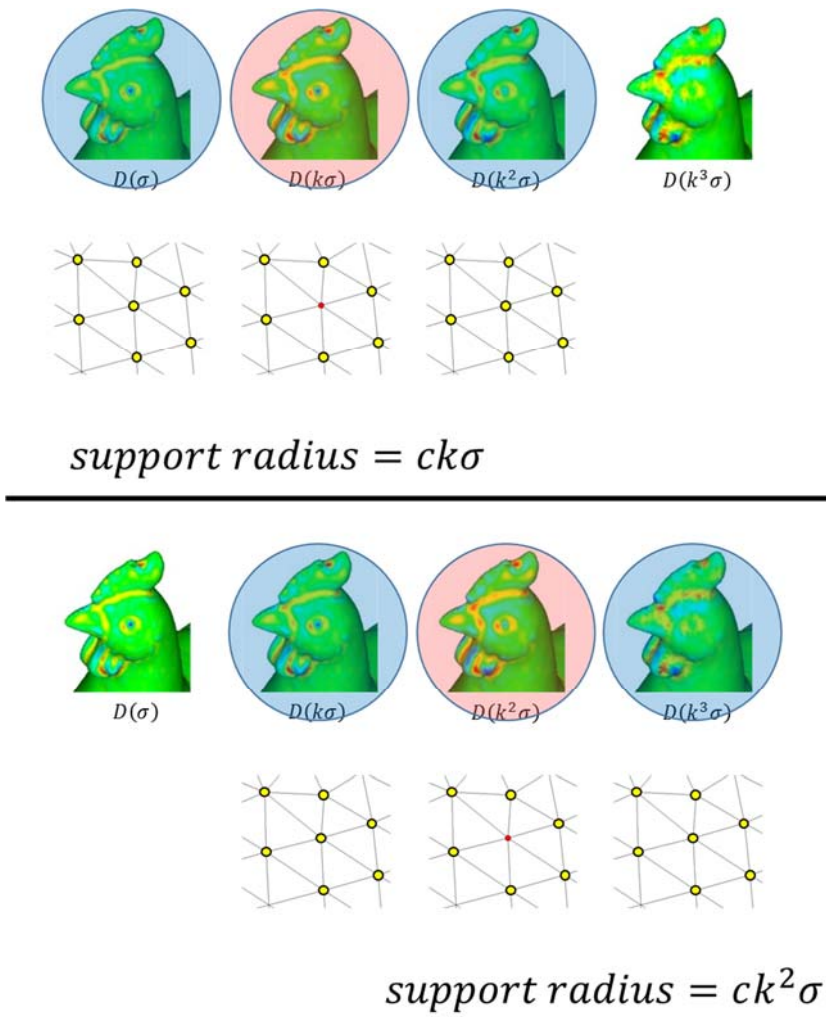
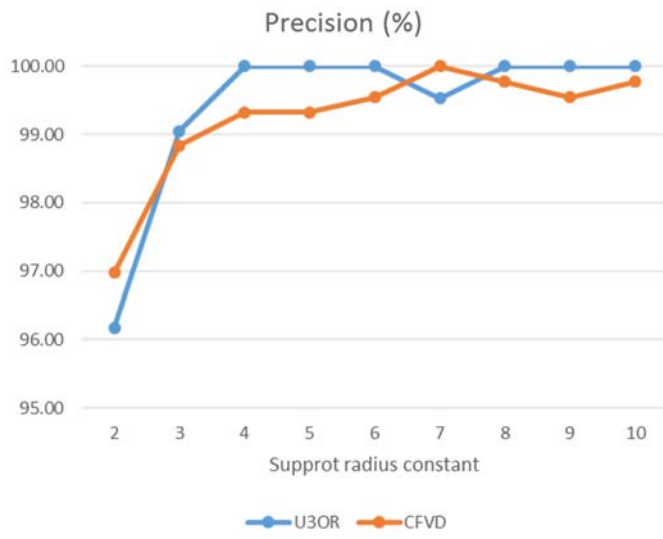


Fig. 4.4 Local extrema detection in the scale space.



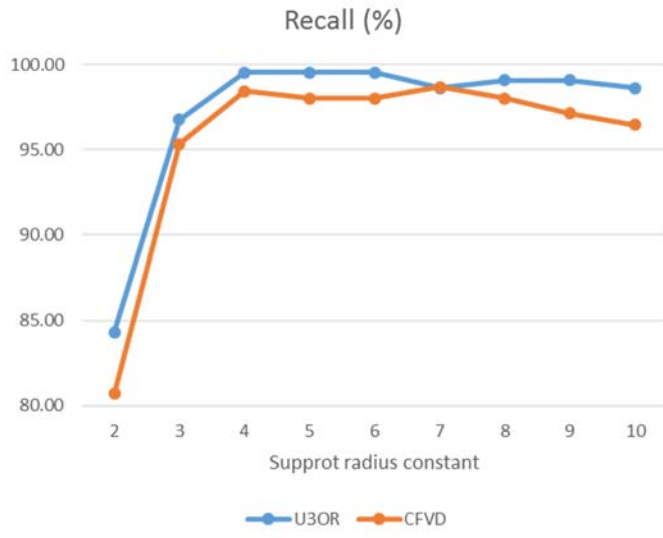


Fig. 4.5 Change of precision and recall value according to changes in parameter c.

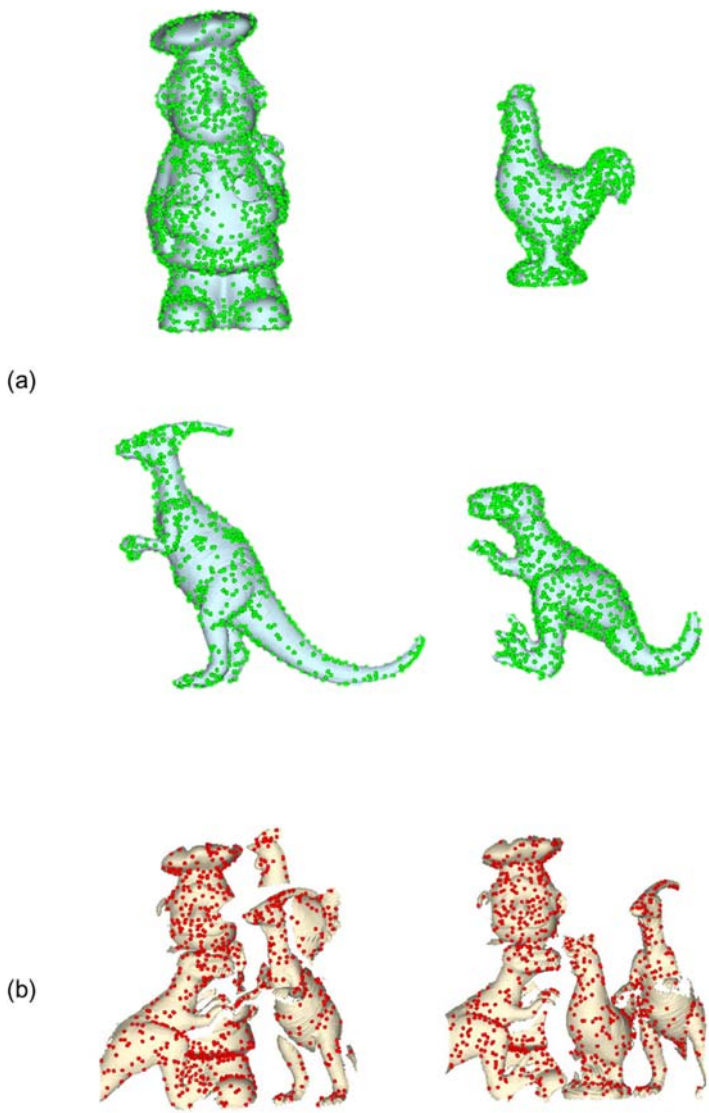


Fig. 4.6 Selected features from (a) the models and (b) the scenes in the U3OR dataset.

CHAPTER 5.
Feature description

In this paper, we propose a new descriptor that uses the gradient of a scalar function defined on a 3D surface. Scalar functions are used in the same manner in the feature selection stage, so the scalar functions best represent the selected features. If the scalar function represents geometric information such as the curvature, then an existing descriptor generated with geometry can be used. However, if the scalar function does not represent the geometric information as the object's texture information, then you cannot expect good results.

5.1 LRF construction

Ensuring that the selected features have rotational invariant characteristic requires calculating a unique and repeatable LRF/A. Most recent local descriptors use LRF/A; our LRF is based on the previously presented LRF in [6, 8, 44].

Given an arbitrary feature point p and a support radius r, a sphere with center p and radius r is called a support region. If there are n_F faces in this region, then any point q_i on the i-th face (Fig. 5.1) can be defined as:

$$q_i(\alpha, \beta) = v_{i1} + \alpha(v_{i2} - v_{i1}) + \beta(v_{i3} - v_{i1}),$$

where $0 \leq \alpha, \beta \leq 1$, and $\alpha + \beta \leq 1$. v_{i1}, v_{i2} and v_{i3} represent three of the face's vertices.

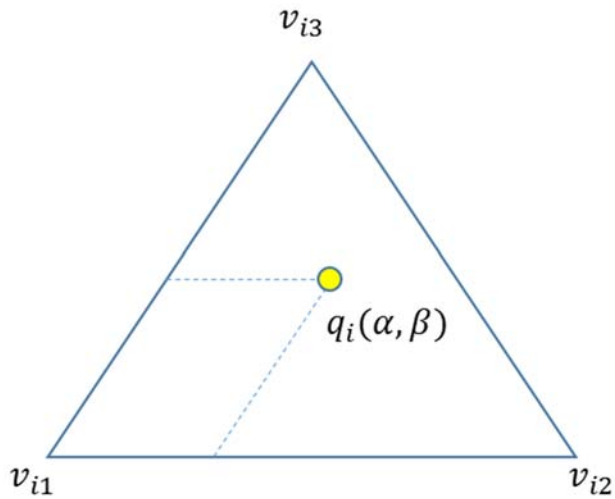


Fig. 5.1 The i-th face of support region.

The scatter matrix $\mathbf{C}_i$ of the i-th face can then be expressed as:

$$\mathbf{C}_i = \int_0^1 \int_0^{1-\beta} (q_i(\alpha,\beta) - p)(q_i(\alpha,\beta) - p)^T d\alpha d\beta$$

$$= \frac{1}{12} \sum_{m=1}^{3} \sum_{n=1}^{3} (v_{im} - p)(v_{in} - p)^T + \frac{1}{12} \sum_{m=1}^{3} (v_{im} - p)(v_{im} - p)^T.$$

The scatter matrix C for all n_{F} faces inside the support region is:

$$\mathbf{C} = \sum_{i=1}^{n_F} w_{i1} w_{i2} \mathbf{C}_i.$$

Here, w_{i1} is the ratio between the area of the i-th face and the total area of the n_F faces:

$$w_{i1} = \frac{\|(v_{i2} - v_{i1}) \times (v_{i3} - v_{i1})\|}{\sum_{j=1}^{n_F} \|(v_{j2} - v_{j1}) \times (v_{j3} - v_{j1})\|}.$$

w_{i2} is a weight that is related to the distance from the feature point to the centroid

of the i-th face, that is:

$$w_{i2} = \left(r - \left\| p - \frac{v_{i1} + v_{i2} + v_{i3}}{3} \right\|\right)^2.$$

Performing an eigenvalue decomposition on the overall scatter matrix $\mathbf{C}$ gives us the three eigenvalues $\lambda_1 > \lambda_2 > \lambda_3$ of the matrix $\mathbf{C}$, and the three corresponding orthogonal eigenvectors $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ which respectively denote the x, y, and z axes of the LRF.

The next step in [6] is to eliminate sign ambiguity among the eigenvectors. However, since our study only uses $\mathbf{e}_3$, we simplified this step using the normal vector of the feature point $\mathbf{n}_p$. The unambiguous vector $\tilde{\mathbf{e}}_3$ is defined as:

$$\tilde{\mathbf{e}}_3 = \mathbf{e}_3 \cdot \text{sign}(\mathbf{e}_3 \cdot \mathbf{n}_p).$$

We only use $\tilde{\mathbf{e}}_3$ because it has higher repeatability than the other two eigenvectors. The following tests were carried out to quantitatively identify this; first, features were extracted from all models and scenes of the U3OR dataset and all model features were transformed by ground truth transformation. If the distance from the moved model feature to the nearest scene feature was less than a certain threshold, the two features were paired. The threshold value was twice the e_{avg} of the model. We calculated the angular difference of each axis of LRF and the surface normal for all pairs.

Table 5.1 The average angular difference of each axis of LRF and surface normal in 4 models of U3OR dataset. (degree)

	Chef	Chicken	Para	Trex	AVG
Avg. axis x diff	7.18	8.90	7.73	7.65	7.87
Avg. axis y diff	7.26	9.12	8.04	7.87	8.07
Avg. axis z diff	**2.13**	**1.95**	**2.06**	**2.09**	**2.06**
Avg. normal diff	6.74	8.44	7.59	6.97	7.43

As shown in Table 5.1, the repeatability of the z-axis of LRF (i.e. $\tilde{\mathbf{e}}_3$) was the most stable. In this paper, a new orientation could be determined by replacing the other two eigenvectors with the gradient of a scalar function; we call this orientation "feature orientation."

5.2 Feature orientation assignment

As described in Chapter 2.2.1, the SHOT descriptor [34, 35] uses the entire spherical support region. However, the interior of the support region is mostly empty because the local surface inside the support region is generally a thin disk. Therefore, using the entire region is inefficient. Thus, we define the plane that best represents the local surface without using the entire space and define the descriptor on this plane.

Now, we can define a feature plane P that passes through feature point p and whose normal direction is $\tilde{\mathbf{e}}_3$. Assigning one or more feature orientation on the feature plane gives the proposed descriptor robust rotation-invariant properties; this requires first calculating the gradient of the scalar function. The gradient of scalar function at i-th face F_i can be defined as:

$$\nabla f(F_i) = -\frac{1}{2A}\ \mathbf{n}_{F_i} \times (\ f(v_{i1})\mathbf{e}_{23} + f(v_{i2})\mathbf{e}_{31} + f(v_{i3})\mathbf{e}_{12}\ ),$$

$$\mathbf{e}_{lm} = v_{im} - v_{il}, \quad l, m = 1,2,3.$$

We show example gradient vectors in Fig. 5.2.

The next step is to project the gradient vector of all faces in the local surface to P. An orientation histogram that has 36 bins that span 360° of the orientation is formed from the projected gradient vectors. Each gradient vector is weighted with its magnitude and distance to the feature point as follows.

$$w_i = \|\nabla f(F_i)\| \times g\left(\left\|p - \frac{v_{i1} + v_{i2} + v_{i3}}{3}\right\|, \sigma_i\right),$$

where σ_i is 0.5 times the feature's support radius.

The resulting histogram is smoothed three times with the Laplacian smoothing

technique for robustness. In the smoothed histogram, the angle to the highest peak determines the principal direction of local gradients. In addition, after the highest peak in the histogram has been detected, local peaks with heights above 80% of the highest peak are also selected. This means that there could be multiple features at the same location and support radius with different feature orientations.

The final step is to fit a quadratic function to the histogram using the neighboring bins of a peak to get the feature orientation at sub-bin precision. The equation is expressed as follows.

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \end{bmatrix}^{-1} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

$$\text{feature orientation} = -\frac{b}{2a}$$

Here, b and c are coefficients of the quadratic function, x_i $(i = 1,2,3)$ is the angles of the peak and the neighboring bins, and y_i $(i = 1,2,3)$ is the histogram height.

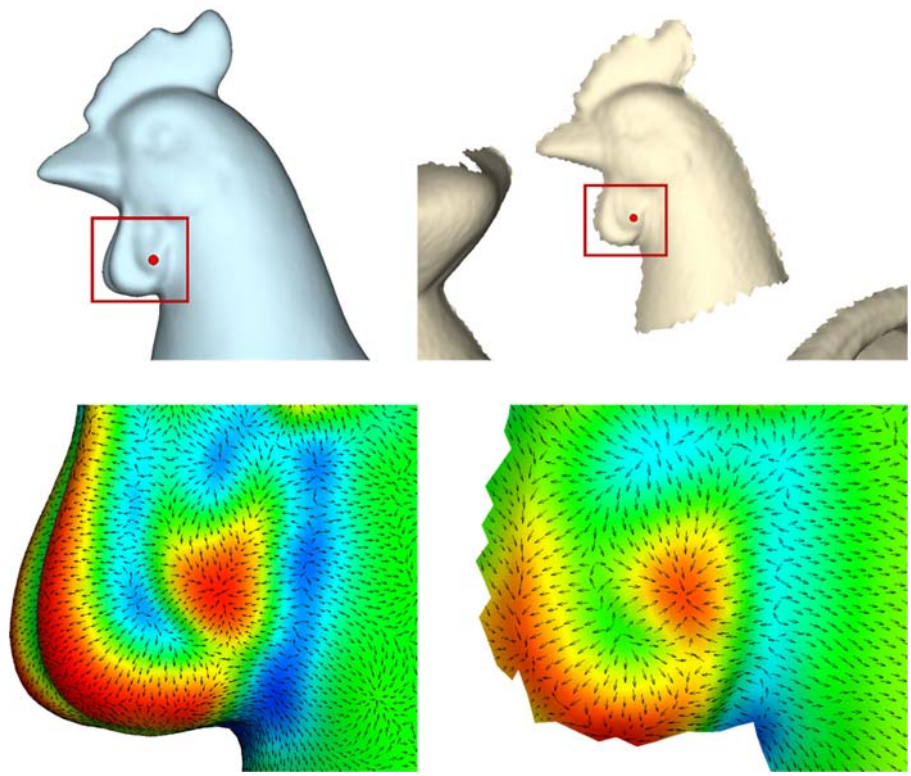


Fig. 5.2 Examples of scalar functions and gradient vectors around features.

5.3 Feature vector generation

Defining the feature orientation in Chapter 5.2 allows us to represent the gradients of the scalar function projected on the feature plane as relative angles to the feature orientation. This can be used to represent the entire local surface as a single histogram, but does not guarantee sufficient descriptiveness. In existing studies such as SIFT [3] and SHOT [34, 35], the support region is divided into several sections and the set of local histograms calculated in each section is used to enhance the descriptiveness.

In this study, the feature plane is divided into the azimuth and radial directions. The total number of sections is 16, which results from eight azimuth divisions and two radial divisions. The grid is arranged along the feature orientation to maintain its rotation-invariant property. We show the location and order of each section in Fig. 5.3.

Now, as when calculating the feature orientation, a local histogram is generated using the angular difference between the gradient of the scalar function and the feature orientation in each section. The local histogram has eight bins and the gradient vector is weighted with its magnitude and distance for the feature point.

As pointed out in [3, 34, 35], it is important to avoid boundary effects, because our descriptor is based on local histograms. The boundary effect means that small changes in feature orientation can have a large impact on the descriptor. Therefore, for each gradient vector accumulated into a specific local histogram bin, we perform trilinear interpolation with its neighbors. Therefore, each bin is incremented by a

weight of $1-d$ for each dimension. d is the current entry's distance from the central value of the bin. In the azimuth dimension, d is the angular distance, and d is the Euclidean distance in the radial dimension. In addition, d is normalized by the distance between two neighboring bins. Fig. 5.4 shows the trilinear interpolation process.

After all local histograms have been generated, they are normalized to have their Euclidean norm equal to 1 to ensure their robustness to point density variation. Finally, the local histograms are concatenated in a single feature vector, the length of which is $2 \times 8 \times 8 = 128$.

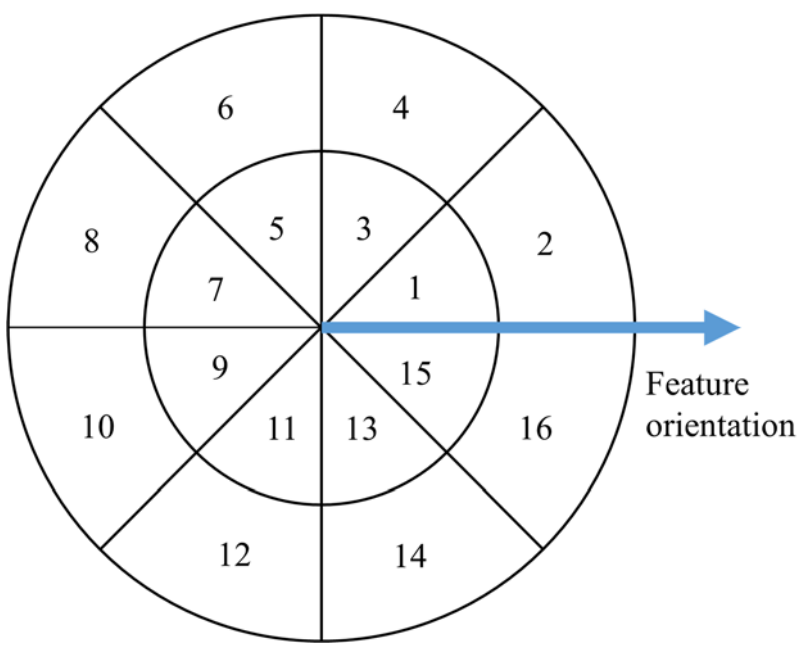
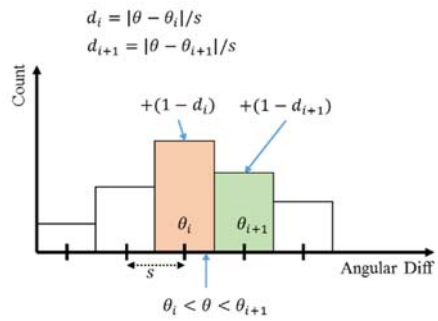
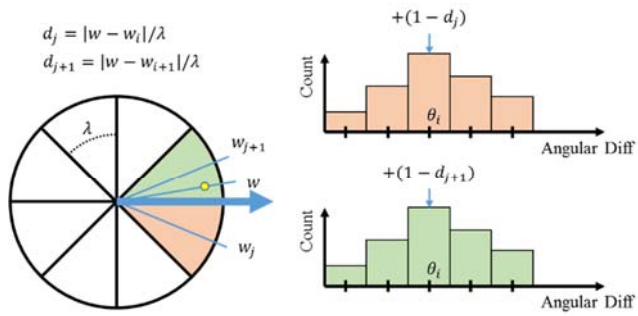


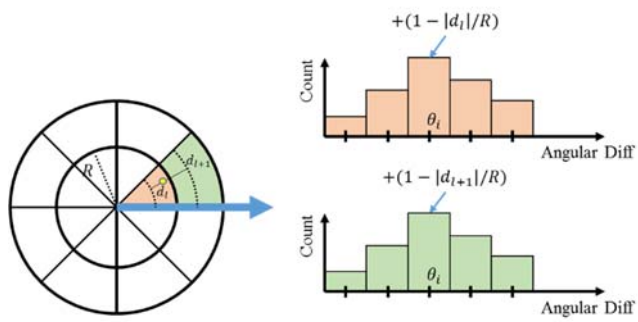
Fig. 5.3 The location and order of each section according to the feature orientation.



(a) Interpolation on angular difference



(b) Interpolation on azimuth dimension



(c) Interpolation on radial dimension

Fig. 5.4 The trilinear interpolation process.

CHAPTER 6.
3D object recognition

This chapter describes the process for achieving scale-invariant 3D object recognition in detail using the feature selection method and feature descriptor described above. The approximate procedure for the 3D object recognition system is as shown in Fig. 1.1.

6.1 Offline processing

First, we assume that we have several model meshes that we want to find in the scene, and that we need to construct a model library for these models.

Given a model, first we extract the feature points using the feature selection algorithm described in CHAPTER 4 for this model. Then, we generate feature vectors for each feature point using the feature descriptor described in CHAPTER 5. We repeat the above procedure for all models, then collect all the feature vectors of all models and represent them using the k-d tree method [45] to improve the efficiency of online recognition. While the subsequent process should be performed online, the process of creating the model library can be performed offline.

6.2 Matching

After constructing the model library, you must perform recognition work on the input scene in earnest. First, we extract feature points and feature vectors from the input scene in a similar manner as with the model library. Then, we should test each model included in the model library one at a time to see whether the model is included in the scene and what its exact location and size are if it is. The important factor here is the test order of the models. It is more efficient to test a model that is more likely to be included first than to test a model that has not been included in the scene.

The order can be determined by comparing all feature vectors extracted from the scene with all the model feature vectors contained in the model library using the NNDR method. Since all the model library's feature vectors are indexed using the k-d tree, we can easily find the closest model feature and the second closest model feature for a given scene feature. If the ratio between the closest distance and second closest is less than a threshold τ_M $(0 < \tau_M < 1)$, the scene feature and its closest model feature are considered as corresponding and the correspondence votes for the model. The closer the τ_M is to 1, the more correspondence pairs are obtained, and the smaller the τ_M, the smaller the number of correspondence pairs that will be obtained. If τ_M is too small then an insufficient number of correspondence pairs can be secured, so the algorithm's recognition rate will be low. If it is too large then both the number of correspondence pairs and the probability of false positives will increase. Fig. 6.1 shows the change in the precision and recall value according to the

change of τ_M when the algorithm of this study is tested with the U3OR dataset.

In other words, the recall value is the same as the recognition rate. As described above, when τ_M is 0.75, the recall value is lowered, and when τ_M is 0.95, the precision value is lowered. In this study, 0.9 was used as the value of τ_M.

Repeat this for all scene features and sort the models according to the number of votes received; now we will find the exact poses and sizes of the models in the scene in sorted order.

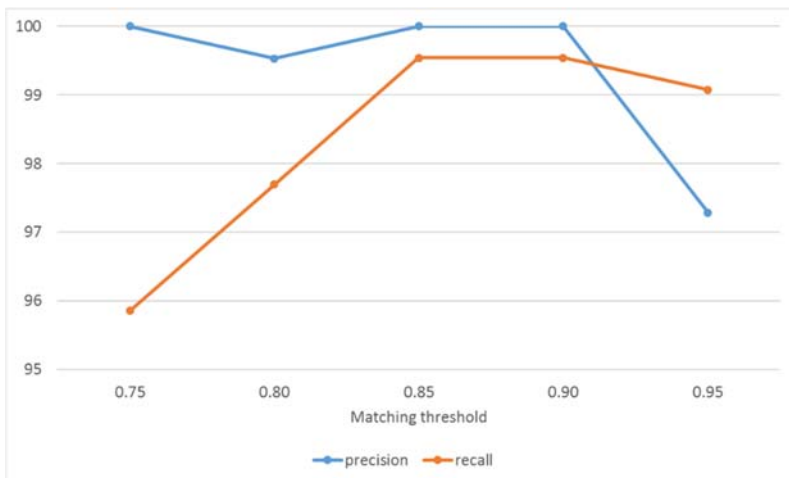


Fig. 6.1 Change of precision and recall value according to change of matching threshold in the U3OR dataset.

6.3 Transformation hypothesis generation

Once the model to be tested has been determined, the feature correspondences should be generated again, similar to the previous step. The difference is that it does not compare with all the features in the model library, only the features of the model to be tested. Using these new feature correspondences, transformation hypotheses that represent the model's posture and size are generated.

In this paper, a RANSAC-based algorithm is used to generate hypotheses. Unlike previous studies, this study assumes that the scales of the model and the scene differ, so we need to find a similarity transformation in consideration of the uniform scale for a rigid transformation. A similarity transformation can be obtained using three correspondences. Let each vertex position of three randomly sampled correspondences be m_1, m_2, m_3, s_1, s_2, and s_3. First, we need to measure the scale difference between the model and the scene. The scale ratio ρ_{scale} of the scene for the model can be obtained as follows:

$$\rho_{scale} = \frac{\sum_{i=1}^{3} \|s_i - c_s\|}{\sum_{i=1}^{3} \|m_i - c_m\|},$$

$$c_s = \frac{s_1 + s_2 + s_3}{3},$$

$$c_m = \frac{m_1 + m_2 + m_3}{3}.$$

Then, the new model vertices m_1', m_2' and m_3' are obtained sing ρ_{scale} as follows:

$$m_i' = \rho_{scale} m_i, \quad i = 1,2,3.$$

The rigid transformation $\mathbf{T}_r$ that transforms m_1', m_2', and m_3' into s_1, s_2, and s_3 can easily be obtained using the singular value decomposition technique. The reliability of $\mathbf{T}_r$ can be confirmed using the root mean square error e_{rms}, which is measured as follows:

$$e_{rms} = \frac{\sum_{i=1}^{3} \|\mathbf{T}_r m_i' - s_i\|}{3}$$

If e_{rms}/ρ_{scale} is greater than a certain threshold τ_d then the same operation is repeated by sampling other correspondences. The reason for dividing e_{rms} by ρ_{scale} is to compensate for the scale difference between the model and scene. If $e_{rms}/\rho_{scale} < \tau_d$, the reliability check is performed again using the normal vector. Let the vertex normal vector that corresponds to m_i be $\mathbf{n}_{m_i}$ and the vertex normal vector that corresponds to s_i be $\mathbf{n}_{s_i}$. Then, we can obtain $\mathbf{n}'_{m_i}$ by transforming $\mathbf{n}_{m_i}$ by $\mathbf{T}_r$. If the angle difference between $\mathbf{n}'_{m_i}$ and $\mathbf{n}_{s_i}$ is smaller than τ_a, the similarity matrix $\mathbf{T}_s$ is finally obtained by multiplying the diagonal component of $\mathbf{T}_r$ by ρ_{scale}.

We now return to the RANSAC algorithm, and should get at least one transformation hypothesis, randomly sampled from a set of correspondences. The number of correspondences sampled simultaneously is three because a similarity transformation can be obtained using three correspondences. We increase the efficiency in the random sampling process by first calculating the combination list of the correspondence set. If there are n_C correspondences, the total number of cases that can be selected in threes is ${}_{n_c}C_3$. If you create a full list of combinations

in this manner and then shuffle the list, you can effectively perform sampling without duplication, rather than randomly sample three correspondences one at a time. Once the three correspondences have been obtained in this manner; they are used to calculate the similarity transformation $\mathbf{T}_s$. If the obtained e_{rms} of the $\mathbf{T}_s$ is too large or one of the three correspondence fails in the normal vector direction check, the next three correspondences are sampled again. If not, we can calculate the consensus number of the $\mathbf{T}_s$ using the rest of the correspondences. If the model feature point of the i-th correspondence is m_i and the scene feature point is s_i, the consensus number of $\mathbf{T}_s$ increases by one when the condition $\|\mathbf{T}_s m_i - s_i\| < \tau_d$ is satisfied and the correspondence is saved. If the consensus number of $\mathbf{T}_s$ is greater than one, a new transformation matrix $\widetilde{\mathbf{T}}_s$ can be recalculated using all stored correspondences. As in the calculation of $\mathbf{T}_s$, this time, the root mean square error check and the normal direction check are performed using all the correspondences that were used to calculate $\widetilde{\mathbf{T}}_s$. Then, $\widetilde{\mathbf{T}}_s$ is saved as transformation hypothesis when the tests pass.

6.4 Verification and segmentation

After obtaining several transformation hypotheses through the RANSAC algorithm for a given model, the next step is verifying each hypothesis.

Before starting the verification, the transformation hypotheses should be sorted according to the consensus number. If the consensus numbers are the same, they are sorted in ascending order of e_{rms}. The verification is performed by applying ICP to the transformed model by the ordered transformation hypotheses.

The following parameters exist in the ICP algorithm: the sampling ratio, the maximum iteration number, and the convergence condition. Basically, we assume that the model mesh is well aligned on the scene with respect to the transformations that are true. Therefore, we use a sampling ratio of 0.01, the maximum iteration number of 30, and the convergence condition of $0.5e_{avg}$. If the ICP algorithm converges, then the transformation hypothesis is a true transformation and the model is verified. Conversely, if no transform hypotheses converge, then the model is considered not included in the scene and the next model is then tested continuously. When the ICP algorithm converges, the scene segmentation and visible proportion calculations of the model are also performed. The reason for scene segmentation is to remove the already found model from the scene to reduce the searching space and improve the efficiency when examining the next model. The method of scene segmentation is as follows: All of a scene's feature points are projected onto a model mesh that is arranged by a verified transformation matrix. If the distance between

the projected point and the scene feature point is less than $2e_{avg}$ of the model mesh, label the scene feature. Fig. 6.2 (d) shows the state of the scene after segmentation; the red color is the visible part of the model in the scene and the scene feature points in this region are excluded from the next model test (Fig. 6.3).

The model's visible proportion is calculated to apply a similar process to scene segmentation to the model mesh. This time, all vertices of the aligned model mesh are projected onto the scene mesh. As before, if the distance from the projected point is less than $2e_{avg}$, label the vertex. Then, for all faces of the model mesh, calculate the total area of the faces for which all three points are labeled and divide this by the total area of the model mesh. The resulting value is the model's visible proportion. In Fig. 6.2 (e), the red area of the model represents the part of the scene in which the model is visible. We can use the visible proportion value to perform additional validation; if the visible proportion is too small, then the model cannot be considered to have been properly found. In our study, $\tilde{\mathbf{T}}_s$ is only considered true if the visible proportion is greater than $\tau_{visible}$. We show a total segmentation example in Fig. 6.4

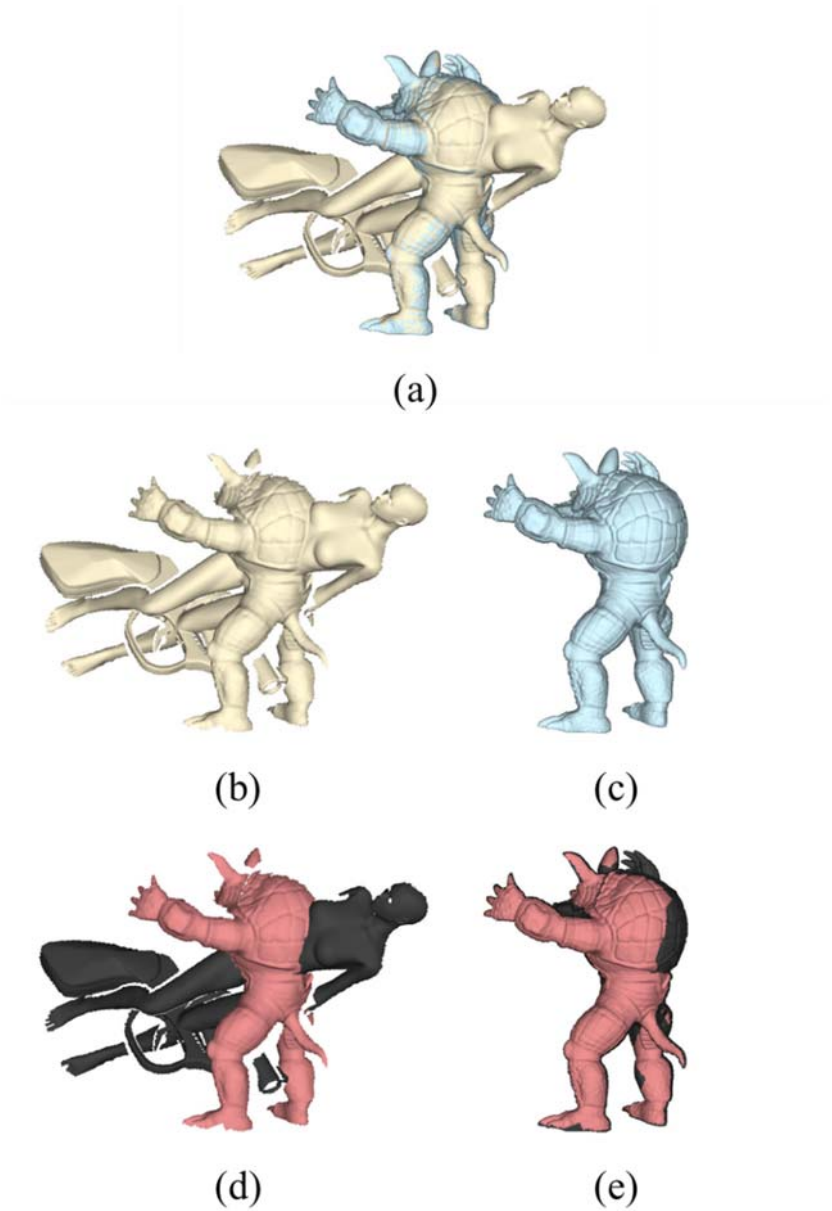


Fig. 6.2 The example of scene segmentation and the visible proportion of the model. (a) Well aligned model, (b) scene mesh, (c) model mesh, (d) scene mesh after segmentation and (e) model mesh after visible proportion calculation.

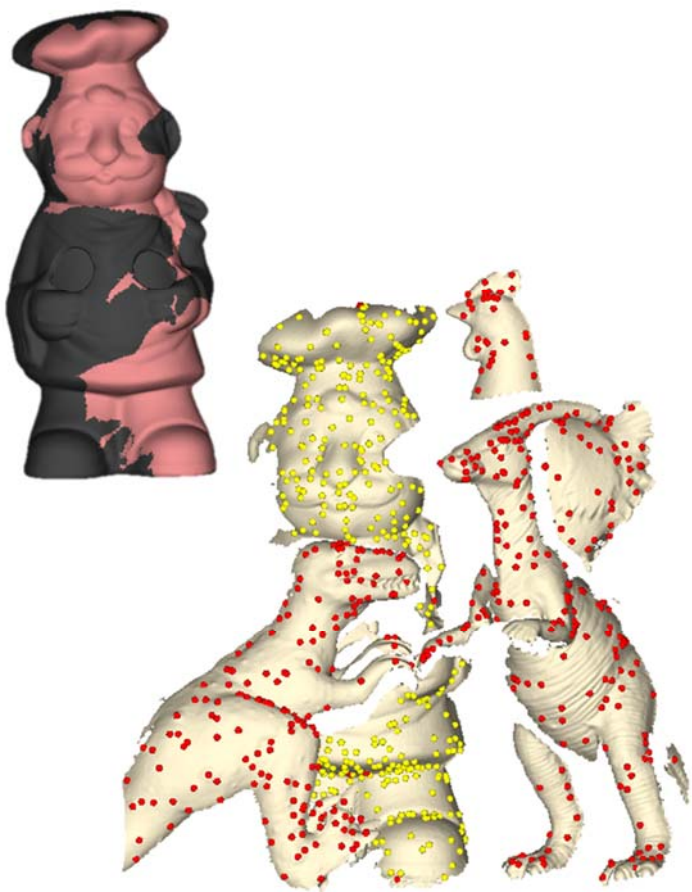


Fig. 6.3 An example of feature points segmentation of a scene. Among the feature points in the scene, the overlapping part of the Chef model was segmented in yellow.

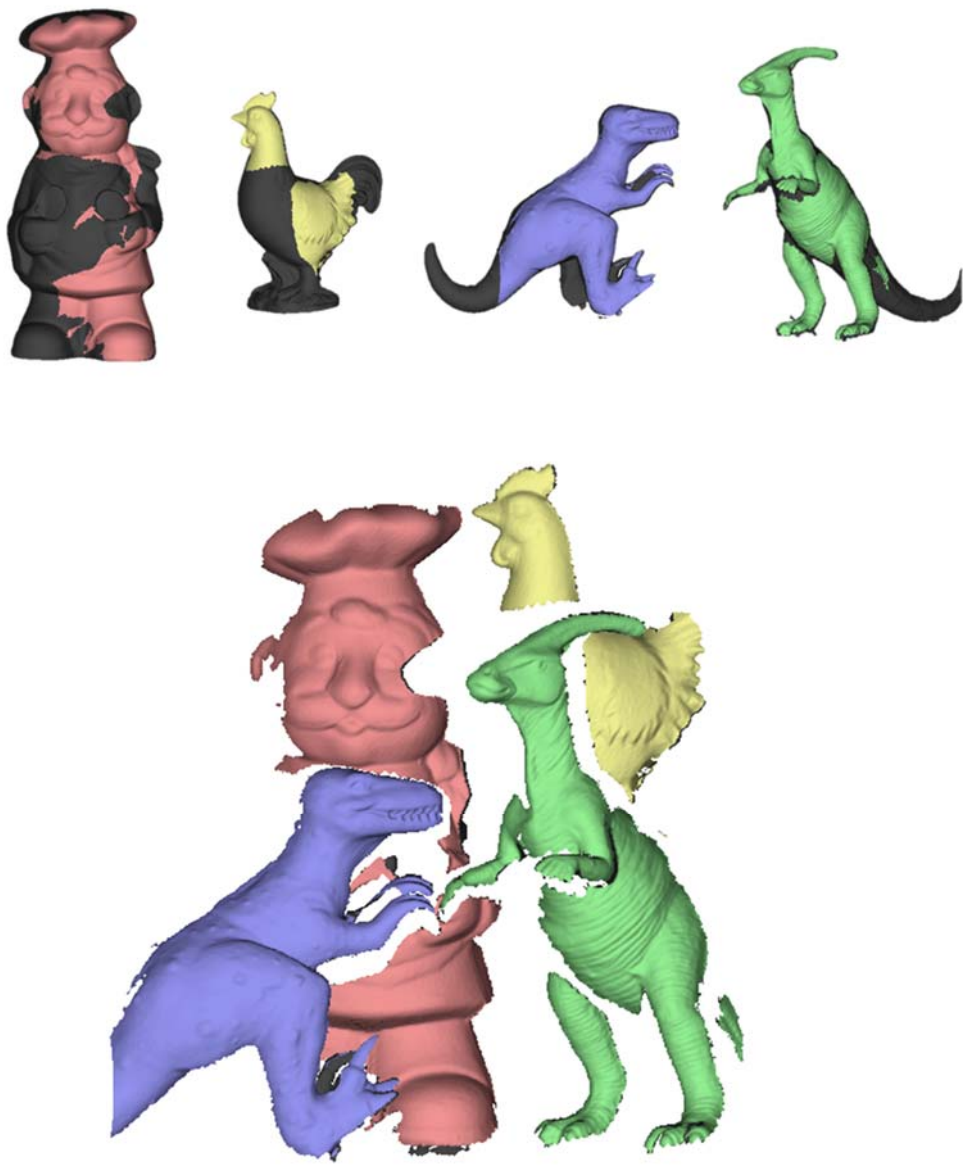
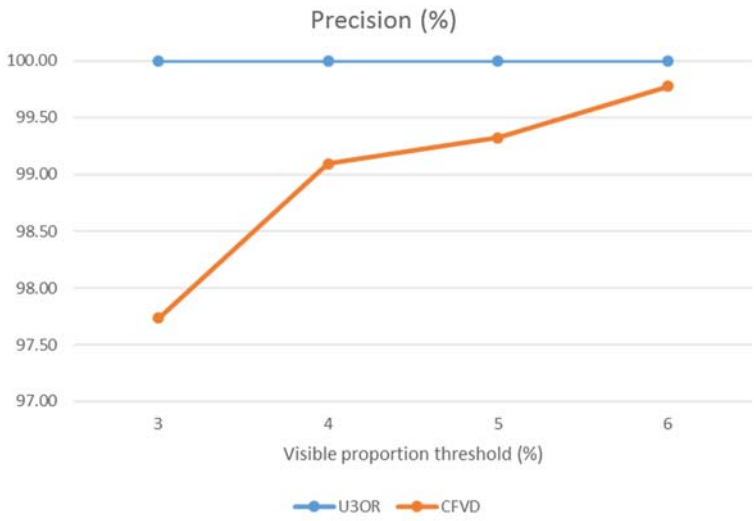


Fig. 6.4 Total segmentation results.

The effect of the $\tau_{visible}$ value on the results is investigated by measuring the precision and recall by varying the value of $\tau_{visible}$ from 0.03 to 0.06 (Fig. 6.5). When the $\tau_{visible}$ value was low, the precision of the CFVD dataset was drastically reduced. As the $\tau_{visible}$ value increased, the precision tended to increase. However, if $\tau_{visible}$ is too large, the recall may decrease because the visible proportion of the object may be smaller than $\tau_{visible}$. In fact, Fig. 6.5 shows that the recall of the U3OR dataset is reduced when $\tau_{visible}$ is 0.06. In this study, the value of $\tau_{visible}$ was chosen as 0.05 because it was almost impossible to find the correct answer if the visible proportion was less than 0.05.



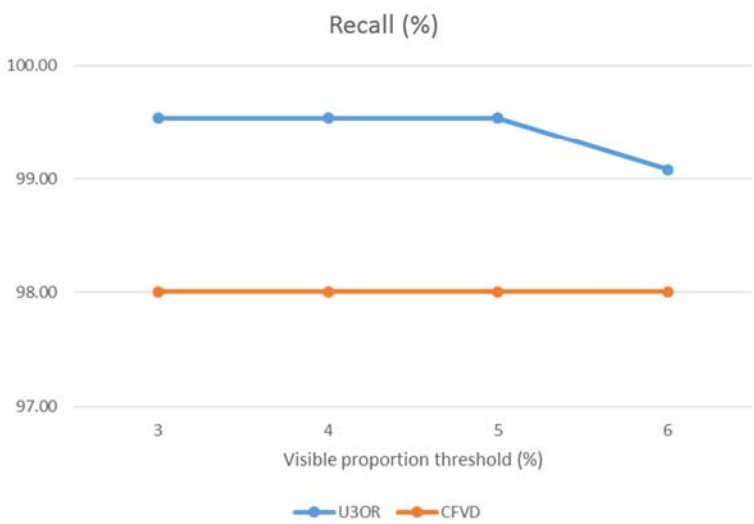


Fig. 6.5 Change of precision and recall value according to change of $\tau_{visible}$.

CHAPTER 7.
Experiments

The performance of the proposed 3D object recognition algorithm was evaluated and experiments were performed using the most famous U3OR data set and CFVD data set among publicly available datasets. These datasets contained multiple objects in each scene in the presence of occlusion and clutter; a more detailed description of these datasets has already been discussed in CHAPTER 3.

7.1 Results for the U3OR dataset

Since many existing studies have excluded the rhino model from the U3OR dataset, the experiment was carried out without the rhino model, and an experiment including the rhino model was also performed.

With the exception of the Rhino model, the average recognition rate of the proposed algorithm was 100% and there were no false positives. We found all 188 objects in the 50 scenes, which is an improvement over previous research results. Table 7.1 shows the comparison with other state-to-the art techniques. The second highest recognition rate was 99.35% for 3D-Vor [9]. Compared to the results of other studies, achieving 100% recognition rate is a very meaningful result.

Table 7.1 Comparison with major 3D object recognition systems on the U3OR dataset.

No.	Method (year)	Average Recognition Rate (%)
1	Spin Image [3] (1999)	82.8
2	3Dtesnsor [2] (2006)	94.6
3	Integral invariant [46] (2009)	86
4	Keypoint based [41] (2010)	88.5
5	VD-LSD [5] (2011)	79
6	EM [20] (2012)	97.5
7	RoPS [6] (2013)	98.8
8	TriSI [8] (2015)	95.7
9	3D-Vor [9] (2016)	99.35
10	Proposed	100

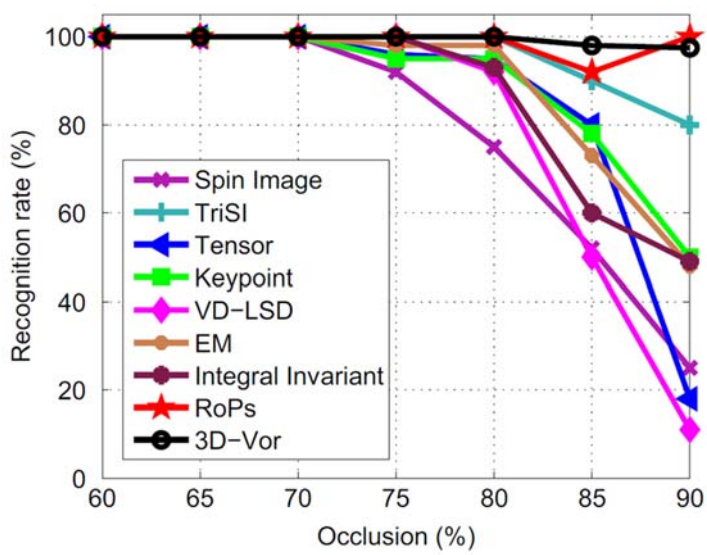


Fig. 7.1 The change in recognition rate due to occlusion in the U3OR dataset [9].

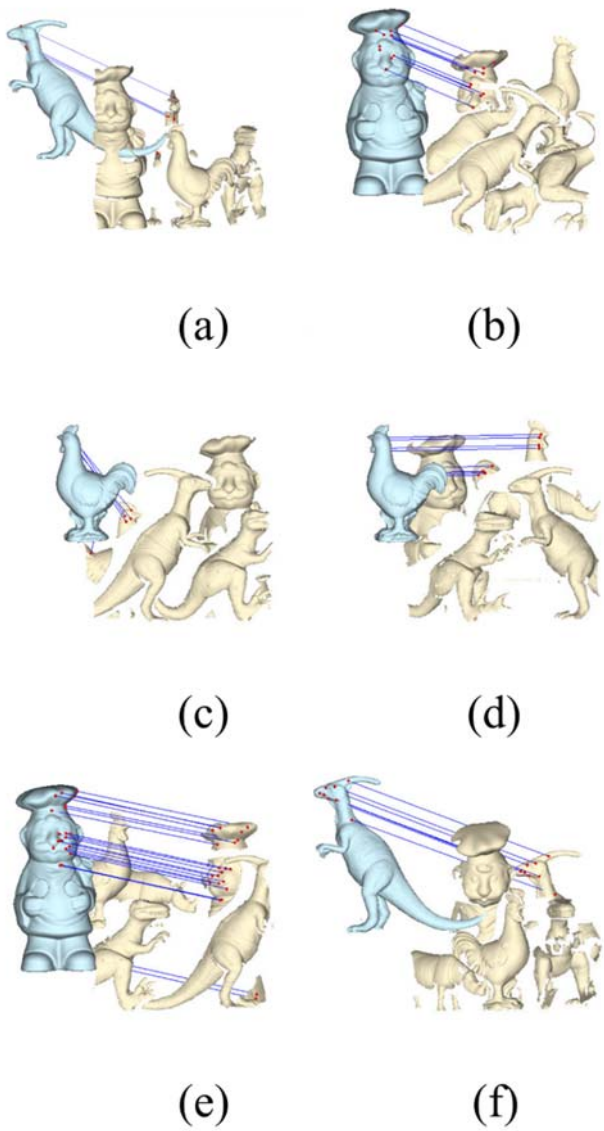


Fig. 7.2 Examples of highly occluded models. (a) Parasaurolophus, 91.4%, (b) Chef, 91.3%, (c) Chicken, 89.7%, (d) Chicken, 89.5%, (e) Chef, 89.4%, (f) Parasaurolophus, 89%.

Fig. 7.1 shows recognition rate according to the occlusion change in other studies. In these experiments, occlusion was defined according to Johnson and Hebert [1].

$$occlusion = 1 - \frac{model\ \ surface\ \ patch\ \ area\ \ in\ \ scene}{total\ \ model\ \ surface\ \ area}$$

Most other studies show a sharp decline in recognition rate for 80–90% occlusion. Even the most recent study, 3D-Vor, was unable to achieve a 100% recognition rate. Among the 188 cases that exclude the Rhino model, there were six cases in which occlusion was 89% or greater, which are shown in Fig. 7.2. In all of these cases, we successfully found the posture and size of the model, including the case with the highest occlusion rate of 91.4%.

For the test that included the Rhino model, the proposed algorithm achieved an average recognition rate of 99.54% and no false positives. We found 216 objects out of a total of 217 in the 50 scenes. Fig. 7.3 shows six examples of successful recognition in the U3OR dataset that include the Rhino model.

Fig. 7.4 shows the only case that failed. In the figure, the Rhino model between the Chef and Parasaurolophus was not recognized. In this case, the Rhino model had a very high occlusion of almost 95%.

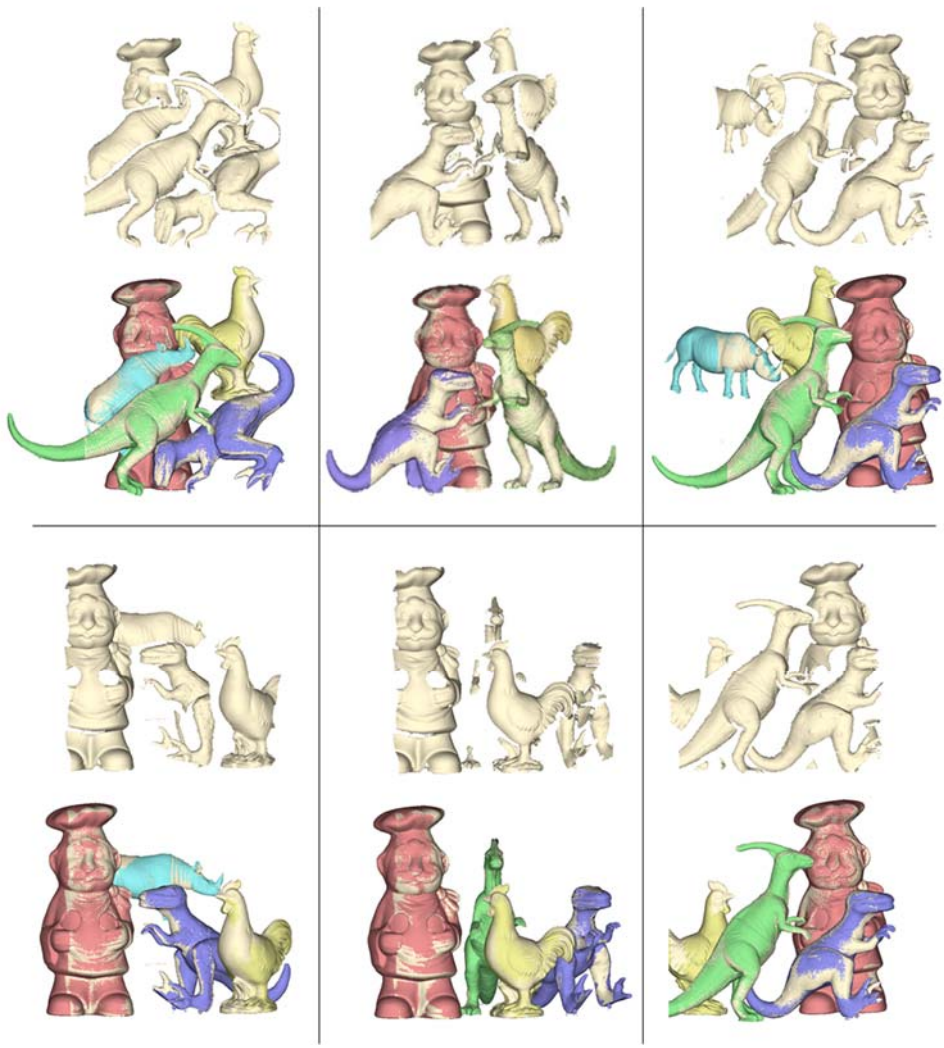


Fig. 7.3 Six examples of successful recognition in the U3OR dataset.

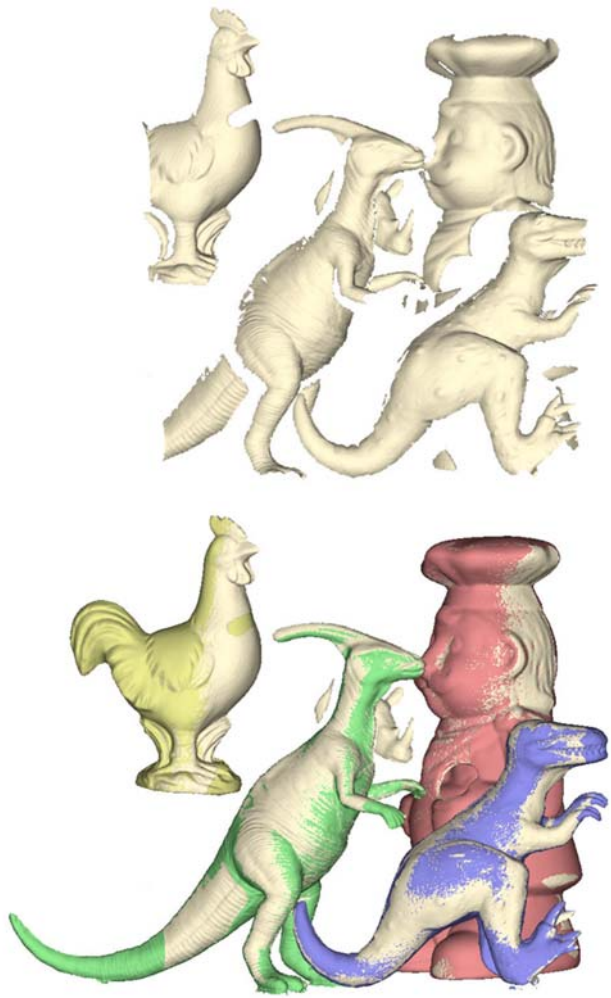


Fig. 7.4 Failed example in the U3OR dataset. The exact posture was not found for the Rhino model in the middle.

One of the main contributions of our study is the correlation between the feature selection method and feature descriptors. The feature selection method and the feature descriptors must match each other well to attain a better performance. The feature selection method proves this by using the method proposed in this study, and recognition tests are performed by only changing the descriptor. Since the most recently published 3D-Vor descriptor was closely related to their feature selection method, the experiment was instead performed using the TriSI descriptor. The test method was as follows; each model was matched to the scene containing the model and the number of matched feature pairs that was used to calculate the resultant transformation in the model was saved. The better the descriptor's performance, the greater the number of matched pairs, the higher the model's occlusion, and the smaller the number of matched feature pairs. Fig. 7.5 shows the test results for each model. As expected, the model's higher occlusion results in a smaller number of matching pairs. In addition, the performance of the proposed descriptor is superior to TriSI in all models.

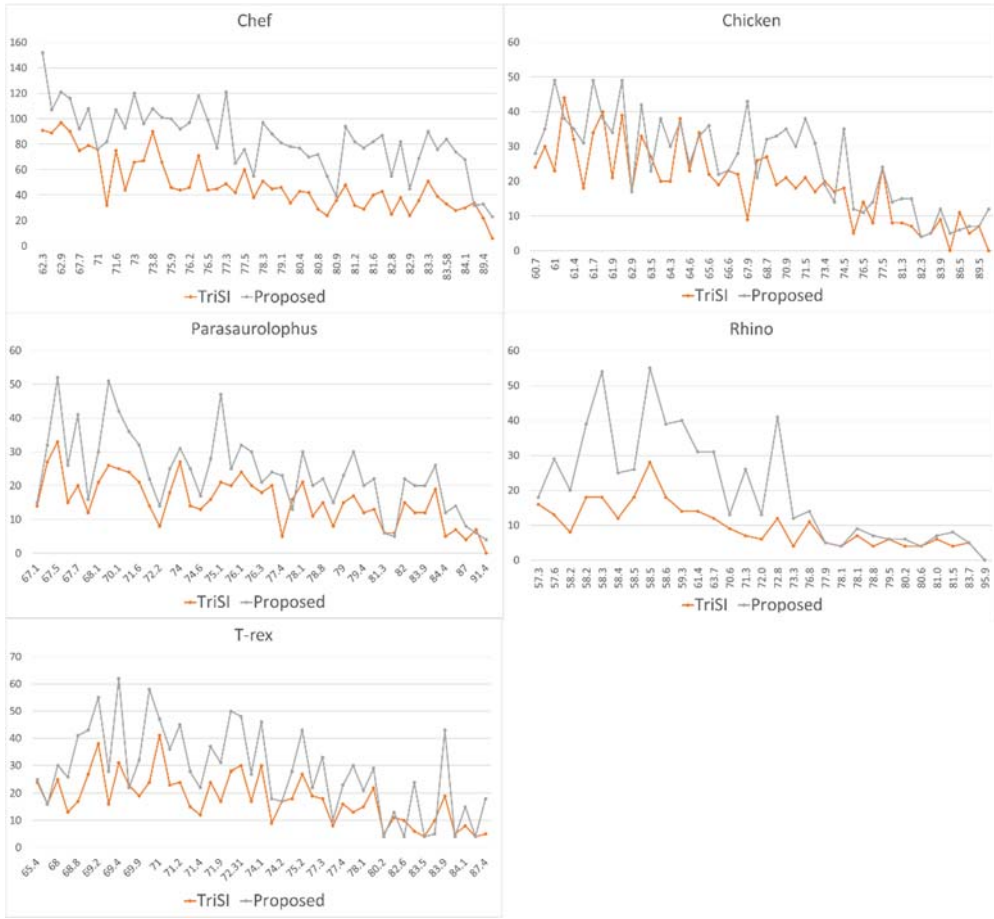


Fig. 7.5 The number of matched feature pairs according to each model's occlusion. The orange line is the result of TriSI and the gray line is the result of our descriptor.

We also tested for situations in which the model and the scene had different mesh resolutions. The e_{avg} of the half-size chef model was 0.4, and e_{avg} of the first scene was 0.84.

In Fig. 7.6, we graphically plotted the number of matched features by changing the average edge length of the chef model to 0.75 and 1.14. As e_{avg} changed from 0.4 to 0.75 and 1.14, the number of vertices of the model decreased from 176,560 to 50,738 and 21,928, respectively. The coarsest model and the original first scene are shown in Fig. 7.7.

As shown in Fig. 7.6, the lower the mesh's resolution, the greater the influence on n_O. Without downsampling, the number of matched feature pairs dropped to less than half when e_{avg} was 0.75. When the average edge length of the model was 1.14, it will fail to recognize whether downsampling has been performed. In Fig. 7.8, we show the recognition results according to the mesh resolution changes.

As the model becomes coarser and the mesh resolution difference from the scene becomes larger, the recognition difficulty increases. Increasing the number of intervals is one way to address this situation; however, increasing the number of octaves is more efficient than increasing the number of intervals. Downsampling is one advantage of the algorithm proposed in this paper because it can effectively cope with situations in which the resolution difference between the model and the scene is large.

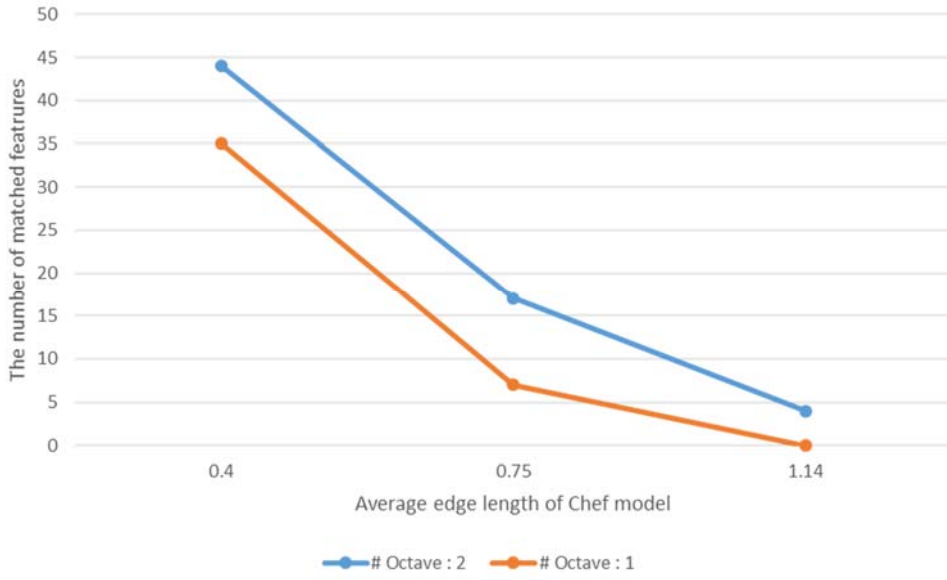


Fig. 7.6 The number of matched feature pairs according to change in average edge length.

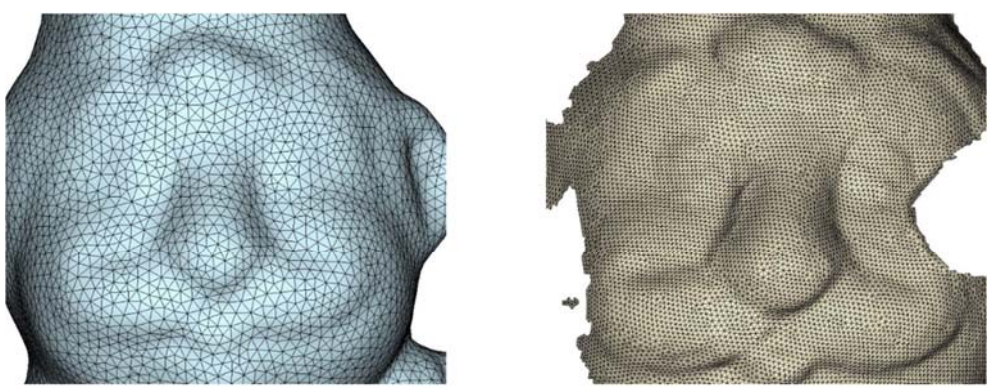


Fig. 7.7 The difference in the mesh resolution between the model and the scene. (a) $e_{avg} = 1.14$, (b) $e_{avg} = 0.84$.

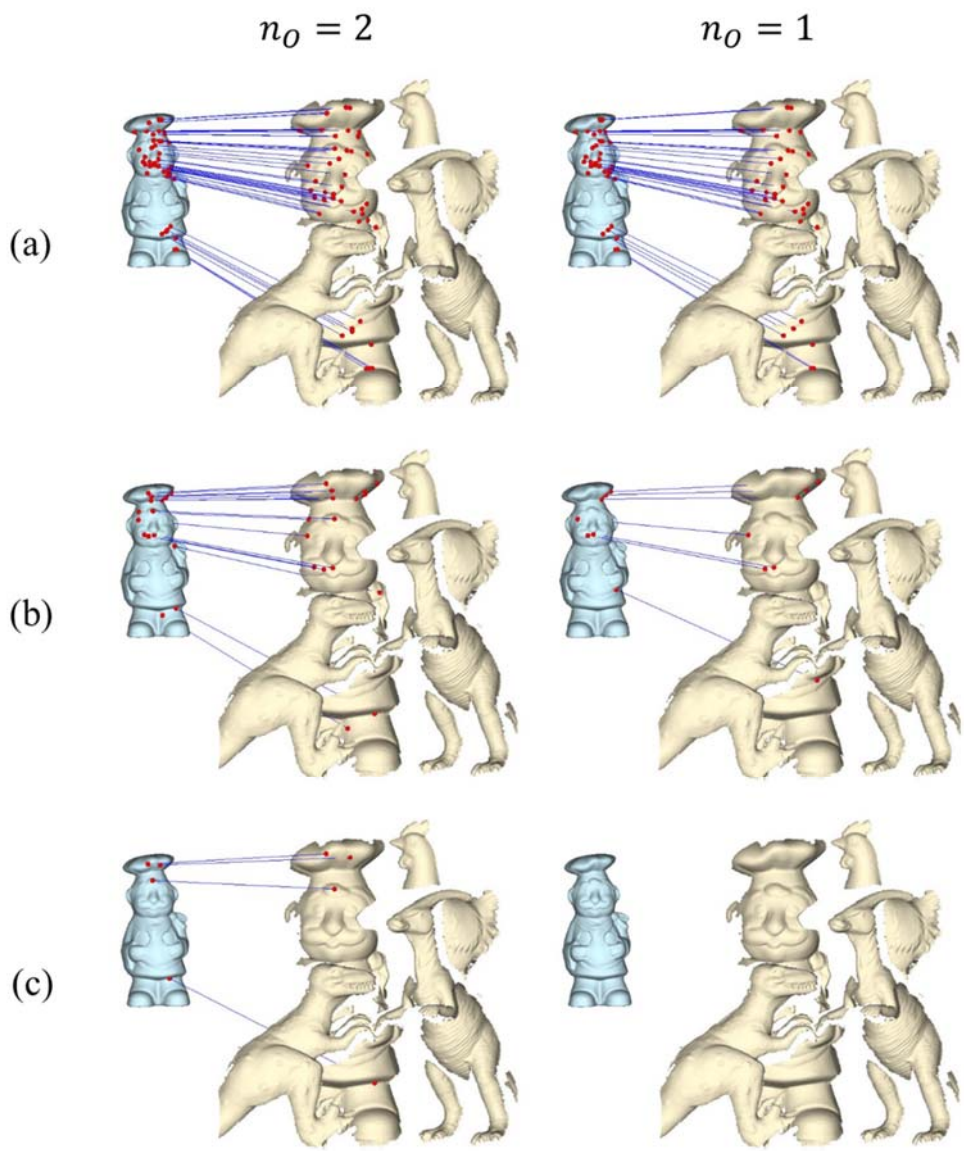


Fig. 7.8 Recognition results according to the e_{avg} of the Chef model. First column: $n_O = 2$, second column : $n_O = 1$, (a) $e_{avg} = 0.4$, (b) $e_{avg} = 0.75$, (c) $e_{avg} = 1.14$.

Finally, it should be noted that the feature selection method proposed in this study can extract certain features and support radii regardless of the mesh scale, so scale-invariant recognition can be performed; this is the main contribution of our feature selection algorithm. Few studies have achieved scale-invariant 3D object recognition [10, 20, 41] and these scale-invariant recognition systems tend to be less recognizable than fixed-scale recognition systems. Unlike previous studies, our algorithm is robust at any scale (Fig. 7.9).

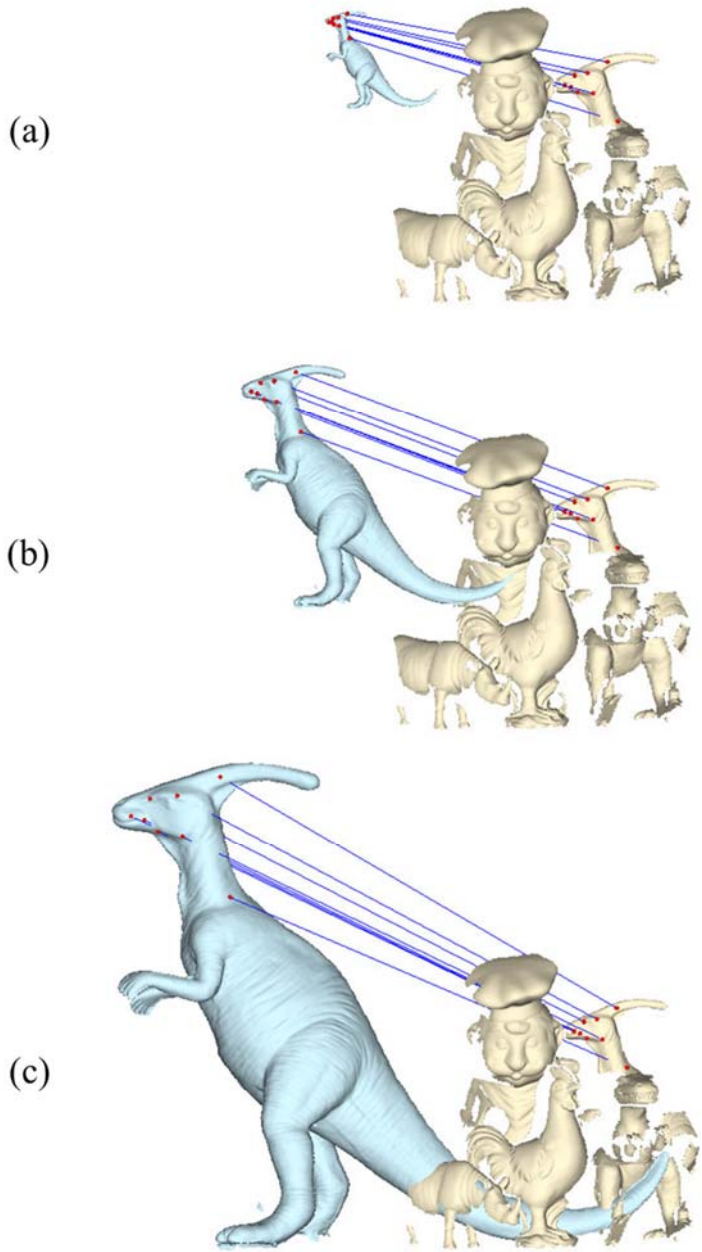


Fig. 7.9 Examples of scale-invariant recognition. The recognition results are the same even if the model scale has changed. The model scale is (a) 0.4, (b) 1.0, (c) 2.0.

7.2 Results on the CFVD dataset

As in other studies [6, 8-10], we tested 150 scenes using the 18 out of 20 models in the CFVD dataset. The CFVD dataset is characterized by the fact that there are 3–5 objects in each scene, while there are 18 tested models. Therefore, many false positive cases occur if the performance of the feature descriptor and matching algorithm is insufficient. In addition, the dataset contains several models with large flat and featureless areas and very similar shapes [10]. Thus, testing with the CFVD dataset is more challenging than with other datasets.

The average recognition rate of the proposed algorithm was 97.8% and there are only one false positive; we found 441 objects out of 451 in 150 scenes. Table 7.2 shows the precision and recall values for each model when tested with the proposed algorithm. Table 7.3 and Table 7.4 show the results of comparison with other studies (3D-Vor [9], TriSI [8], RoPS [6], SHOT + Game [10]) for precision and recall values, respectively. As we can see from these two tables, the proposed algorithm outperformed other studies. Fig. 7.10 shows six examples of successful recognition in the CFVD dataset.

Table 7.2 Object recognition test results for the CFVD datasets.

Model	Total Matches	True Positive	False Positive	Precision (%)	Recall (%)
Armadillo	30	28	0	100.0	93.3
Bunny	35	35	0	100.0	100.0
Cat1	25	24	0	100.0	96.0
Centaur1	24	24	0	100.0	100.0
Chef	13	13	0	100.0	100.0
Chicken	27	27	0	100.0	100.0
Dog7	22	21	0	100.0	95.5
Dragon	19	19	0	100.0	100.0
Face	22	22	1	95.7	100.0
Genesha	24	24	0	100.0	100.0
Gorilla0	23	23	0	100.0	100.0
Horse7	34	34	0	100.0	100.0
Lioness13	22	22	0	100.0	100.0
Para	32	30	0	100.0	93.8
Rhino	22	21	0	100.0	95.5
Trex	39	39	0	100.0	100.0
Victoria3	19	16	0	100.0	84.2
Wolf2	19	19	0	100.0	100.0
Total	**451**	**441**	**1**	**99.8**	**97.8**

Table 7.3 Comparison of the precision values with other studies in the CFVD test.

Model	Proposed	3D-Vor	TriSI	RoPS	SHOT+Game
Armadillo	100	100	100	97	100
Bunny	100	100	100	100	100
Cat1	100	94	100	100	78
Centaur1	100	100	100	100	96
Chef	100	100	93	100	93
Chicken	100	98	100	97	93
Dog7	100	99	100	100	95
Dragon	100	99	100	100	100
Face	95.7	100	100	100	91
Genesha	100	100	100	100	89
Gorilla0	100	97	100	100	95
Horse7	100	100	97	100	97
Lioness13	100	95	100	100	88
Para	100	96	97	97	97
Rhino	100	98	100	96	91
Trex	100	100	100	100	97
Victoria3	100	100	100	100	83
Wolf2	100	98	100	100	82
Total	**99.8**	**98.3**	**99.3**	**99.1**	**93**

Table 7.4 Comparison of the recall values with other studies in the CFVD test.

Model	Proposed	3D-Vor	TriSI	RoPS	SHOT+Game
Armadillo	93.3	100	100	100	97
Bunny	100	100	100	100	97
Cat1	96.0	70	60	44	82
Centaur1	100	100	100	100	100
Chef	100	100	100	100	100
Chicken	100	100	100	100	100
Dog7	95.5	89	91	91	86
Dragon	100	92	100	100	89
Face	100	99	100	100	95
Genesha	100	100	96	100	100
Gorilla0	100	100	100	100	91
Horse7	100	100	100	100	100
Lioness13	100	100	96	100	100
Para	93.8	96	100	97	94
Rhino	95.5	97	100	100	91
Trex	100	100	100	100	97
Victoria3	84.2	97	95	95	83
Wolf2	100	93	100	100	95
Total	**97.8**	**96.2**	**96.7**	**96**	**94.7**

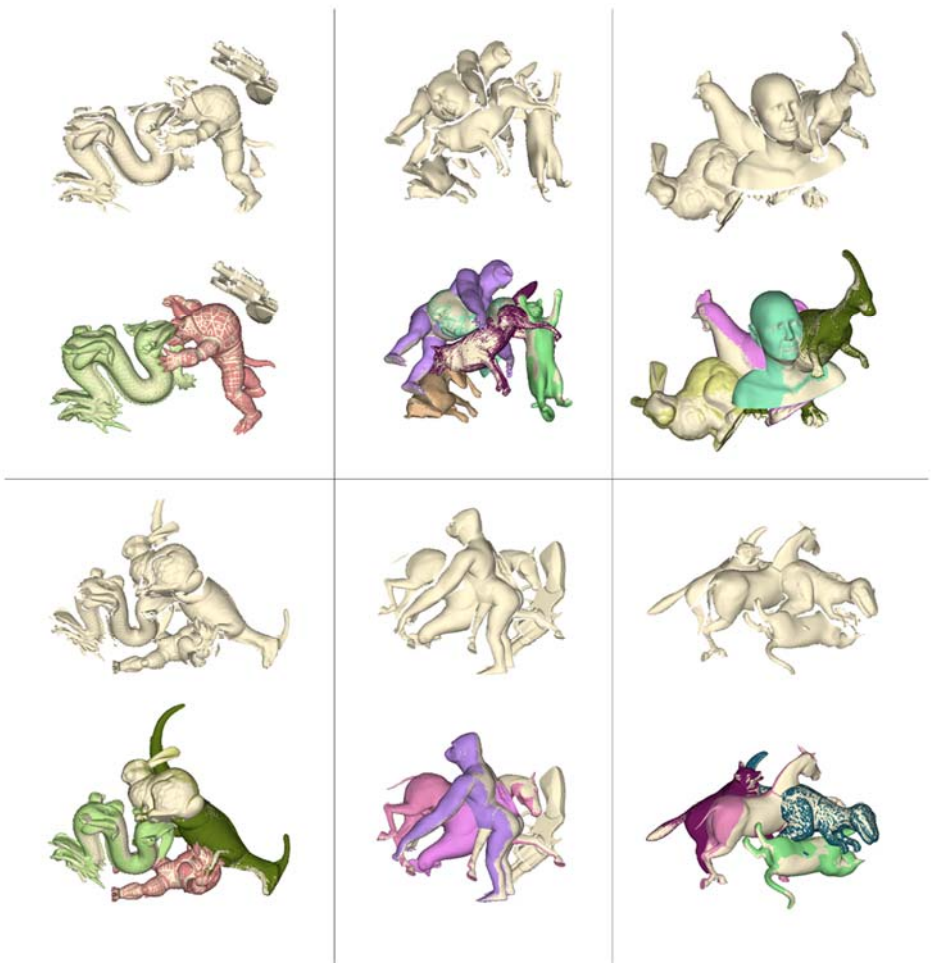


Fig. 7.10 Six examples of successful recognition in the CFVD dataset.

Another notable point is the number of selected features. Table 7.5 shows the average number of features selected in the U3OR and CFVD datasets. Compared with RoPS [6] and 3D-Vor [9], our study is meaningful as it uses fewer features and yet still achieves higher recognition rates. In particular, the number of average feature points in a scene is less than half that of 3D-Vor for the U3OR dataset, and less than one-third for the CFVD dataset. This can be a great advantage, as it takes a long time to calculate a feature. This statistic also means that the feature selection method and the feature descriptor proposed in this study are superior to existing methods.

Table 7.5 The average number of selected feature points in a scene and a model.

U3OR dataset

	scenes	models
RoPS	2,259	4,247
3D-Vor	1,947	2,324
Proposed	**910**	**1,518**

CFVD dataset

	scenes	models
RoPS	2,210	5,000
3D-Vor	1,702	1,977
Proposed	**485**	**1,141**

CHAPTER 8.
Conclusion

This paper proposes new algorithms for each element technique in 3D object recognition. First, we presented a novel scale-invariant feature selection method that successfully extended the scale-invariant nature of the 2D SIFT algorithm to the 3D surface. By effectively calculating the Gaussian and DoG pyramids, it is able to detect highly repeatable features and obtain the support radius irrespective of the mesh scale.

We also proposed a new feature descriptor using the gradient of the scalar function. Because the scalar function was also used in the feature selection method, the proposed descriptor can best represent information around a feature. In addition, by projecting a gradient vector on a 2D plane, the surrounding information can be represented more compactly. We increased the robustness of the descriptor using LRF and by allowing more than one descriptor vector at the same feature point.

In the feature matching step, we proposed a new RANSAC-based transform hypothesis generation algorithm that finds the correct correspondence groups among many correspondence pairs by calculating the similarity transformation with only three vertices. It also applies the vertex normal rejection step to reduce false positives.

We have developed a robust 3D object recognition system that can effectively cope with large differences in scale between models and scenes by combining the above algorithms.

We tested our algorithm on the U3OR and CFVD datasets, which are the most widely used in 3D object recognition. The experimental results show that the proposed technique achieved recognition rates of 99.5% and 97.8%, respectively, which is beyond that of state-of-the-art studies.

Another advantage of the proposed algorithm is that it uses arbitrary scalar functions as defined in the surface mesh. If the scalar function represents the surface information such as the curvature, geometric feature points can be found. Similarly, the same algorithm can even be applied when the scalar function represents texture information. As the 3D surface data including the color and texture information becomes more generalized, the algorithm proposed in this study can be applied more broadly.

Finally, the proposed algorithm can also be applied in a data-driven approach. The next step in the field of object recognition is to find similar shapes in real scenes by building a large database rather than by looking for shapes that exactly match an already known template. Such an approach will require finding the features of an object and describing them effectively. From this perspective, the algorithm proposed in this study will be valuable as the underlying technology of a data-driven approach.

REFERENCES

1. Johnson, A.E. and M. Hebert, *Using spin images for efficient object recognition in cluttered 3D scenes.* IEEE transactions on pattern analysis and machine intelligence, 1999. **21**(5): p. 433-449.

2. Mian, A.S., M. Bennamoun, and R. Owens, *Three-dimensional model-based object recognition and segmentation in cluttered scenes.* IEEE transactions on pattern analysis and machine intelligence, 2006. **28**(10): p. 1584-1601.

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

4. Taati, B., M. Bondy, P. Jasiobedzki, and M. Greenspan. *Variable Dimensional Local Shape Descriptors for Object Recognition in Range Data*. in *2007 IEEE 11th International Conference on Computer Vision*. 2007.

5. Taati, B. and M. Greenspan, *Local shape descriptor selection for object recognition in range data.* Computer Vision and Image Understanding, 2011. **115**(5): p. 681-694.

6. Guo, Y., F. Sohel, M. Bennamoun, M. Lu, and J. Wan, *Rotational projection statistics for 3D local surface description and object recognition.* International Journal of Computer Vision, 2013. **105**(1): p. 63-86.

7. Guo, Y., M. Bennamoun, F. Sohel, M. Lu, and J. Wan, *3d object recognition in cluttered scenes with local surface features: A survey.* IEEE transactions on pattern analysis and machine intelligence, 2014. **36**(11): p. 2270-2287.

8. Guo, Y., F. Sohel, M. Bennamoun, J. Wan, and M. Lu, *A novel local surface feature for 3D object recognition under clutter and occlusion.* Information Sciences, 2015. **293**: p. 196-213.

9. Shah, S.A.A., M. Bennamoun, and F. Boussaid, *A novel feature representation for automatic 3D object recognition in cluttered scenes.* Neurocomputing, 2016. **205**: p. 1-15.

10. Rodolà, E., A. Albarelli, F. Bergamasco, and A. Torsello, *A scale independent selection process for 3d object recognition in cluttered scenes.* International Journal of Computer Vision, 2013. **102**(1-3): p. 129-145.

11. Mokhtarian, F., N. Khalili, and P. Yuen, *Multi-scale free-form 3D object recognition using 3D models.* Image and Vision Computing, 2001. **19**(5): p. 271-281.

12. Yamany, S.M. and A.A. Farag, *Surface signatures: an orientation independent free-form surface representation scheme for the purpose of objects registration and matching.* IEEE transactions on pattern analysis and machine intelligence, 2002. **24**(8): p. 1105-1120.

13. Gal, R. and D. Cohen-Or, *Salient geometric features for partial shape matching and similarity.* ACM Transactions on Graphics (TOG), 2006. **25**(1): p. 130-150.

14. Matei, B., Y. Shan, H.S. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert, *Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation.* IEEE transactions on pattern analysis and machine intelligence, 2006. **28**(7): p. 1111-1126.

15. Zhong, Y. *Intrinsic shape signatures: A shape descriptor for 3d object recognition*. in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*. 2009. IEEE.

16. Sipiran, I. and B. Bustos, *Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes.* The Visual Computer, 2011. **27**(11): p. 963-976.

17. Harris, C. and M. Stephens. *A combined corner and edge detector*. in *Alvey vision conference*. 1988. Citeseer.

18. Castellani, U., M. Cristani, S. Fantoni, and V. Murino. *Sparse points matching by combining 3D mesh saliency with statistical descriptors*. in *Computer Graphics Forum*. 2008. Wiley Online Library.

19. Darom, T. and Y. Keller, *Scale-invariant features for 3-D mesh models.* IEEE

Transactions on Image Processing, 2012. **21**(5): p. 2758-2769.

20. Bariya, P., J. Novatnack, G. Schwartz, and K. Nishino, *3D geometric scale variability in range images: Features and descriptors.* International Journal of Computer Vision, 2012. **99**(2): p. 232-255.

21. Novatnack, J. and K. Nishino. *Scale-dependent 3D geometric features.* in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. 2007. IEEE.

22. Zaharescu, A., E. Boyer, K. Varanasi, and R. Horaud. *Surface feature detection and description with applications to mesh matching*. in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. 2009. IEEE.

23. Chua, C.S. and R. Jarvis, *Point signatures: A new representation for 3d object recognition.* International Journal of Computer Vision, 1997. **25**(1): p. 63-85.

24. Malassiotis, S. and M.G. Strintzis, *Snapshots: A novel local surface descriptor and matching algorithm for robust 3D surface alignment.* IEEE transactions on pattern analysis and machine intelligence, 2007. **29**(7).

25. do Nascimento, E.R., G.L. Oliveira, A.W. Vieira, and M.F. Campos, *On the development of a robust, fast and lightweight keypoint descriptor.*

Neurocomputing, 2013. **120**: p. 141-155.

26. Frome, A., D. Huber, R. Kolluri, T. Bülow, and J. Malik, *Recognizing objects in range data using regional point descriptors*. Computer vision-ECCV 2004, 2004: p. 224-237.

27. Tombari, F., S. Salti, and L. Di Stefano. *Unique shape context for 3D data description*. in *Proceedings of the ACM workshop on 3D object retrieval*. 2010. ACM.

28. Guo, Y., F.A. Sohel, M. Bennamoun, M. Lu, and J. Wan. *TriSI: A Distinctive Local Surface Descriptor for 3D Modeling and Object Recognition*. in *GRAPP/IVAPP*. 2013.

29. Chen, H. and B. Bhanu, *3D free-form object recognition in range images using local surface patches.* Pattern Recognition Letters, 2007. **28**(10): p. 1252-1262.

30. Chen, H. and B. Bhanu, *Human ear recognition in 3D.* IEEE transactions on pattern analysis and machine intelligence, 2007. **29**(4).

31. Koenderink, J.J. and A.J. Van Doorn, *Surface shape and curvature scales.* Image and Vision Computing, 1992. **10**(8): p. 557-564.

32. Rusu, R.B., N. Blodow, Z.C. Marton, and M. Beetz. *Aligning point cloud views using persistent feature histograms*. in *Intelligent Robots and Systems*,

2008. IROS 2008. IEEE/RSJ International Conference on. 2008. IEEE.

33. Rusu, R.B., N. Blodow, and M. Beetz. *Fast point feature histograms (FPFH) for 3D registration*. in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. 2009. IEEE.

34. Salti, S., F. Tombari, and L. Di Stefano, *Shot: Unique signatures of histograms for surface and texture description.* Computer Vision and Image Understanding, 2014. **125**: p. 251-264.

35. Tombari, F., S. Salti, and L. Di Stefano. *Unique signatures of histograms for local surface description*. in *European Conference on Computer Vision*. 2010. Springer.

36. Mikolajczyk, K. and C. Schmid, *A performance evaluation of local descriptors.* IEEE transactions on pattern analysis and machine intelligence, 2005. **27**(10): p. 1615-1630.

37. Besl, P.J. and N.D. McKay. *Method for registration of 3-D shapes*. in *Robotics-DL tentative*. 1992. International Society for Optics and Photonics.

38. Papazov, C. and D. Burschka, *An efficient ransac for 3d object recognition in noisy and occluded scenes.* Computer Vision–ACCV 2010, 2011: p. 135-148.

39. Papazov, C., S. Haddadin, S. Parusel, K. Krieger, and D. Burschka, *Rigid 3D*

geometry matching for grasping of known objects in cluttered scenes. The International Journal of Robotics Research, 2012. **31**(4): p. 538-553.

40. Candès, E.J., J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information.* IEEE Transactions on information theory, 2006. **52**(2): p. 489-509.

41. Mian, A., M. Bennamoun, and R. Owens, *On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes.* International Journal of Computer Vision, 2010. **89**(2): p. 348-361.

42. Witkin, A. *Scale-space filtering: A new approach to multi-scale description*. in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84.* 1984. IEEE.

43. Botsch, M. and L. Kobbelt. *A remeshing approach to multiresolution modeling*. in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*. 2004. ACM.

44. Guo, Y., M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N.M. Kwok, *A comprehensive performance evaluation of 3D local feature descriptors.* International Journal of Computer Vision, 2016. **116**(1): p. 66-89.

45. Bentley, J.L., *Multidimensional binary search trees used for associative searching.* Communications of the ACM, 1975. **18**(9): p. 509-517.

46. Pottmann, H., J. Wallner, Q.-X. Huang, and Y.-L. Yang, *Integral invariants for robust geometry processing.* Computer Aided Geometric Design, 2009. **26**(1): p. 37-60.

ABSTRACT (Korean)

최근 들어서 3차원 스캐닝 기술이 발전함에 따라 다양한 3차원 표면 데이터를 쉽게 얻을 수 있게 되었고, 이와 더불어서 3차원 데이터에 대한 정합 및 인식 기술에 대한 수요도 나날이 증가하고 있는 추세이다. 특히 물체의 많은 부분이 가려지고 여러 물체가 존재하는 복잡한 배경에서 3차원 물체의 정확한 위치를 찾아내는 기술은 산업 현장의 검사, 의료 영상, 게임 등 여러 곳에서 필요한 중요한 기술이다.

기존의 많은 연구들은 물체의 크기가 알려지지 않은 경우나 가려진 부분이 클 경우에는 원하는 만큼의 성능을 내지 못하였다. 본 연구에서는 기존 연구들의 성능을 뛰어넘는 새로운 물체 인식 알고리즘을 제안하였다. 복잡한 배경에서 3차원 물체의 위치 및 크기를 인식하는 과정은 크게 특징점 선택, 특징점 기술, 매칭의 3단계로 이루어진다.

본 연구에서는 2D SIFT 알고리즘을 3차원으로 확장하여 완벽한 규모 불편의 특성을 갖는 특징점 선택 알고리즘을 제안하였다. 이 특징점 선택 알고리즘은 3차원 표면 데이터에 정의된 스칼라 함수에 대한 Difference of Gaussian 피라미드를 구성하여 반복성이 높고 규모 불변한 성질을 갖는 특징점들을 추출해낸다. 이렇게 선택된 특징점들은 본 연구에서 제안한 새로운 형상 기술자에 의해서 효과적으로 주변 정보를 묘사할 수 있

다. 기존의 형상 기술자들과 달리 3차원 표면 데이터에 정의된 스칼라 함수의 Gradient를 이용함으로써 본 연구에서 제안한 특징점 선택 알고리즘과 결합 하였을 때 높은 인식률을 달성할 수 있다. 또한 새로운 RANSAC 기반의 Transform Hypothesis 생성 알고리즘을 제안함으로써 검색 공간을 줄이고 False Positive의 확률을 낮췄다.

본 연구에서 제안된 3차원 물체 인식 알고리즘을 기존의 연구에서 많이 사용되었던 U3OR 데이터와 CFVD 데이터로 테스트 한 결과, 각각 99.5%와 97.8% 의 인식률을 얻었으며, 이는 기존 연구들의 결과를 뛰어넘는 수치이다.

주요어: 3차원 물체 인식, 규모 불변의 특징점, 규모 불변의 물체 인식, 3차원 특징 기술자, RANSAC 매칭

학번: 2010-23229