HAL open science

New Graph Distance for Deformable 3D Objects Recognition based on Triangle-Stars Decomposition

Kamel Madi, Éric Paquet, Hamamache Kheddouci

▶ **To cite this version:**

Kamel Madi, Éric Paquet, Hamamache Kheddouci. New Graph Distance for Deformable 3D Objects Recognition based on Triangle-Stars Decomposition. Pattern Recognition, 2019, 10.1016/j.patcog.2019.01.040 . hal-02005169

HAL Id: hal-02005169

https://hal.science/hal-02005169v1

Submitted on 21 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

Version of Record: https://www.sciencedirect.com/science/article/pii/S0031320319300627
Manuscript_0b665982c1f34b10cf6f278f17d46fa0

New Graph Distance for Deformable 3D Objects Recognition based on Triangle-Stars Decomposition

Kamel Madi[a,c,*], Eric Paquet[b], Hamamache Kheddouci[a]

[a] *Université de Lyon, CNRS, Université Lyon 1, LIRIS, UMR5205, F-69622, France*
[b] *National Research Council Canada, Ottawa, Canada*
[c] *Umanis, Levallois-Perret, 92300, France*

Abstract

We address the problem of comparing deformable 3D objects represented by graphs such as triangular tessellations. We propose a new graph matching technique to measure the distance between these graphs. The proposed approach is based on a new decomposition of triangular tessellations into triangle-stars. The algorithm ensures a minimum number of disjoint triangle-stars, provides improved dissimilarity by covering larger neighbors and allows the creation of descriptors that are invariant or at least oblivious under the most common deformations. The present approach is based on an approximation of the Graph Edit Distance, which is fault-tolerant to noise and distortion, thus making our technique particularly suitable for the comparison of deformable objects. Classification is performed with supervised machine learning techniques. Our approach defines a metric space using graph embedding and graph kernel techniques. It is proved that the proposed distance is a pseudo-metric. Its time complexity is determined and the method is evaluated against benchmark databases. Our experimental results confirm the performances and the accuracy of our system.

Keywords: Graph matching, Graph edit distance, Graph decomposition, Graph embedding, Graph metric, Graph classification, Pattern recognition, 3D object recognition, Deformable object recognition, Metric learning

*Corresponding author
Email address: kamel.madi@liris.cnrs.fr (Kamel Madi)

© 2019 published by Elsevier. This manuscript is made available under the CC BY NC user license
https://creativecommons.org/licenses/by-nc/4.0/

1. Introduction

Comparing 3D objects is one of the most important tasks in 3D object recognition. Objects represented by graphs such as triangular tessellations (Definitions 1, 2), may be compared using graph matching techniques. In a graph, properties are associated with vertices while relationships are represented by edges. Vertices, edges and their attributes are specified according to the underlying application; for instance, vertices may represent points, regions of interest or any other substructure obtained by applying a data reduction process such as segmentation. Edges are associated with vertex connectivity, defining a topological relationship between them, such as proximity, adjacency, etc. Graph matching is the process of finding a correspondence between vertices and edges of two graphs that satisfies a set of constraints, ensuring that substructures in one graph are mapped to similar substructures in the other. Several approaches have been proposed to solve the graph matching problem [1, 2, 3, 4]. Graph edit distance is one of the most celebrated measures for determining such a distance [5, 6, 7, 8]. It is defined as the minimum-cost sequence of edit operations that transforms one graph into another. Tolerance to noise and distortion is one of the advantages of edit distance. Unfortunately, it has a high computational complexity which grows exponentially with the number of vertices [9]. In this paper, we address the problem of comparing deformable 3D objects. The shapes are represented by graphs. We propose a new distance for comparing deformable 3D objects. This distance is based on the decomposition of triangular tessellations into a set of new substructures that we call *triangle-stars*. A triangle-star is a connected component formed by the union of a triangle and the set of its neighboring triangles, depending on the neighborhood order considered. The proposed decomposition offers a parameterizable triangle-stars representation, which is determined by the order assigned to the neighborhood (Definition 7). The resulting number of triangle-stars is much smaller than the number of vertices as well as being smaller than the number of classical stars [10, 11] (Definition 2). As a result, the computational complexity is re-

duced. The proposed dissimilarity measure ensures a better approximation of the Graph Edit Distance. Indeed, considering triangle-stars structures makes it possible to cover a larger neighborhood area and presents a richer local descriptor compared to classical stars [10, 11]. From the proposed distance, it is possible to construct a set of descriptors which are either invariant or at least oblivious under the most common deformations.

A preliminary version of our work appeared in [12]. In the present version, the graph dissimilarity measure and the triangle-stars decomposition method are both improved. Classification is performed using supervised machine learning techniques. Our approach defines a metric space which describes the various objects. Five classifiers were used, namely: the naïve Bayes classifier, the random forest classifier, the gradient boosted trees approach, support vector machine (SVM) and logistic regression. The complexity analysis section has been extended. A new set of experiments has been performed with recent benchmark databases for deformable shape comparison. The remainder of the paper is organized as follows. In Section 2, we briefly review some related works. The proposed decomposition is described in Section 3 while the proposed distance is introduced in Section 4. Time complexity is determined in Section 5. In Section 6, we present and discuss our experimental results while comparing them with some benchmark shape-matching algorithms, for two benchmark databases. Finally, Section 7 concludes the paper.

2. Related works

In this section, we briefly review some 3D object recognition methods. We mainly focus on graph-based approaches as they are the most relevant for our technique. For a more exhaustive review, we refer the reader to [13] for 3D object recognition methods and to [9, 14, 15, 16, 17] for pattern recognition and graph matching techniques. Then, we briefly present the algorithms with which our method has been compared.

2.1. 3D object recognition related works

Three main categories can be distinguished for 3D object comparison techniques [13]: feature-based methods, graph-based methods and others. In the case of feature-based methods, objects are compared using features associated with their geometrical and topological properties. These features may be global, local or spatial maps. In the case of global features and spatial maps, the entire shape is characterized by a unique descriptor while, in the case of local features, a descriptor is either associated with the vertices, the triangles or any substructure of interest. Various feature-based approaches have been proposed in the literature [18, 19, 20, 21, 22]. Graph-based methods are powerful tools for establishing correspondences between objects and partial shapes as well as for their invariant description. According to the type of graph considered, several graph-based techniques have been proposed for 3D object comparison [13]. For instance, in some approaches, shapes are reduced to skeletons through a thinning process [23, 24, 25]. The resulting skeletons are compared using graph matching techniques. Other approaches rely on Reeb graphs, which are constructed from mapping functions defined on the shape manifolds [26, 27, 28]. Various segmentation techniques have been proposed in the literature, in which the shapes are segmented into a finite set of components from which a graph is constructed: the segments correspond to the vertices while their topological relationships are represented by the edges. These graphs may be compared with graph matching techniques [29, 30, 31]. Additional methods have been proposed [13] such as: view-based similarity, volumetric error based similarity, and weighted point set based similarity, among others. In the case of view-based similarity, two 3D objects are considered to be similar if their projections are similar from all possible viewpoints [32, 33, 34]. For volumetric error based similarity, the distance between two shapes is calculated by estimating the volumetric error between them [35, 36]. On the other hand, in the case of weighted point set based similarity, a distance between two shapes is calculated from a set of descriptors, which consist of weighted 3D points. First, the shapes are decomposed into substructures. Then, each substructure is represented by a

weighted point. Finally, the weighted points are matched. The matching process depends on the weight considered. In [37], the weight represents the volume of the component, in [38], a measure for curvature is used as a weight, while in [39], the authors consider a hierarchy of weighted point sets, representing spherical object approximations.

2.2. ***Benchmark for shape-matching algorithms***

The proposed approach has been compared to various benchmark algorithms associated with SHREC competitions. These algorithms correspond to the two benchmark databases used for their evaluations, namely the TOSCA database [40, 41] and the SHREC11 watertight database [42]. Consequently, our method has been compared with the following algorithms: **CAM** [43]: An approach in which surfaces are represented by 3D curves extracted around feature points. **GeodesicD2**: A global description which consists of distribution of the geodesic distances associated with a given 3D shape. This approach is an extension of Euclidean D2 [44]. **DSR** [45]: The Hybrid Feature Vector is a combination of two view-based descriptors: a depth buffer for the silhouette and a radialized extent function descriptor. **RSH** [46]: The Ray-based approach with Spherical Harmonic Representation is a method which aligns the models into a canonical position, determines the maximal extents and applies a spherical harmonic decomposition. **TD** [47]: The Temperature Distribution descriptor is a shape descriptor based on the heat kernel. The L2 norm is used to evaluate the distance between the descriptors. **Shape-DNA** [48]: The Shape-DNA is a numerical fingerprint obtained by evaluating the eigenvalues of the Laplace-Beltrami operator associated with the manifold. The matching between two objects is obtained by comparing their respective eigenvalues. **SRCP-TD** [49]: The SRCP-TD is a method based on a sparse representation of a scale-invariant heat kernel. The authors use Laplace-Beltrami eigenfunctions to detect critical points on the manifold. The descriptor is constructed from the heat kernel values at these points. A sparse representation is employed to reduce the dimensionality of the descriptor.

3. Algorithm description: new decomposition into triangle-stars

In this section, we propose a new decomposition of triangular tessellations into connected components that we call *triangle-stars*. This decomposition aims to reduce the number of components while covering larger neighborhoods. The extent of the neighborhood associated with a triangle-star is determined by its order N_k. From this representation, it is possible to define a description which is invariant or at least oblivious under the most common deformations. Prior to decomposition, a strict total order on the triangles must be established. This order aims to reduce the number of triangle-stars that is generated while guaranteeing the uniqueness of the decomposition.

3.1. ***Triangle-star***

We propose to decompose graphs such as triangular tessellations into a set of connected components that we call triangle-stars (TS). Triangle-stars are defined from the following set of definitions:

Definition 1 (Graph). A graph G is a set of vertices connected by a set of edges. Formally, a graph G is a four tuple $G = (V, E, \alpha, \beta)$, where: V and $E \subseteq V \times V$ are the vertices and edges. $\alpha : V \to L_V$ and $\beta : E \to L_E$ are labeling functions associated with the vertices and edges, respectively [50].

Definition 2 (Star). A classical star S is a labeled, single-level, tree formed by a root vertex r, the set of leaves or mono-degree vertices L and the set of edges E connecting the root vertex with any vertex belonging to L. Figure 1 shows all the classical stars that may be constructed from a given graph G [10, 11].

Definition 3 (Triangular tessellations). A triangular tessellation G_{tr} is a graph defined by a set of vertices, edges and triangles. Formally, G_{tr} is a graph defined by a six tuple $G_{tr} = (V, E, T, \alpha, \beta, \theta)$, where: V, E and T are the vertices, edges and triangles, respectively. $\alpha : V \to L_V$, $\beta : E \to L_E$ and $\theta : T \to L_T$ are their corresponding labeling functions.

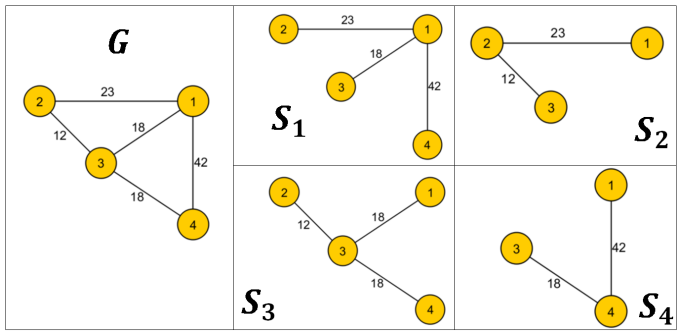


Figure 1: Classical stars associated with a graph.

Definition 4 (neighborhood of a triangle). Two triangles are neighbors if they share at least one common vertex. Let t_1 and t_2 be two triangles and $V(t_1)$ and $V(t_2)$ their respective vertices. Then, t_1 and t_2 are *neighbors* $\Leftrightarrow \|V(t_1) \cap V(t_2)\| > 0$. In other words, the neighborhood (N) of a triangle t consists of all the triangles sharing at least one common vertex with t.

Definition 5 (N_k-neighborhood of a triangle). Two triangles t_0 and t_k are N_k-neighbors if between t_0 and t_k there is a chain of at most $(k-1)$ distinct triangles, which are pairwise consecutive neighbors. Formally, t_0 and t_k are N_k-*neighbors* $\Leftrightarrow \exists\, t_{i=1...k-1}$ *where* : $\forall\, i \in 1...(k-1)$, t_i *and* t_{i+1} *are neighbors.* In the case of $k = 1$, the N_k-neighborhood is reduced to a simple neighborhood (Definition 4).

Definition 6 (triangle-star). A triangle-star ts is a labeled subgraph, defined by a triangle and the set formed by its neighbors. Formally, a triangle-star ts is a three tuple $ts = (t_r, T', \theta)$, where: t_r is the root triangle, T' is the set of adjacent triangles and $\theta : T \rightarrow L_T$ is the triangle labeling function, while L_T is a set of labels.

Definition 7 (N_k-triangle-star). A N_k-triangle-star N_k-ts is a triangle-star defined by a triangle and the set of its N_k-neighbors. In the case of $k = 1$, the N_k-triangle-star is a simple triangle-star (Definition 6). See, for instance, the following example.

Example 1. Consider a graph-tessellation G_{tr} containing 17 triangles $t_{0...16}$. Table 1 shows the graph-tessellation G_{tr} and the corresponding N_k-triangle-stars associated with the triangle t_0 according to the order $N_{k=0...2}$ of the corresponding neighborhood. In the case $N_{k=0}$, the graph-tessellation is decomposed into 17 triangle-stars. Each triangle-star consists of a single triangle which means that no neighborhood is considered. Therefore, the triangle-star N_0-TS associated with the triangle t_0 is the very same triangle t_0. When $N_{k=1}$, only the closest neighborhood is employed for the construction of the triangle-stars (See Definition 4). Thus, the triangle-star N_1-TS associated with the triangle t_0 is formed by the triangle t_0 and its N_1-neighbors (direct neighbors), *i.e*, $T(N_1\text{-}TS) = \{t_{0...7}\}$. In the case $N_{k=2}$, the N_2-neighborhoods are considered, which means that the graph-tessellation is decomposed into a set of triangle-stars formed by the root triangle and its N_2-neighbors (See Definition 5). As a result, the triangle-star N_2-TS associated with the triangle t_0, is formed by the triangle t_0 and its N_2-neighbors (second order neighborhood). $T(N_2\text{-}TS) = \{t_{0...16}\}$.

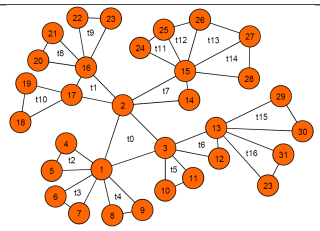
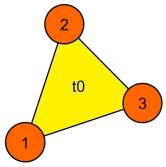
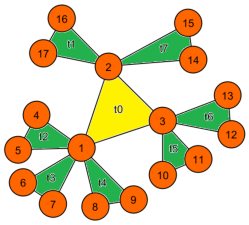
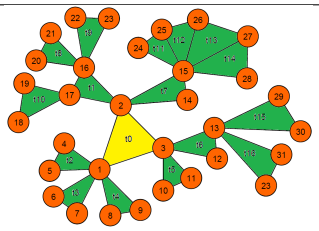
Graph G_{tr}	N_0-TS of triangle $t0$
	
N_1-TS of triangle $t0$	N_2-TS of triangle $t0$
	

Table 1: Example of N_k-triangle-stars.

Triangle-star features. A six-tuple $t_j = (v_1, v_2, v_3,\ e_1, e_2, e_3)$ is associated with each triangle t_j. The vertices v_i are labeled by their respective Cartesian coordinates $v_i = (x, y, z)$, while the edges $e_k = (v_p, v_w)$ are labeled (weighted) with the Euclidean in distance between their respective vertices (v_p, v_w). The triangles are labeled with a three-tuple $t_j = (id, Area, Perimeter)$, where id is a number. Each triangle-star is characterized by a set of descriptors, allowing for evaluation of the dissimilarity between triangle-stars. We consider the following descriptors: Area of triangle-star, Perimeter of triangle-star, Area of the triangles forming the triangle-stars, their Perimeters, the Weights associated with their edges, and the Degrees of their vertices. Our choice of descriptors is justified by the fact that these quantities are oblivious under the most common deformations.

Triangle-star vector representation. A vector is associated with each triangle-star. This vector consists of the global area AG and the global perimeter PG of the triangle-star, the area A and the perimeter P of each triangle belonging to the triangle-star, the weights associated with their edges W, as well as the degrees Deg associated with their vertices. This vector is given by:

$$\{AG(ts),\ PG(ts),\ \{A(t_i),\ P(t_i),\ W(t_{i,\ j=1\ldots3}),\quad Deg(t_{i,\ j=1\ldots3})\}_{i=1}^{i=\|T(ts)\|}\}$$

The various variables are described in Table 2. The triangles belonging to the triangle-star ts are ranked according to their areas in descending order. The weights and the degrees are ranked in descending order as well. All triangle-stars TS vectors have the same size: $size = 2 + (8\ \ \Gamma)$, where Γ is the maximum number of triangles in the triangle-stars. If a triangle-star ts has a number of triangles less than Γ, the unassigned entries are completed with $zeros$.

Definition 8 (Disjoint triangle-stars). Two triangle-stars ts_i and ts_j are $disjoints$ if they do not share, at least, a common triangle. Let $i \neq j,\ if\ ts_i$ and ts_j are $disjoints \Rightarrow T(ts_i) \cap T(ts_j)\ =\ \varnothing$.

Symbol	Description
$t_{i,l}$	Triangle t_l belonging to the triangle-star ts_i: $t_l \in ts_i$
$W_{i,l,k}$	Weight (Euclidean distance) of edge e_k belonging to triangle $t_l \in ts_i$
$Deg_{i,l,k}$	Degree of vertex v_k belonging to triangle $t_l \in ts_i$
Γ	Max number of triangles in the triangle-stars
$\alpha \in \mathbb{R}_+^6$	Parameters associated with the descriptors where $\|\alpha\|_1 = 1$
$A(t_i)$	Area of triangle i.
$P(t_i)$	Perimeter of triangle i.
$AG(ts_i)$	Area of triangle-star i. $AG(ts_i) = \sum_{j=1}^{j=\|T(ts_j)\|} A(t_j)$
$PG(ts_i)$	Perimeter of triangle-star i. $DG(ts_i) = \sum_{j=1}^{j=\|T(ts_j)\|} D(t_j)$

Table 2: Quantities associated with the dissimilarity measure and their description.

3.2. ***Triangle ordering***

The proposed decomposition generates disjoint triangle-stars (Definition 8), which significantly reduces the number of components ($\|TS\| \ll \|V\| < \|T\|$) as well as the number of comparisons involved when matching two graphs. However, depending on the ordering considered, the triangle-stars obtained may differ (see Example 2). Indeed, the same triangular tessellation may generate different sets of triangle-stars if the ordering of the triangles is not identical.

In order to ensure the **uniqueness** of the decomposition and to further reduce the number of triangle-stars, a strict descending total order must be established on the set of triangles prior to their decomposition into triangle-stars. Considering a $\|Neighbors\ (Triangles)\|$ with *descending order* results in the smallest number of triangle-stars TS (as demonstrated in the experimentations), thus further contributing to the reduction of computational complexity.

In order to establish a strict descending total order on the set of triangles, each triangle t_i is represented by a vector $\{\|N(t_i)\|, \{x_{i,j}, y_{i,j}, z_{i,j}\}_{j=1}^{j=3}\} \in \mathbb{R}^{10}$ which corresponds to the number of neighbors $\|N(t_i)\|$ and to the Euclidean coordinates x, y, z associated with the vertices forming the triangle t_i. Naturally, the coordinates and consequently the ordering may be affected by a rotation of the object. In order to eliminate this problem, the coordinates are expressed in the reference frame defined by the Eigen vectors of the tensor of inertia associated with the vertices. The number of neighbors $\|N(t_i)\|$ is used to further

reduce the number of triangle-stars. If two triangles have the same number of neighbors, the vertex coordinates are used to ensure the uniqueness of the decomposition. The vertices in the vector (10-tuple) are lexicographically ordered according to their coordinates.

Example 2. Let us consider a triangle t_0 with n neighbors $t_{1\ldots n}$ (Figure 2).
If we consider a triangle order based on the number of neighbors with descending order, we obtain only one triangle-star, else three triangle-stars are obtained.

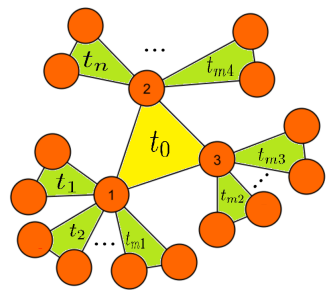


Figure 2: A triangle t_0 with n triangular neighbors.

3.3. ***Triangle-stars decomposition***

Once the strict total order of the triangles has been established, the decomposition of the graph into triangle-stars may be performed. This process is described in Algorithm 1. According to the order established for the triangles (strict descending order), the first N_k-triangle-star is constructed from the first triangle and its corresponding N_k-neighbors (Definition 7); that is, the triangles not belonging to any other N_k-triangle-stars. Then, the set of triangles and the set of the resulting N_k-triangle-stars are updated. The process is repeated until a N_k-triangle-star is associated with each triangle.

The proposed decomposition generates a reduced number of triangle-stars ts as opposed to the number of vertices $\|TS\| \ll \|V\|$. The resulting triangle-stars are disjoints (Definition 8) and cover a larger local area than the classical stars (Definition 2). This decomposition is also parameterizable through the neighborhood order. Indeed, the higher the neighborhood order, the smaller the number of triangle-stars ($\|N_{k+1}\text{-}TS\| \leq \|N_k\text{-}TS\|$) which, in turn, cover a larger neighborhood ($\|T(N_{k+1}\text{-}TS)\| \geq \|T(N_k\text{-}TS)\|$). In addition, the proposed decomposition is unique.

Algorithm 1 Graph decomposition into N_k-triangle-stars.

Inputs: A graph G_{tr} and the neighborhood order N_k.
Outputs: A set of N_k-triangle-stars (N_k-TS).
Begin
    Apply a **descending** strict total order on the set of triangles of G_{tr};
    N_k-$TS = \varnothing$;
    while $(T(G_{tr}) \neq \varnothing)$ **do**
        $t_i = T(G_{tr})[0]$
        $T(N_k\text{-}ts_i) = t_i \cup N_k$-**neighbors**$(t_i)$;
        N_k-$TS = N_k$-$TS \cup N_k$-ts_i ;
        $T(G_{tr}) = T(G_{tr}) - T(N_k\text{-}ts_i)$;
    end while
    return N_k-TS ;
End

Example 3. We consider a triangular tessellation defined as follows: $G_{tr} = \{16\ vertices,\ 20\ triangles\ t_{1...20}\}$ as shown in Table 3. The decomposition into triangle-stars (N_1-TS) begins with the construction of the first triangle-star TS_1 using the triangle t_{13} as the root triangle and its corresponding N_1-neighborhood. The triangle t_{13} is selected as the root triangle since it has the maximum number of neighbors, which is 12 in this particular case. The next triangle with the maximum number of neighbors (seven) is t_1; this triangle is employed for the construction of the second triangle-star TS_2. TS_2 is constructed using t_1 and its 3 neighbors. The third triangle-star TS_3 is formed by t_{11} and its neighbors: t_{11} had 5 neighbors which is the maximum number of neighbors in the remaining set of triangles. TS_3 is constructed using t_{11} and its 2 remaining neighbors. In the case of the N_2-neighborhood, we obtain one triangle-star N_2-TS_1 formed by the triangle t_{13} and its N_2-neighborhood which consists of all the triangles belonging to G_{tr}. Therefore, N_2-TS_1 is equivalent to the triangular tessellation G_{tr} in this particular case.

Example 4. Table 4 shows the correspondence in between N_2-triangle-stars of two poses centaur0 and centaur3 of the same object centaur belonging to the TOSCA Database.

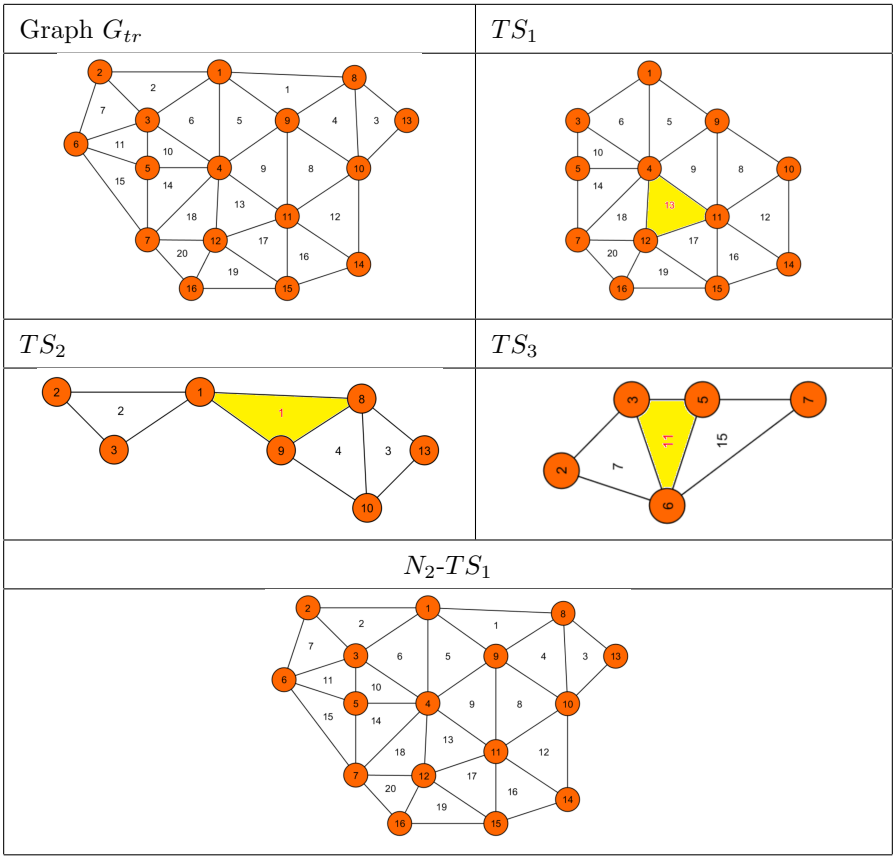


Table 3: Decomposition of a graph into a set of N_1 and N_2-triangle-stars.

4. Algorithm description: new distance for triangular tessellations

In this section we propose a novel distance between the triangle-stars of two triangular tessellations to address their matching. We prove that the proposed distance (TSM for *Triangle-Star Measure*) is a pseudo-metric.

4.1. ***Edit distance between triangle-stars***

We first introduce the graph edit distance between triangle-stars. The proposed dissimilarity measure is defined to be applicable to deformable objects. As a result, the set of descriptors must be invariant or at least oblivious under the most common deformations. The dissimilarity measure is based on the fol-

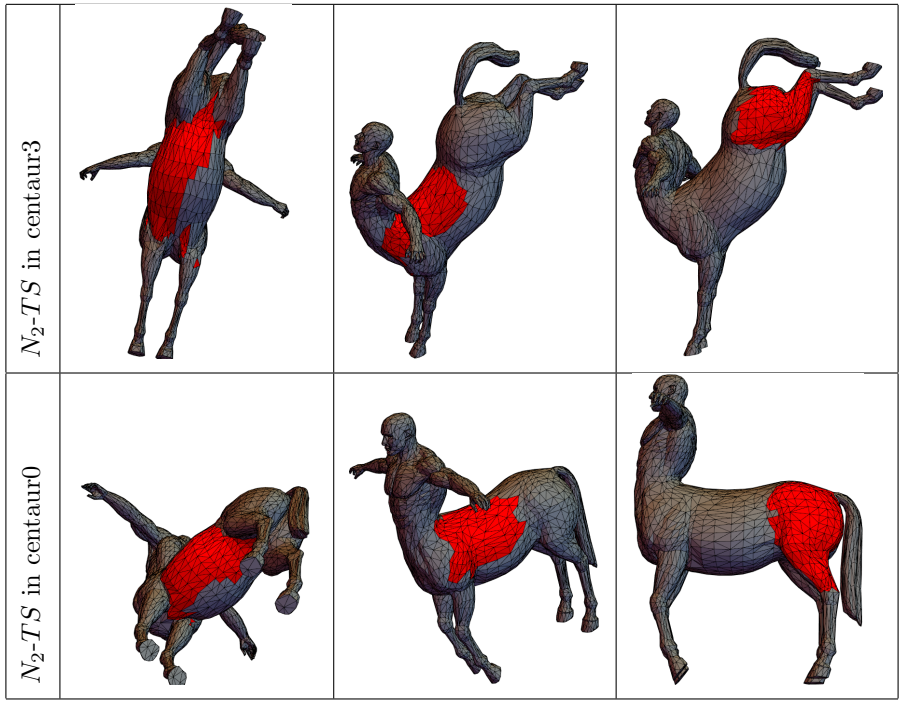


Table 4: The correspondence between N_2-triangle-stars (colored in red) of two poses centaur0 and centaur3 of the same object centaur belonging to the TOSCA Database.

lowing parameters: *Area* and *Perimeter* of triangle-star, *Area* and *Perimeter* of triangles, *Weights* of edges and *Degrees* of vertices. Formally a triangle-star is represented as follows: $\{AG(ts),\ PG(ts),\ \{A(t_i),\ P(t_i),\ W(t_{i,\ j=1...3}),\ Deg(t_{i,\ j=1...3})\}_{i=1}^{i=\|T(ts)\|}\}$. The dissimilarity measure d between two triangle-stars ts_i and ts_j is defined as:

$$d(ts_i, ts_j) = \sum_{k=1}^{k=6} dsim_k\ (ts_i, ts_j) \tag{1}$$

The dissimilarity measure d is normalized ($0 \leq d \leq 1$) and requires the def-

inition of six auxiliary functions $dsim_k$. These functions are defined as follows:

$$dsim_k\ (ts_i, ts_j) = \begin{cases} \alpha_1 & \frac{|AG(ts_i)\ -\ AG(ts_j)|}{AG_{MAX}} & \text{if } k = 1 \\ \alpha_2 & \frac{|PG(ts_i)\ -\ PG(ts_j)|}{PG_{MAX}} & \text{if } k = 2 \\ \alpha_3 & \frac{\sum_{l=1}^{l=\Gamma} |\ A(T(ts_i)_l)\ -\ A(T(ts_j)_l)\ |}{A_{MAX}\quad \Gamma} & \text{if } k = 3 \\ \alpha_4 & \frac{\sum_{l=1}^{l=\Gamma} |\ P(t_{i,l})\ -\ P(t_{j,l})\ |}{P_{MAX}\quad \Gamma} & \text{if } k = 4 \\ \alpha_5 & \frac{\sum_{l=1}^{l=\Gamma} \sum_{k=1}^{k=3} |\ W_{i,l,k}\ -\ W_{j,l,k}\ |}{3\quad W_{MAX}\quad \Gamma} & \text{if } k = 5 \\ \alpha_6 & \frac{\sum_{l=1}^{l=\Gamma} \sum_{k=1}^{k=3} |\ Deg_{i,l,k}\ -\ Deg_{j,l,k}\ |}{3\quad Deg_{MAX}\quad \Gamma} & \text{if } k = 6 \end{cases} \tag{2}$$

$dsim_1(ts_i, ts_j)$ and $dsim_2(ts_i, ts_j)$ compare respectively the Area AG and the Perimeter PG of two triangle-stars ts_i and ts_j. $dsim_3(ts_i, ts_j)$ and $dsim_4(ts_i, ts_j)$ compare respectively the Area A and the Perimeter P of their respective triangles. While $dsim_5(ts_i, ts_j)$ and $dsim_6(ts_i, ts_j)$ compare respectively the Weights W associated with their respective edges and the Degree Deg of the corresponding vertices. The symbols associated with the dissimilarity measure are defined in Table 2.

Graph Edit Distance (GED) is based on a set of classical edit operations namely: insertion, deletion and substitution of vertices and/or edges. The proposed dissimilarity measure $d(ts_i, ts_j)$ (Eq. 1) is equivalent to a substitution edit operation of two triangle-stars ts_i and ts_j. However, the insertion edit operation of a triangle-star ts_i is equivalent to replacing an empty triangle-star ε by ts_i which, in turn, is equivalent to the dissimilarity measure $d(\varepsilon, ts_i)$. Analogously, the deletion edit operation of a triangle-star ts_i is equivalent to replacing ts_i by an empty triangle-star ε, which is equivalent to the dissimilarity measure $d(ts_i, \varepsilon)$. The six auxiliary functions $dsim_k$ (Eq. 2) are equivalent to a substitution of two triangle-stars ts_i and ts_j $dsim_k(ts_i, ts_j)$, a deletion or an insertion of a triangle-star ts_i $dsim_k(ts_i, \varepsilon)$ by considering, separately, one of the six parameters (AG, PG, A, P, W and Deg). Consequently, the dimensions of the cost matrix D are reduced and we obtain a better time complexity, see Section 5.

Scale invariance. The dissimilarity measure d (Eq. 1) is sensitive to scale, which means that if two triangle-stars ts_i and ts_j have the same structure

but a different scale, they are not considered to be similar. In order to obtain scale invariance, the weights W must be normalized, which means formally that $\forall i, W_i = \frac{W_i}{W_{MAX}}$. Consequently, by normalizing weights W, the parameters AG, PG, A and P are also normalized.

4.2. ***Edit distance between two triangular tessellations***

The calculation of the distance between two triangular tessellations represented by triangle-stars constitutes the last step of our algorithm. We call this dissimilarity measure *Triangle-Star Measure* TSM. This measure determines the best possible matching between two sets of triangle-stars. The dissimilarity between two sets of triangle-stars is defined as follows:

Definition 9 (*TSM*). Let g_{Tr1} and g_{Tr2} be two triangular tessellations, TS_1 and TS_2 their corresponding sets of triangle-stars, M the set of all possible matchings between TS_1 and TS_2, and $m \in M$ the mapping function. The dissimilarity measure $TSM(TS_1, TS_2)$ (normalized dissimilarity) is defined as:

$$TSM(TS_1,\ TS_2) = \frac{\min_{m \in M} \sum_{ts_i\ \in\ TS_1,\ m(ts_i)\ \in\ TS_2} d(ts_i,\ m(ts_i))}{\max(\|TS_1\|,\ \|TS_2\|)} \tag{3}$$

The computation of $TSM(TS_1, TS_2)$ is equivalent to solving the assignment problem, which is one of the fundamental combinatorial optimization problems that aim to find the minimum/maximum weight matching in a weighted bipartite graph. To solve this assignment problem, we define a $n \times n$ matrix D, where n is given by $n = \max(\|TS_1\|,\ \|TS_2\|)$. Each element $D_{i,j}$ of the matrix represents the dissimilarity measure $d(ts_i,\ ts_j)$ (Eq. 1) between a triangle-star ts_i in TS_1 and a corresponding triangle-star ts_j in TS_2. In the case of $\|TS_1\| \neq \|TS_2\|$, the smallest set of triangle-stars is completed by $(max(||TS_1||, ||TS_2||) - min(||TS_1||,\ ||TS_2||))$ empty triangle-stars ε. The distance between an empty triangle-star ε and a triangle-star ts is computed by Eq. 1 and corresponds to the cost of adding ts to the small set of triangle-stars (or deleting ts from the large set of triangle-stars).

We apply the Hungarian algorithm [51] to the matrix D to find the best assignment in $\mathcal{O}(n^3)$ time. The evaluation of the distance between two graphs (triangle tessellations), is summarized in Algorithm 2.

The classification is then performed using supervised machine learning techniques. Our approach defines a metric space which describes the various objects, using graph embedding and graph kernel techniques: Each object (the set of its triangle-stars) TS_i is mapped onto a vector space, where it is represented by a vector of distances $TSM(TS_i, TS_{j=1\ldots n})$, between TS_i and the set of other objects $TS_{j=1\ldots n}$.

Algorithm 2 The distance between two graphs using TSM.

Inputs: Two graphs g_1 and g_2.
Outputs: The distance between g_1 and g_2.
Begin
    Decomposition of g_1 and g_2 into sets of triangle-stars TS_1 and TS_2, (Algo. 1).
    Construct a matrix of distance D.
    For each $ts_i \in TS_1$ and $ts_j \in TS_2$ do
        $D_{i,j} = d(ts_i,\ ts_j)$ (Eq. 1);
    end For each
    Solving (Eq. 3), by applying the Hungarian algorithm [51] on the matrix D.
    return the distance $TSM(TS_1,\ TS_2)$;
End

4.3. The Pseudo-metric

In this section we prove that the proposed distance is a **pseudo-metric**.

Definition 10 (Pseudo-metric) Let X be a set of objects and $x, y, z \in X$. Let f be a function defined as follows $f : X \times X \longrightarrow \mathbb{R}$. Let the following set of properties: (1) **non-negativity**: $f(x, y) \geq 0$, (2) **symmetry**: $f(x, y) = f(y, x)$, (3) **triangle inequality**: $f(x, y) \leq f(x, z) + f(z, y)$ and (4) **uniqueness**: $f(x, y) = 0 \Rightarrow x = y$. The function f is a metric if f satisfies these four properties, while f is a pseudo-metric if f satisfies only the first three properties (1, 2 and 3).

Since f is a pseudo metric, a distance function may be defined between each pair of graphs. As a result, the dissimilarity between graphs may be efficiently determined. [52, 53].

Lemma: The proposed similarity measure TSM (Eq. 3) between two sets of triangle-stars TS_1 and TS_2 is a **pseudo-metric**.

Proof: From (Eq. 3) it may be concluded that if TSM is a pseudo-metric then d (Eq. 1) is a pseudo-metric, which implies that $dsim_k$ (Eq. 2) are pseudo-metrics. Consequently, we shall prove that $dsim_k$ (Eq. 2) are pseudo-metrics. Proving that $dsim_k$ (Eq. 2) are pseudo-metrics is equivalent to ensuring that the first three properties are satisfied. The functions $dsim_k$ are defined as follows: $dsim_k = \alpha_k * \frac{|x_1 - x_2|}{\beta}$ with $x_1, x_2, \in \mathbb{R}_+,\ \alpha_k, \beta \in \mathbb{R}_+^*$.
1) **non-negativity**: $TSM(TS_1, TS_2) \geq 0$. We have $dsim_k \geq 0 \Rightarrow TSM \geq 0$ Thus TSM is non-negative. 2) **symmetry**: $TSM(TS_1, TS_2) = TSM(TS_2, TS_1)$. The proposed decomposition is unique and the TSM is only based on symmetrical operations (addition and subtraction in absolute value). Consequently, TSM is symmetrical. 3) **triangle inequality**: $TSM(TS_1, TS_2) \leq TSM(TS_1, TS_3) + TSM(TS_3, TS_2)$. We have the triangle inequality verified in: $|x_1 - x_2| \leq |x_1 - x_3| + |x_3 - x_2|$. Thus, the triangle inequality is verified in $dsim_k$ and we have: $TSM(TS_1, TS_2) \leq TSM(TS_1, TS_3) + TSM(TS_3, TS_2)$. Consequently, the triangle inequality is verified in TSM.

5. Computational complexity

The most demanding part of the algorithm, in terms of time complexity, is the one solving the assignment problem. As mentioned earlier, the Hungarian algorithm [51] is employed to find the best assignment in $\mathcal{O}(n^3)$ time, where n is the *maximum* number of components in the two graphs. Let $n = \max(\|V_1\|,\ \|V_2\|)$ and $n' = \max(\|TS_1\|,\ \|TS_2\|)$, where V_i is the set of vertices and TS_i is the set of triangle-stars in $g_{tr\ i}$. In the proposed decomposition, any triangle-star has at least one triangle. Consequently, in the

worst case, we have $n' = \frac{n}{3}$, which means that complexity is of the order of $\mathcal{O}(0.037 \quad n^3)$. However, the number of triangle-stars depends on the structure of the underlying graph as well as on the neighborhood order N_k. Indeed, the number of triangle-stars decreases when the neighborhood order N_k increases. The computational complexity, for the TOSCA database [40, 41], is of the order of $\mathcal{O}(\alpha \quad [\frac{n}{\log(n)}]^3)$, where $\alpha \in [1.80 * 10^{-7},\ 0.74]$ for $N_{k=1...6}$: $\mathcal{O}(0.74 \ [\frac{n}{\log(n)}]^3)$ for N_1, $\mathcal{O}(0.0001 \ [\frac{n}{\log(n)}]^3)$ for N_2, $\mathcal{O}(5.6115 * 10^{-6} \ [\frac{n}{\log(n)}]^3)$ for N_3, $\mathcal{O}(9.84 \ * \ 10^{-7} \ [\frac{n}{\log(n)}]^3)$ for N_4, $\mathcal{O}(3.59 \ * \ 10^{-7} \ [\frac{n}{\log(n)}]^3)$ for N_5 and $\mathcal{O}(1.80 \ * \ 10^{-7} \ [\frac{n}{\log(n)}]^3)$ for N_6. As shown above, the complexity decreases when the neighborhood order N_k increases: the neighborhood order $N_{k=1}$ has the highest complexity while the neighborhood order $N_{k=6}$ has the lowest.

Computing the exact value of the graph edit distance GED is NP-Hard for general graphs which implies an exponential computational complexity [11]. As a result, several algorithms approximating GED in polynomial time have been suggested. We propose a new graph distance, which is an approximation of GED (bipartite graph matching) based on triangle-star (TS) decomposition. The proposed dissimilarity measure $d(ts_i, ts_j)$ (Eq. 1) is equivalent to a substitution edit operation of two triangle-stars ts_i and ts_j. The insertion and deletion edit operation is equivalent to replace an empty triangle-star ε by ts_i $d(\varepsilon, ts_i)$ (for insertion) and replacing ts_i by an empty triangle-star ε $d(ts_i, \varepsilon)$ (for deletion), respectively. Consequently, the dimensions of the cost matrix D are reduced to $n \times n$, where n is given by $n = \max(\|TS_1\|,\ \|TS_2\|)$ ($\|TS\| \ll \|V\|$). Therefore, we obtain a better time complexity than the state of the art. Indeed, in [10] and [11], the complexity achieved is $\mathcal{O}((n+m)^3)$, where n and m are the number of nodes in the two graphs. In [54] and [55], the obtained complexity is $\mathcal{O}((\max(n,\ m))^3)$. However, the Jonker-Volgenant linear solver [56] has some convergence problems for some specific cost matrices. In [57], the proposed algorithm has a quadratic computational cost $\mathcal{O}(n * m)$. However, it does not generate a bijective correspondence between the nodes of the two graphs. In [58], the complexity achieved is: $\mathcal{O}((\min(n,\ m))^2 \ \max(n,\ m))$. Our approach obtains a better time complexity than [58], except from when

$||n - m|| \in]1600, 4000[$, for which the complexity is $\mathcal{O}(0.037 \quad n^3))$ and, when $||n - m|| \in]1700, 3900[$ and $N_{k=1}$, for which the complexity is $\mathcal{O}(0.74\ [\frac{n}{\log(n)}]^3)$. However, these intervals are not present in our experimentations (TOSCA and SHREC databases).

6. Experimental results

To evaluate the proposed approach, we undertook a set of experimentations in which we compared our approach with some benchmark algorithms for shape-matching, under different evaluation criteria, for two benchmark databases, namely the TOSCA database [40, 41] and the SHREC11 watertight database [42].

*6.1. **Database description***

In this section, we describe the two benchmark databases utilized in our evaluation: TOSCA and SHREC11 watertight databases.

The TOSCA database [40, 41] consists of 148 three-dimensional objects. Each object is represented by a triangular tessellation. The database consists of 12 classes. Each class is composed of the same object submitted to isometric or quasi-isometric deformations. The database is unbalanced (from 3 to 24 objects per class). On average, the number of vertices is 3154 while the number of triangles is 6220.

The SHREC11 watertight database [42] consists of a relatively large collection of non-rigid 3D shapes. This database was created from several publicly available databases such as the McGill database [59], the TOSCA database [40, 41] and the Princeton Shape Benchmark [60]. The SHREC11 watertight database [42] consists of 600 non-rigid objects represented by triangular tessellations. The database is balanced: it is formed by 30 classes, each consisting of 20 poses of the same object.

6.2. *Experimental results*

In this section, we compare and discuss our results with those obtained with the following benchmark databases: TOSCA [40, 41] and SHREC11 watertight [42]. The proposed distance TSM is parameterized through the parameters α_k, which determine the weights assigned to the various similarity measures. The default value for these parameters is: $\alpha_k = 1/6, \forall\, k$. As proved earlier, the TSM distance is a pseudo-metric. In order to classify the various objects belonging to the benchmark databases, the metric is learned with machine learning techniques. This is in contrast with the standard approach in which an invariant descriptor is associated with each object and classification is performed with machine learning techniques through supervised learning. In our approach, the TSM distance defines a metric space which describes the various objects. Therefore, in order to perform classification, the metric must be learned using a supervised learning process. Five classifiers were evaluated in order to determine their suitability for metric learning, namely: the naïve Bayes classifier, the random forest classifier, the gradient boosted trees approach, support vector machine (SVM) and logistic regression. These classifiers have been extensively described in the literature [61, 62]. Therefore, we shall limit ourselves to a high level description. In what follows, the features refer to the metric. The naïve Bayes (NB) classifier [61, 62] is a probabilistic classifier based on the Bayes' theorem. It assumes that the features are generated independently given the class and uses the Bayes theorem to predict the class. The random forest (RF) classifier [61, 62] uses a set of decision trees to predict the class. Each decision tree has been trained on a random subset of the training set, and only uses a random subset of the features. The gradient boosted tree (GBT) classifier [61, 62] predicts labels by iteratively training a sequence of decision trees on training data and combining them. The support vector machine (SVM) classifier [61, 62] separates the training set into two classes using a maximum-margin hyperplane. The multi-class classification problem is reduced to a set of binary classification problems. Finally, the logistic regression (LR) classifier [61], also known as the maximum entropy classifier, models class probabilities with logistic functions of

linear combinations of features.

For each classifier, a small validation set was automatically generated in order to evaluate the classifier parameters. The sole parameter for the naïve Bayes classifier was the smoothing parameter, which was typically around 0.2. For the random forest classifier, four parameters were required, namely: features fraction, leaf size, number of trees, and distribution smoothing, which were set to $1/2\sqrt{37}$, 2, 100 and 0.5 respectively. This algorithm was implemented using the Intel Data Analytics Accelerations Library (DAAL) [63]. The gradient boosted trees approach requires numerous parameters, including the boosting method. This is based on the gradient, the maximum number of training rounds (50), the number of leaves (13), the learning rate (0.1), the maximum number of bins (255), the number of threads (20), the maximum depth (6), the leaf size (15), the feature fraction (1), and the bagging fraction (1) among others. For the SVM classifier, the following parameters were determined: kernel type (radial basis function), gamma scaling parameter (0.00725065), soft margin parameter (3), bias parameter (1), multiclass strategy (one versus one), and kernel cache size (100). Finally, the logistic regression classifier was optimized by means of the limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algorithm with a quadratic regularization term (0.001). Four metrics were employed to evaluate the performance of our system, namely: accuracy, precision, recall and F-measure. These metrics were chosen because of their performance and in order to compare our approach with those utilized in the benchmark databases. Since these metrics are well known and extensively used, we refer the reader to [64, 65] for more details. In addition, in some cases, we added the confusion matrix and the accuracy-rejection rate curve in order to further assert the performance of the system as well as its sensitivity with respect to the detection threshold probability. Here, the rejection rate refers to the classification probability of a given result: if this probability is below a certain threshold, the outcome of the classification process is considered undetermined and, consequently, not employed in evaluation of the performance metrics. Since the benchmark databases are relatively small, they are not really suitable for cross-

validation. Therefore, to evaluate the metric, we employed a bootstrapping or bagging technique. Bagging assumes that the dataset is representative of its real distribution. The training and validation sets are uniformly randomly sampled with replacement from the original dataset in order to train the classifiers. The process was repeated ten times. Accuracy, precision, recall and F-measure correspond to the mean of these ten iterations.

Whenever possible, the classifiers were implemented on the GPU. The calculations were performed on a workstation with two Xeon processors with 40 cores, 64 GB or RAM and a NVIDIA Quadro GP-100 GPU with 3584 CUDA cores and 16 GB of memory.

6.2.1. ***Experimental results for the TOSCA and SHREC11 Watertight Databases***

The TSM distance between each pair of tessellated models was evaluated for the first six neighborhoods. These distances form the metric which must be learned. For each neighborhood order and for each classifier: accuracy, precision, recall, F-measure, confusion matrix, and accuracy-rejection rate curve were determined. Only the results associated with the three best classifiers are reported in Table 5 as these results are substantially better than the others.

For the second neighborhood order, the best results were obtained with the logistic regression classifier, while, for the first and the last four neighborhood orders, the best results were obtained with either the random forest classifier or the gradient boosted trees classifier. For the best classifiers, the lowest accuracy was 76.35% while the highest accuracy was 88.51%. Both were obtained with the first neighborhood order using the gradient boosted trees classifier.

The accuracy-rejection rate curve and the confusion matrix are reported for the first neighborhood order in Figures 3 and 4 respectively. This accuracy-rejection rate curve clearly demonstrates that an accuracy of 95% may be easily achieved just by imposing a rejection rate of 0.1. The confusion matrix (Figure 4) further demonstrates the performance of our approach.

N_k	Accuracy			Precision			Recall			F-measure		
	GBT	RF	LR	GBT	RF	LR	GBT	RF	LR	GBT	RF	LR
1	**88.51**	83.78	85.81	**91.80**	85.12	88.34	**89.64**	80.41	87.41	**0.90**	0.80	0.86
2	71.62	79.05	**79.73**	72.31	**81.70**	81.60	74.40	80.41	**82.04**	0.72	0.80	**0.81**
3	**81.76**	79.73	77.27	**85.39**	79.87	77.41	**81.47**	78.50	77.52	**0.82**	0.78	0.76
4	75.00	**76.35**	61.49	77.35	**78.98**	68.07	75.40	**76.50**	64.26	0.74	**0.76**	0.62
5	**77.70**	72.30	62.21	**81.04**	74.29	66.06	**76.75**	71.10	68.20	**0.76**	0.70	0.68
6	**80.41**	79.05	62.21	**83.34**	80.18	66.06	**79.94**	78.96	68.20	**80.41**	0.78	0.68

Table 5: accuracy, precision, recall and F-measure obtained with the gradient boosted trees (GBT), the random forest (RF) and the logistic regression (LR) classifiers for the TOSCA Database.

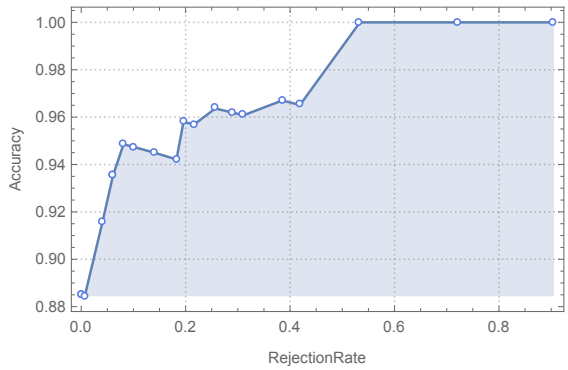


Figure 3: Accuracy-rejection rate curve for the first neighborhood order $N_{k=1}$ for the gradient boosted trees classifier.

We also compared our method TSM with four other SHREC benchmark algorithms, namely CAM, GeodesicD2, DSR and RSH. Table 6 compares the precision reported by these approaches with the precision achieved by our method for the first neighborhood order with the gradient boosted trees classifier. Our approach clearly outperforms the others with a precision of 91.80% as opposed to a maximum of 30% for the four other approaches.

In addition, we compared our approach for all neighborhood orders with the TD, Shape-DNA and SRCP-TD methods in terms of F-measure. The results are reported in Table 7. Once more, our approach outperformed the others

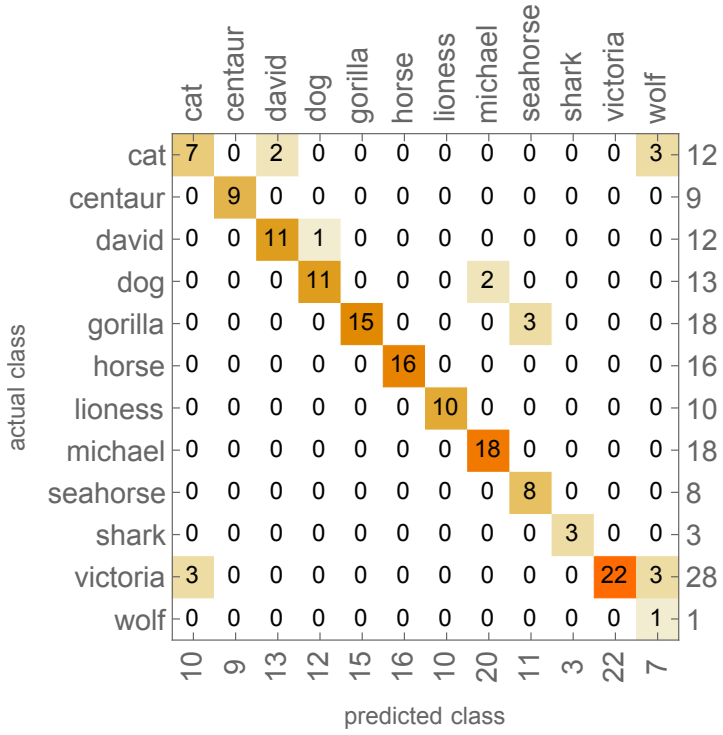


Figure 4: Confusion matrix for the first neighborhood order $N_{k=1}$ for the gradient boosted trees classifier.

Method	Precision (%)	Recall (%)
TSM, $N_{k=1}$, GBT	**91.80**	**89.64**
CAM	30	89.64
Geodesic2D	26	89.64
DSR	24	89.64
RSH	21	89.64

Table 6: Comparison of our method TSM (first neighborhood order $N_{k=1}$) in terms of precision with the CAM, GeodesicD2, DSR and RSH approaches for the TOSCA Database.

irrespective of the neighborhood order considered.

Finally, we report the results obtained with the SHREC11 Watertight Database. The experimental methodology was the same as that used for TOSCA. The best

Method	N_k	Classifier	F-measure
TSM	1	GBT	**0.90**
TSM	2	LR	**0.81**
TSM	3	GBT	**0.82**
TSM	4	RF	**0.76**
TSM	5	GBT	**0.76**
TSM	6	GBT	**0.80**
TD	N/A	N/A	0.67
Shape-DNA	N/A	N/A	0.45
SRCP-TD	N/A	N/A	0.44

Table 7: F-measure for the best classifier associated with a given neighborhood order compared with the F-measure obtained with the TD, Shape-DNA and SRCP-TD algorithms for the TOSCA Database.

results in terms of accuracy, for neighborhood orders two and six, were 65.80% and 69.16%, respectively; these were obtained with the random forest classifier.

6.2.2. ***Experimental results for the TOSCA Database with respect to re-meshing***

We investigated the robustness of our approach TSM with respect to mesh reduction and re-meshing. This is of paramount importance since our method is based on the mesh or graph associated with the 3D models. The TOSCA Database [40, 41] was employed for the evaluation. The number of triangles was reduced by 10% and 20% in order to generate the TOSCA_90 and TOSCA_80 Databases respectively. The triangular reduction was realized with quadratic edge collapse decimation [66], which uses iterative contractions of vertex pairs to simplify the model and maintains surface error approximations using quadric matrices. By contracting arbitrary vertex pairs (not just edges), the algorithm is able to join unconnected regions of models. This may facilitate far better approximation, both visually and with respect to geometric error. The results we evaluated in terms of accuracy, precision, recall and F-measure for the TOSCA_80 and TOSCA_90 Databases are reported in Table 8.

Mesh reduction and re-meshing have a direct incidence on our approach since

N_k	Classifier		Accuracy (%)		Precision (%)		Recall (%)		F-measure	
	T80	**T90**	**T80**	**T90**	**T80**	**T90**	**T80**	**T90**	**T80**	**T90**
1	RF	RF	84.45	84.46	86.97	87.44	84.29	85.44	0.84	0.85
2	GBT	GBT	71.62	77.70	75.20	78.00	72.50	78.26	71.49	0.78
3	RF	GBT	71.62	73.65	73.50	76.85	74.09	75.29	0.73	0.75
4	RF	RF	79.73	75.00	81.97	78.32	80.95	77.08	0.80	0.76
5	GBT	RF	77.70	76.35	80.51	77.34	78.77	77.82	0.79	0.76
6	GBT	RF	74.32	73.65	80.82	74.74	75.59	73.06	0.76	0.73

Table 8: TSM accuracy, precision, recall and F-measure obtained with the gradient boosted trees (GBT) and the random forest (RF) classifiers for $N_{k=1\ldots6}$ for the TOSCA_80 (T80) and the TOSCA_90 (T90) Databases.

the latter is graph-based. Nevertheless, TSM is based on an approximation of Graph Edit Distance which is fault-tolerant to noise and distortion. Furthermore, our approach uses a set of descriptors which are invariant or at least oblivious under the most common geometrical deformations. As a result, TSM is robust with respect to mesh reduction and re-meshing. Despite the fact that the triangular reduction was relatively great (10% to 20%), the algorithm displayed a surprisingly high resilience to mesh reduction: accuracy was as high as 84.5% for the first neighborhood order for both a 10% and 20% mesh reduction as opposed to 88.51% without mesh reduction. These results were all obtained with the random forest classifier.

Our experimentation with the TOSCA, the SHREC11 watertight and the mesh-reduced TOSCA databases, as described earlier, reveals the performances and robustness of our approach.

Indeed, we obtained excellent results in terms of accuracy, precision, recall and F-measure for the TOSCA and the SHREC11 watertight databases. Our predicted time complexity was systematically confirmed by our experimentations. Our method outperforms CAM, GeodesicD2, DSR and RSH in terms of accuracy, precision and recall for the TOSCA database. The same remarks apply when our approach is compared to other benchmark methods in terms of F-measure.

7. Conclusions

In this paper, we presented a new matching algorithm for addressing the problem of comparing deformable 3D objects represented by graphs (triangular tessellations). The proposed approach is based on a new decomposition of triangular tessellations into triangle-stars. The resulting triangle-stars are used to determine the distance between triangular tessellations using the Hungarian algorithm. The proposed algorithm ensures a minimum number of disjoint triangle-stars, offers a better dissimilarity by covering a larger neighborhood in triangle-stars and uses a set of descriptors which are invariant or at least oblivious under the most common deformations. The proposed approach is based on an approximation of Graph Edit Distance, which is fault-tolerant to noise and distortion, making our approach suitable for comparing deformable objects. Classification is performed using supervised machine learning techniques. Our approach defines a metric space using graph embedding and graph kernel techniques. We proved that the proposed distance TSM is a pseudo-metric. Our experimental results, as obtained from various benchmark databases for deformable shapes (TOSCA and SHREC11), confirm the performances and the accuracy of our algorithm. In a near future, we plan to enrich the triangle-stars description while reducing the time complexity and improving the performances of the proposed graph matching algorithm. We also plan to combine our approach with deep learning.

Acknowledgments

Many thanks to Dr.**Hamida Seba** for her expertise in [12].
This work was partially supported by the program "Avenir Lyon Saint-Etienne".

References

[1] R. Myers, R. C. Wison, E. R. Hancock, Bayesian graph edit distance, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (6) (2000) 628–635.

[2] H. Bunke, A. Münger, X. Jiang, Combinatorial search versus genetic algorithms: A case study based on the generalized median graph problem, Pattern Recognition Letters 20 (11-13) (1999) 1271–1277.

[3] M. Gori, M. Maggini, L. Sarti, Exact and approximate graph matching using random walks, IEEE Trans. Pattern Anal. Mach. Intell. 27 (7) (2005) 1100–1111.

[4] K. Madi, H. Seba, H. Kheddouci, O. Barge, A graph-based approach for kite recognition, Pattern Recognition Letters 87 (2017) 186–194.

[5] H. Bunke, K. Shearer, A graph distance metric based on the maximal common subgraph, Pattern Recognition Letters 19 (3-4) (1998) 255–259.

[6] S. Sorlin, C. Solnon, J. Jolion, A generic graph distance measure based on multivalent matchings, in: Applied Graph Theory in Computer Vision and Pattern Recognition, 2007, pp. 151–181.

[7] A. Sanfeliu, K.-S. Fu, A distance measure between attributed relational graphs for pattern recognition, Systems, Man and Cybernetics, IEEE Transactions on SMC-13 (3) (1983) 353–362.

[8] A. Papadopoulos, Y. Manolopoulos, Structure-based similarity search with graph histograms, in: 10th International Workshop on Database & Expert Systems Applications, Florence, Italy, September 1-3, 1999, Proceedings., 1999, pp. 174–178.

[9] D. Conte, P. Foggia, C. Sansone, M. Vento, Thirty years of graph matching in pattern recognition, IJPRAI 18 (3) (2004) 265–298.

[10] K. Riesen, H. Bunke, Approximate graph edit distance computation by means of bipartite graph matching, Image Vision Comput. 27 (7) (2009) 950–959.

[11] Z. Zeng, A. K. H. Tung, J. Wang, J. Feng, L. Zhou, Comparing stars: On approximating graph edit distance, PVLDB 2 (1) (2009) 25–36.

[12] K. Madi, E. Paquet, H. Seba, H. Kheddouci, Graph edit distance based on triangle-stars decomposition for deformable 3d objects recognition, in: 2015 International Conference on 3D Vision, 3DV 2015, Lyon, France, October 19-22, 2015, 2015, pp. 55–63.

[13] J. W. H. Tangelder, R. C. Veltkamp, A survey of content based 3d shape retrieval methods, Multimedia Tools Appl. 39 (3) (2008) 441–471.

[14] H. Bunke, K. Riesen, Recent advances in graph-based pattern recognition with applications in document analysis, Pattern Recognition 44 (5) (2011) 1057–1067.

[15] P. Foggia, G. Percannella, M. Vento, Graph matching and learning in pattern recognition in the last 10 years, IJPRAI 28 (1).

[16] M. Vento, A long trip in the charming world of graphs for pattern recognition, Pattern Recognition 48 (2) (2015) 291–301.

[17] J. Yan, X.-C. Yin, W. Lin, C. Deng, H. Zha, X. Yang, A short survey of recent advances in graph matching, in: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, ICMR '16, 2016, pp. 167–174.

[18] E. Paquet, M. Rioux, A. Murching, T. Naveen, A. Tabatabai, Description of shape information for 2-d and 3-d objects, Signal processing: Image communication 16 (1) (2000) 103–122.

[19] M. Yu, I. Atmosukarto, W. K. Leow, Z. Huang, R. Xu, 3d model retrieval with morphing-based geometric and topological feature maps, in: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003), 16-22 June 2003, Madison, WI, USA, 2003, pp. 656–661.

[20] M. Kazhdan, B. Chazelle, D. Dobkin, T. Funkhouser, S. Rusinkiewicz, A reflective symmetry descriptor for 3d models, Algorithmica 38 (1) (2004) 201–225.

[21] M. Körtgen, G.-J. Park, M. Novotni, R. Klein, 3d shape matching with 3d shape contexts, in: The 7th central European seminar on computer graphics, Budmerice, Slovakia., Vol. 3, Budmerice, 2003, pp. 5–17.

[22] J. Xie, Y. Fang, F. Zhu, E. Wong, Deepshape: Deep learned shape descriptor for 3d shape matching and retrieval, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1275–1283.

[23] H. Sundar, D. Silver, N. Gagvani, S. J. Dickinson, Skeleton based shape matching and retrieval, in: 2003 International Conference on Shape Modeling and Applications (SMI 2003), 12-16 May 2003, Seoul, Korea, 2003, pp. 130–142, 290.

[24] S. Biasotti, S. Marini, M. Mortara, G. Patanè, An overview on properties and efficacy of topological skeletons in shape modelling, in: 2003 International Conference on Shape Modeling and Applications (SMI 2003), 12-16 May 2003, Seoul, Korea, 2003, pp. 245–256, 297.

[25] Skeleton graph matching vs. maximum weight cliques aorta registration techniques, Computerized Medical Imaging and Graphics 46, Part 2 (2015) 142 – 152, information Technologies in Biomedicine.

[26] M. Hilaga, Y. Shinagawa, T. Komura, T. L. Kunii, Topology matching for fully automatic similarity estimation of 3d shapes, in: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001, Los Angeles, California, USA, August 12-17, 2001, 2001, pp. 203–212.

[27] S. Biasotti, S. Marini, M. Mortara, G. Patanè, M. Spagnuolo, B. Falcidieno, 3d shape matching through topological structures, in: Discrete Geometry for Computer Imagery, 11th International Conference, DGCI 2003, Naples, Italy, November 19-21, 2003, Proceedings, 2003, pp. 194–203.

[28] V. Barra, S. Biasotti, 3d shape retrieval using kernels on extended reeb graphs, Pattern Recognition 46 (11) (2013) 2985–2999.

[29] E. Kalogerakis, S. Chaudhuri, D. Koller, V. Koltun, A probabilistic model for component-based shape synthesis, ACM Transactions on Graphics (TOG) 31 (4) (2012) 55.

[30] H. Laga, M. Mortara, M. Spagnuolo, Geometry and context for semantic correspondences and functionality recognition in man-made 3d shapes, ACM Transactions on Graphics (TOG) 32 (5) (2013) 150.

[31] Y. Kleiman, O. van Kaick, O. Sorkine-Hornung, D. Cohen-Or, SHED: shape edit distance for fine-grained shape similarity, ACM Trans. Graph. 34 (6) (2015) 235.

[32] D. Chen, X. Tian, Y. Shen, M. Ouhyoung, On visual similarity based 3d model retrieval, Comput. Graph. Forum 22 (3) (2003) 223–232.

[33] Y. Gao, Q. Dai, View-based 3d object retrieval: Challenges and approaches, IEEE MultiMedia 21 (3) (2014) 52–57.

[34] S. Zhao, H. Yao, Y. Zhang, Y. Wang, S. Liu, View-based 3d object retrieval via multi-modal graph learning, Signal Processing 112 (2015) 110–118.

[35] M. Novotni, R. Klein, A geometric approach to 3d object comparison, in: Proceedings International Conference on Shape Modeling and Applications, 2001, pp. 167–175.

[36] H. Sánchez-Cruz, E. Bribiesca, A method of optimum transformation of 3d objects used as a measure of shape dissimilarity, Image and Vision Computing 21 (12) (2003) 1027–1036.

[37] T. K. Dey, J. Giesen, S. Goswami, Shape segmentation and matching with flow discretization, in: Algorithms and Data Structures, 8th International Workshop, 2003, Ottawa, Ontario, Canada, July 30 - August 1, 2003, Proceedings, 2003, pp. 25–36.

[38] J. W. Tangelder, R. C. Veltkamp, Polyhedral model retrieval using weighted point sets, International journal of image and graphics 3 (01) (2003) 209–229.

[39] A. Shamir, A. Sharf, D. Cohen-Or, Enhanced hierarchical shape matching for shape transformation, International Journal of Shape Modeling 9 (2) (2003) 203–222.

[40] A. M. Bronstein, M. M. Bronstein, R. Kimmel, Efficient computation of isometry-invariant distances between surfaces, SIAM J. Scientific Computing 28 (5) (2006) 1812–1836.

[41] A. M. Bronstein, M. M. Bronstein, R. Kimmel, Calculus of nonrigid surfaces for geometry and texture manipulation, IEEE Trans. Vis. Comput. Graph. 13 (5) (2007) 902–913.

[42] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. V. Nguyen, R. Ohbuchi, Y. Ohkita, Y. Ohishi, F. Porikli, M. Reuter, I. Sipiran, D. Smeets, P. Suetens, H. Tabia, D. Vandermeulen, SHREC '11 track: Shape retrieval on non-rigid 3d watertight meshes, in: Eurographics Workshop on 3D Object Retrieval 2011, Llandudno, UK, April 10, 2011. Proceedings, 2011, pp. 79–88.

[43] H. Tabia, M. Daoudi, J. Vandeborre, O. Colot, A new 3d-matching method of nonrigid and partially similar models using curve analysis, IEEE Trans. Pattern Anal. Mach. Intell. 33 (4) (2011) 852–858.

[44] R. Osada, T. A. Funkhouser, B. Chazelle, D. P. Dobkin, Shape distributions, ACM Trans. Graph. 21 (4) (2002) 807–832.

[45] D. Vranic, 3d model retrieval, Ph.D. thesis, University of of Leipzig (may 2004).

[46] D. Saupe, D. V. Vranic, 3d model retrieval with spherical harmonics and moments, in: Pattern Recognition, 23rd DAGM-Symposium, Munich, Germany, September 12-14, 2001, Proceedings, 2001, pp. 392–397.

[47] Y. Fang, M. Sun, K. Ramani, Temperature distribution descriptor for robust 3d shape retrieval, in: IEEE Conference on Computer Vision and

Pattern Recognition, CVPR Workshops 2011, Colorado Springs, CO, USA, 20-25 June, 2011, 2011, pp. 9–16.

[48] M. Reuter, F. Wolter, N. Peinecke, Laplace-beltrami spectra as 'shape-dna' of surfaces and solids, Computer-Aided Design 38 (4) (2006) 342–366.

[49] M. Abdelrahman, M. T. El-Melegy, A. A. Farag, Heat kernels for non-rigid shape retrieval: Sparse representation and efficient classification, in: Ninth Conference on Computer and Robot Vision, CRV 2012, Toronto, Ontario, Canada, May 28-30, 2012, 2012, pp. 153–160.

[50] J. A. Bondy, Graph theory with applications.

[51] H. W. Kuhn, The hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1-2).

[52] J. Vleugels, R. C. Veltkamp, Efficient image retrieval through vantage objects, Pattern Recognition 35 (1) (2002) 69–80.

[53] J. E. Barros, J. C. French, W. N. Martin, P. M. Kelly, T. M. Cannon, Using the triangle inequality to reduce the number of comparisons required for similarity-based retrieval, in: Storage and Retrieval for Still Image and Video Databases IV, San Diego/La Jolla, CA, USA, January 28 - February 2, 1996, 1996, pp. 392–403.

[54] F. Serratosa, Fast computation of bipartite graph matching, Pattern Recognition Letters 45 (2014) 244–250.

[55] F. Serratosa, Speeding up fast bipartite graph matching through a new cost matrix, IJPRAI 29 (2).

[56] R. Jonker, A. Volgenant, A shortest augmenting path algorithm for dense and sparse linear assignment problems, Computing 38 (4) (1987) 325–340.

[57] A. Fischer, C. Y. Suen, V. Frinken, K. Riesen, H. Bunke, Approximation of graph edit distance based on hausdorff matching, Pattern Recognition 48 (2) (2015) 331–343.

[58] S. Bougleux, B. Gauz'ere, D. B. Blumenthal, L. Brun, Fast linear sum assignment with error-correction and no cost constraints, Pattern Recognition Lettersdoi:https://doi.org/10.1016/j.patrec.2018.03.032.

[59] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, S. J. Dickinson, Retrieving articulated 3-d models using medial surfaces, Mach. Vis. Appl. 19 (4) (2008) 261–275.

[60] P. Shilane, P. Min, M. M. Kazhdan, T. A. Funkhouser, The princeton shape benchmark, in: 2004 International Conference on Shape Modeling and Applications (SMI 2004), 7-9 June 2004, Genova, Italy, 2004, pp. 167–178.

[61] K. P. Murphy, Machine learning - a probabilistic perspective, Adaptive computation and machine learning series, MIT Press, 2012.

[62] I. J. Goodfellow, Y. Bengio, A. C. Courville, Deep Learning, Adaptive computation and machine learning, MIT Press, 2016.

[63] J. Reinders, Intel data analytics acceleration library.
URL https://software.intel.com/en-us/blogs/daal

[64] T. Fawcett, An introduction to ROC analysis, Pattern Recognition Letters 27 (8) (2006) 861–874.

[65] D. M. W. Powers, Evaluation: from precision, recall and f-factor to roc, informedness, markedness and correlation, School of Informatics and Engineering Technical Reports (SIE-07-001).

[66] M. Garland, P. S. Heckbert, Surface simplification using quadric error metrics, in: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1997, Los Angeles, CA, USA, August 3-8, 1997, 1997, pp. 209–216.