Research Article

Automatic Detection of Repetitive Components in 3D Mechanical Engineering Models

Laixiang Wen,1 Jinyuan Jia,2 Shuang Liang,2 and Jianhua Zhang3

1 School of Electronics and Information Engineering, Tongji University, Shanghai 201804, China
2 School of Software Engineering, Tongji University, Shanghai 201804, China
3 College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China

Correspondence should be addressed to Shuang Liang; shuangliang@tongji.edu.cn

Received 15 August 2013; Accepted 23 September 2013

Academic Editor: Shengyong Chen

Copyright © 2013 Laixiang Wen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We present an intelligent method to automatically detect repetitive components in 3D mechanical engineering models. In our work, a new Voxel-based Shape Descriptor (VSD) is proposed for effective matching, based on which a similarity function is defined. It uses the voxels intersecting with 3D outline of mechanical components as the feature descriptor. Because each mechanical component may have different poses, the alignment before the matching is needed. For the alignment, we adopt the genetic algorithm to search for optimal solution where the maximum global similarity is the objective. Two components are the same if the maximum global similarity is over a certain threshold. Note that the voxelization of component during feature extraction and the genetic algorithm for searching maximum global similarity are entirely implemented on GPU; the efficiency is improved significantly than with CPU. Experimental results show that our method is more effective and efficient than that existing methods for repetitive components detection.

1. Introduction

Pattern recognition plays an important role in artificial intelligence. In mechanical engineering, there are always repetitive patterns (components) in designs. The ability to detect such repetitive components automatically from the general input of 3D mechanical models is important for many applications in mechanical engineering. For example, finding out such repetitive components provides a natural way of model reduction [1, 2], and it facilitates intelligent editing of mechanical models, such as propagating edits to multiple instances of one mechanical component.

Our paper addresses the problem of detect repetitive components in 3D mechanical models, such as the wheel rims and the tires of a car, shown in Figure 1. This means we are looking for components in the model that can be reused under orthogonal mappings (rotation and mirroring) and translations to assemble the original model more compactly. Detecting such repetitive components in 3D models is a challenging work, since we typically have no prior knowledge of the size, shape, location, or orientation of the individual components. It is essential to normalize the pose of each component into a canonical coordinate system before the similarity measuring between each other. The pose alignment [3–5] is a series of geometry-invariant transformations, including translation, rotation, and scaling. After the pose alignment, the feature descriptors [5–11] are extracted to represent each component. The distance between the feature descriptors indicates the similarity between the two components.

Content-based 3D model retrieval [8, 12–15] also faces the same problem of pose alignment and similarity measuring. The most well-known approach to compute the pose alignment in 3D model retrieval is the Principal Component Analysis (PCA) method [9, 16], which is based on the computation of moments of the components. In most cases, 3D models are represented by geometric meshes and corresponding attributes which are used as input of PCA, such as vertices (CPCA) [1, 5], normals (NPCA) [5], and voxels (VPCA) [11]. However, PCA often faces severe inaccuracies regarding the alignment of similar models [10, 14]. Similarity measuring can be separated from pose alignment if it is based on PCA...
method. Feature descriptor used for similarity measuring differs from one method to another. Chen et al. [17] proposed the light field descriptor with good discriminative power but it has long extraction and matching time as a disadvantage, generated by 2D images-projections of the 3D model. Zhang et al. [11] use hybrid descriptors being extracted from these depth images to represent the 3D model. These feature descriptors usually are based on pose alignment which could be separated from similarity measuring. There are also some optimization methods [18, 19] which mix the pose alignment and the similarity measuring. They are dedicated to finding an optimal solution of the pose alignment under which the maximum similarity can be achieved. The pose is adjusted through the resulting similarity. These optimization methods outputs better results than PCA-based methods, but have higher time cost. To minimize the time cost, Martinek and Grosso [20] propose a GPU-based similarity measuring function using improved depth images to accelerate optimal pose searching.

The inaccuracy of PCA-based pose alignment affects severely the similarity measuring, while the time efficiency is the problem for optimization methods but it has better accuracy. Our aim is to detecting repetitive components in 3D mechanical engineering models, so that the accuracy is the highest priority with the consideration of time efficiency. In our work, a new Voxel-based Shape Descriptor (VSD) is proposed for the effective matching, based on which a similarity function is defined. It uses the voxels intersecting with 3D outline of mechanical components as the feature descriptor. Because each mechanical component may have different pose, the alignment before the matching is needed. For the alignment, we adopt the Genetic Algorithm (GA) to search the optimal solution where the maximum global similarity is the objective. If the maximum global similarity is over 0.99 allowing the error 0.01, we tell that the two components are same. Note that the voxelization of component during feature extraction and the GA for searching maximum global similarity are entirely implemented on GPU; the efficiency is improved significantly than on CPU. With GPU, the time efficiency of our method is very close to that of nonoptimization methods.

2. Related Work

2.1. Pose Alignment. Pose alignment of 3D models is used for preprocessing models before 3D shape matching and is commonly used in 3D model retrieval [8, 13, 21]. To conduct such alignment, a concatenation of isometrics in 3D space must be selected to determine the canonical coordinate system, and then the related model is aligned under it. The most well-known approach used to compute the canonical coordinate system of 3D models is the Principal Component Analysis method (PCA) [9], which is based on the computation of moments of 3D models.

After translating the 3D model according to its center of mass to the origin of the world coordinate system, three principal axes computed with PCA are used to determine the orientation. Some PCA-based methods, such as CPCA [9, 12] and NPCA [5], mainly focus on the different selection of samples; the samples of CPCA are the weighted vertices, whose precision is affected by the distribution of the vertex on the surface. While NPCA uses the area-weighted normal of each triangle as the sample, its precision depends on the distribution of the normal. Since PCA is a statistic method, its precision, regarding to the alignment of similar models with small difference [10, 14], is affected by the quality of the samples.

There are also other different solutions. First, symmetry-based approach, Chaouch and Verroust-Blondet [4], is based on two types of symmetries of the models: the reflective symmetry [22] and the local translational symmetry along a direction. Second, optimization-based approach, Kazhdan [18] decomposes the 3D optimization alignment problem into two subproblems of 2D and 1D optimization aligning respectively. Martinek and Grosso [20] introduced a GPU-based optimal 3D model alignment approach. It is similar to Kazhdan’s method, but a GPU-based similarity function using improved depth images to accelerate approximately optimal pose alignment searching and high efficiency is obtained. Third, projection area-based approach, Johan et al. [3] proposed an algorithm to find the alignment axes based on minimum projection area by noting that many objects have a minimum projection area when we orthogonally project them in the direction parallel to one of the axes of the ideal canonical coordinate frame.

2.2. Feature Descriptor. Feature descriptor is extremely important for model matching, model classification, model indexing, or model recognition. Because 3D space can be parameterized using spherical functions, Vranic and Saupe [12] proposed the ray-based descriptor which characterizes a 3D model by a spherical extent function capturing the furthest intersection points of the model’s surface with rays emanating from the origin. Spherical harmonics and moments representation of the spherical extent function were tested in [6], showing better performance than the former. Chen et al. [17] proposed the light field descriptor with good discriminative power but it has long extraction and matching time as a disadvantage, generated by 2D images-projections of the 3D model. In [23], the generalized radon transform is introduced, where the descriptor is extracted by combining the radial integration transform which captures the radial information of a 3D model and the spherical integration transform which captures its spherical information. Cornea
et al. [7] proposes the use of curve skeletons produced by application of the generalized distance field to the volumetric representation of a 3D model. The hybrid descriptor proposed by [5] is a combination of many different methods in order to achieve better performance. Zhang et al. [11] use hybrid descriptors being extracted from these depth images to represent the origin 3D model shape. More other methods were surveyed in [13].

3. Overview of the Proposed Approach

We proposed a high-efficient and accurate approach to detect the repetitive components. Matching method for 3D geometries is our main problem to be solved. Voxel-based Shape Descriptor (VSD) which represented each component is used for our similarity measuring. Pose alignment is the preprocessing for the similarity measuring. Our approach uses Genetic Algorithm (GA) to achieve the maximum similarity by searching optimal pose of two components. But the time cost is the fatal weakness of using genetic algorithms directly on CPU. GPU has the highly parallel architecture and can be used for our proposed VSD, shown in Figure 2 (central part). Limiting the searching domain is very significant to time efficiency and accuracy of the GA method. Section 4.1 describes the definition of the domain.

4. Similarity Measuring

4.1. Pose Definition. In order to find the best pose alignment between A and B, we let A reside in its original pose and assign a pose P only to B. First of all, both of the two components are normalized by their AABB bounding box. The quality of the pose alignment can be measured by means of the similarity function using VSD defined in the next section which we will also refered to as objective function.

The pose in 3D space can be composed of three parts: rotation, translation, and scale. Translation uses the center position of the 3D component as the reference point, rotation, translation, and scale. Translation uses the center of the translation and scale, respectively, are confined into a subdomain of the 3D space. For the translation, if B is moved out of the range of the AABB of A, the two components will never overlap. So we can limit the translation vector \( t \) by the domain \( x, y, z \in [-1.0, 1.0] \). The same to the scale, if \( B \) is too small or too large, the alignment result will obviously be the solution, so the domain of the scale can be confined to \( s \in [0.5, 2.0] \).

For the rotation, it can be represented by an orthogonal \( 3 \times 3 \) matrix which is not suitable for carrying out numerical optimization procedure. We parameterize the space of rotations according to Euler angles \( \theta \in [-0.5\pi, 0.5\pi] \) and \( \phi, \psi \in [-\pi, \pi] \), which are similar to the ways used in [18]. It expresses every rotation around the \( y \)-axis, \( r(\theta, \phi, \psi) \). Now we obtain the pose definition of the component in 3D space by \( P \):

\[
P = \{r, t, s\}
\]

\[
\begin{bmatrix}
\cos \phi \cos \theta + \sin \phi \sin \theta \sin \psi & \sin \phi \cos \theta - \sin \phi \cos \psi & \sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi & 0 \\
-\cos \phi \sin \theta + \sin \phi \cos \theta & \cos \phi \cos \theta & -\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi & 0 \\
\sin \phi \cos \theta & -\sin \theta & \cos \psi \sin \phi + \cos \phi \sin \theta \cos \phi & 0 \\
\end{bmatrix}
\begin{bmatrix}
\cos \phi \cos \theta + \sin \phi \sin \theta \sin \psi \\
-\cos \phi \sin \theta + \sin \phi \cos \theta \\
\sin \phi \cos \theta \\
\end{bmatrix}
\begin{bmatrix}
\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi \\
-\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi \\
\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi \\
\end{bmatrix}
\begin{bmatrix}
\sin \phi \cos \theta & -\sin \theta & \cos \phi \cos \theta & 0 \\
\end{bmatrix}
\begin{bmatrix}
\sin \phi \cos \theta & -\sin \phi \cos \psi & \sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi & 0 \\
\cos \phi \cos \theta & \cos \phi \cos \psi & -\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi & 0 \\
\sin \theta & -\sin \psi \cos \phi + \cos \psi \sin \theta \cos \phi & 0 \\
\end{bmatrix}
\begin{bmatrix}
t_x \\
t_y \\
t_z \\
s
\end{bmatrix}
\]
The goal of our approach is to find the pose alignment $P$ which applied to $B$ could get the maximum similarity.

4.2. Voxel-Based Shape Descriptor. Voxelization is a widely used technique to adjust the state of optimal solution of similarity using GA and capture the main feature of the 3D model. Many 3D model retrieval approaches [24, 25] use it as a basic way to extract the feature descriptor. For our algorithm, we propose a similarity function using VSD for adjusting population of poses, which is highly accurate and efficient due to the fact that we implement the method entirely on the GPU.

Initially, the smallest AABB circumscribing the component is constructed, and then both the model and this AABB are normalized into unit cube. Decompose this AABB into $N \times N \times N$ uniform grid of voxels and set each voxel with the initial state as transparent. Here, we use voxel$(x, y, z, s)$ to describe the state of one voxel, where $(x, y, z)$ is the spatial position of this voxel and $s$ is the scale size of it. If the initial voxel located at $(x, y, z)$ is regarded as an opaque voxel, if there are geometric primitives intersecting this voxel, there is a voxel $(x, y, z, 1) = 1$. Otherwise, it is regarded as transparent, voxel$(x, y, z, 1) = 0$. All the opaque voxels will be gathered to form a union to describe the shape of original 3D component and be used to perform the similarity measuring:

$$V = \{\text{voxel}(x, y, z, s) | \text{voxel}(x, y, z, s) = 1\}, \quad (2)$$

where $1 \leq x, y, z \leq N$. We denote the voxels collections of $A$ and $B$ as the sets $A_V$ and $B_V$, respectively. We define a voxel$(x, y, z)$ to belong to the union of $A_V$ and $B_V (A_V \cap B_V)$, only if this voxel that belonged to $A_V$ intersects with at least one voxel in $B_V$. Then the similarity measuring can be defined as

$$\text{similarity}(A, B) = \frac{|A_V \cap B_V|}{|A_V \cup B_V|}. \quad (3)$$

For each potential pose of $B$, the spatial position must be applied to $B$ and update the variables of similarity function. Our voxel-based similarity function does not depend on the rendering pipeline and could apply the pose to the voxels union quickly and independently. For the voxel voxel$(x, y, z) \in B_V$, if the current pose of $B$ is $P$, we should apply this pose to the voxel as follows:

$$\text{voxel}' = \text{voxel} \times P$$

$$= [x, y, z, 1]$$

$$\times \begin{bmatrix}
\cos \phi \cos \varphi + \sin \phi \sin \varphi \sin \theta & \sin \phi \cos \theta & -\sin \phi \cos \varphi + \cos \phi \sin \varphi \sin \theta & \sin \theta \\
-\cos \phi \sin \varphi + \sin \phi \sin \theta \cos \varphi & \cos \phi \cos \varphi & \sin \phi \sin \varphi + \cos \phi \sin \theta \cos \varphi & \cos \theta \\
\sin \phi \cos \theta & -\sin \phi \cos \varphi + \cos \phi \sin \theta \sin \varphi & \cos \phi \cos \varphi & \sin \phi \sin \varphi + \cos \phi \sin \theta \cos \varphi & \sin \theta + \cos \phi \cos \varphi \sin \theta \\
t_x & t_y & t_z & s
\end{bmatrix}, \quad (4)$$

where voxel$'$ is the new position of the voxel under the pose $P$ and $s$ is the size of the voxel. Since each voxel is a spatial axis-aligned cube with different size, the similarity computing become the intersection testing between these two voxels collections of the two objects. The whole voxels set of one object could also be divided into small parts to do the intersection testing independently, this makes our approach high parallelized and convenient to implement on GPU.

5. Optimal Solution of Similarity Using GA

The problem of finding the optimal pose alignment between two 3D components can now be formulated as a global optimization problem, where the task is to determine the pose $P$ to achieve the maximum similarity value in the limited 3D space. Blinded searching in the potential solution domain has a very low efficiency, even if the solution domain is limited. One more efficient way to do the heuristic searching is the optimal algorithms, so here we use the genetic algorithm.

5.1. Our GA on GPU. Genetic algorithm is a powerful, domain independent search technique inspired by Darwinian Theory. In general, GA employs selection, mutation, and crossover to generate new search points in a state space. A genetic algorithm starts with a set of individuals that forms a population. Usually, the initial population is generated randomly using a uniform distribution or some heuristic strategy. On every iteration of the algorithm, each individual is evaluated using the fitness function and the termination function is invoked to determine whether the termination criteria have been satisfied. The algorithm ends if acceptable solutions have reached. Otherwise, the individuals in the population are manipulated by applying different evolutionary operators such as mutation and crossover. The consecutive process of replacement forms a new population for the next generation.

Although GA is very effective in solving many practical optimal problems, its execution time can become a limiting factor for some huge problems due to a lot of candidate solutions. Fortunately, the most time-consuming fitness evaluation can be performed independently for each individual in the population using various types of parallelization [26]. In our work, we implement the GPU-based GA to do the optimal aligning pose finding. The structure of our GA mapping onto the GPGPU architecture is shown in Figure 3.

In the structure on GPU shown in Figure 3, each potential pose of component $B$ is the individual in the GA’s population.
For each block (block\_i) in GPU, it computes the similarity under one individual pose in the population independently. In each block, the similarity is divided into many subsimilarity computations (SC\_ij) mapping to each thread (T\_ij) in the block. All the threads are synchronized to compute each final similarity under the corresponding individual pose. The work of CPU performs the evolution process of GA to generate new generation of pose population, including selection, mutation, and crossover, according to similarities of last generation. For our problem, if the any similarity is over 0.99, the process ends. Otherwise, the maximum count of iteration is reached.

5.2 Heuristic Initial Population. In general, if the generations of the GA are enough, it could find the optimal solution in theory. But in order to reduce the iterations of generation, we could use some heuristic strategies to generate the better
initial population. If the initial population contains more solutions contributing potentially to the optimal solution, the GA searching will terminate more sooner. For the pose of $B$, there are some potential poses which may be the optimal pose:

1. The original pose of component $B$: the original pose of the component may be in accordance with the perceptive system of the human, so in the pose the two components may have the largest union. Consider the following:

$$ P = \{ r(0,0,0), t(0,0,0), s(1) \}; \quad (5) $$

2. The PCA pose of component $B$, transforming the pose of $B$ from the original pose of $B$ to the pose of $A$ using PCA method: PCA is a quick and efficient way to approximately align the components into a canonical coordinate system. It is usually the potential solution of pose alignment to get the maximum similarity. Consider the following:

$$ P = r_{\text{pca}}^A, p_{\text{pca}}^B; \quad (6) $$

3. The special rotation poses of component $B$, such as rotations around $x, y, z$ axes with angles, $0.5\pi, \pi, 2\pi$, and so forth: before the components are put into the database or used, many of their transformations are simply changed with the directions of the axes or special rotation angles. Hence, the poses to inverse these operations are gathered to form the thirdly initial poses. Consider the following:

$$ P = \{ r(\alpha, \beta, \gamma), t, s \}, \quad (7) $$

where $\alpha, \beta, \gamma \in \{\pi/6, \pi/4, \pi/3, \pi/2, \ldots, \pi\}$. These initial poses are combined randomly, inserted into the initial population of GA.

### 6. Experimental Results

To evaluate our proposed method, there are three aspects including VSD and its time efficiency, optimal pose alignment using GA method, and the ability to detect repetitive components in mechanical engineering models. The Princeton 3D Shape Benchmark database [27] is used for the first two tests, which is a model database dedicated to evaluate 3D model matching method. Note that one model in PSB stands for one component named in this paper. Models collected freely from the Internet are used for the third test that detect repetitive components. The hardware configuration we use is Intel Core 2 Duo 2.0 GHz, 1 G RAM, and Nvidia Quadro FX3800M.

#### 6.1. Our VSD and Time Efficiency

Since we use the VSD for similarity computation, the resolution of voxelization affects the accuracy of representation of a component. So what resolution we should adopt needs to be determined. And the time efficiency is also an important aspect of our optimization method using GA for optimal pose alignment. Under different resolution of voxelization, the time cost of execution on CPU and GPU, speedup of transplanting from CPU to GPU, and similarity value for each matching between two components are recorded. We choose 100 models randomly from PSB, and the average results are shown in Table 1. From the results, we can see that the average and the worst of similarity value tend to be stable when the resolution of voxelization exceeds 128, so we use resolution $128 \times 128 \times 128$ in the following experiments. One hundred models chosen randomly from PSB were used to compute the similarity between each other under different voxel resolution of the VSD. The low resolution means the 3D outline of component is represented by the coarse voxels. Along with raising the resolution, the VSD describes the 3D outline more and more accurate. So each comparison could reach its maximum similarity value after the resolution of VSD exceeds some value. And the average similarity value and the worst of all comparisons' results tend to be stable. That is the way we determine that resolution value $128 \times 128 \times 128$ is sufficient to the VSD to describe the component accurately. From Table 1, the average time cost of execution based on GPU is fairly small to be an optimization method using GA. And we can also find that it gains a high speed-up of transplanting from CPU to GPU.

#### 6.2. Evaluations on Our Optimal Pose Alignment

A comparison of our pose alignment is restricted to the most commonly used normalization technique, the PCA. Currently, there are two typical implementations of PCA, CP, and NPCA. The former uses the vertices of the 3D meshes to conduct the PCA. The latter uses the normal of each face or each vertex of 3D meshes to conduct the PCA. As to solving the PCA, the eigenvalues and eigenvectors of the covariance matrix of the samples, vertices (CP), or normals (NPCA) are needed to compute. Letting the eigenvalues be $\delta_1, \delta_2, \delta_3, \delta_1 \geq \delta_2 \geq \delta_3$, the corresponding eigenvectors to eigenvalues are $e_1, e_2, e_3$. Because three principal axes are not oriented, firstly, these eigenvalues are sorted by descending order to make it

| Resolution | CPU (sec) Average | GPU (sec) Average | Speedup (GPU/CPU) Average | Similarity (%) Average/worst |
|------------|------------------|------------------|--------------------------|-------------------------------|
| 32         | 71.179           | 0.470            | 151.582                  | 33.7/173.2                   |
| 64         | 289.718          | 1.051            | 275.705                  | 69.5/51.6                    |
| 128        | 679.578          | 1.785            | 380.686                  | 90.1/70.0                    |
| 256        | 2652.107         | 4.741            | 432.824                  | 90.1/70.0                    |
correspond to the $x$, $y$, and $z$ axes, and then, we compute the similarity values for the rest of the six permutations of the axes' orientation. Among them, we take the orientation which has the maximum similarity value as the estimated PCA alignment. The major drawback of PCA is that the axes react sensibly to small changes in the geometry. This happens when the model has many small decoration parts. Furthermore, when the PCA confronts the degenerated cases, like sphere and cylinder, three principal axes are not certain [1, 2]. The visual results of some examples using different pose alignment methods are shown in Figure 4. Obviously, our method can get better pose alignment than the PCA methods. And our method can be applied to not only the man-made objects but also the natural objects.

The PCA results are shown in Table 2. It explains the unsatisfactory worst-case results and shows that our approach could provide a better alignment. In order to test our approach more objectively, let the population size and the generation size be large enough so that we can get the optimal solution of pose alignment to achieve maximum similarity in theory. The resolution of the voxelization is $128 \times 128 \times 128$, the size of pose population is 1024, and the max iteration count of generation is 150. We select several typical classes of models from PSB to test our method. The PSB offers some standard class files for the researchers to benchmark their methods. From the standard class file, we choose part of classes for our test and each class contains certain number of models determined by PSB. The average similarity and the worst case on each class of models are used to indicate the accuracy of pose alignment between two models. The best case of similarity means align with itself, so its value is always 1.0. The average similarity indicates the overall effect of the pose alignment method. The worst case significantly indicates if the pose alignment method has a strong ability. The result data is shown in Table 2. From the average similarity of each alignment method, our method is more stable under different cases than the other two methods. The worst case of each method shows that our method can always get an optimal pose alignment.

### Table 2: Resulting data on several typical classes of models in PSB.

| Classes of models     | Model number | CPCA (%)          | NPCA (%)          | Our method (%)       |
|-----------------------|--------------|-------------------|-------------------|----------------------|
|                       |              | Best/average/worst| Best/average/worst| Best/average/worst   |
| Vehicle               | 70           | 1.0/94.7/66.0     | 1.0/95.9/67.4     | 1.0/98.1/75.3        |
| Airplane              | 99           | 1.0/71.7/11.9     | 1.0/89.6/54.3     | 1.0/97.5/90.1        |
| Liquid container      | 24           | 1.0/65.4/37.1     | 1.0/66.8/34.7     | 1.0/98.2/80.5        |
| Tables                | 35           | 1.0/80.2/13.4     | 1.0/81.8/14.7     | 1.0/94.3/87.5        |
| Animals               | 124          | 1.0/91.5/55.7     | 1.0/89.6/54.3     | 1.0/95.1/70.2        |
| Chairs                | 50           | 1.0/67.9/29.5     | 1.0/71.7/30.2     | 1.0/96.5/84.3        |
| Body part             | 55           | 1.0/51.5/32.7     | 1.0/54.5/33.9     | 1.0/81.4/71.1        |

6.3. Ability to Detect Repetitive Components. Table 3 shows experimental results to measure our method and the method proposed by Papadakis et al. [5]. The models used in the experiments and the corresponding repetitive components searched out are shown in Figure 5.

To detect the repetitive components, the similarity measuring is of utmost importance. We propose a method based on GA to adjust the pose, so that the global maximum similarity is obtained. According to the count of the components before and after removing repetition using the two algorithms shown in Table 3, ours is more able to detect repetitive components of a model than Cai’s. In other words, our method of similarity measuring is more accurate.
Table 3: Result data of detecting repetitive components using our method and Cai’s.

| Models  | Component count (original model) | Component count (ours) (removing repetition) | Component count (Cai’s) (removing repetition) |
|---------|---------------------------------|-----------------------------------------------|-----------------------------------------------|
| Car1    | 360                             | 145                                           | 236                                           |
| Car2    | 407                             | 208                                           | 300                                           |
| Motorbike | 487                           | 236                                           | 307                                           |
| Engine  | 189                             | 50                                            | 121                                           |
| Tank    | 1653                            | 199                                           | 280                                           |
| Robot   | 55                              | 30                                            | 43                                            |

There are two main reasons that why our method is more accurate than Cai’s.

1. Our optimal pose alignment method has better efficacy than the PCA methods. The optimal pose alignment is obtained only if the two components have the global maximum similarity. It is not influenced by the resolution and the vertex distribution of the two components, while the PCA methods are sensitive to them.

2. Our proposed feature descriptor VSD can be used to recognize the same components but different resolution, and can exactly represent the 3D outline of each component. While the vertex-to-vertex matching through nearest searching is used by Cai, it has a higher requirement of effect on pose alignment where the PCA methods usually fail when the two components are not exactly the same.

For the method for finding repetitive components, due to the elimination of redundancy, file size of model is obviously reduced and we can use lesser required components to restore the original model such that high compression ratio is obtained. It is worth noting that the compression ratio of a 3D model is determined by the number of repetitive components being found and whether these components contain a large scale of vertices and planes. Another application of this automatic detection of repetitive component may be the intelligent editing, and we believe that the potential applications of it are not confined to these.

7. Conclusions

In this paper, we introduced an automatic detection method of repetitive components in 3D mechanical engineering models but not limited to it. The global similarity between components is computed through a GA-based optimization method. It adjusts the pose to get an optimal pose to achieve the maximum global similarity. For similarity computation, a new VSD is proposed, using the 3D outline of components as the feature descriptor. To solve the time efficiency problem of the optimization method using GA, we implement on GPU for large data computation. And some heuristic strategies for initializing the population are also used to improve the quality of the results and to speedup the searching. The experimental results show that time efficiency, the effect of pose alignment, and the ability of repetitive components detection are all remarkably superior to existing methods. In the future, more accurate similarity computation will be researched, such as using the segmentation techniques to flag different parts of the 3D object with different weights. On the other hand, taking the advantage of GPU, more parallelized algorithms will be imported into the domain of mechanical engineering.
Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this article.

Acknowledgments

This research work was supported by the National Twelfth Five-year Plan Major Science and Technology Project of China (no. 2012BAC11B01-04-03). The authors also would like to thank the anonymous reviews for their valuable comments and suggestions.

References

[1] D. Shikhare, S. Bhakar, and S. Mudur, “Compression of large 3D engineering models using automatic discovery of repeating geometric features,” Signal Processing, vol. 19, no. 20, p. 15, 2001.

[2] K. Cai, W. Wang, Z. Chen, Q. Chen, and J. Teng, “Exploiting repeated patterns for efficient compression of massive models,” in Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry (VRCAI ’09), pp. 145–150, December 2009.

[3] H. Johan, B. Li, Y. Wei, and I. Iskandarsyah, “3D model alignment based on minimum projection area,” Visual Computer, vol. 27, no. 6–8, pp. 565–574, 2011.

[4] M. Chaouch and A. Verroust-Blondet, “Alignment of 3D models,” Graphical Models, vol. 71, no. 2, pp. 63–76, 2009.

[5] P. Papadakis, I. Pratikakis, I. Theoharis, G. Passalis, and S. Perantonis, “3D object retrieval using an efficient and compact hybrid shape descriptor,” in Proceedings of the Eurographics Workshop on 3D Object Retrieval, 2008.

[6] D. Saupe and D. V. Vranici, “3D model retrieval with spherical harmonics and moments,” in Pattern Recognition, pp. 392–397, Springer, New York, NY, USA, 2001.

[7] N. D. Cornea, M. Fatih Demirci, D. Silver, A. Shokoufandeh, S. J. Dickinson, and P. B. Kantor, “3D object retrieval using many-to-many matching of curve skeletons,” in Proceedings of the International Conference on Shape Modeling and Applications (SMI ’05), pp. 368–373, June 2005.

[8] Y.-B. Yang, H. Lin, and Q. Zhu, “Content-based 3D model retrieval: a survey,” Chinese Journal of Computers, vol. 27, no. 10, pp. 1297–1310, 2004.

[9] E. Paquet, M. Rioux, A. Murching, T. Naveen, and A. T. Ali Tabatabai, “Description of shape information for 2-D and 3-D objects,” Signal Processing, vol. 16, no. 1, pp. 103–122, 2000.

[10] B. Bustos, D. A. Keim, D. Saupe, T. Schreck, and D. V. Vranici, “Feature-based similarity search in 3D object databases,” ACM Computing Surveys, vol. 37, no. 4, pp. 345–387, 2005.

[11] Q. Zhang, J. Jia, and H. Li, “A GPU based 3D object retrieval approach using spatial shape information,” in Proceedings of the IEEE International Symposium on Multimedia (ISM ’10), pp. 212–219, Taichung, Taiwan, December 2010.

[12] D. V. Vranic and D. Saupe, “3D model retrieval,” in Proceedings of the Spring Conference on Computer Graphics and its Applications (SCCG ’00), 2000.

[13] J. W. H. Tangelder and R. C. Veltkamp, “A survey of content based 3D shape retrieval methods,” Multimedia Tools and Applications, vol. 39, no. 3, pp. 441–471, 2008.

[14] T. Funkhouser, P. Min, M. Kazhdan et al., “A search engine for 3D models,” ACM Transactions on Graphics, vol. 22, no. 1, pp. 83–105, 2003.

[15] K. Guo, W. Liu, and H. Song, “A novel mechanical component retrieval approach based on differential moment,” Advances in Mechanical Engineering, vol. 2013, Article ID 401846, 5 pages, 2013.

[16] D. V. Vranic, D. Saupe, and J. Richter, “Tools for 3D-object retrieval: karhunen-loeve transform and spherical harmonics,” in Proceedings of the IEEE fourth Workshop on Multimedia Signal Processing, pp. 293–298, Cannes, France, October 2001.

[17] D. Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouyoung, “On visual similarity based 3D model retrieval,” in Computer Graphics Forum, 2003.

[18] M. Kazhdan, “An approximate and efficient method for optimal rotation alignment of 3D models,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 7, pp. 1221–1229, 2007.

[19] M. Chaouch and A. Verroust-Blondet, “A novel method for alignment of 3D models,” in Proceedings of the IEEE International Conference on Shape Modeling and Applications (SMI ’08), pp. 187–195, Stony Brook, NY, USA, June 2008.

[20] M. Martinek and R. Grosso, “Optimal rotation alignment of 3D objects using a GPU-based similarity function,” Computers and Graphics (Pergamon), vol. 33, no. 3, pp. 291–298, 2009.

[21] D. V. Vranic, “3D model retrieval [Ph.D. thesis],” University of Leipzig, Leipzig, Germany, 2004.

[22] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser, “A planar-reflective symmetry transform for 3D shapes,” in Proceedings of the ACM (SIGGRAPH ’06), pp. 549–559, August 2006.

[23] P. Daras, D. Zarpalas, D. Tzovaras, and M. G. Strintzis, “Efficient 3-D model search and retrieval using generalized 3-D radon transforms,” IEEE Transactions on Multimedia, vol. 8, no. 1, pp. 101–114, 2006.

[24] J.-L. Shihi and H.-Y. Chen, “A 3D model retrieval approach using the interior and exterior 3D shape information,” Multimedia Tools and Applications, vol. 43, no. 1, pp. 45–62, 2009.

[25] P. Daras and A. Axenopoulos, “A 3D shape retrieval framework supporting multimodal queries,” International Journal of Computer Vision, vol. 89, no. 2–3, pp. 229–247, 2010.

[26] P. Pospichal, J. Jaros, and J. Schwarz, “Parallel genetic algorithm on the cuda architecture,” in Applications of Evolutionary Computation, pp. 442–451, Springer, New York, NY, USA, 2010.

[27] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, “The prince- ton shape benchmark,” in Proceedings of the Shape Modeling International (SMI ’04), pp. 167–178, June 2004.