3D Object Recognition Based on Description of Geometric Relations among Features Extracted from Surface Models

Makoto Maeda, Kousuke Kumamaru and Katsuhiro Inoue
Kyushu Institute of Technology, 680-4 Kawatsu, Iizuka, Fukuoka 820-8502, Japan

E-mail: {maeda, kumamaru, inoue}@ces.kyutech.ac.jp

Abstract

Geometric hashing method is well-known as one of techniques that realize 3D model matching procedure as quickly as possible. The matching process of the method uses hash-tables prepared off-line for the surface models of all 3D objects to be registered in a database. In order to index the model information into the hash-table, a hash function is required. It has to provide an index value that uniquely describes geometric relations among features extracted from the surface models. In this paper, two hash functions are proposed, in which they are defined by using several geometric features such as normal vectors and curvatures. Furthermore, in order to improve the efficiency of the matching process, a rehashing procedure is introduced so that the calculated indexes may be scattered within the hash table as uniformly as possible. Through experiments using several surface models, it has been verified the geometric hashing method introducing the novel hash functions and the rehashing procedure is effective in 3D object recognition.

1 Introduction

Object recognition is a central issue in computer vision. Especially, the object recognition that is achieved by matching an observed model in a scene with stored object models is called model-based object recognition.

In the technique, some geometric features such as feature points, surface curvatures and surface normals are extracted from each model, so that both the object model and the observed scene can be represented by sets of these features. Hence, the recognition of an object in a scene amounts to the discovery of a match between a subset of the scene features and a subset of the features of some object models stored in memory.

Geometric hashing method is well-known as one of techniques that realize such model matching procedure as quickly as possible [1, 2]. The matching process of the method uses hash-tables prepared off-line for the surface models of all 3D objects to be registered in a database.

In order to index the model information into the hash-table, a hash function is required. It has to provide an index value that uniquely describes geometric relations

among features extracted from the surface models. The performance of this description, however, depends on the detection accuracy of feature points, such as the positions of feature points and their number. Hence, it is desirable for the hash function to calculate the index value robustly, even though some unreliable features are obtained due to noise. Consequently, one of the central issues of the geometric hashing technique is how the hash function can stably calculate the index values that have reliable distinction between the different objects.

In this paper, two hash functions are proposed, in which they are defined by using several geometric features such as normal vectors and curvatures. Furthermore, in order to improve the efficiency of the matching process, a rehashing procedure is introduced so that the calculated indexes may be scattered within the hash table as uniformly as possible. Through experiments using several surface models, it has been verified the geometric hashing method introducing the novel hash functions and the rehashing procedure is effective in 3D object recognition.

2 Detection of Feature Points

2.1 Surface model and curvature estimation

Surface curvature has been widely utilized, since it provides a unique viewpoint invariant description of local surface shape[11]. Once a parametric surface $f(u, v)$ with coordinates u and v which has at least C^2 continuity is obtained, surface curvatures can be easily computed. Since the surface model utilized in this paper is defined by 4th order B-spline functions which have C^3 continuity, it is easily applied to the estimation of surface curvatures.

On the other hand, a stored object model needs to have the information on the entire circumferential shape of the object so that the matching may be performed even if the observed scene including a part of the object is given. Therefore the modeling process to obtain surface models from the entire circumferential range data of the object is required. However, it is not so easy to perform such a modeling, since the parametrization is generally complex [10, 12].

Our study has developed a modeling method to obtain the surface models of 3-D circumferential shape by using a segmentation method that divides the entire

circumferential range data into several significant domains [8, 9]. It is important to obtain surface models of 3-D circumferential shape, but here I limit the discussion to how the model-based object recognition can be realized, given such surface models.

2.2 Feature points

3-D object recognition system using geometric hashing has been developed by Mokhtarian et al. [2], in which local maxima of Gaussian and mean curvatures are chosen as feature points, since these curvatures are non-directional quantities and more intuitive. In our method, the local maxima of Gaussian curvatures are also chosen as feature points. Although there is still room for argument about what should be chosen as feature points, the discussion is limited to the case choosing the local maxima of Gaussian curvatures as feature points in this paper.

Fig. 1: Diagram of detection of feature points.

Let us consider $f_0, \dots, f_8$ are 9 points sampled from the surface model as shown in Fig.1 and Gaussian curvature at each point is computed. If the curvature at the point f_0 on the surface is higher than the ones at all its neighbors, f_0 is marked as a feature point.

However, it is not easy to estimate curvatures without being influenced by noise. As the resolution of surface model becomes high, the number of feature points tends to increase. The detection accuracy of feature points, such as the positions of feature points and their number, may affect the matching process.

3 Geometric Hashing Method

Geometric hashing method is well-known as one of techniques that realize model-based object recognition. Some geometric features such as feature points are extracted from the object, so that both the model object and the observed scene can be represented by sets of these features. Consequently, the recognition of an object in a scene amounts to the discovery of a match between a subset of the scene features and a subset of the features of some model objects.

An important point to emphasize in the geometric hashing technique is that the model information is indexed into a hash-table using minimal transformation

invariant features. The invariant features are calculated by a hash function so that they may reflect geometric relations among feature points. The matching algorithm employs the hash-tables prepared for model objects. Given feature points from a scene, one tries to match the measurements derived at the scene points to those memorized in the hash-table. It can be said that the geometric hashing method is the efficient technique to realize the matching based on feature points.

However, the performance of the method depends on the accuracy of the detection of feature points, such as the position of feature points and their number. Hence, it is desirable for the hash function to calculate the index value robustly, even though some unreliable features are obtained due to noise. Consequently, one of the central issues of the geometric hashing technique is how the hash function can stably calculate the index values that have reliable distinction between the different objects.

3.1 Hash functions

Two hash functions are proposed, in this paper, to describe geometric relations among features extracted from surface model. They are defined by using several geometric features such as normal vectors and curvatures.

Fig. 2: Diagram of features extracted from a surface model.

Fig.2 shows the diagram of features extracted from a surface model. Choose an arbitrary ordered triplet of non-collinear points A , B and C from feature points that are detected from the model, so that a triangle ABC is formed. κ_a , κ_b and κ_c denote the curvature values computed at the points A , B and C . n_a , n_b and n_c denote the unit normal vectors of the surface model at the points A , B and C , and the edge lengths AB , BC and AC as d_1 , d_2 and d_3 , respectively. Let κ_a and d_1 be maximum curvature value and edge length, then calculate the index value IV_1 or IV_2 to establish the hash-table as follows.

$$IV_1 = \left(\frac{d_2 + d_3}{d_1} \right) \left(\frac{c_1 + n_a \cdot n_b}{c_1 + n_b \cdot n_c} \right) \left(\frac{c_1 + n_a \cdot n_c}{c_1 + n_b \cdot n_c} \right),$$

$$IV_2 = \left(\frac{\text{sgn}(\kappa_b)}{\text{sgn}(\kappa_a)} \right) \left(\frac{\text{sgn}(\kappa_c)}{\text{sgn}(\kappa_a)} \right) \left(\frac{c_2 + |\kappa_b|}{c_2 + |\kappa_a|} \right) \left(\frac{c_2 + |\kappa_c|}{c_2 + |\kappa_a|} \right),$$

where c_1 and c_2 are arbitrary constants that satisfy $|c_1| > 1$ and $c_2 \neq 0$. The constants are added in order to avoid the problem of a zero divide. In this paper, since c_1 and c_2 are set to 2 and 1, respectively, it is clear that IV_1 is in $(1/9, 18]$ and IV_2 is in $[-1, 1]$. Consequently, the set of model information at the points A , B and C is indexed into the hash-table.

3.2 Rehashing procedure

Since the range of IV_1 and IV_2 are bounded as mentioned above, if given the specific number m of division, the range is uniformly divided into sets of small ranges r_k ($k = 1, \dots, m$). If the index value that is calculated for a triangle ABC is within the range r_k , the set of model information at the points A , B and C is indexed into the k -th record in a hash-table.

However, in the case that many index values are similar and indexed into same record, the matching process using such a hash-table is not efficient since there arise many collisions in searching. In order to deal with this issue, each range r_k ($k = 1, \dots, m'$) is appropriately adjusted according to the number of collisions. Hence, the calculated indexes may be scattered within the hash table as uniformly as possible.

3.3 Geometric hashing algorithm

In this section, the geometric hashing algorithm is briefly explained. First, in the registration stage (Stage-1), the model information of a 3D object is indexed into a hash-table using minimal transformation invariant features. This process is performed off-line for the surface models of all 3D objects, so that many hash-tables are obtained as a database. Next, given a scene, scene features are extracted from the local scene models. After that, the matching stage (Stage-2) tries to match the indexes calculated using the scene features to those memorised in the hash-tables.

3.3.1 Stage-1 : registration stage

Given the surface models of all 3D objects to be registered in a database, the process for indexing the model information into the hash-table is performed as follows.

- (Step-1) Extract local maxima of Gaussian curvatures from each 3D object model as feature points.
- (Step-2) Choose an arbitrary ordered triplet of non-collinear points A , B and C from the feature points detected in Step-1, so that a triangle ABC is formed. Calculate the index value IV_1 or IV_2 , so that the set of model information at the points A , B and C is indexed into a hash-table. As long as all possible triplets of feature points that are not collinear exist, the indexing procedure is repeated for different triplets of feature points. Consequently, a hash-table into that all

the model information is indexed is established for a 3D object.

- (Step-3) Perform the rehashing procedure so that the calculated indexes may be scattered within the hash table as uniformly as possible.

3.3.2 Stage-2 : matching stage

Given a scene that may include several objects, scene features are extracted from the local scene models. Notice that the input 3D objects can either be complete or incomplete. After that, the process for matching the indexes calculated using the scene features to those memorised in the hash-tables is performed as follows.

- (Step-4) Detect feature points by using the obtained scene features. Choose an arbitrary ordered triplet of non-collinear points A , B and C from the feature points, then calculate the index value IV_1 or IV_2 . After that, we try to match the index value as well as the shape of the triangle ABC to those memorised in the hash-tables that are established by Step-3. If the appropriate entry is found in the hash-tables, vote for the entry. As long as all possible triplets of feature points that are not collinear exist, repeat the matching process for different triplets of feature points.
- (Step-5) Finally, tally up the votes for each hash-table, and then we can decide that the target object is the 3D object corresponding to the hash-table that scores a large number of votes.

3.3.3 Stage-3 : verification stage

Several objects may score a large number of votes close to each other. The voting scheme may yield more than one candidate solution with very close scores. In this case, the most likely candidate will be chosen using an object localization applied in Step-6.

The object localization is the process of determining the position, orientation, and scale of a scene model. In order to localize model objects in a target scene, correspondences must be found between features in the object model and features in the scene model so that a transformation T can be computed which maps the object model to the scene model.

- (Step-6) Estimate the seven transformation parameters that define T . If the scene model is correctly recognized, the transformation parameters between the corresponding triplets of feature points can be similarly estimated, so that they form some large clusters in the parameter space. Consequently, the model objects with the largest clusters are then chosen as the most likely objects present in the scene.

4 Experimental results

This section presents some experiments that test the performance of our matching technique using the pro-

Fig. 3: Object models and the extracted feature points.

posed hash functions. All experiments were carried out by using a PC with PentiumD 2.1GHz CPU. There are 10 different object models which are constructed by NURBS surfaces in our database. In order to verify the robustness of the technique in the presence of noise, Gaussian noise was added to 10 % control points of each model. The standard deviation of the noise was set to 1% of the minimum length of each object.

Fig.3 shows feature points extracted from each surface model. The most feature points were extracted from Model05 and the number of them was 83. On the contrary the feature points extracted from Model02 were the least and the number of them was 16. Hence, hash-tables for Model05 and Model02 were created by using 91881 triangles and 560 triangles, respectively.

4.1 Experiment-1

The first experiment verified the effectiveness of the rehashing procedure. The experiment investigated how frequently the collisions of hash values computed by IV_1 in creating a hash-table for Model05 arise.

Fig.4 shows the histogram of the collisions of hash values. Fig.4(a) shows the histogram of the collisions of hash-table created by geometric hashing under the condition that the ranges for hash values are uniformly divided. On the contrary Fig.4(b) shows the histogram of the collisions after the rehashing procedure for the hash-table was adopted. From these figures, it turned out that the number of collisions can be reduced by the rehashing procedure. The matching process using hash-table to which the rehashing procedure was adopted was about 1.9 times faster. Therefore, the rehashing procedure can improve the efficiency of the matching process.

4.2 Experiment-2

In this experiment, the geometric hashing method using either IV_1 or IV_2 was applied to the input object, in

which arbitrary 3D transformations, such as 3D rotation and translation, were applied to the input object. Furthermore rehashing procedures were applied to the created hash-tables in order to reduce the collisions.

Note that if one of the models received a vote count that was substantially higher than the vote counts for all other objects, then it can be selected as the correct object. Otherwise, two or more models received high vote counts that were relatively similar. In this case, the verification approach using the object localization is applied only to the surviving models in order to select one of them.

Table 1 and Table 2 show the results of geometric hashing for the 3D transformed input objects using IV_1 and IV_2 , respectively. The numbers denote the percentages of triangles on each input object which received votes. The numbers in righthand column denote the computation time, but they are not CPU time.

Fig. 4: Histogram of the collisions of hash values.

Table 1: Results of geometric hashing using IV_1 .

Inputs/Models	M01	M02	M03	M04	M05	M06	M07	M08	M09	M10	Time(sec)
M01	100	10	5	0	0	1	3	1	27	0	12
M02	13	100	9	0	2	20	22	0	0	0	10
M03	0	1	100	2	5	11	4	0	0	0	138
M04	0	0	0	100	66	1	0	0	0	0	1416
M05	0	0	0	63	100	3	0	0	0	0	1900
M06	0	0	7	23	42	100	1	0	0	0	104
M07	1	8	17	0	3	18	100	0	1	0	30
M08	0	0	0	0	0	0	0	100	10	9	15
M09	15	0	1	0	1	2	2	3	100	6	39
M10	0	0	2	0	1	0	0	5	8	100	21

Table 2: Results of geometric hashing using IV_2 .

Inputs/Models	M01	M02	M03	M04	M05	M06	M07	M08	M09	M10	Time(sec)
M01	100	12	5	0	0	2	45	0	10	0	13
M02	8	100	11	0	0	0	3	0	0	0	17
M03	0	1	100	5	3	7	1	0	0	0	141
M04	0	0	0	100	64	1	0	0	0	0	2290
M05	0	0	0	65	100	2	0	0	0	0	2272
M06	0	0	5	17	23	100	1	0	0	0	98
M07	2	15	11	5	1	23	100	0	1	0	50
M08	0	0	0	0	0	0	0	100	6	12	12
M09	0	0	0	0	0	0	0	2	100	1	51
M10	0	0	1	0	0	0	0	14	32	100	17

These tables prove clearly that all objects can be recognised correctly, since one of the models received a vote count that was substantially higher than the vote counts for all other objects. Even if either IV_1 or IV_2 was used it turned out that they can be recognized in the almost same accuracy. However, the matching time based on the hash-table created by using IV_1 was 1.35 times faster than using IV_2 . Consequently, the hash function IV_1 is superior to describe geometric relations among shape features. Incidentally, the reason why the computation time of Model04 and Model05 is much longer than the others may be that there are too many feature points. Hence, in order to realize the matching efficiently, the technique that chooses significant feature points and preferentially uses them may be required.

Let us focus attention on the matching result (IV_1) for Model05 as the input model in Table 1. While Model05 received 100% votes, Model04 also received 63% votes. In this case, the verification approach using the object localization was applied only to Model04 and Model05.

Fig.5 shows the histogram of the estimated 3D transformation parameters. The parameters were estimated based on 57426 triangles matched between Model04 and Model05. From this figure, it can be seen that variance of each parameter distribution is large. In this case, since Model04 can be discarded as the incorrect object, Model05 can be recognized as the correct object.

Fig. 5: Histogram of the estimated 3D transformation parameters (IV_1 : Model05 $\rightarrow$ Model04).

Finally, we investigated if the object localization can be correctly determined. Model03 used as the input model was transformed as follows,

3D translation : (25[mm], 10[mm], 13[mm])

3D rotation : (45[deg], 0[deg], 60[deg]).

Fig.6 shows the histogram of the estimated 3D transformation parameters. The parameters were estimated

Fig. 6: Histogram of the estimated 3D transformation parameters (IV_1 : Model03 $\rightarrow$ Model03).

based on 7056 triangles matched between Model03 (input/transformed model) and Model03. From this figure, it turned out that the distribution of each parameter concentrates near each parameter of the ground-truth transformation. Consequently, we can determine the position, orientation, and scale of an input object.

5 Conclusion

In this paper, we proposed two hash functions defined by using several geometric features such as normal vectors and curvatures in order to describe geometric relations among features extracted from the surface models. Furthermore, in order to improve the efficiency of the matching process, a rehashing procedure was introduced so that the calculated indexes may be scattered within the hash table as uniformly as possible.

We demonstrated that the number of collisions can be reduced by the rehashing procedure. The matching process using hash-table to which the rehashing procedure was adopted was about 1.9 times faster than using the normal hash-table.

Through the experiment that the geometric hashing method was applied to 10 object models, it was confirmed that they can be recognized correctly in the almost same accuracy, even if either IV_1 or IV_2 was used. However, the matching time based on the hash-table created by using IV_1 was 1.35 times faster than using IV_2 . Consequently, the hash function IV_1 is superior to describe geometric relations among shape features.

On the other hand, it was confirmed that the computation time for the matching tends to become much longer in the case that there are too many feature points. Hence, in order to realize the matching efficiently, the technique that chooses significant feature points and preferentially uses them may be required. Furthermore, we plan additional experiments using various other models that are defined by polygon meshes.

References

- [1] Y. Lamdan and H. J. Wolfson, "Geometric hashing: a general and efficient model-based recognition scheme," *IEEE International Conference on Computer Vision*, pp. 238-249, 1988.
- [2] F. Mokhtarian, N. Khalili and P. Yuen, "Multi-scale free-form 3D object recognition using 3D models," *Image and Vision Computing*, Vol. 19, pp. 271-281, 2001.
- [3] T. Denton, A. Shokoufandeh, J. Novatnack and K. Nishino, "Canonical subsets of image features," *Computer Vision and Image Understanding*, Vol. 112, pp. 55-66, 2008.
- [4] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," *Proc. IEEE Int. Conf. on Computer Vision*, pp. 525-531, 2001.
- [5] G. Taubin, "Estimation of planar curves, surfaces, and nonplanar space curves, defined by implicit equations with applications to edge and range image segmentation," *IEEE Trans. PAMI*, Vol. 13, No. 11, pp. 1115-1138, 1991.
- [6] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with application in computer vision," *IEEE Trans. PAMI*, Vol. 21, No. 5, pp. 450-465, 1999.
- [7] Xinming Yu, T. D. Bui and A. Krzyżak, "Robust estimation for range image segmentation and recognition," *IEEE Trans. PAMI*, Vol. 16, No. 5, pp. 530-538, 1994.
- [8] M. Maeda, K. Kumamaru and K. Inoue, "Segmentation of Range Data Based on A Stochastic Clustering Method with Competitive Process," *Proc. Int. Conf. on Pattern Recognition*, pp. 624-627, 2004.
- [9] M. Maeda, K. Kumamaru and K. Inoue, "Shapes modeling of 3-D objects based on a hybrid representation using extended B-spline surface model," *Proc. Int. Conf. on Pattern Recognition*, pp. 656-659, 2002.
- [10] F. S. Cohen, W. Ibrahim and C. Pintaviroo, "Ordering and Parameterizing Scattered 3D Data for B-spline Surface Approximation," *IEEE Trans. PAMI*, Vol. 22, No. 6, pp. 642-648, 2000.
- [11] P. J. Besl, *Surfaces in Range Image Understanding*, Springer-Verlag, 1988.
- [12] D. Decarlo and D. Metaxas, "Shape evolution with structural and topological changes using blending," *IEEE Trans. PAMI*, Vol. 20, No. 11, pp. 1186-1205, 1998.