3D Shape Matching Using Skeleton Graphs

Angela Brennecke Tobias Isenberg

Department of Simulation and Graphics
Otto-von-Guericke University of Magdeburg

Abstract

We describe a method to match 3D polygonal geometry models using their internal skeleton graphs. These graphs carry information about the overall shape of the model. In order to make assumptions about the similarity of the 3D models, we search for the biggest common subgraph. As internal skeleton graphs normally do not consist of many edges, the problem of exponential time complexity due to NP completeness does not pose a problem. To measure similarity, we calculate a scalar match quality value based mainly on the edge ratios. In addition, we present an empirical study in order to evaluate the approach by trying to match a number of example models. It demonstrates that the skeleton based matching algorithm allows us to effectively compare two meshes based on their overall shape.

Keywords: internal skeletons from progressive meshes, graph similarity, shape matching, polygonal models.

1 Introduction

In this paper we examine the suitability of internal skeleton extraction to determine the similarity of two polygonal meshes. We show how to compute a similarity measure for 3D polygonal geometry models by comparing their internal skeleton graphs. Such a similarity measure for shapes is useful, for example, for searching in 3D model databases. Our work is motivated by the observation that internal skeleton graphs carry information about the overall shape of a model. By comparing skeletons instead of meshes we may draw conclusions on the 3D shapes' similarity. Thus, not the whole polygonal model has to be used in the comparison.

To do the actual skeleton comparison, the amount of data is reduced by simplifying the skeleton graphs first. In a second step, the results are compared by applying a backtracking algorithm to them. Then, a similarity measure is computed depending on the found corresponding edges and nodes.

To demonstrate the feasibility of the approach, we present an empirical study using a number of exemplary models. This study confirms our assumptions. Models having a branched geometry are more advantageous than models having a compact geometry, as their internal skeletons correspond better to their external form. Nevertheless, even the results for those compact shapes back up our conjecture because we act on the assumption that internal skeletons represent the overall geometry only.

In the following Section 2, we first present an overview of related work in the field of similarity analysis of graphs and 3D objects in general. We then explain the basics of internal

skeleton extraction in Section 3 and discuss graph similarity in more detail in Section 4. Afterwards, we give the details of our own approach in Section 5 and explain the backtracking algorithm from which we derive the similarity measure. This is followed by a discussion of an empirical study and its results in Section 6. Finally, we conclude the paper and present ideas for future work in Section 7.

2 Related Work

The problem of determining the similarity of shapes in form of geometric models as well as the search in model databases has been studied intensively in recent years. There is a wide variety of different approaches to this problem. They will be reviewed below. However, since our approach is based on silhouette graphs, we first discuss a number of graph similarity measures.

In graph theory, *edit distances* are used for comparing graphs (see, e. g., [KKV90]). So-called *edit operations* are used to transform one graph $\mathcal{G}$ into a second graph $\mathcal{G}'$ by means of *edge delete* and *edge insert* operations as well as *node delete* and *node insert* operations. Then, the distance $\delta(\mathcal{G}, \mathcal{G}')$ between two graphs is characterized by the number of operations separating them (see [CKS98]). However, it can be shown that the transfer of $\mathcal{G}$ into $\mathcal{G}'$ by means of edit-operations is NP complete [KKV90]. In order to use this method nevertheless, ZHANG et al. [ZWS95, WZCS02] suggest applying edit operations only to those nodes whose degree does not exceed two. Their work was motivated by the need for efficiently comparing molecules in bio-chemistry. Another metric comes from BUNKE and SHEARER who developed a distance measure based on the maximal common subgraph [BS98].

Apart from shape matching algorithms based on comparing skeleton graphs, there are other approaches that are based on different concepts. In the field of object recognition, algorithms based on *geometric hashing* are employed more often. These algorithms are used to find objects in 2D or in 3D independent of affine transformations. This requires the preceding calculation of properties of the objects that are to be searched. These properties are stored in a hash table. The first algorithms based on this idea were developed by LAMDAN et al. [Lam89, LSW90]. An additional discussion of geometric hashing approaches can be found, e. g., in [HB94] and [WR97].

An obvious application of geometric hashing is shape matching. In this case, a feature vector that specifies important properties of the sought objects is used as a comparative structure [HKSV02]. It is important that the feature vectors are robust against modifications such as rotation, translation, scaling, and similar transformations. Properties that can be used in feature vectors are, e. g., surface points, number of edges and nodes, volume, or the curvature of an object. With these properties, relatively good results were achieved (see, for example, [HID95]). In addition, two classes of feature vectors can be distinguished: geometry-based and image-based feature vectors. The latter can be obtained, for example, by means of using different projections of the model. VRANIC and SAUPE worked on the development of such comparative criteria specifically in the context of multimedia-databases [VS00, VS01].

A different approach that employs *multi-resolution Reeb graphs* (MRGs) that are based on geodesic distance measurement—the distance between two points on the surface—was presented by HILAGA et al. [HSKK01]. The MRG method hierarchically constructs a Reeb graph for each model by a successive re-partitioning of regions. Hence, the MRG converges to the original Reeb graph. The major advantage of their *topology matching* method is that it is invariant to transformations such as rotation or translation. The MRG captures the important features of objects. Moreover, the similarity search can take place in the feature space. Therefore, the MRG can be used as a key for a geometric model database. Since this algorithm is fast and efficient it is specifically qualified for interactive requests. CHEN and OUHYOUNG show how to apply this MRG based shape matching to 3D object retrieval systems [CO02]. They discuss how to apply pre-processing, for example, to the handling of models with objects that have previously not been connected.

Another way of comparing 3D objects is by means of a form based approach. For this, the external form of the objects is examined. In this context, ANKERST et al. suggest the usage of shape histograms as intuitive feature vectors [AKKS99]. They decompose the 3D space into cells around the center of an object. The *shape histogram*'s values correspond to the number of object points in each cell. Then, a quadratic form distance function is applied in order to find the most similar objects. Their work also aims at developing suitable criteria for the search in molecule-databases.

OSADA et al. discuss a slightly different method to compute shape histograms called *shape distributions* that also allows for comparing 3D shapes [OFCD02]. Their method can deal with partially erroneous geometries which is very advantageous when dealing with arbitrary shapes. FUNKHOUSER et al. use a shape descriptor based on spherical harmonics [FMK+03]. Although it cannot be used for object reconstruction, it can be used to determine the distance between two shapes and is orientation invariant.

3 Internal Skeletons of Geometric Models

All of the previously discussed algorithms for shape matching use some kind of shape abstraction. Similar to these approaches, internal skeleton graphs are able to capture the overall form of objects. These graphs can be generated using an algorithm based on the progressive meshes [Hop96]. In this section, we briefly summarize the progressive mesh technique and discuss how it can be extended to generate internal skeletons of polygonal meshes. In addition, we evaluate the resulting skeletons in terms of the application for shape matching.

The progressive mesh representation is a concept for storing and transferring polygonal meshes. Typical methods for progressive coding of triangle meshes use, in general, the mesh optimization technique by HOPPE et al. [HDD+93, Hop96] that is based on the edge collapse operation. This algorithm successively selects edges from the mesh and collapses them. Thus, the complexity of the object's tessellation is successively reduced. The selection of edges to be collapsed is usually based on a criterion that changes the shape of the object as little as possible, e. g., by minimizing an energy function. The edge collapse process is stopped once a sufficiently small mesh has been obtained. This prevents the mesh

from degenerating. The progressive mesh consisting of the series of edge collapse steps and the coarse mesh can be decoded by applying the reverse operation—the vertex split—in reverse order [Hop96].
RAAB suggests using the edge collapse based simplification method to generate internal skeletons for polygonal meshes [Raa98]. Instead of stopping the algorithm before the mesh becomes degenerate, RAAB continues to apply edge collapse operations. The modified algorithm only stops for an edge if this edge is not attached to a valid polygon anymore. The remaining degenerate edges form the internal skeleton of the original polygonal mesh. Most significant of the properties of the edge collapse based skeleton is that the generated internal skeleton captures the overall form of a shape. It does not, however, preserve certain local features on the surface of objects. This, in turn, makes it ideal to be used in a shape matching application that aims for comparing the global shape of objects. In addition, graph based internal skeletons of three-dimensional shapes are well suited for representing tubular shapes. Internal skeleton extraction is not as well suited for shapes with a more compact or flat nature. For some applications, a potential problem with skeletons generated with the edge collapse based algorithm is that the skeleton edges are not necessarily located exactly in the middle of the original model part. However, this property is not relevant for skeleton based shape matching. Finally, in edge collapse based skeleton extraction algorithms the exact position of generated skeleton edges and the decomposition of the skeleton graph into skeleton edges is dependent on the order in which the edges are selected for being collapsed. Thus, it is dependent on the tessellation of the object. However, this potential problem can be avoided by applying pre-processing to the skeletons as described later on.

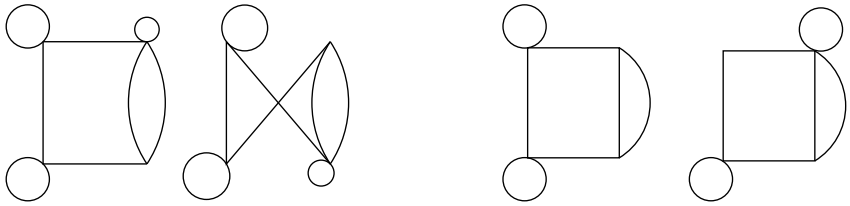
4 Graph Similarity

Based on the internal skeletons that are extracted using the method described above, we are now able to define a similarity measure for geometric models. Instead of comparing the meshes, we can only compare their skeletons. These capture the overall shape of the models. However, before explaining the details of this method, we first examine similarity measures for graphs.

4.1 Graphs

A graph $\mathcal{G}$ consists of a set of nodes $\mathcal{V}$, a set of edges $\mathcal{E}$, and a function ψ that connects every edge $e_i \in \mathcal{E}$ to the corresponding nodes $v_i, v_j \in \mathcal{V}$, more specifically $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \psi\}$. Moreover, the adjacency degree of a node $v_k \in \mathcal{V}$ is the number of edges that are connected by v_k with other nodes from $\mathcal{V}$.
In order to be able to make statements about the similarity of two graphs $\mathcal{G}$ and $\mathcal{G}'$, one can use the concepts of *homomorphism* and *isomorphism*. More specifically, it is necessary to know the features of these two mapping functions. If the mapping is surjective, which means an unambiguous mapping of the values of $\mathcal{G}$ into the amount of values of $\mathcal{G}'$, and if also $\tau(\psi(e)) = \psi'(\tau(e))$ is valid, $\mathcal{G}$ and $\mathcal{G}'$ are homomorphic. If τ is bijective they are isomorphic. This means for every value (node or edge) of $\mathcal{G}$ there must be a corresponding

value of $\mathcal{G}'$ and vice versa. Therefore, an isomorphism as well is an equivalence relation. Isomorphy is the mathematical concept of equivalence. Hence, two isomorphic graphs are equivalent to one another. To prove that two graphs are isomorphic is the mathematical answer to the shape matching problem. However, the proof of isomorphy is NP complete and known as the *isomorphy problem*. Therefore, isomorphy cannot be easily determined, except for specific cases.
In addition, there is another complication that occurs when considering shape similarity. Figure 1 shows sets of two more or less similar graphs. The first set of two graphs shown in Figure 1(a) are isomorphic. Due to transformations, they do not seem to be similar to a human observer. On the other hand, the two graphs in Figure 1(b) seem to be much more similar but are only homomorphic. This demonstrates that for shape similarity as perceived by a human observer one cannot solely rely on the mathematical proof of isomorphy as a similarity measure. Thus, it is necessary to find other means for determining similarity or difference of two graphs.



(a) Isomorphic graphs. (b) Homomorphic graphs.

Figure 1: Isomorphic and homomorphic graphs.

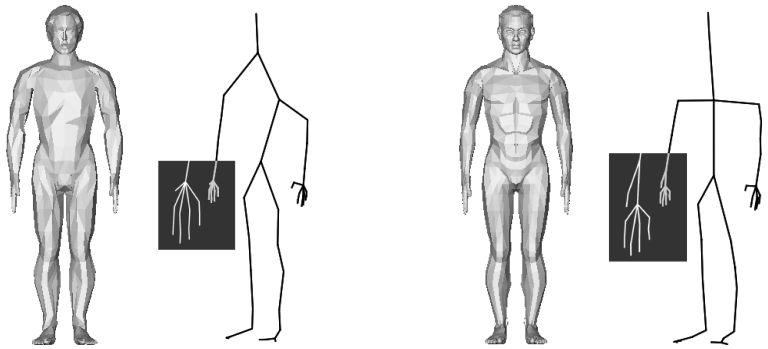
4.2 Similarity Definitions

Questions on similarity of graphs as subjectively perceived by a human observer usually cannot be answered unambiguously, since the transition is often fluid and complex. However, in order to still be able to make statements about the similarity of objects, it is essential to define these in some form.
When looking at the examples in Figures 2(a) and 2(b), one can observe that the skeletons of a sphere and of a cube are isomorphic. Although the skeleton graphs are mathematically equivalent, the source models differ significantly and are only similar with respect to their overall form. Thus, simple geometry model skeletons do not serve for a similarity definition, as they do not offer any information about their own differences, in most cases.
The proof of isomorphy and in particular the determination of similarity by means of edit distances is difficult with regard to complex graphs. Not only is the problem of giving the proof for isomorphy NP complete, but also finding the shortest chain of edit operations to determine the similarity of objects cannot efficiently be solved by an algorithm. For simple graphs, though, as they consist of a small number of edges only, even an exponential algorithm can be used.
As has been argued in Section 3, the skeleton extraction algorithm based on progressive

(a) Shape and skeleton of a sphere. (b) Shape and skeleton of a cube.

Figure 2: Comparing shape and skeleton of a sphere with those of a cube.

mesh simplification is sensitive to the particular tessellation of a mesh and is somewhat numerically instable. This may result in different skeletons being generated from visually very similar meshes (see example in Figure 3).



(a) Low resolution model. (b) High resolution model.

Figure 3: Model of a human and generated skeleton compared at different resolutions.

Hence, it is also important to integrate the global overall shape of the model in the similarity definition and examination. Therefore, for implementing skeleton based shape matching we combine certain features of the above mentioned similarity definitions into our algorithm. This will be introduced and discussed in detail in the following.

5 A Similarity Measure Based on Backtracking

The internal skeleton graphs computed according to RAAB have the property that even for finely tessellated meshes they usually do not consist of very many edges. In addition, they do not contain small branches caused by noise on the surface (e. g., tessellation artifacts in polygonal meshes). Instead, they represent a simplified global shape of an object and disregard surface details.

Therefore, based on the algorithm for generating internal skeletons from polygonal meshes and the previous considerations on graph similarity when applied to shape matching, we suggest finding the largest common subgraph of the skeleton graphs of the two models.

First, we look for corresponding edges and nodes similar to the procedure for proving isomorphy. In addition, we take the angle between edge pairs into account as well as the edge lengths and edge ratios. Moreover, we weight every edge's length of skeleton S_1 with the average edge length of skeleton S_2 and vice versa. This means that we can keep track of the model's global shape and can also compare models having different sizes.
Specifically, we propose the following heuristic method to compare two 3D meshes:

1. Compute the internal skeleton of both meshes as summarized in Section 3.
2. Simplify each skeleton by removing insignificant nodes of degree two.
3. Starting with one randomly selected node in each graph, do a backtracking search for the best common subgraph.
   (a) In each step, compute a similarity measure based on the size of the subgraph and the edge ratios.
   (b) Keep only the subgraph that represents the best match. If there are several equal similarity measures choose the one having the lowest average value of length and angle differences.

5.1 Skeleton Graph Simplifications

As previously noted by ZHANG et al. [ZWS95, WZCS02], nodes of degree two do not seem to play an important role when trying to match graphs because they are often only artifacts from skeleton extraction. In addition, in internal skeleton graphs generated by edge collapse operations the occurrence of such nodes is dependent on the tessellation of the original mesh and on the sequence of the edge collapse steps.
Therefore, all nodes of the degree two are examined more closely. On the one hand, very short edges (compared to the rest of the edges) at the ends of the graph are eliminated. They result most probably from tessellation artifacts rather than represent the overall shape of the object. Edges being removed connect a node with the degree two with a node with the degree one. In addition to this process, it is possible to fuse two edges into a common edge if the angle in between does not exceed a user-specified limit. These edges being originally split also results from the particular tessellation and does not represent the shape of objects. This skeleton graph simplification stage, done as a pre-processing step, has two important effects for the proposed backtracking algorithm. First of all, it makes it easier to find a common subgraph, if as in most cases only nodes of degree three and above have to be considered. In addition, the simplification of the skeleton graph leads to a smaller overall size of the graph, which, in turn, reduces the computation time that is needed for the backtracking algorithm.

5.2 Skeleton Comparison Using Backtracking

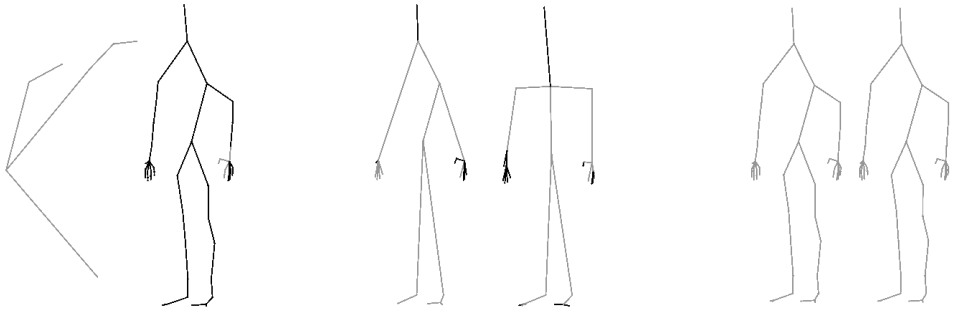
As laid out in Section 4.2, the comparison of the skeleton graphs must primarily account for the topological structure of the two graphs. Furthermore, additional methods have to be used to be able to make more explicit statements about the models to be examined. Therefore, first we create a comparison configuration by which the skeletons can be compared.

This comparison configuration comprises, e. g., the maximal value for differences in edge length as well as the maximal value for angle differences.
Moreover, the configuration of the comparison is adjustable interactively so that the combination of comparative criteria can be selected freely. For example, in a specific case only the degrees of nodes or only the lengths of edges may be used. This makes it possible to compare and evaluate specific configurations.
The backtracking algorithm proposed for comparing the two skeletons and finding a common subgraph naturally has exponential time complexity because of the NP completeness of the problem. However, as explained above, the size of the skeleton graphs is typically very small, in particular, after the simplification step. Therefore, the complexity of the matching algorithm does not pose a big problem for the execution time of the method.
Starting with a randomly selected node in each skeleton, the algorithm recursively compares the two graphs trying to match the skeletons. For each specific attempt to match two subgraphs, a scalar result that denotes the quality of the match is computed. Afterwards, the backtracking recursion is continued and a different path is chosen.
The scalar match quality is calculated in the interval between zero and one. Depending on the chosen configuration, the following factors are taken into account during this computation: the ratio of the found edges to all edges in a graph as well as the ratio of both skeleton graphs. The length, angle, and degree differences are used indirectly only. The best found pair of subgraphs is stored in a separate data structure. After each comparison, the newly computed match quality is compared to the one of the previous best match. If the new value indicates a better match, the previously best subgraph is replaced by the new one.
In order to compare the skeletons on the basis of the factors named above, the factors could now be weighted equally. However, this contradicts the objective of having a global similarity measure. For example, a subgraph consisting of seven out of ten edges but with 70% difference in regard to angle, length, and degree ratios, still is a better match than a subgraph consisting of two out of ten edges but with only 10% difference. Therefore, more emphasis has to be put on a good ratio of edges rather than on the other factors. The differences in length, position, and degree play a role, but these factors are mainly used in the search for edges. For most flexibility, the specific weights for each factor are left for the user to specify.

6 Empirical Study

In order to test the feasibility of the new skeleton based similarity measure we conducted a small empirical study with 13 different models. We found that our algorithm works well for models having a branched and tubular topology as well as for models with a complex shape. The comparison results comply with what we expected, namely that the algorithm yields similar subgraphs in accordance to the overall geometry.
Figure 4 shows the results of our comparison method for three selected model combinations. First, we have a combination of the skeletons of a bunny and a human in Figure 4(a). Our successful attempt to put emphasis on the edge ratios for global similarity can clearly be seen there. The bunny's skeleton is contained entirely in the common subgraph but only

roughly in the hand of the man's skeleton. This results in a very low match quality.
This also holds for the second Figure 4(b). Here, two simplified skeletons of different human models have been compared. The applied simplification preserves the overall geometry of the models and leads to a great number of similar edges. Based on that observation, the resulting match quality of 0.454816 may seem very low. However, experiments have shown that values larger than 0.5 denote two fairly similar models. The reason for low mesh quality in such cases lies in the fact that less significant parts of the skeleton are also considered in the calculation. These are not as important for representing the overall shape of the model but are still part of the skeleton, e. g., the parts of the graphs that represent the hands in this particular example.
Finally, Figure 4(c) shows the result of a comparison of identical graphs which, of course, leads to the resulting shape matching quality value of 1.0.



(a) Similarity 0.059311. (b) Similarity 0.454816. (c) Similarity 1.0.

Figure 4: Three matching results. The gray edges denote the common subgraph.

In all three comparisons we used the same specifications. In particular, the skeletons were not examined for differences in node degrees. However, the difference of the angles in between two edges had to be less or equal to 90 degrees and the difference of the edge lengths had to be less or equal to 50 length units. Experiments have shown that these values lead to good results when tested with several different models.
Figure 5 shows the results for the comparison of the 13 different models mentioned above. The configuration specification is nearly the same as above. The only difference is that we did not specify an edge length difference value. The gray values in the comparison matrix indicate the quality of the match. Black is a perfect match while white represents no match at all. The matrix illustrates the performance of our algorithm, working very well for the overall structure of the models. This is demonstrated, in particular, by four result groups that will be discussed in detail below.
The first three comparisons of tubular models result in fairly good matches. Even though the combination of the *David* model with the two preceding man models does not result in a very good match, these are the only significant matches for the *David* model. Except for the *Elephant*, all other model combinations lead basically to no match at all. The same is true for the two human models. The reason for the *Elephant* matching these models is its

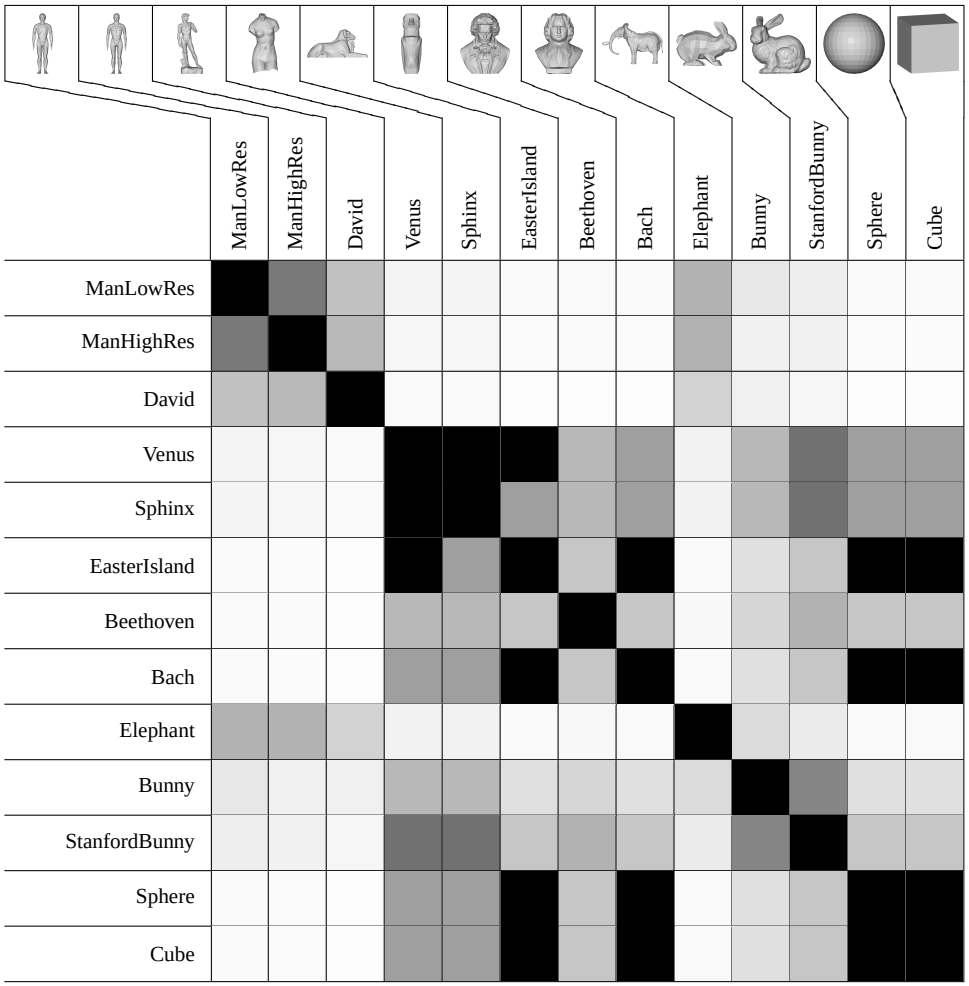


Figure 5: Comparison matrix for a number of example shapes. The gray value indicates the quality of the match, black is a perfect match and white no match at all.

similarly branched shape, as its skeleton also contains four extremities and a spine.
Another bundle of good matches can be found when looking at the *Venus*, *Sphinx*, *Easter-Island*, and *Bach* models. These all have a similar overall shape leading to a similar skeleton graph mostly consisting of one or two edges. However, even here difference nuances are recognizable. The *Venus*, *Sphinx*, and *EasterIsland* models have a long and thin overall shape that is clearly reflected in the matching results.
A third and fourth interesting result group is the two *Bunny* models as well as the *Cube* and *Sphere* combination. Both groups have results being close or equal to 1.0. They show on the one hand that our algorithm works well for complex overall shapes with small extremity features (e. g., the *Bunnies* and their ears). On the other hand, it also shows that we can

make assumptions about the overall form without being able to exactly elucidate its special features (e. g., *Cube* and *Sphere*).
As previously mentioned, the exponential character of the algorithm does not pose a problem because the skeletons are simplified and usually contain a few edges only. For the models used in this empirical study, the average number of edges of the simplified skeleton graph has been 11 although the models were fairly complex. Even though the most complex model (*David*) used in our study had approximately 30.000 triangles, its simplified skeleton only consisted of 37 edges. In addition, on an AMD Duron 600 MHz machine, one comparison with our algorithm required at most 70 milliseconds while the average comparison took only 4.2 milliseconds.

7 Conclusion and Future Work

The shape similarity measure presented in this paper is based on computing internal skeletons from polygonal meshes. Using these skeletons, their similarity is not only computed based on the mathematical equivalence of the generated skeleton graphs but also takes into account length ratios and angles between edges in the graph. In addition, pre-processing removes most of the instability inherent in the progressive mesh skeleton extraction by removing most insignificant nodes of degree two.
Using this new similarity measure, we achieved exactly what we expected. It is possible to compare three-dimensional computer graphic models on the basis of their skeleton graphs. Hence, assumptions about the similarity of the model's global shape can be made. However, the approach does not allow a comparison of local features of the models.
The presented algorithm could be further improved in the future by making use of, for example, a quadric error metric for computing the match quality instead of using length ratios that may lead to certain inaccuracies. In order to confirm that the algorithm works well, we also would like to apply it to a broader field of models, e. g., by using it in a 3D shape database. In addition, it would be interesting to consider the distance between the skeleton edge and the external model surface. This would increase the performance of the algorithm for simple geometry models. In this context, a threshold could be introduced to achieve more exact results.
Likewise, it would be possible to include image-space methods that could provide a preprocessing of the models. In order to achieve results that are similar not only in their external structure but also in their surface details, other information would have to be included. This could, for example, be achieved by using an external skeleton of the models.

References

[AKKS99] M. Ankerst, G. Kastenmüller, H.-P. Kriegel, and T. Seidl. 3D Shape Histograms for Similarity Search and Classification in Spatial Databases. In *Advances in Spatial Databases SSD'99*, volume 1651, pages 207–228, Hong Kong, 1999. Springer.

[BS98] H. Bunke and K. Shearer. A Graph Distance Metric Based on the Maximal Common Subgraph. In *Pattern Recognition Letters*, volume 19, pages 255–259, 1998.

[CKS98] G. Chartrand, G. Kubicki, and M. Schultz. Graph Similarity and Distance in Graphs. In *Aeuqations Mathematicae*, volume 55(1-2), pages 129–145, Basel, 1998. Birkhäuser Verlag AG.

[CO02] D.-Y. Chen and M. Ouhyoung. A 3D Object Retrieval System Based on Multi-Resolution Reeb Graph. In *Proc. Computer Graphics Workshop*, page 16, Tainan, Taiwan, 2002.

[FMK+03] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, and D. Jacobs. A Search Engine for 3D Models. *acm Trans. on Graphics*, 22(1):83–105, January 2003.

[HB94] Y. Hecker and R. Bolle. On Geometric Hashing and the Generalized Hough Transform. In *IEEE Trans. On Systems, Man, And Cybernetics, 24*, pages 1328–1338, September 1994.

[HDD+93] H. Hoppe, T. DeRose, T. Duchamp, J. McDondald, and W. Stuetzle. Mesh Optimization. In *Proc. SIGGRAPH 93*, pages 19–26, New York, 1993. ACM Press.

[HID95] M. Hebert, K. Ikeuchi, and H. Delingette. A Spherical Representation for Recognition of Free-Form Surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(7):681–690, 1995.

[HKSV02] M. Heczko, D. Keim, D. Saupe, and D. Vranic. Verfahren zur Ähnlichkeitssuche auf 3D-Objekten. *Datenbank-Spektrum*, 2(4):54–63, Februar 2002.

[Hop96] H. Hoppe. Progressive Meshes. In *Proc. SIGGRAPH 96*, pages 99–108, Reading, MA, 1996. Addison Wesley.

[HSKK01] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. Kunii. Topology Matching for Fully Automatic Similarity Estimation of 3D Shapes. In *Proc. SIGGRAPH 2001*, pages 203–212, New York, 2001. ACM Press.

[KKV90] E. Kubicka, G. Kubicki, and I. Vakalis. Using Graph Distance in Object Recognition. In *Proc. 1990 ACM Ann. Conf. on Cooperation*, pages 43–48. ACM Press, 1990.

[Lam89] Y. Lamdan. Object Recognition by Geometric Hashing, 1989.

[LSW90] Y. Lamdan, J. Schwartz, and H. Wolfson. An Affine Invariant Model-Based Object Recognition. In *IEEE Trans. Robotics and Automation*, volume 6, pages 578–589. IEEE, October 1990.

[OFCD02] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape Distributions. *acm Trans. on Graphics*, 21(4):807–832, October 2002.

[Raa98] A. Raab. *Techniken zur Exploration und Visualisierung geometrischer Modelle*. PhD thesis, Otto-von-Guericke University Magdeburg, Germany, 1998.

[VS00] D. Vranic and D. Saupe. 3D-Model Retrieval. Research article, Univ. of Leipzig, 2000.

[VS01] D. Vranic and D. Saupe. A feature vector approach for retrieval of 3d objects in the context of mpeg-7. Research article, Univ. of Leipzig, 2001.

[WR97] H. Wolfson and I. Rigoutsos. Geometric Hashing: An Overview. *IEEE Computational Science & Engineering*, pages 10–21, October 1997.

[WZCS02] J. Wang, K. Zhang, G. Chang, and D. Shasha. Finding Approximate Patterns in Undirected Acyclic Graphs. *Pattern Recognition*, 35(2):473–483, 2002.

[ZWS95] K. Zhang, J. Wang, and D. Shasha. On the editing distance between undirected acyclic graphs and related problems. In *Proc. Symp. on Combinatorial Pattern Matching*, 937, pages 395–407, Berlin, 1995. Springer-Verlag.