Vis Comput (2010) 26: 1269–1281
DOI 10.1007/s00371-010-0521-3

ORIGINAL ARTICLE

Partial matching of real textured 3D objects using color cubic higher-order local auto-correlation features

Asako Kanezaki · Tatsuya Harada · Yasuo Kuniyoshi

Published online: 30 July 2010

Abstract In recent years, the need for retrieving real 3D objects has grown significantly. However, various important considerations must be taken into account to solve the real 3D object retrieval problem. Three-dimensional models obtained without the use of special equipment such as engineered environments or multi-camera systems are often incomplete. Therefore, the ability to perform partial matching is essential. Moreover, the time required for the matching process must be relatively short, since the operation will need to be performed repeatedly to deal with the dynamic nature of day-to-day human environments. Furthermore, real models often include rich texture information, which can compensate for the limited shape information. Thus, the descriptors of the 3D models have to consider both shape and texture patterns. In this paper, we present new 3D shape features which take into account the object's texture. The additive property of these features enables efficient partial matching between query data and 3D models in a database. In the experiments, we compare these features with conventional features, namely Spin-Image, Textured Spin-Image, and CHLAC features using a dataset of real textured objects. Furthermore, we demonstrate the retrieval performance of these features on a real color 3D scene.

Keywords Real object retrieval · Shape and texture features · Partial matching · Real-time 3D processing · Object detection

A. Kanezaki (✉) · T. Harada · Y. Kuniyoshi
The University of Tokyo, Tokyo, Japan
e-mail: kanezaki@isi.imi.i.u-tokyo.ac.jp

T. Harada
e-mail: harada@isi.imi.i.u-tokyo.ac.jp

Y. Kuniyoshi
e-mail: kuniyosh@isi.imi.i.u-tokyo.ac.jp

1 Introduction

Recently, the state of the art of 3D scanning has advanced dramatically, allowing us to obtain precise 3D models of various objects with associated textures. It has also become possible to obtain 3D descriptions of large scenes. The availability of such precise 3D models has motivated research into the problem of real 3D object retrieval.

Various important considerations must be taken into account when addressing the problem of real 3D object retrieval. First, the ability to perform partial matching is essential. Real 3D models are often incomplete since they are measured from only a few directions. Also, when searching for a query object in a 3D scene, partial matching is inevitable because objects are rarely segmented in the scene obtained with a 3D scanner. Furthermore, the matching method needs to be fast, for it to be applicable to large databases or large 3D scenes. If the matching is to be performed against a changeable dataset such as a 3D scene in a day-to-day environment, the feature extraction process must also be fast. Second, whereas artificial 3D models can be textured according to the users' preference, real 3D models have their own unique color textures containing rich information about the visual "feel" of the objects' surfaces. This texture information can supplement shape information and provide an additional clue for retrieval. In particular where the amount of occlusion is large, the texture of the target object can more easily be observed than its shape characteristics. Thus, real 3D object retrieval techniques should take into account texture information as well as shape information.

In this paper, we propose "Color Cubic Higher-order Local Auto-Correlation (Color-CHLAC) Features", which accumulate local descriptors of both shape and texture patterns of the objects' surfaces, and which can then be used to match the query object against models of real textured 3D objects.

These features can be extracted from partial models of any shape or size and their computation is fairly fast. Moreover, the feature vector of any part of the target model can be calculated simply by summing the feature vectors of its subparts. This additive property allows high-speed matching of partial models of various sizes.

The rest of this paper is organized as follows: Section 2 discusses related work on 3D features, while our 3D features are presented in Sect. 3. Section 4 describes the matching method. Experimental results are given in Sect. 5, while Sect. 6 presents a search demonstration in a 3D scene. Finally, Sect. 7 summarizes our method and proposes future work.

2 Related work

Expressiveness of descriptors is critically important for object retrieval. Furthermore, descriptors and matching methods are so closely related that the question whether fast matching can be applied depends on the property of the descriptors. In this section we discuss 3D descriptors from the point of view of expressiveness and the ability to apply a fast partial matching method. We first discuss conventional descriptors which are useful for artificial 3D object retrieval, and then refer to Spin-Image and Textured Spin-Image, which can be applied to real 3D object retrieval.

2.1 Descriptors for artificial 3D object retrieval

In the field of artificial 3D object retrieval, various 3D shape descriptors have been proposed [1, 2]. As discussed in [2], these can be classified as histogram-based, transform-based, graph-based, or 2D view-based. Histogram-based [3] and transform-based descriptors [4, 5] rely on the center point of 3D models, making it difficult to apply these techniques to partial matching. Graph-based descriptors can be used to perform partial matching [6–9], however, it is difficult to apply such a descriptor to real object retrieval since only those parts with segments matching the predefined sub-parts of the models in the database can be compared. Similarly, although [10] enables fast object retrieval by priority-driven partial matching, the matching is limited to salient segments, which are not necessarily observed in a real environment. 2D view-based descriptor [11] computed from multiple range images can also be used for partial matching, however, the memory requirement and computation cost are both high since a query part must be compared with all views of each database model.

Furthermore, these descriptors do not take texture information into consideration. In [12], shape descriptors are used based on the curvature of surface patches and color descriptors represented by the average, maximum, and minimum values of R, G, and B. However, in this approach, it is difficult to balance shape information and texture information. Also, this method does not take into account shape and texture co-occurrence patterns.

2.2 Spin-image and textured spin-image

Spin-Image (SI) [13] is a computational tool able to describe local shape patterns and to perform partial matching, and as such can be applied to object recognition in real environments containing clutter and occlusion. A SI is created for an oriented point at a vertex in the surface mesh by projecting points onto cylindrical coordinates. It is computed using the following equation:

$$(\alpha, \beta) = \left(\sqrt{\|\boldsymbol{x} - \boldsymbol{p}\|^2 - (\boldsymbol{n} \cdot (\boldsymbol{x} - \boldsymbol{p}))^2}, \boldsymbol{n} \cdot (\boldsymbol{x} - \boldsymbol{p}) \right)$$

where $\boldsymbol{p}$ is the position of the oriented point, $\boldsymbol{n}$ is its normal, and $\boldsymbol{x}$ is the position of another point in the surface mesh. Then the SI is calculated as a 2D accumulator indexed by α and β. Finally, the set of SIs calculated for all points in the surface mesh of an object is used as the description of the object.

Textured Spin-Image (TSI) [14] is an extension of SI. TSIs are computed from a surface mesh, the points of which have luminance information. Therefore, they can take into account shape and texture co-occurrence patterns. In practice, a TSI is simply a stack of standard spin-images SI(l), where each layer $l \in [1, \ldots, L]$ corresponds to a given level of luminance. The dimension of a TSI is L times larger than that of a SI.

In the recognition process based on these descriptors, SIs/TSIs of randomly selected points in the query mesh are created, and then the nearest SIs/TSIs to these points are found from all SIs/TSIs of each database model. Although an efficient nearest neighbor search algorithm [15] is used in this step, the large number of points in a surface mesh means that the overall computation time is substantial.

3 Color-CHLAC features

The proposed Color-CHLAC features are an extension of the CHLAC features [16]. CHLAC features are integrals of the local autocorrelation of 3D voxel data. In our approach, Color-CHLAC features are computed by measuring the autocorrelation function of color 3D voxel data at specific points, represented by local patterns. In this extension, we innovated a new description of color voxel data. Each voxel has a 6-dimensional status which represents both occupation and color information, so that local descriptors can be represented by the co-occurrence of their shape and colors. Because the feature vector consists of the sums of each pattern, it is insensitive to minor changes, noise, and loss

of data. Moreover, these features enable partial matching of various sized models by choosing an appropriate integral interval.

3.1 Outline

The concept of Color-CHLAC feature extraction is illustrated in Fig. 1. A given element of the feature vector is calculated by summing the response of a particular local pattern over all the color voxels. Local patterns are expressed by the relative position of neighboring voxels, for example, two voxels in a row. By summing the response of each pattern, we obtain a rough measure of some property of an object's surface. The voxels of the local patterns are also differentiated by color, so that information about texture patterns can also be obtained. Because of this color sensitivity, the features can be used to discriminate between objects with the same shape, if their textures differ.

3.2 Method for creating color 3D voxel data

To extract Color-CHLAC features, color voxel data corresponding to the 3D model must be found. The simplest way of creating voxel data from a measured point cloud is to divide the 3D space at regular intervals and judge whether or not a given voxel includes a measured point. However, voxel data created in this approach has many holes on view direction because measurement points tend to cluster on the object plane perpendicular to the view direction. In this paper, we first create a surface mesh from a point cloud and then transform the mesh into dense voxel data.

Figure 2 illustrates the method for transforming the surface mesh into voxel data. First, the 3D space is divided into sufficiently small intervals, e.g. 1 mm × 1 mm × 1 mm. Letting A, B, and C be the vertices of a mesh triangle, the collision of each voxel and the line AB is detected, and a voxel is marked as "occupied" if a collision occurs. Next, lines are drawn from the voxel collision points on line AB to the line AC (these lines are parallel to BC). Then the collision of these lines and each voxel is detected and voxels are marked as occupied when appropriate. This process is repeated for all triangles in the surface mesh.

Finally, voxel data are resized to a proper resolution. For example, to transform voxel data of size 1 mm × 1 mm × 1 mm to voxel data of size 4 mm × 4 mm × 4 mm, 4 × 4 × 4 voxels are joined together in one voxel. In this work, each voxel has R, G, and B color values. When resizing, the color values of each voxel in the output, voxel data are computed by averaging over the corresponding region of the input voxel data.

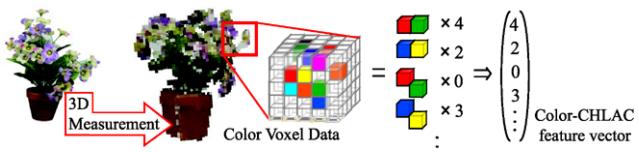

Fig. 1 Illustration of Color-CHLAC features. The resulting feature vector consists of the sums of local patterns of color voxel data

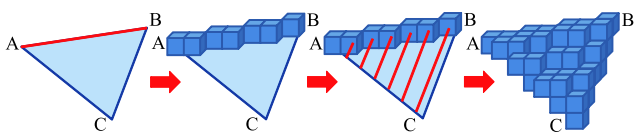

Fig. 2 Illustration of transformation from mesh data into voxel data. See text for details

3.3 Color-CHLAC feature extraction

Letting $\boldsymbol{x} = (x, y, z)^{\mathrm{T}}$ be the position of a voxel, we use the notation $p(\boldsymbol{x}) = 1$ if the voxel is occupied, and $p(\boldsymbol{x}) = 0$ otherwise. When $p(\boldsymbol{x}) = 1$, the voxel has RGB color values. We represent these, normalized between 0 and 1, as $r(\boldsymbol{x})$, $g(\boldsymbol{x})$, and $b(\boldsymbol{x})$, respectively. By defining $r'(\boldsymbol{x}) \equiv 1 - r(\boldsymbol{x})$, $g'(\boldsymbol{x}) \equiv 1 - g(\boldsymbol{x})$ and $b'(\boldsymbol{x}) \equiv 1 - b(\boldsymbol{x})$, the voxel status $\boldsymbol{f}(\boldsymbol{x}) \in R^6$ is defined as follows:

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{cases} (r(\boldsymbol{x})\ r'(\boldsymbol{x})\ g(\boldsymbol{x})\ g'(\boldsymbol{x})\ b(\boldsymbol{x})\ b'(\boldsymbol{x}))^{\mathrm{T}} & (p(\boldsymbol{x}) = 1) \\ (0\,0\,0\,0\,0\,0)^{\mathrm{T}} & (p(\boldsymbol{x}) = 0) \end{cases}$$

During pre-processing for feature extraction, $r(\boldsymbol{x})$, $g(\boldsymbol{x})$, and $b(\boldsymbol{x})$ can be binarized. If these are binarized, the resulting voxel status $\boldsymbol{f}(\boldsymbol{x})$ can be categorized into 9 patterns as shown in Fig. 3. Since $\boldsymbol{f}(\boldsymbol{x})$ contains not only RGB values but also their reverse values, the function is able to differentiate between low RGB value voxels and empty voxels.

Color-CHLAC features are integral to $\boldsymbol{f}(\boldsymbol{x})$ or correlations of $\boldsymbol{f}(\boldsymbol{x})$ between neighboring voxels and are calculated according to the following equations:

$$z = \sum \boldsymbol{f}(\boldsymbol{x}), \tag{1}$$

$$z(\boldsymbol{a}) = \sum \boldsymbol{f}(\boldsymbol{x})\boldsymbol{f}^{\mathrm{T}}(\boldsymbol{x} + \boldsymbol{a}) \tag{2}$$

The dimension of Color-CHLAC features calculated by (1) is 6. 14 patterns are used for the displacement vectors $\boldsymbol{a}$ in (2) (Fig. 4). Note that not only is the $\boldsymbol{f}(\boldsymbol{x})$ correlation between two neighboring voxels integrated, but also the correlation between two elements of $\boldsymbol{f}(\boldsymbol{x})$ of one voxel. Excluding redundant elements, the dimension of the Color-CHLAC

State Variable $f(x)$	Voxel State		State Variable $f(x)$	Voxel State	
$(0\,0\,0\,0\,0\,0)^T$	Empty		$(1\,0\,1\,0\,0\,1)^T$	Yellow	
$(0\,1\,0\,1\,0\,1)^T$	Black		$(0\,1\,1\,0\,1\,0)^T$	Cyan	
$(1\,0\,0\,1\,0\,1)^T$	Red		$(1\,0\,0\,1\,1\,0)^T$	Magenta	
$(0\,1\,1\,0\,0\,1)^T$	Green		$(1\,0\,1\,0\,1\,0)^T$	White	
$(0\,1\,0\,1\,1\,0)^T$	Blue		Others	× forbidden	

Fig. 3 Patterns of binarized color voxel status

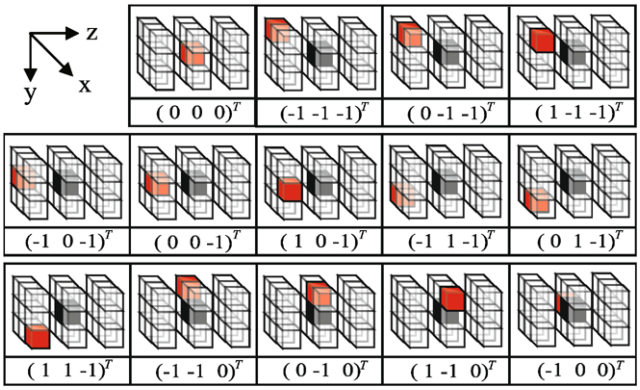

Fig. 4 Patterns of displacement vectors. Position of the center voxel is $\boldsymbol{x}$, while position of the other highlighted voxel is $\boldsymbol{x} + \boldsymbol{a}$

features calculated by (2) is 480, if color values are binarized, and 489 otherwise.

With color binarization, the patterns of neighboring voxels whose colors differ from one another are emphasized and correctly detected. Also, robustness to small changes in light intensity is achieved. On the other hand, if the colors of the target object are near the thresholds, the features can be sensitive to light variations. Moreover, continuous color values include richer information than binary color values. In this paper, we extract Color-CHLAC features from both binarized color voxel data and the original color voxel data. Then the dimension of the Color-CHLAC feature vector is 981 ($= 6 + 480 + 6 + 489$). We set the threshold of color binarization to 0.5.

3.4 Color-CHLAC feature compression

The dimension of a Color-CHLAC feature vector is 981, which is rather large. The greater the feature dimension the longer is the computational requirements for calculating similarity. In this paper, we compress Color-CHLAC feature vectors by Principal Component Analysis (PCA), using feature vectors extracted from all subdivisions of all database objects. This is the same idea used in SI [13] and TSI [14].

PCA is effective in this study because it is a linear transformation. Our method requires that the feature vectors have the additive property. Since PCA is a linear transformation, so the additive property is retained in the compressed feature vectors.

4 Matching method

4.1 Color-CHLAC features cache

It is important for 3D features to enable matching of partial models of any size in real 3D object retrieval. It is also essential that the time required for the matching process

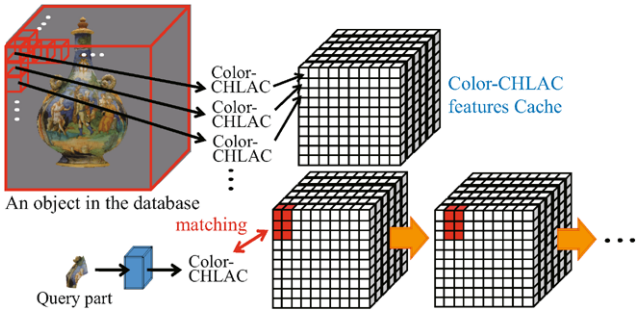

Fig. 5 Illustration of matching method using the Color-CHLAC features cache. See text for details

be short. By choosing an appropriate integration interval, Color-CHLAC features can be calculated from any region in each 3D model. Furthermore, the full feature vector of a partial model can be calculated simply by summing the feature vectors of its sub-parts. Another merit of Color-CHLAC features for partial matching is that they are translation invariant. Consequently, the feature vector is insensitive to small linear position changes, and thus fast rough scanning should not degrade the results too much.

The illustration of the proposed matching method is shown in Fig. 5. As a pre-processing step for object retrieval, we calculate and store the Color-CHLAC features of subdivided parts of models in a database. For example, suppose that the color voxel data of every object is a $300 \times 300 \times 300$ grid and Color-CHLAC features are computed from subdivisions consisting of $30 \times 30 \times 30$ voxels. Then at most, $10 \times 10 \times 10$ Color-CHLAC feature vectors are obtained. In the retrieval process, given a query part of a certain size, e.g. $40 \times 20 \times 80$ voxels, we first choose an appropriate matching area size. This corresponds to the integral interval of Color-CHLAC feature computation in one operation of matching. The size of the matching area is chosen to be larger than that of the query part and an integral multiple of the size of the subdivisions of the database models. In this case, it is a $60 \times 30 \times 90$ grid. The similarity measure is simply the dot product of the query's Color-CHLAC feature vector and the corresponding feature vector of the given model with the same area. The matching process is fast because the Color-CHLAC features of each matching model area can be calculated by simply adding some of the pre-computed and cached feature vectors (in this case, 6 ($= 2 \times 1 \times 3$) vectors).

4.2 Summed-area tables

Since Color-CHLAC features are integral features, they can be computed in constant time regardless of the matching area size, using the Summed-Area Table (SAT) [17]. To deal with 3D voxel data, we extend the SAT from 2D to 3D. The original definition of the SAT is the sum of the image pixels of the upright rectangle stretching from the top

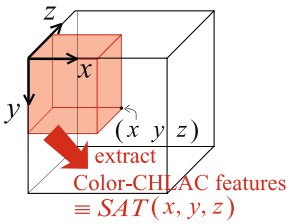

Fig. 6 Summed-Area Tables (SAT) in our implementation

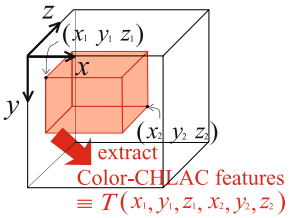

Fig. 7 Color-CHLAC features computation using the SAT

left corner to the bottom right corner. In our implementation, $SAT(x, y, z)$ is defined as the Color-CHLAC feature vector extracted from the voxel area ranging from $(0, 0, 0)$ to (x, y, z) (see Fig. 6). As shown in Fig. 7, let the Color-CHLAC feature vector of the voxel area with x ranging from x_1 to x_2, y ranging from y_1 to y_2, and z ranging from z_1 to z_2, be $T(x_1, y_1, z_1, x_2, y_2, z_2)$. This is computed by the following equation:

$$
\begin{aligned}
&T(x_1, y_1, z_1, x_2, y_2, z_2) \\
&\quad = SAT(x_2, y_2, z_2) - SAT(x_1, y_2, z_2) \\
&\qquad - SAT(x_2, y_1, z_2) - SAT(x_2, y_2, z_1) \\
&\qquad + SAT(x_1, y_1, z_2) + SAT(x_1, y_2, z_1) \\
&\qquad + SAT(x_2, y_1, z_1) - SAT(x_1, y_1, z_1)
\end{aligned}
$$

Using the SAT, $T(x_1, y_1, z_1, x_2, y_2, z_2)$ can always be computed by adding the 8 cached feature vectors, regardless of the size of the query part. Note that this is not effective when the number of subdivisions included in the matching area is smaller than 8.

4.3 Achieving robustness to rotation

In principle, any feature based matching scheme can fail if the feature vector produced by an object changes significantly when the object is rotated. There are two basic approaches to solving this problem. One is to use a rotation-invariant descriptor, and the other is to rotate objects before the matching process. We opt for the latter approach, repeating the matching process with various orientations of the query model. The feature vector of an object rotated 90 degrees can be obtained rapidly through a simple exchange of the elements of the feature vector in the initial orientation. This is possible because each displacement vector in Fig. 4 is equivalent to another, rotated 90 degrees. This method is used together with smaller rotations of the object (e.g. 30 and 60 degrees) to obtain a set of poses which are tested against the model database. To obtain tight matching, the step size of the rotation should be small, but this increases computation time. A good matching can be obtained with reasonable step sizes. In this work, we set the step size to 30 degrees, resulting in 504 different orientations.

Fig. 8 67 objects in the database

5 Retrieval with real color 3D models

We evaluated the performance of the proposed approach for the part-in-a-whole matching task using real color 3D models. First, we analyzed how the recognition accuracy and the computation time decreased when the subdivision-size parameter decreased. Second, we compared our method with conventional approaches using Spin-Image (SI) [13], Textured Spin-Image (TSI) [14] and CHLAC features [16]. Finally, we tested the retrieval performance of our method on a large database of 3D models.

5.1 Retrieval performance versus subdivision size

5.1.1 Setup

We used the "Telecom Paris Image-based Digitized 3D Models Archive" [18], a dataset of real textured 3D objects. The database consists of 67 complete 3D models shown in Fig. 8. Ten query parts (Fig. 9) were manually obtained from the models, randomly rotated in the range 0 to 359 degrees, and used as input to the matching procedure. We transformed each model into 1 mm × 1 mm × 1 mm color voxel data using the method described in Sect. 3.2. In this experiment, we used compressed Color-CHLAC feature vectors whose dimension was 25.

5.1.2 Results

In our method, the subdivision size should be small to obtain tight matching, but this increases computation time. Results

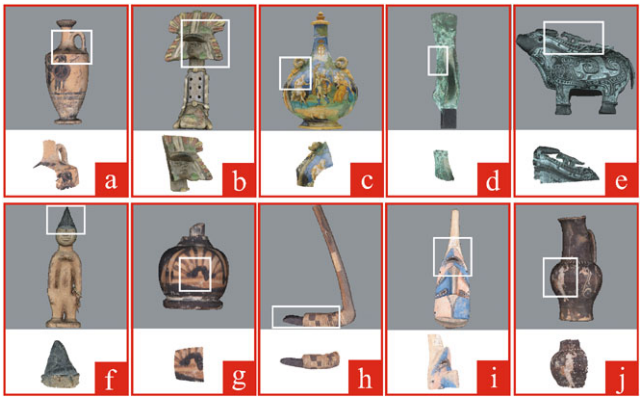

Fig. 9 10 query parts

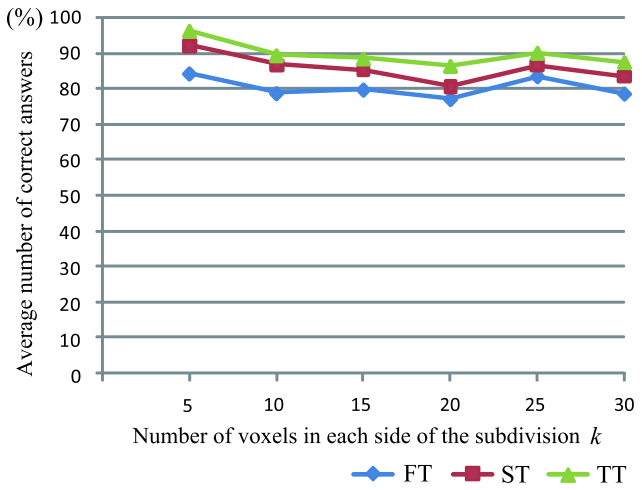

Fig. 10 Average correct rate versus subdivision size

of the average correct rate in 100 trials versus subdivision size are shown in Fig. 10. In each trial, the query part was randomly rotated before being input to the matching procedure. FT, ST, and TT represent first-tier, second-tier, and third-tier, respectively. First-tier is the percentage of trials in which the correct object is ranked first among the 11 database objects, whereas second-tier is the percentage where the correct object is ranked first or second, and third-tier is the percentage where the correct object is ranked first, second, or third. Let the number of voxels in each side of the subdivision be k. As shown in Fig. 10, an increase in the subdivision size did not bring about a corresponding decrease in the number of correct answers in this experiment.

On the other hand, matching time increased greatly with a decrease in subdivision size. Table 1 gives the time required for (I) computing the Color-CHLAC features of 504 different orientations of the query part and (II) matching the query part against all sub-parts of the 67 database models. The reported computation time was obtained using a C++ implementation executing on a Pentium D 3.2 GHz with 6.0 GB of main memory.

Table 1 Time taken for (I) feature extraction and (II) retrieval (sec). k is the number of voxels in each side of the subdivision

k	5	10	15	20	25	30
(I)	0.91	0.91	0.91	0.91	0.91	0.91
(II)	510	45	10.4	3.6	1.7	0.85

Fig. 11 Examples of first retrieved parts. Query parts are in the *top rectangles*, correct answers in the *middle rectangles*, and wrong answers in the *bottom rectangles*

Examples of the first retrieved model for a given query object are shown in Fig 11. Query parts are shown in the top rectangles, correct answers in the middle rectangles, and wrong answers in the bottom rectangles. In this case, the subdivision size was set to $30 \times 30 \times 30$ voxels. As shown in Fig. 11, the correct model was retrieved even if the exact same area was not matched against the query part. We observed that a query part with a strong color characteristic is correctly retrieved even if it was fairly small. On the other hand, mistakes sometimes occurred if there was another model in the database with a similar shape (see (h) in Fig. 11) or texture (see (i) in Fig. 11).

5.2 Comparison with other methods

In this section, we compared our Color-CHLAC features with Spin-Image (SI) [13], Textured Spin-Image (TSI) [14], and CHLAC features [16] from the viewpoint of noise resistance and computation time. We used the same database as that used in Sect. 5.1. Note that we tested CHLAC fea-

tures with the same partial-matching method as that used with Color-CHLAC features. For SI and TSI extraction, the resolution of each surface mesh was changed by mesh resampling [19] so that the average length of edges in the surface mesh was around 1 mm. For Color-CHLAC features and CHLAC features extraction on the other hand, we transformed each model into 1 mm × 1 mm × 1 mm voxel data (with or without colors) using the method described in Sect. 3.2.

5.2.1 SI and TSI parameters

Important parameters in SI generation are the *bin size*, *image width*, and *support angle*. Details of these are given in [13].

According to [13], choosing a bin size parameter equal to the mesh resolution creates a descriptive SI. We set the bin size to 0.9 mm, which was the average length of the edges in all the surface meshes of objects in the database. Image width was set to 15 and the support angle to 60 degrees, similar to the values used in [13]. The height of the SI was also set to 15 as in [13].

In the recognition step, a fraction of the oriented points are selected at random from each query part. Let the number of selected points be K and the number of points in the query part be N_q, we set $K = N_q/100$. Then K SIs are generated and matched against all SIs for each object in the database. To find the closest points, we used the efficient closest point search structure [15], in the same way as in [20]. In this approach, a reference point is a potential closest point only if it is less than a predetermined distance ϵ from the query point. Choosing a small ϵ allows faster lookup of closest points, but decreases the likelihood of finding the correct closest point. In this paper, we searched experimentally for a suitably small value for ϵ that does not substantially reduce the recognition rate, resulting in the value 10.

After the closest point search, the similarity between each point in the query part and the closest point in the ith object in the database is computed. The definition of the similarity measure between two SIs follows that given in [20]. In [20], the query part in a 3D scene and each object in a database are tightly matched by geometric matching using groups of point correspondences. However, unlike [20], the objective of our work is not to compute a transformation from model to scene, but to recognize objects in the scene fast enough for application in an online system. In this paper, we define the similarity between a query part and the ith object in the database, that is y_i, as the summation of the similarity measures between each point in the query part and the closest point in the i-th object. Let the number of SIs for the ith object be N_i, these SIs be $\boldsymbol{p}_n$, and the SIs generated from the query part be $\boldsymbol{q}_k$, then y_i is given by

$$y_i = \sum_{k=1}^{K} \max_{n=1,\ldots,N_i} \left((\operatorname{atanh}(R(\boldsymbol{p}_n, \boldsymbol{q}_k)))^2 - \lambda \left(\frac{1}{c_{nk} - 3} \right) \right),$$

Table 2 SI and TSI parameters

Parameter	SI	TSI
Bin size	0.9	0.9
Image width	15	15
Support angle [degree]	60	60
Number of points (K)	$N_q/100$	$N_q/100$
Search distance (ϵ)	10	10
Luminance level (L)	–	4
Compressed dimension (d)	25	25

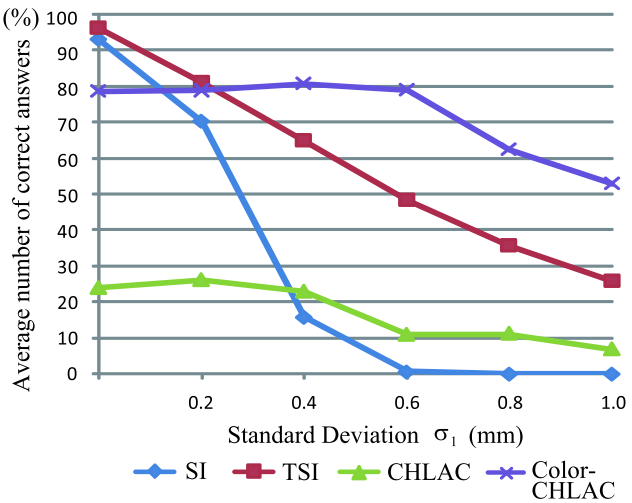

Fig. 12 Comparison of average correct rate (First-Tier) versus noise in coordinates of point clouds

where c_{nk} is the number of overlapping pixels used in the computation of correlation coefficient R. We set λ to 3, as is the case in [20].

According to [14], the value of the luminance level L in TSI should be small, ranging between 3 and 8. We set L to 4. The SI and TSI feature vectors are compressed using PCA. We set the dimension of the compressed feature vector d to 25. Note that d is set to 25, both in [13] and [14].

Table 2 gives the SI and TSI parameters decided above.

5.2.2 Results

A comparison of the average correct rate in 100 trials is shown in Figs. 12, 13, 14, 15, 16 and 17. FT, ST, and TT represent first-tier, second-tier, and third-tier, respectively. Where SI or TSI was used, matching was tested using a different choice for the K selected points as the query part. When using CHLAC features or our Color-CHLAC features, the query part was randomly rotated each time. Note that the subdivision size was set to 30 × 30 × 30 voxels for both CHLAC features and Color-CHLAC features.

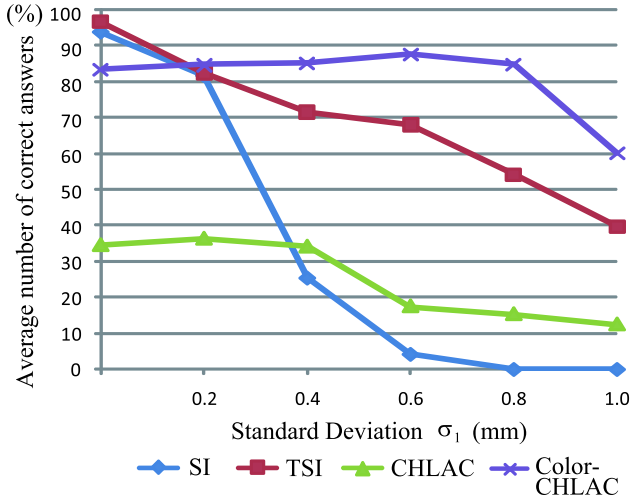

Fig. 13 Comparison of average correct rate (Second-Tier) versus noise in coordinates of point clouds

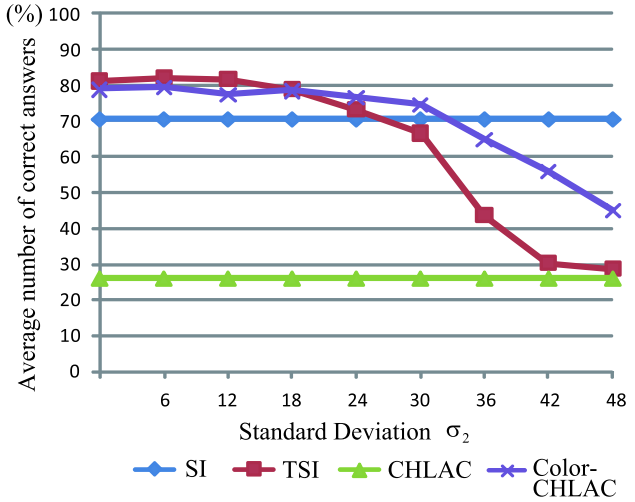

Fig. 15 Comparison of average correct rate (First-Tier) versus noise in the color of point clouds

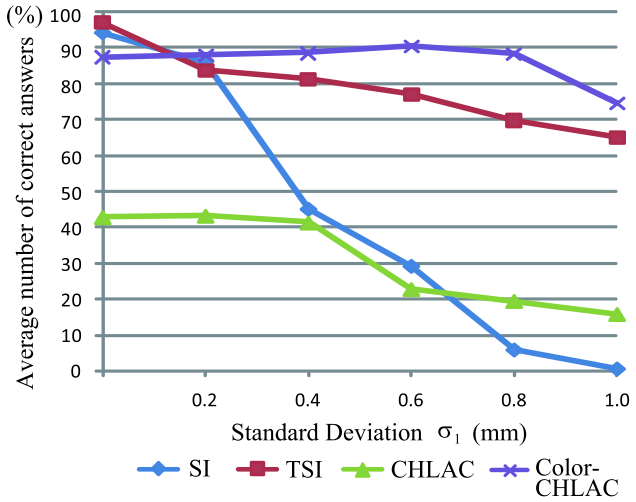

Fig. 14 Comparison of average correct rate (Third-Tier) versus noise in coordinates of point clouds

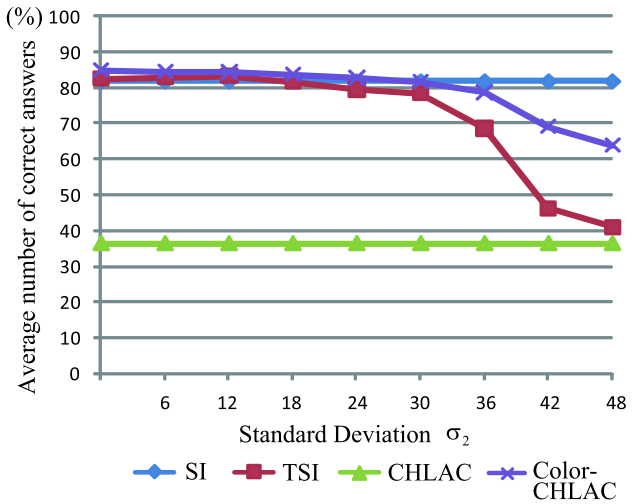

Fig. 16 Comparison of average correct rate (Second-Tier) versus noise in the color of point clouds

Figures 12, 13 and 14 show a comparison of the number of average correct answers versus the Gaussian noise in the coordinates of the query point clouds. Note that no noise was added to color values. The horizontal axis represents the standard deviation σ_1 (mm). As seen in Figs. 12, 13 and 14, SI and TSI were superior to the proposed method when no noise was added to the test data. This was because the changes in voxel data brought by the rotation of models affected the retrieval performance of our method. However, SI and TSI seem to be sensitive to noise, so that their retrieval performances were surpassed by that of the Color-CHLAC features when σ_1 was more than 0.2 mm.

Color-CHLAC features are robust to noise not only in the points' positions but also in their RGB values. Figures 15, 16 and 17 give a comparison of the average correct rate versus the Gaussian noise in the RGB color values for query point clouds. The horizontal axis represents standard deviation σ_2, and RGB values range between 0 and 256. Note that Gaussian noise with $\sigma_1 = 0.2$ mm is added to the coordinates. The retrieval performance of Color-CHLAC features were superior to that of TSI when σ_2 was greater than 12.

Table 3 gives the time required for (I) extracting features of a query part and (II) calculating all the similarities between the query part and the 67 objects in the database. CHLAC and Color-CHLAC features exhibited substantially faster performance than SI and TSI in the time required for (II). Since the time for (II) increases linearly as the number of objects in the database increases, the shortness of this computation time is important in dealing with a large database.

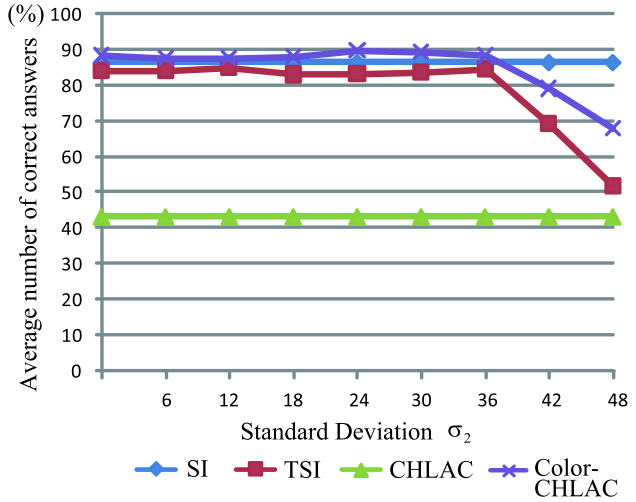

Fig. 17 Comparison of average correct rate (Third-Tier) versus noise in the color of point clouds

Table 3 Computation time (sec)

Feature	SI	TSI	CHLAC	Proposed
(I) Feature extraction	0.14	0.51	0.21	0.99
(II) Similarity computation	27	46	1.6	0.85

Let the number of subdivisions in the ith object in the database be M_i. The computation complexity of calculating the similarity between the query part and the ith object in the database is $O(dM_i)$ with the proposed method, and $O(dK \log_2 N_i)$ with the others. As shown in Sect. 5.1, when using our features, M_i can be reduced by increasing the subdivision size with almost no decrease in the correct rate. For SI or TSI on the other hand, the computation time of the similarity computation can be decreased by decreasing K, but this will adversely affect the average correct rate. In the case of using SI or TSI, the search distance parameter ϵ also affects the time required for the similarity computation. In short, the proposed method also has the advantage that choosing parameters is easier.

Regarding the pre-processing feature extraction of the database, our method was considerably faster than the others. The ith object in the database has N_i feature vectors when using SI or TSI, while it has M_i feature vectors in our approach. Since N_i is the number of points in the ith object's surface mesh, N_i can only be reduced by reducing the resolution of the surface mesh. In this experiment, the pre-processing feature extraction of the database took more than 11 hours when using SI, more than 16 hours when using TSI, and less than 2 minutes in our approach. Consequently, our method can be used when the database changes frequently, such as when it represents a 3D scene in an everyday environment.

Fig. 18 Some examples of voxel data in the large database

5.3 Retrieval performance on a large database

5.3.1 Setup

In this experiment, we evaluated the retrieval performance of our method on a larger database. We added 497 color 3D models downloaded from Archive3D.net[1] to the database used in Sects. 5.1 and 5.2. The models downloaded from Archive3D.net consist of 71 objects in "Home Appliances" category, 165 in "Kitchen Ware" category, 63 in "Toys and ChildRoom" category, 156 in "Decoration" category, and 42 in "House Plants" category. To align the size of objects in the database, we transformed each object into color voxel data so that the longest side of the bounding box might be 200 voxels. Some examples of voxel data in the database are shown in Fig. 18.

We created 564 query parts by clipping a rectangular solid from each database object. Let the ratio of the number of voxels in each query part to the number of voxels in the corresponding database model be α. We set α to 0.2, 0.4, 0.6, 0.8, and 1. Note that no noise was added to the color values or the coordinates of query parts.

Query parts were rotated by β degrees before being cut from database models. We tested 5 choices for β: (a) 0, (b) 10, (c) 30, (d) 45, and (e) a random number in the range 0 to 359. Also in this experiment the subdivision size was set to $30 \times 30 \times 30$ voxels.

5.3.2 Results

Average correct rate (FT, ST, and TT) versus α is shown in Figs. 19, 20 and 21. The performance in the case (a) was so high that FT was nearly 80% when $\alpha = 0.2$. On the other hand, the average correct rate decreased as the degrees of rotation of query parts increased, and the performance on (e) was from 20% to 30% lower than that on (a). This was because the topological changes in voxel data brought by the rotation of models affected the retrieval performance. In addition, since there were a lot of models with a single color

[1] http://archive3d.net.

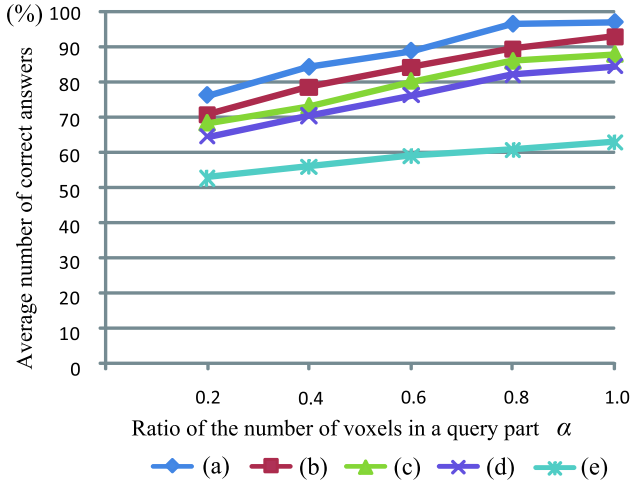

Fig. 19 Average correct rate (FT) versus α. The value of β is (*a*) 0, (*b*) 10, (*c*) 30, (*d*) 45, and (*e*) a random number in the range 0 to 359

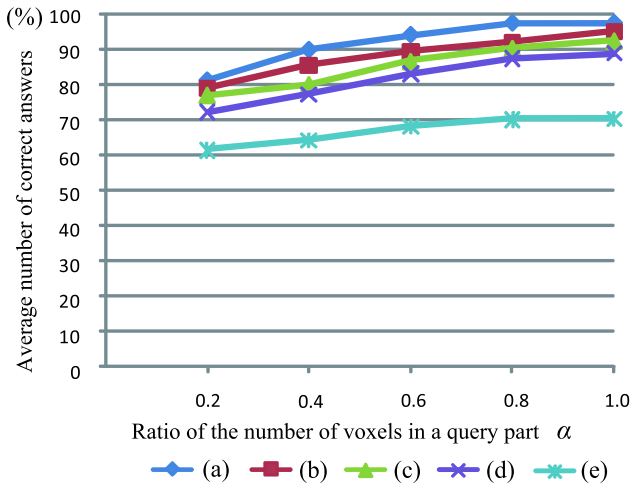

Fig. 20 Average correct rate (ST) versus α. The value of β is (*a*) 0, (*b*) 10, (*c*) 30, (*d*) 45, and (*e*) a random number in the range 0 to 359

(e.g. black or white) in the database, the topological changes in voxel data could make it difficult to distinguish such models.

Average computation time required for (I) extracting features of a query part, (II) calculating all the similarities between the query part and the 564 objects in the database, and the sum of the time for (I) and (II) versus α is shown in Fig. 22. When α increases, the time for (I) increases, while the time for (II) decreases because the total number of times of the matching between a query and a database model decreases. The sum of the time for (I) and (II) was shortest when $\alpha = 0.6$, which was 4.42 sec.

6 Search demonstration in 3D scene

In this section, we demonstrate the results of using the system to find a target object in our lab room. This system

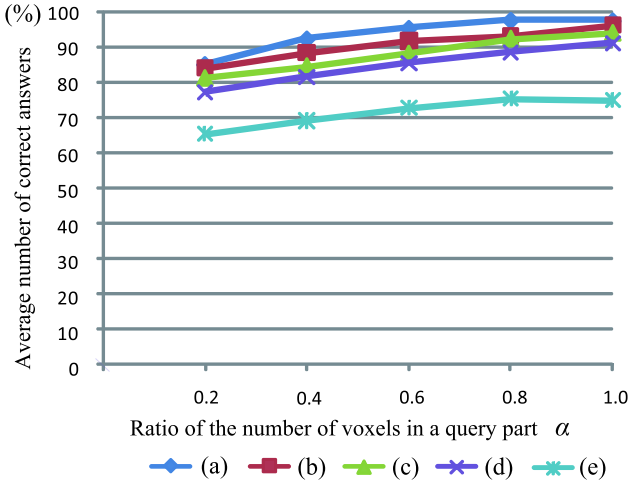

Fig. 21 Average correct rate (TT) versus α. The value of β is (*a*) 0, (*b*) 10, (*c*) 30, (*d*) 45, and (*e*) a random number in the range 0 to 359

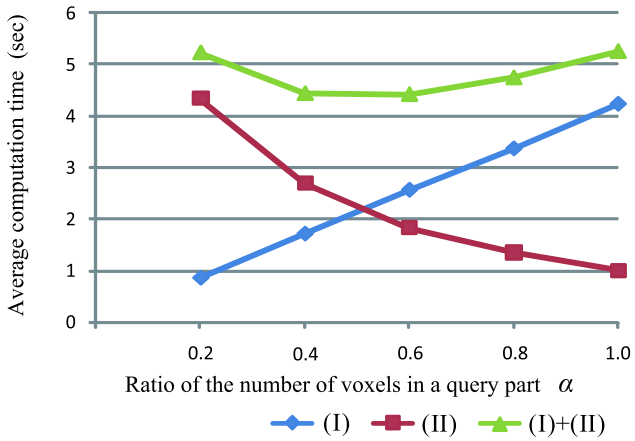

Fig. 22 Average computation time versus α. (I) Feature extraction of a query part and (II) similarity calculation between the query part and all the 564 objects in the database

aims at automatically searching for objects (e.g. those lost by someone) in a normal indoor environment.

6.1 Setup

In this experiment, we used an IEEE1394 camera and a infrared ranger (SwissRanger, SR3000)[2] to obtain color 3D voxel data (Fig. 23). We fixed the optical camera on the laser ranger, and used the overlapping area of the images acquired from both camera and laser ranger. The resolution of a range image obtained by the SR3000 is 144 × 176, which is rather low. Consequently, we set the size of a voxel to 10 mm × 10 mm × 10 mm in this experiment.

[2]MESA Imaging AG, http://www.mesa-imaging.ch/index.php.

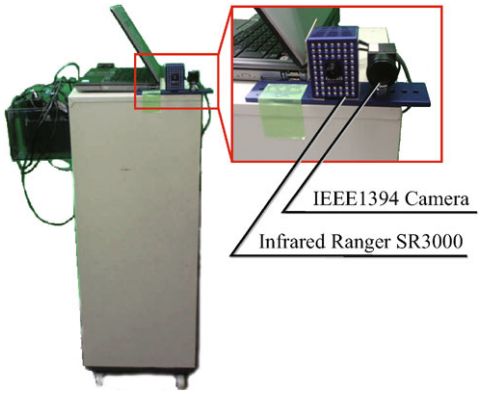

Fig. 23 Sensors on the cart used to measure the 3D scene

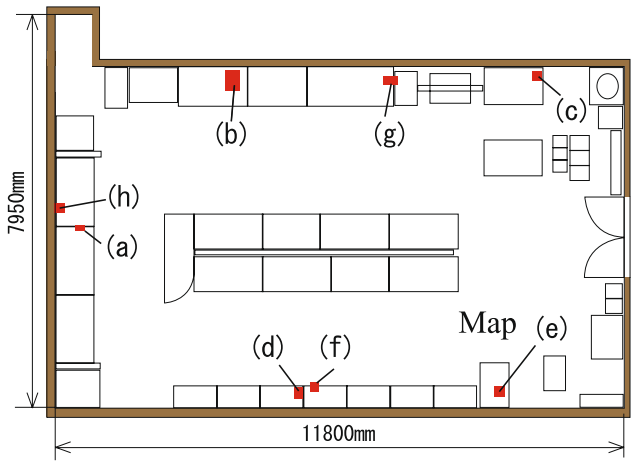

Fig. 24 The layout of the target 3D scene. The *highlighted rectangles* represent the areas where the target objects (Fig. 26 (a)–(h)) are

The target environment is our laboratory room whose size is 7950 mm (height) × 11800 mm (width) × 2700 mm (depth) (Fig. 24). We moved a cart (Fig. 23) manually around the room looking outward and obtained 45 color images (Fig. 25) and range images together. The obtained 3D data was aligned manually an d transformed into 867 × 1281 × 230 voxel data. We set the subdivision size of the voxel data to 10 × 10 × 10 grid, resulting in obtaining 87 × 128 × 23 subdivisions. There were 9345 subdivisions in the 3D scene which included one or more voxels, which was 3.6% of the total number of the subdivisions.

We put eight target objects (Fig. 26) on various locations in the room (Fig. 24). The matching area size of the doll (Fig. 26(h)) was the minimum of that of all the target objects, which was 20 × 10 × 10 grid, while that of the dolphin (Fig. 26(b)) was the maximum, which was 50 × 50 × 50 grid. Each object was measured from C directions, and then the scanned data was transformed into color voxel data, respectively. For each object the value of C was different, which was small for a monotonous-looking object (e.g. Fig. 26(g)), ranging from 1 to 8.

Fig. 25 Color images of the target 3D scene

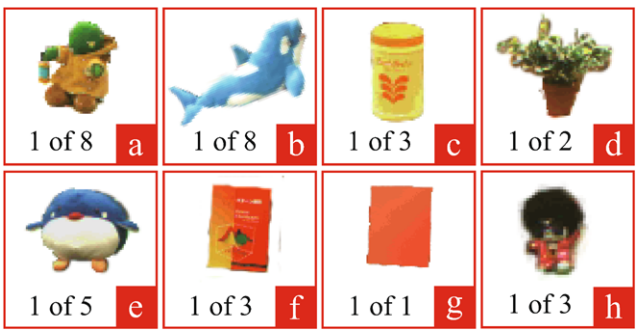

Fig. 26 Color images of the target objects

Object detection process consists of the following steps. First, the system performs partial matching between C query parts of a target object and the 3D scene of the room in 504 different orientations. Then it stores the maximum value of the similarities between the feature vector of each local area of the scene and 504 × C feature vectors of the target object. Finally, the system outputs three local areas, which have the largest similarity to the target object, the second largest similarity, and the third largest similarity, respectively. We tested 10 different values: 20, 40, . . . , 200 as the dimension of the compressed Color-CHLAC feature vectors.

Table 4 Object detection results. The trial with 120-dimensional features reported the best performance

dim	20	40	60	80	100	120	140	160	180	200
(a)	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}
(b)	1_{st}	1_{st}	1_{st}	×	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}
(c)	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}
(d)	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}
(e)	2_{nd}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}
(f)	2_{nd}	2_{nd}	2_{nd}	2_{nd}	2_{nd}	2_{nd}	3_{rd}	3_{rd}	3_{rd}	3_{rd}
(g)	×	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}	1_{st}
(h)	×	×	×	×	×	3_{rd}	3_{rd}	3_{rd}	3_{rd}	×

Table 5 Average computation time required for object detection (sec). (I) Feature extraction of 504 different orientations of a query part and (II) matching the query part against all local areas of the 3D scene

dim	20	40	60	80	100	120	140	160	180	200
(I)	0.06	0.07	0.09	0.10	0.11	0.13	0.15	0.16	0.20	0.22
(II)	5.2	5.6	5.9	6.5	6.8	7.2	7.5	7.9	8.2	8.6

6.2 Results

The object detection results are shown in Table 4. 1_{st} represents the case when the correct area was detected as the first-ranked area, 2_{nd} as the second-ranked area, 3_{rd} as the third-ranked area, and × represents the case when the object detection trial failed. The results vary depending on the dimension of feature vectors. The best results were reported when the dimension was 120. This indicates that setting the dimension to a proper value is important.

A search result example is shown in Fig. 27. The object-detected box area in the 3D scene is shown in the left upper rectangle, while a color image captured near the area is shown in the bottom. The matching reported the exact position where the query object (Fig. 26(e)) could be found.

Table 5 shows the average time required for (I) computing the Color-CHLAC features of 504 different orientations of a query part and (II) matching the query part against all local areas of the 3D scene. Note that the total time required for the detection of one object is approximately C times longer than (I)+(II). Since the average number of C was 4, the average computation time required for the detection of one object was 30 seconds when the dimension of feature vectors was 120.

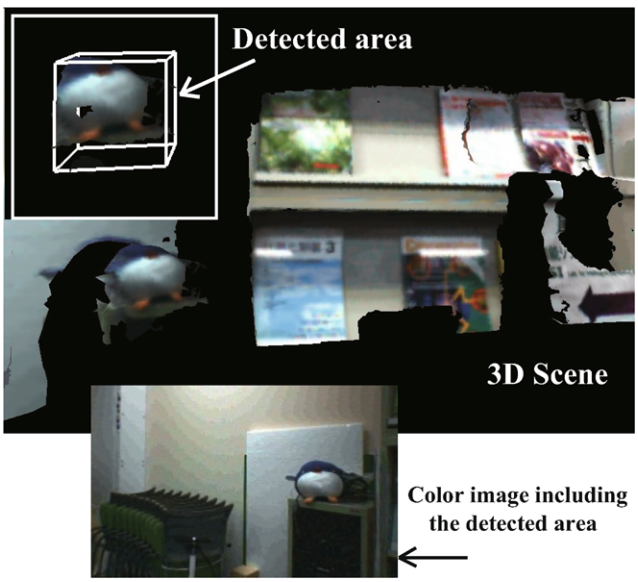

Fig. 27 A result example of the detection of (e) in Fig. 26. The detected area is shown in *the left upper rectangle*

7 Conclusion

In this paper, we proposed new features which describe the co-occurrence of shape and colors of an object's surface. These features enable efficient partial matching of real textured 3D objects.

The retrieval performance was evaluated with a database of real textured 3D models. Color-CHLAC features reported much higher accuracy than CHLAC features. Furthermore, considerably faster matching is possible using our method than using either Spin-Image or Textured Spin-Image. Our features are also robust to Gaussian noise both in the position of measured points and in their color values. However, smoothing of the query part's surface was not applied in our experiment, so there is the possibility that the retrieval performances of SI and TSI may be improved by smoothing. On the other hand, the results of the experiment in Sect. 5.3 indicate that the topological changes in voxel data brought

by the rotation of models affect the retrieval performance. It is our future work to achieve the robustness to the topological changes in voxel data.

Finally, we demonstrated the possibility of using the system to search for objects in the real world. It is expected that our proposed approach may contribute to real 3D object retrieval. In future work, we apply our system to searching for more varied objects in a large environment.

References

1. Bustos, B., Keim, D.A., Saupe, D., Schreck, T., Vranic, D.V.: Feature-based similarity search in 3D object databases. ACM Comput. Surv. **37**(4), 345–387 (2005)
2. Tangelder, J.W.H., Veltkamp, R.C.: A survey of content based 3D shape retrieval methods. In: Proceedings of IEEE Shape Modeling International, pp. 145–156 (2004)
3. Zaharia, T., Preteux, F.: Shape-based retrieval of 3D mesh models. In: Proceedings of ICME (2002)
4. Vranic, D.V., Saupe, D.: 3D shape descriptor based on 3D Fourier Transform. In: Proceedings of ECMCS, pp. 271–274 (2001)
5. Kazhdan, M., Funkhouser, T., Rusinkiewicz, S.: Rotation invariant spherical harmonic representation of 3D shape descriptors. In: Proceedings of Symp. on Geometry Processing (2003)
6. Biasotti, S., Marini, S., Spagnuolo, M., Falcidieno, B.: Sub-part correspondence by structural descriptors of 3D shapes. Comput. Aided Des. **38**(9), 1002–1019 (2006)
7. Mademlis, A., Daras, P., Tzovaras, D., Strintzis, M.G.: On 3D partial matching of meaningful parts. In: Proceedings of ICIP (2007)
8. Zhang, J., Siddiqi, K., Macrini, D., Shokoufandeh, A., Dickinson, S.J.: Retrieving articulated 3-D models using medial surfaces and their graph spectra. EMMCVPR, pp. 285–300 (2005)
9. Demirci, M.F., Shokoufandeh, A., Keselman, Y., Bretzner, L., Dickinson, S.J.: Object recognition as many-to-many feature matching. Int. J. Comput. Vis. **69**(2), 203–222 (2006)
10. Funkhouser, T., Shilane, P.: partial matching of 3D shapes with priority-driven search. In: Proceedings of Symposium on Geometry Processing (2006)
11. Ohbuchi, R., Osada, K., Furuya, T., Banno, T.: Salient local visual features for shape-based 3D model retrieval. In: Proceedings of IEEE Int. Conf. on Shape Modeling and Applications (2008)
12. Park, S., Guo, X., Shin, H., Qin, H.: Surface completion for shape and appearance. Vis. Comput. **22**(3), 168–180 (2006)
13. Johnson, A.E., Hebert, M.: Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans. Pattern Anal. Mach. Intell. **21**, 433–449 (1999)
14. Cortelazzo, G.M., Orio, N.: Retrieval of colored 3D models. In: Proceedings of Third Int. Sym. on 3DPVT (2006)
15. Nene, S., Nayar, S.: Closest point search in high dimensions. In: Proceedings of IEEE Conf. CVPR, pp. 859–865 (1996)
16. Kobayashi, T., Otsu, N.: Action and simultaneous multiple-person identification using cubic higher-order local auto-correlation. In: Proceedings of ICPR, vol. 4, pp. 741–744 (2004)
17. Crow, F.: Summed-area tables for texture mapping. In: Proceedings of SIGGRAPH, vol. 18, pp. 207–212 (1984)
18. Esteban, C.H., Schmitt, F.: Silhouette and stereo fusion for 3D object modeling. Comput. Vis. Image Underst. **96**(3), 367–392 (2004)
19. Johnson, A.E., Hebert, M.: Control of polygonal mesh resolution for 3-D computer vision. Graph. Models Image Process. **60**, 261–285 (1998)
20. Johnson, A.E., Hebert, M.: Surface matching for object recognition in complex three-dimensional scenes. Image Vis. Comput. **16**, 635–651 (1998)

Asako Kanezaki received the M.S. degree in Mechano-Informatics from the University of Tokyo, Japan in 2010. She is currently a Ph.D. candidate in the Graduate School of Information Science and Technology, the University of Tokyo. Her research interests include real world description, object recognition and retrieval, as well as applications.

Tatsuya Harada received the M.S. degree in mechanical engineering from the University of Tokyo, Japan in 1998 and the Ph.D. degree (2001) in mechanical engineering from the University of Tokyo, Japan. Dr. Harada was an assistant professor in the Intelligent Cooperative Systems laboratory of the University of Tokyo in 2001–2006. He is currently an associate professor at the University of Tokyo, Japan. His research interests include real world intelligent system, object and scene recognition, as well as data mining methods in time-series data.

Yasuo Kuniyoshi is a Professor at the Department of Mechano-Informatics, School of Information Science and Technology, The University of Tokyo, Japan. He received M. Eng. and Ph.D. degrees from the University of Tokyo, Japan, in 1988 and 1991, respectively. From 1991 to 2000, he was a Research Scientist and then a Senior Research Scientist at Electrotechnical Laboratory, AIST, MITI, Japan. From 1996 to 1997 he was a Visiting Scholar at MIT AI Lab. In 2001, he was appointed as an Associate Professor at the University of Tokyo. Since 2005, he is a Professor at the same university. His research interests include emergence and development of embodied cognition, humanoid robot intelligence, machine understanding of human actions and intentions. He published over 400 technical papers and received IJCAI 93 Outstanding Paper Award, Best Paper Awards from Robotics Society of Japan, Sato Memorial Award for Intelligent Robotics Research, Okawa Publications Prize, Gold Medal "Tokyo Techno-Forum21" Award, and other awards. For further information about his research, visit http://www.isi.imi.i.u-tokyo.ac.jp/.