Computer-Aided Design 64 (2015) 22–32

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Analysis of an EMST-based path for 3D meshes[☆]

Vincent Itier[a,b], Nicolas Tournier[a,b], William Puech[a,*], Gérard Subsol[a], Jean-Pierre Pedeboy[b]

[a] LIRMM, Univ. Montpellier 2, CNRS, 161 rue Ada, 34 095 Montpellier Cedex 5, France
[b] Strategies S.A, 41-43 rue de Villeneuve, Parc des Affaires SILIC - BP 80429, 94583 Rungis, France

HIGHLIGHTS

- We analysed sensitivity of the EMST structure to obtain a more robust synchronization.
- We computed how a vertex can be moved without changing the connections.
- We present a new theoretical analysis and a way to visualize EMST robustness.
- We detect fragile area and to predict the 3D object robustness.
- Keywords are Euclidean minimum spanning tree, Sensitivity analysis, Synchronization.

ARTICLE INFO

Article history:
Received 9 April 2014
Accepted 22 February 2015

Keywords:
Euclidean minimum spanning tree
Sensitivity analysis
Synchronization

ABSTRACT

For several 3D data applications such as data-hiding or compression, data ordering is a major problem. We need to know how to achieve the same 3D mesh path between the coding and decoding stages. Various algorithms have been proposed in recent years, but we focus on methods based on Euclidean Minimum Spanning Trees (EMST). In this paper, we analyse the sensitivity of the EMST structure to obtain a more robust synchronization. We present a new theoretical analysis and a way to visualize EMST robustness. Moreover, this analysis can be useful in 3D data-hiding in order to detect fragile area and to predict the 3D object robustness during transmission on a noisy channel.

1. Introduction

Internet is very useful for broadcasting multimedia information. There are more and more 3D object exchanges in computer graphics, CAO and video games. Therefore, it is essential to produce efficient techniques for protecting, visualizing, sharing, printing and modifying these 3D objects. For these applications, it could be important to have a single 3D mesh path which orders the vertices. In this domain, this vertex ordering step is often called the synchronization step. Indeed, contrary to the 2D imaging field, where there is a trivial path with rows and columns, the case of 3D objects is more complex, even though the 3D mesh is semi-regular or regular. For example, in 3D data-hiding, it is essential to locate where the binary embedded data is distributed. This kind of path is used for synchronization with the aim of keeping the same order at inserting and extracting stages. A survey of 3D watermarking techniques has been proposed by Wang et al. [1]. Watermarking techniques are interesting to protect the file content and also to embed meta-data. Data-hiding may be a way to have new functionalities, *i.e.* keeping the standard format without increasing the size. Another example of the benefits of having a single 3D mesh path is to produce a deterministic traversal of the mesh for 3D compression, as presented in a survey by Peng et al. [2].

The step which gives a mesh order is one of the main difficulties in compression, watermarking or visualization. Furthermore, in some specialized areas (medicine, industry), the position of vertices and the connectivity between them (in the 3D mesh) should not be affected by the path building process. Various methods have been proposed, but we are interested by the Euclidean Minimum Spanning Tree (EMST) method proposed by Amat et al. [3]. The authors proposed a scheme which does not move any mesh vertex. They used an EMST to be able to scan the mesh in a unique manner for data synchronization.

Since the method is fragile, an EMST sensitivity analysis is necessary to determine the robustness threshold of an EMST-based

[☆] This paper has been recommended for acceptance by Hiromasa Suzuki.
[*] Corresponding author.
E-mail addresses: vincent.itier@lirmm.fr (V. Itier), nicolas.tournier@lirmm.fr (N. Tournier), william.puech@lirmm.fr (W. Puech), gerard.subsol@lirmm.fr (G. Subsol), jp.pedeboy@cadwin.com (J.-P. Pedeboy).

http://dx.doi.org/10.1016/j.cad.2015.02.005
0010-4485/© 2015 Elsevier Ltd. All rights reserved.

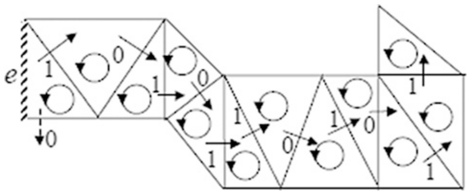

Fig. 1. Scheme of the path used for the data-hiding method proposed by Ohbuchi et al. [11].

on a 3D mesh. In this paper, we analysed the EMST sensitivity by computing how a vertex can be moved without changing the connections in the EMST. Since the problem is very difficult, we made some assumptions in order to make it tractable. It may also be interesting to quantify the EMST robustness. A sensitive analysis can be useful for some applications such as 3D reconstruction [4], 2D object recognition and classification [5], watermarking [3,6], compression [7] and segmentation [8]. For example, the proposed analysis could be used to improve the choice of the mark vertex selecting proposed by Wang et al. [9].

The rest of the paper is organized as follows. In Section 2, we deal with the 3D path building issue. We present various classes of techniques, such as the synchronization by data structure (EMST). In Section 3, we present the problem of EMST sensitivity. Then, in Section 4, we propose a new approach to this problem by analysing the displacement of the vertex at each step of Prim's algorithm [10]. We assessed our theory in order to quantify the displacement of the vertices. The results are given in Section 5 and our approach is validated. The discussion is concluded in Section 6 and future work directions are mentioned.

2. Defining a 3D mesh path

3D processing is a comparatively new multimedia research field. One of the main problems in these applications is the ordering of 3D model data. In this section, we present ordering techniques based on a single path of 3D model. First, we present methods which only perform a path on a part of the mesh. Then we introduce some methods which order the patches created on the mesh. Finally, we present techniques that define a path along all vertices of the mesh.

2.1. Band of vertices

One of the first paths for data-hiding watermarking techniques was proposed by Ohbuchi et al. [11]. The algorithm and synchronization of the embedded message are quite simple. The idea, illustrated in Fig. 1, consists of duplicate facets of the mesh along a band that encodes the message. To start the duplication, a starting edge and an orientation of the triangles must be defined. To encode a '0', from the current edge, they have to duplicate the first edge they meet during the exploration, and for embedding a '1' they have to duplicate the second edge. The duplicated edge becomes the current one and the algorithm continues until all bits of the message are embedded.

With this approach, the mark is robust to geometrical modifications such translation, rotation and scaling. But it is visible and easily detectable, *i.e.* the algorithm is not secure. Moreover, vertices and facets are added and the size of the mesh increases as a function of the message size. Nevertheless, it is one of the first paths proposed for 3D blind data-hiding methods.

The idea behind creating a band, or performing a scan on the mesh to synchronize the hidden message is classic reasoning. For example, Mao et al. [12] and Cayre et al. [13] proposed almost the same approach to scan the mesh.

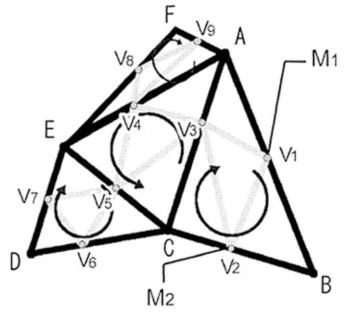

Fig. 2. Scheme of the synchronization method proposed by Mao et al. [12].

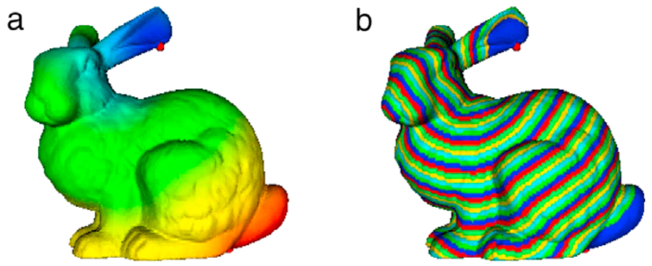

Fig. 3. Scheme of a mesh splitting into regions proposed by Luo and Bors [15]: (a) Geodesic map. (b) Iso-geodesic mesh strip generation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

In the approach proposed by Mao et al., they built a triangle path in the mesh to synchronize the data to be embedded, as illustrated in Fig. 2. They selected a starting triangle Δ, an edge e of Δ, and an orientation of this triangle. At each step, they choose one of the neighbour triangles as a function of the orientation and the previous step. From the current edge c, they scan the triangle in the direction of the orientation and select the last edge e of the current triangle Δ. Then they select the neighbouring triangle Δ', if it exists, such that e is the common edge between Δ and Δ'. Then Δ' becomes the current triangle, and will be scanned in the opposite direction. The orientation to scan the triangle alternates as illustrated in Fig. 2. The operation continues until we do not need triangles to embed the hidden message.

In principle, this method does not create geometric error. However, the watermark can be detectable because it is a high density area in the mesh. Furthermore, such methods are obviously not robust against connectivity attacks.

2.2. Patch ordering

In this section, we present some methods that create patches on the 3D mesh. This is an interesting approach to deal with certain constraints in the 3D mesh area such as malicious attacks or visible deformations induced by watermarking for example. In segmentation areas, a 3D mesh is partitioned following the model semantics. For example, Tierny et al. [14] proposed a method based on construction of the skeleton of the mesh that produces a small number of patches. These patches are quite semantically correct, but there are few of them and they are not ordered. Defining a path in a mesh requires a higher number of patches and a very deterministic algorithm.

Luo and Bors [15] proposed an efficient watermarking scheme based on regions of equal geodesic distance. These distances are calculated from a chosen vertex (in red in Fig. 3).

Each strip in Fig. 3(b) is used for embedding a single bit, while the region around the source, which is shown in blue, is trimmed away. Patches are created thanks to their geodesic distance from a vertex, so they are ordered by construction.

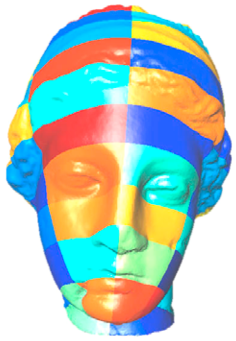

Fig. 4. Scheme of the set of patches proposed by Wang et al. [16].

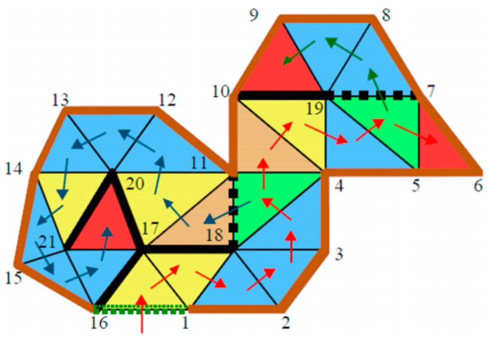

Fig. 5. Scheme of the *Edgebreaker* [18] synchronization method.

Wang et al. [16] proposed another synchronization for their watermarking method that is based on generating a cylindrical system patch by discretizing its h and θ domains. Patches are ordered according to their spatial locations. In another paper, Wang et al. [17] proposed to use manifold harmonics and to quantify the amplitudes of some low frequencies coefficients in order to hide a 16-bit watermark (see Fig. 4).

2.3. Mesh traversal

The *Edgebreaker* mesh compression algorithm proposed by Rossignac [18] is a mono-resolution algorithm, *i.e.* it does not allow access before full load. It puts the vertices and facets in order at the same time without redundancy, as illustrated in Fig. 5. The scan is also unique, as it depends on the first vertex and the first triangle chosen.

This technique is based on region growing that incrementally encodes facets and their relations. Thanks to this algorithm, we obtain a unique sequence of vertices that totally defines the mesh.

To illustrate the need for a well defined order in a 3-D mesh, we present two watermarking techniques which are based on previous synchronization methods. The watermarking of Bogomjakov et al. [19] is based on swapping elements in the file. The synchronization step can be done by any kind of deterministic mesh traversal and they use the traversal performed by the *Edgebreaker* [18] algorithm. This example shows that ordering vertices of a 3-D mesh in a robust way is a relevant goal to produce more and more applications in this area. Another method proposed by Wang and Men illustrates that the synchronization is determined only by the file ordering [20].

In the method proposed by Amat et al. [3], data embedding in the mesh is based on modification of the connection between the vertices in the mesh, without moving the vertices. In order to synchronize the message, an EMST is computed. Fig. 6(a) illustrates the EMST of a horse mesh (504 vertices). The EMST is unique, depending on a seed vertex v_0, while the path of vertices is also unique.

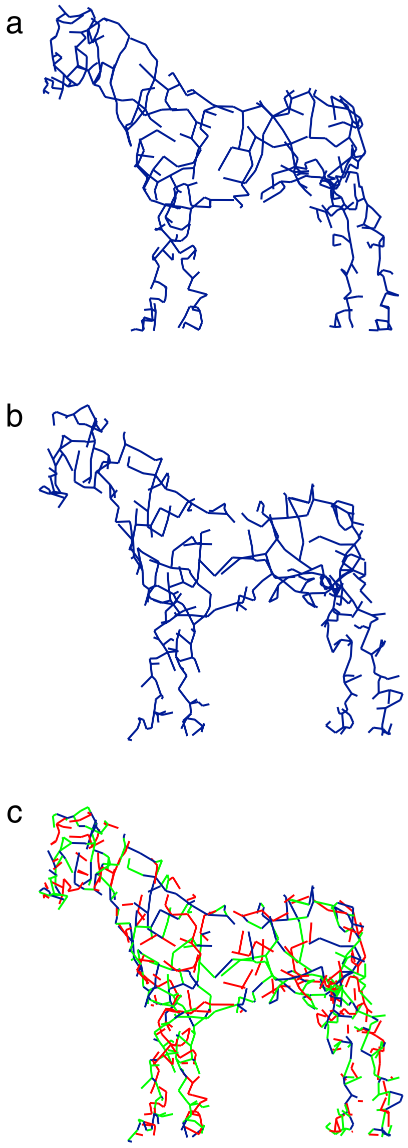

Fig. 6. (a) The EMST of the original mesh. (b) The EMST of a noisy mesh with $\sigma = 10^{-2}$. (c) Comparison between the two EMST with common edges in blue, original edges in red and noisy edges in green. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

This can be a synchronization tool because we can scan the EMST with a single path. Amat et al. [3] selected quadruples, a vertex and its three sons in the EMST to embed one bit per quadruple, if possible. In order to avoid visual distortion and desynchronization, the quadruples must verify three conditions: coplanarity, the measure of the angle formed by the two triangles must be close to 0; convexity, to scan the same geometrical space; and if two quadruples are neighbours or overlapped, then only one of them is used for the embedding.

However, this method produces fragile watermarking, *i.e.* if the mesh is modified then the watermark is not extractable. The problem is to know how this method is stable. Indeed, if it is fragile enough to detect any modification. Conversely, we want to know how the method is robust against vertex displacement in order to find new tracks for robust watermarking. We focus on this original method to quantify the fragility of the data synchronization by studying the robustness of the EMST. Indeed, as we can see Fig. 6(b) which is the EMST of the noisy horse mesh with $\sigma = 10^{-2}$, the EMST computed on a mesh is very sensitive to vertex displacement.

Fig. 6(c) illustrates the comparison between the original horse mesh and the noisy one. The problem is well known in the graph theory for the minimum spanning tree (MST), but has not been very well studied in the geometrical case (*i.e.* EMST). We discuss the state of the art of the analysis of the MST sensitivity in Section 3.

3. Problem of the Euclidean minimum spanning tree sensitivity

In this section, we present how the problem of EMST sensitivity may be defined with various approaches.

3.1. Notations and Prim's algorithm

As per this paper, we have taken a cloud of vertices V and explored how we can move a vertex in space without changing the connections E in the Euclidean minimum spanning tree $T = (V, E)$. Let $G = (V, E)$ be a graph with n nodes and m edges. An EMST is MST which is based on the Euclidean distance, *i.e*, a tree $T = (V, E_T)$ which joins all the vertices of a graph $G = (V, E)$ using the edges $e_i = \{v_s, v_t\} \in E$, with a weight $\omega(e_i) \in \mathbb{R}^+$, that minimizes the total weight $\sum_{e_i \in E_T} \omega(e_i)$.

Our approach is based on Prim's algorithm in order to have an incremental algorithm. At each step we have a sub-tree $T_i = (V_i, E_i)$ of the final EMST. The algorithm starts with a seed vertex $v_0 \in V$. We note $(T_i)_{i \leq n} = ((V_i)_{i \leq n}, (E_i)_{i \leq n})$ the tree sequence representing the EMST construction pattern.

With these notations $T_0 = (\{v_0\}, \emptyset)$. At each step $i > 0$, the algorithm adds the closest vertex $v_i \in \overline{V}_{i-1}$ to V_{i-1} and to E_{i-1} it adds the connection between v_i and the closest vertex of v_i in V_{i-1} that we call its "father" and denote by $f(v_i)$.

Hence, we deduce:

$$V_i = V_{i-1} \cup \{v_i\}; \tag{1}$$

$$E_i = E_{i-1} \cup \{v_i, f(v_i)\}. \tag{2}$$

3.2. MST sensitivity problem

The Minimum Spanning Tree (MST) is a well-known problem in graph theory. It is a polynomial problem that is solved by two famous algorithms, *i.e* Prim's [10] and Kruskal's [21]. For a given MST T, it is interesting to know which connections are fragile, and which are not. The MST sensitivity problem may answer this question. MST sensitivity is also a polynomial problem. In this section, we have presented an approach for solving the problem and we draw a conclusion for our application.

For Gordeev [22,23], the MST sensitivity analysis is an optimization problem based on matroids. The aim is to determine the maximum intensity of a disruptive vector such that the solution of the optimization problem remains a solution after the perturbation. Gordeev considers the following model: let $D_m = \{T_1, \ldots, T_q\}$, with $(q > 1)$ being a system of subsets of E called trajectories; $A = (a_1, \ldots, a_m) \subset \mathbb{R}^m$ such that $\forall i\ a_i = \omega(e_i)$, the weight of the edges of the graph G and $\omega(T_A)$ a functional called the trajectory length for A, such that $\omega(T_A) = \sum_{e_i \in T} a_i$. Therefore, the combinator problem is defined with the pair (E, D_m), A is the variable to optimize in order to minimize the functional $\omega(T_A)$. Gordeev models the MST problem with D_m, the set of all spanning trees of G in which the MST is a trajectory that minimizes the functional $\omega(T_A)$.

Let $\psi(A)$ be the index set i of the optimal trajectories τ_i of the problem for a given A, and $B \in \mathbb{R}^m$, such that for $\epsilon \in \mathbb{R}^*_+$, $\|B\| < \epsilon$ is a perturbation vector. Gordeev talks about ϵ-stability when $\psi(A + B) \subset \psi(A)$. In the MST problem, for a given noise intensity ϵ, some MST are always solutions of the MST problem after the perturbation. He deduces a stability radius $\rho(A) = \sup \epsilon$, such that A is ϵ-stable for the problem. Its algorithm is polynomial and its complexity is $O(n^3 m \log(\frac{n^2}{m}))$ and for a complete graph $O(n^5)$.

For Dixon et al. [24] the sensitivity analysis problem is, for a given graph $G = (V, E)$ and $T = (V, E_T)$ its MST, to know how each edge value can be modified (for $e_i \in E_T$ and $e_i \in E \backslash E_T$) without changing the connections of T. They divide the problem into two parts. For $\forall e_i \notin E_T$, they compute how much they can decrease $\omega(e_i)$ without changing the MST and, for all the edges $e_i \in E_T$, they compute how much they can raise the weight of e_i, $\omega(e_i)$ without changing the MST. This step has a linear time complexity as a function of the number of edges.

The aim of Yaman et al. [25] is to introduce a robust version of the MST where the edge costs are specified as interval numbers. It is a spanning tree such that the weight of the tree minimizes the maximum deviation from the MST. They introduce the notion of a weak tree, and it is an MST for some scenarios. In other words, for different edge valuations, the weak tree is not always an MST. Then, an edge is a weak edge if it lies on some weak tree. On the contrary, an edge is a strong edge if it lies on an MST for all possible edge evaluations. Their goal is to define a robust spanning tree, so they introduce two robustness measures for the MST problem, "absolute robustness" and "relative robustness". These measures are used to characterize the worst case scenario, and then they use mixed integer programming to find a robust spanning tree. This method is very interesting for small graphs but it has huge complexity.

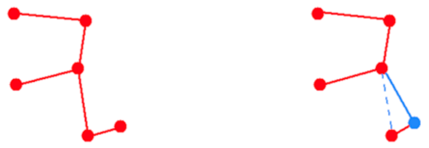
Fig. 7. Example illustrating the EMST sensitivity problem.

3.3. EMST sensitivity problem

For each step of the Prim's algorithm, we want to compute the area in which the vertex $v_i \in V_i$ can be moved without changing the connection in the EMST. This area depends on the seed vertex (denoted v_0) and the vertices selected before v_i. Fig. 7 illustrates that the EMST changes when a vertex moves too much. Distinguish EMST between the vertex ordering is necessary to well understand the analysis. Nevertheless, we point out that the ordering given by Prim's algorithm is dramatically dependent of the EMST stability.

To compute this area we make the following simplification assumptions on the disruption of vertices:

Assumption 3.1. At the step $i > 0$ of Prim's algorithm, we will disturb only the position of the vertex v_i, resulting in the disturbed vertex v^*;

Assumption 3.2. The geometric disruption will be restricted to only be along the half-line $]f(v_i); v_i)$.

After the perturbation, $T^*_i = (V^*_i, E^*_i)$ denotes the graph sequence with $V^*_i = \{v^*_0, v^*_1, \ldots, v^*_i\}$. Suppose that $\forall k;\ k < i$ the EMST at step k is always the same $(T_k = T^*_k)$. This is an important hypothesis, and we note that it is not always verified. Indeed, we simplified the problem by taking into account these assumptions, in order to compute the intrinsic vertex properties. At the step i, supposing that $T^*_k \neq T_k$, $\forall k \neq i$, implies that the radius r_i of v_i depends of the previous ones.

Then, to keep the same connections in the EMST, we need to verify these two conditions:

1. $v^* = v^*_i$, v^* is selected at the ith step of Prim's algorithm;
2. $f(v^*) = f(v^*_i)$, the father of v^* is still the father of v_i.

4. Analysis of the Euclidean minimum spanning tree sensitivity

In this section, we propose to analyse the sensitivity of the EMST by analysing the vertex displacement. In this paper, we limit the vertex displacement analysis along the half-line $]f(v_i); v_i)$. To compute the possible displacement of each vertex without changing the connection in the EMST, we divide the problem into two parts. In Section 4.1, we present how the vertex v_i can come up to $f(v_i)$. We compute a minimum distance limit of $\omega(\{v^*, f(v_i)\})$ denoted by d_i^- and deduce a displacement radius r_i^-. In Section 4.2, we explain how v_i can move away from $f(v_i)$. We also compute a maximum distance limit of $\omega(\{v^*, f(v_i)\})$ denoted by d_i^+ and deduce a displacement radius r_i^+. Then, in Section 4.3 we show how to keep the EMST stable at the step $i+1$. Finally, in Section 4.4, we define a global displacement radius denoted by r.

4.1. Minimum distance limit (r^-)

In this section, we are interested in the approximation of v_i to its father $f(v_i)$. As we know, $f(v_i)$ is the closest vertex of v_i in V_{i-1}, as illustrated in Fig. 8. Obviously, $f(v_i)$ was selected before v_i in Prim's algorithm. Let $f(v_i) = v_k$ and $V_k = \{v_j : j \le k\}$, then we deal with the vertices $v_j \in V_i \backslash V_k$ and the edges $e_j \in E_{i-1} \backslash E_k$ to verify the following properties.

Proposition 4.1. *Let $T_i = (V_i, E_i)$ be the state of the EMST at the ith step of Prim's algorithm, with the previous notations, if $E_{i-1} \backslash E_k \neq \emptyset$, we have:*

$$\forall e_j \in E_{i-1} \backslash E_k; \quad \omega(e_j) < \omega(\{f(v_i), v_i\}). \tag{3}$$

Proposition 4.2. *Let d_i^- be the minimum distance between $f(v_i)$ and v^*, then in order to keep the same connections in the EMST we must have $\omega(v^*, f(v_i)) > d_i^-$ with:*

$$d_i^- = \max\{\omega(e_j) : e_j \in E_{i-1} \backslash E_k\}. \tag{4}$$

This proposition guarantees that $T_{k+1}^* = T_{k+1}, \dots, T_i^* = T_i$, under Assumption 3.1, i.e. $T_j^* = T_j, j \in \{0, \dots, k\}$. We deduce the displacement radius:

$$r_i^- = \omega(\{f(v_i), v_i\}) - d_i^-. \tag{5}$$

Proof. Let $e_l = \{f(v_l), v_l\}$ be the edge, if it exists, verifying $e_l = \max_{e_j \in (E_{i-1} \backslash E_k)} \omega(e_j)$. If this edge does not exist, $d_i^- = 0$, so we can move the vertex v_i as close as we want to $f(v_i)$. We suppose that there is at least one edge e_l:

$$d_i^- = \omega(f(v_l), v_l).$$

It is important to note that $k \le l < i$, in other words, in the chronological vertex selection in Prim's algorithm, v_k is selected before v_l, and v_l before v_i. To demonstrate *reductio ad absurdum*, Proposition 4.2 must be verified to keep the same order in the sequence $(V_j^*)_{j \le i}$.

Supposing:

$\omega(f(v^*), v^*) \le d_i^- \Rightarrow \omega(f(v^*), v^*) \le \omega(f(v_l^*), v_l^*)$. At the $(l-1)$th step of Prim's algorithm, we know $f(v_i^*), f(v_l^*) \in V_{l-1}$. According to the hypothesis $\omega(f(v^*), v^*) \le \omega(f(v_l^*), v_l^*)$, so v^* will be chosen at the $(l-1)^{th}$ step. That is in contradiction with the EMST stability.

Obviously v_i is the closest node of $f(v_i)$ in $V \backslash V_i$, if we move v_i along the half-line $]f(v_i); v_i)$ closer to $f(v_i)$, then the resulting vertex $v^* \in V \backslash V_i$ is the closest node of $f(v_i)$. In conclusion, the father of v_i is also the father of v^* with this displacement. □

4.2. Maximum distance limit (r^+)

Now we are interested in the distance of v_i from its father $f(v_i)$. In order to keep the EMST connections up to step i of Prim's algorithm, we divide this problem into two cases. Firstly, we are looking for the second closest vertex $s(v_i)$ to V_{i-1} to select vertex v^* at the ith step, as illustrated in Fig. 9(a). We consider the distance $\omega((f \circ s)(v_i), s(v_i))$.

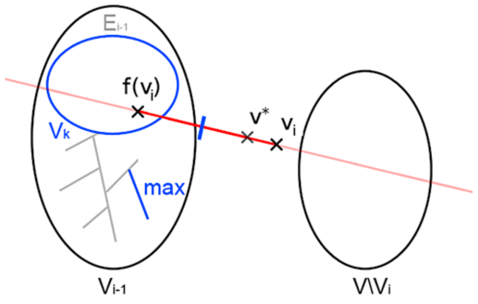

Fig. 8. Scheme of the minimum limit computing.

Proposition 4.3. *Let T_i be the state of the EMST at the ith step of Prim's algorithm, then in order to keep the connection of the EMST at step i we need to verify this first condition:*

$$\omega(f(v_i), v*) < \omega((f \circ s)(v_i), s(v_i)). \tag{6}$$

We denote $d_i^1 = \omega((f \circ s)(v_i), s(v_i))$.

Secondly, to keep the same father $f(v_i)$, we compute the intersection $x(v_k)$ between the half-line $]f(v_i); v_i)$ and the perpendicular bisector of the segment $[f(v_i), v_k]$ $(v_k \in V_{i-1}, v_k \neq f(v_i))$, as illustrated in Fig. 9(b).

Proposition 4.4. *Let T_i be the state of the EMST at the ith step of Prim's algorithm, then in order to keep the connection of the EMST at step i we need to verify this second condition:*

$$\omega(f(v_i), v^*) < \min\{\omega(f(v_i), x(v_k)) : v_k \in V_{i-1}, v_k \neq f(v_i)\}. \tag{7}$$

We denote $d_i^2 = \min\{\omega(f(v_i), x(v_k)) : v_k \in V_{i-1}, v_k \neq f(v_i)\}$.

Proposition 4.5. *Let T_i be the state of the EMST at the ith step of Prim's algorithm, then in order to keep the connection of the EMST at step i we need to verify $\omega(f(v_i), v^*) < d_i^+$, with:*

$$d_i^+ = \min\{d_i^1, d_i^2\}. \tag{8}$$

Under Assumptions 3.1 and 3.2, i.e. $T_j^* = T_j, j \in \{0, \dots, i-1\}$. We deduce the displacement radius:

$$r_i^+ = d_i^+ - \omega(v_i, f(v_i)). \tag{9}$$

Proof. Let us demonstrate *reductio ad absurdum* that $\omega(f(v^*), v^*) < d_i^1$ must be verified to keep the same order in the sequence $(V_i^*)_{0<i<n}$.

We suppose $\omega(f(v^*), v^*) \ge d_i^1 = \omega(s(v_i), (f \circ s)(v_i))$. According to this hypothesis, v_i is the closest vertex of V_{i-1}, and $s(v_i)$ the second one. Then v_i is disturbed in v^* but the other vertices do not move. Moreover $(f \circ s)(v_i), f(v^*) \in V_{i-1}$ and $s(v_i), v^* \in V \backslash V_{i-1}$. Prim's algorithm at step i chooses the closest vertex of V_{i-1} which is $s(v_i)$. v^* is too far from $f(v_i)$, so to verify the condition of our EMST stability problem $\omega(f(v^*), v^*) < d_i^1$. □

Let $v_k \in V_{i-1}$ be a vertex satisfying the relation $\omega(v_k, x(v_k)) = \min_{v_j \in V_{i-1}} \omega(v_j, x(v_j))$. Obviously, on the line $(f(v_i), v_i)$, the vertices $\{f(v_i), v_i, x(v_k)\}$ are aligned in this order.

Moreover, $x(v_k)$ is the equidistant vertex between $f(v_i)$ and v_k. It clearly separates the half-line $]v_i; x(v_k))$ into two parts:

- $\forall v^* \in]v_i; x(v_k)[$, $\omega(f(v_i), v^*) < d(f(v_i), x(v_k))$, v^* is closer to $f(v_i)$ than v_k;
- $\forall v^* \in]x(v_k); \infty)$, $\omega(f(v_i), v^*) > d(f(v_i), x(v_k))$, v^* is closer to v_k than $f(v_i)$.

It proves the proposition. □

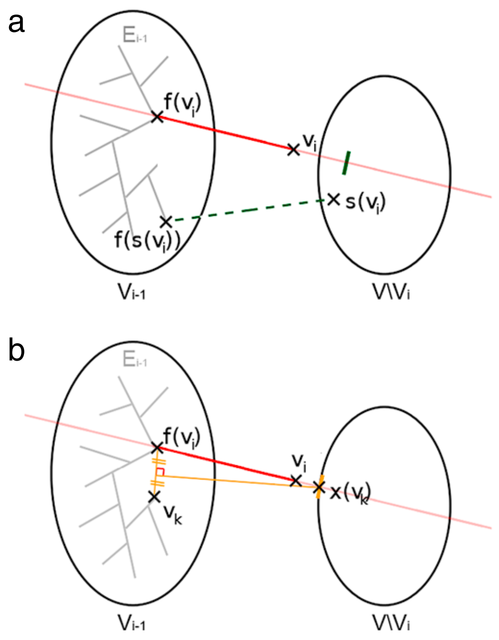

Fig. 9. Schemes of the maximum limit computing: (a) The second closest vertex $s(v_i)$ to V_{i-1}. (b) Intersection $x(v_k)$ between the half-line $]f(v_i); v_i)$ and the perpendicular bisector of the segment $[f(v_i), v_k]$ $(v_k \in V_{i-1})$.

4.3. v_{i+1} verification

The stability of the EMST, requires also that the next vertex is still the same. Suppose that v_i is selected at the ith step of Prim's algorithm, the next chosen vertex is v_{i+1}, $v_{i+1} = \text{argmin}_{v_j} \{w(v_k, v_j) : v_k \in V_i, v_j \in V\backslash V_i\}$. We have shown that under some assumptions $v_i = v_i^*$, in the same way, we show that we can restrict the displacement of v_i to have the same next vertex and maintain the EMST at the $i+1^{th}$ step of Prim's algorithm. We divide this problem in two cases, indeed the next chosen vertex is $s(vi)$, described in the previous section, or is given by $v_{i+1} = \text{argmin}_{v_j} \{w(v_i^*, v_j) : v_j \in V\backslash V_i\}$. Thus this case, is included in the previous analysis.

In the second case, to keep the same next vertex v_{i+1}, we compute the intersection $x(v_k)$ between the half-line $]f(v_i); v_i)$ and the perpendicular bisector of the segment $[v_{i+1}, v_k]$ $(v_k \in V\backslash V_i,$ $v_k \neq v_{i+1})$, as illustrated in Fig. 10.

Proposition 4.6. *Let T_i be the state of the EMST at the ith step of Prim's algorithm, then in order to keep the connection of the EMST at step $i + 1$, we need to verify:*

$$\omega(v^*, v_{i+1}) < \min\{\omega(v_i, x(v_k)) : v_k \in V\backslash V_i, v_k \neq v_{i+1}\}. \tag{10}$$

We denote $d_i^3 = \min\{\omega(v_i, x(v_k)) : v_k \in V\backslash V_i\}$.

Proposition 4.7. *Let T_i be the state of the EMST at the ith step of Prim's algorithm, then in order to keep the connection of the EMST at step i we need to verify $\omega(v^*, v_{i+1}) < d_i^3$, we deduce the displacement radius:*

$$r_i^2 = d_i^3. \tag{11}$$

4.4. Displacement radius (r)

In Sections 4.1 and 4.2, we defined two radii of displacement r^+ and r^-. In order to have a **single** measure of the possible vertex displacement without changing the EMST, we need to verify

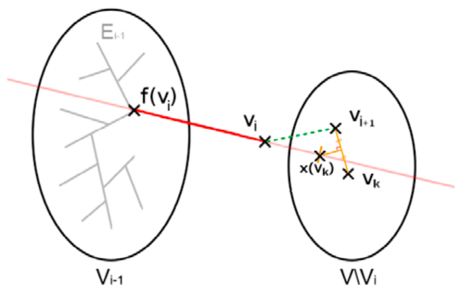

Fig. 10. Scheme of the limit to the next vertex: Intersection $x(v_k)$ between the half-line $]f(v_i); v_i)$ and the perpendicular bisector of the segment $[v_{i+1}, v_k]$ $(v_k \in V\backslash V_i)$.

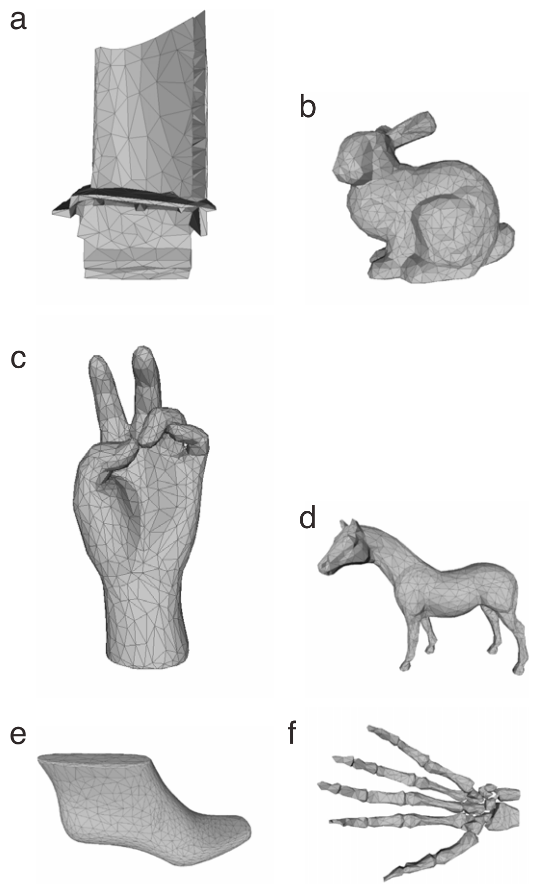

Fig. 11. Selection of 3D objects: (a) Blade, (b) Bunny, (c) Hand, (d) Horse, (e) Shoe, (f) Skeleton.

all the conditions that allow us to compute these parameters independently of the displacement direction.

Let $r_i = r_i = \min\{r_i^+, r_i^-, r_i^2\}$ be the displacement radius of the vertex v_i along the half-line $]f(v_i); v_i)$. Then $x = \frac{1}{\|f(v_i).v_i\|}(v_i - f(v_i))$ denotes the normalized director vector of the half-line $]f(v_i); v_i)$.

Therefore, if $v^* \in]v_i - r \cdot x; v_i + r \cdot x[$ the EMST will not be modified at the ith step of Prim's algorithm. We have defined a scope where the EMST does not change, in this context the vertex ordering done by the Prim algorithm is stable. However, we are aware that modifying the EMST could change dramatically the order defined, even if the percentage of common edge is high.

5. Experimental results

In this section, we experimentally analysed the minimum and maximum distance limits r_i^- and r_i^+ presented in Section 3 for

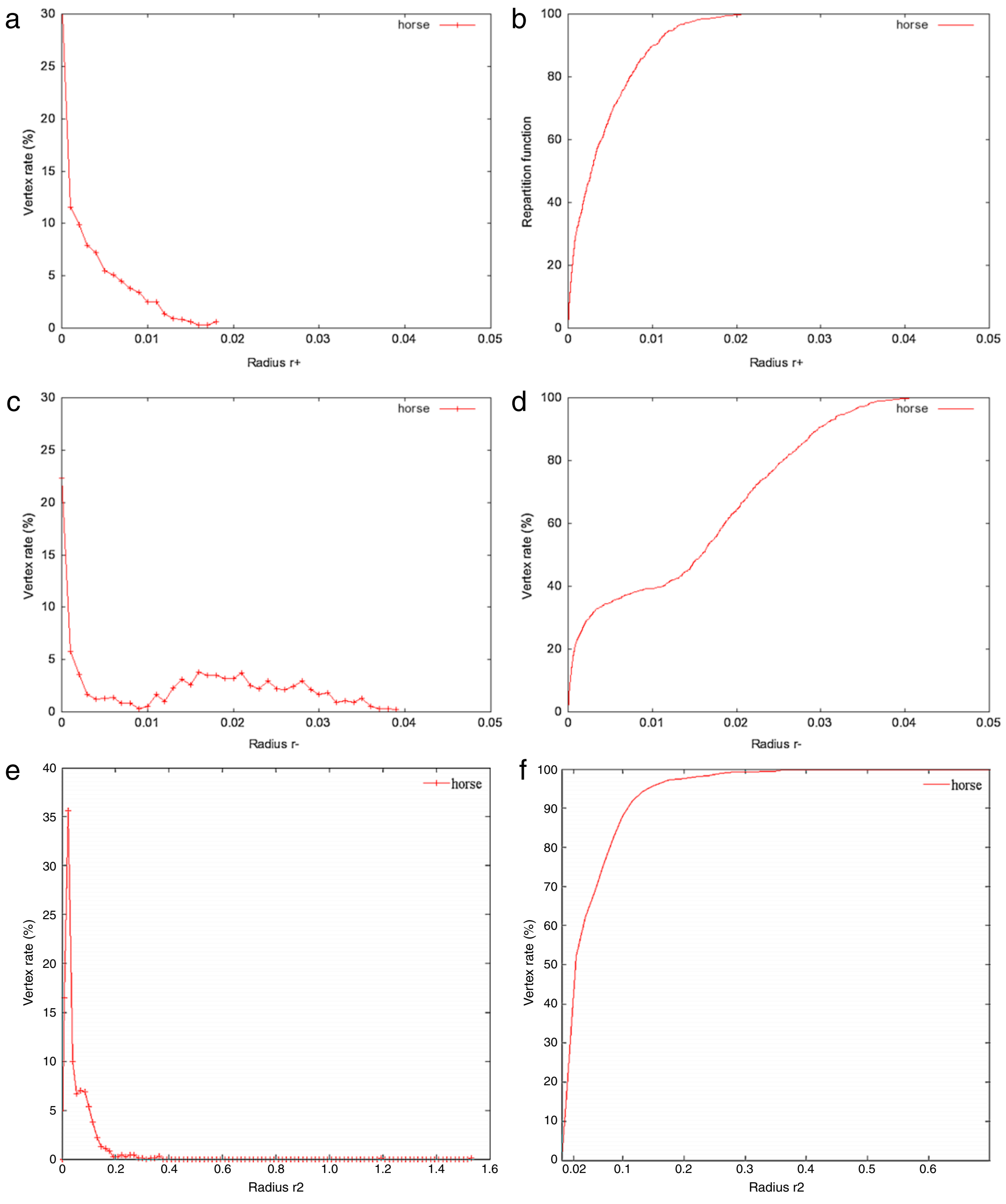

Fig. 12. Distribution of the vertices of the mesh *Horse* as a function of: (a) The Minimum distance limit r^+ (sampled with a step of 10^{-3}). (c) The maximum distance limit r^- (sampled with a step of 10^{-3}). (e) The maximum distance limit r^2 (sampled with a step of 10^{-3}). (b) Cumulative function of r^+. (d) Cumulative function of r^-. (f) Cumulative function of r^2.

each vertex v_i of several 3D objects. In Section 5.1, we describe the experimental conditions. Then, in Section 5.2 we present a full example with the 3D object *Horse* and an analysis of the vertex displacement to comment the results. We also present the result of our analysis by viewing the robust areas on the 3D object *Horse*. Finally, in Section 5.3 we validate the experimental results with the proposed theoretical analysis.

5.1. Experimental conditions

For the experiments, we used a database consisting of more than 20 3D meshes selected from various sources (Stanford University Graphics Laboratory,[1] MADRAS project,[2] Strategies S.A[3] and Aimatshape[4]). Their shapes are very different, as illustrated in Fig. 11, and they are used for different application fields, such as CAD, manufacturing, medicine or entertainment. In this section, we assume that the first vertex is randomly selected. Indeed, if we change the selection of the first vertex $v_0 \rightarrow v_0'$, the same EMST

1 http://www-graphics.stanford.edu.
2 http://www-rech.telecom-lille1.eu/madras.
3 http://www.cadwin.com.
4 http://www.aimatshape.net.

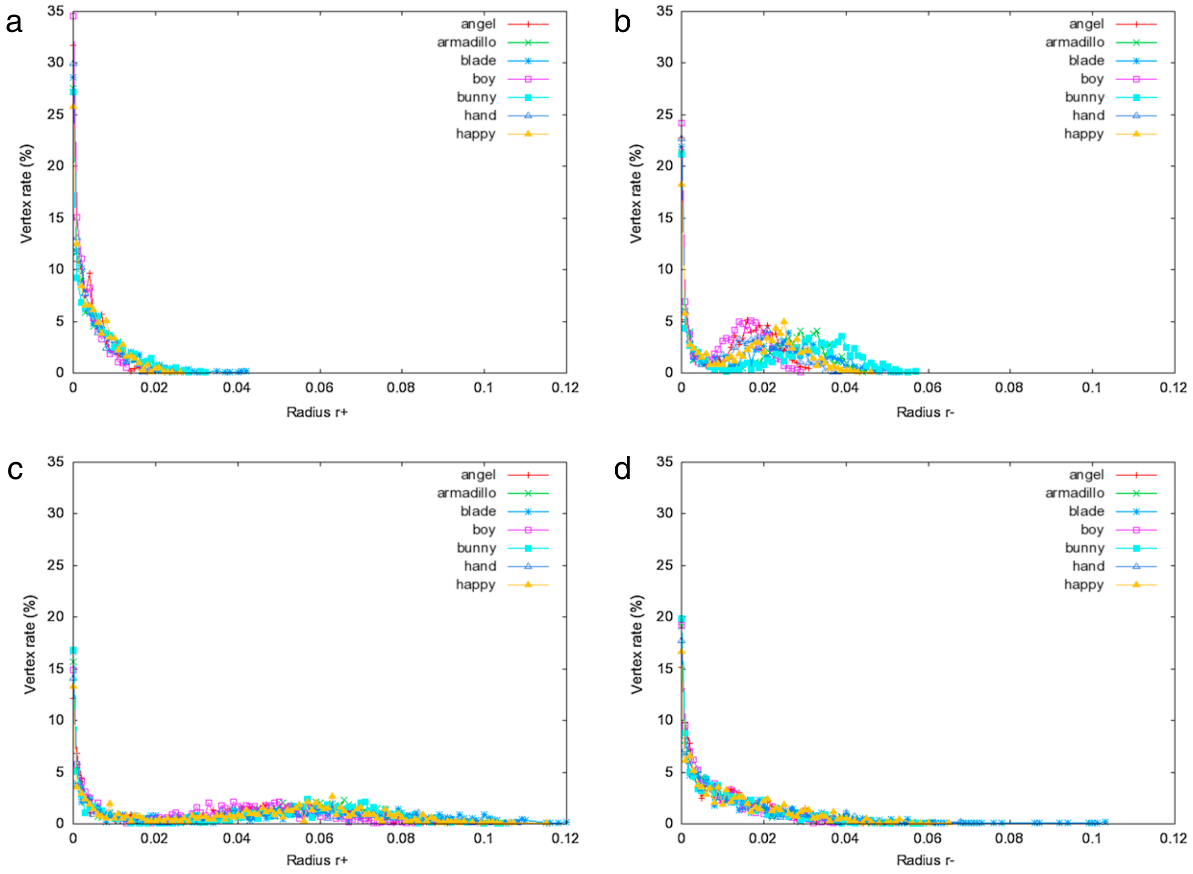

Fig. 13. Distributions of the vertices for seven 3D objects of our database: (a) As a function of r^+ for the normalization (1). (b) As a function of r^- for the normalization (1). (c) As a function of r^+ for the normalization (2). (d) As a function of r^- for the normalization (2).

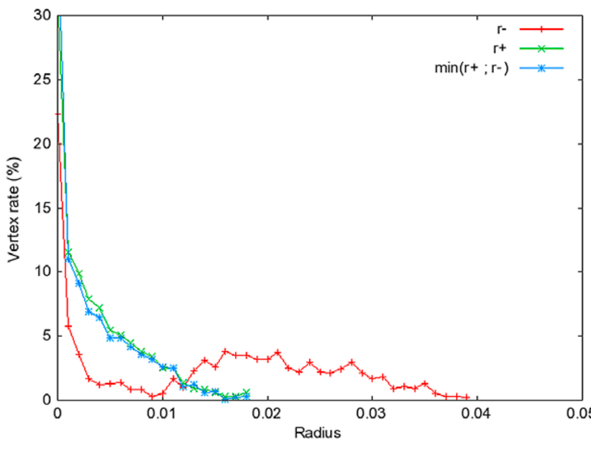

Fig. 14. Distributions of r, r^+ and r^- for the 3D mesh *Horse*.

is generated ($T = T'$). In this case the vertex ordering will be different, so $r_i \neq r'_i,\ i \in [1, n-1]$ in general. But the distribution of the most robust vertices does not change. We choose the first vertex as a parameter, it could be used to define a different order on the EMST. Moreover, we have defined a scope where the EMST does not change, in this context the vertex ordering done by the Prim algorithm is stable. However, we are aware that modifying the EMST could change dramatically the order defined, even if the percentage of common edge is high.

In order to reduce the complexity and to compare these meshes, we sub-sampled them to have approximately 1000 vertices for each 3D mesh. Indeed, for larger 3D objects, we assumed that is always possible to sub-sample them to keep the most interesting vertices. This can be done by a simple decimation, or on a sub-resolution or a clustering method. In fact the complexity of the analysis of EMST algorithm is in $\mathcal{O}(n^3)$. Indeed, as presented in Algorithm 1, the complexity of computing Prim's algorithm is in $\mathcal{O}(n^3)$ since for computing the second closest vertex we should calculate all the distances between the vertices v_j in V_{i-1}, $v_j \neq f(v_i)$ and the vertices in $V \backslash V_i$. Moreover, our radius (given by multiple calculations) can be computed at each step without increasing the upper bound of the complexity, but we add a great factor in each main loop.

Algorithm 1 Prim's algorithm with our computations $\mathcal{O}(n^3)$

```
Require: $G = (V, E)$, $v_0$
 1: $V_T \leftarrow v_0$
 2: $E_T \leftarrow \emptyset$
 3: $EMST = (V_T, E_T)$
 4: while $|E_T| < |V| - 1$ do                                        \\ $\mathcal{O}(n)$
 5:     find $e\{u, v\}$, $e2$ be the two different minimum weighted
        edges between the sets $V_T$ and $V \setminus V_T$,
        $u \in V_T, v \in V \setminus V_T$ compute $d_i^1$ with $e2$      \\ $\mathcal{O}(n^2)$
 6:     $V_T = V_T \cup \{v\}, E_T = E_T \cup \{e\}$
 7:     compute $d_i^2$ and $d_i^-$: loop on $V_T$                      \\ $\mathcal{O}(n)$
 8:     compute $d_i^3$: loop on $V$                                    \\ $\mathcal{O}(n)$
 9:     compute $r_i$: $min\{r_i^+, r_i^-, r_i^2\}$
10: end while
11: return EMST
```

Moreover, in order to be able to compare their disruptions, we normalized these objects and then $k \in \mathbb{R}$ denotes the scaling factor of the normalization. For these experiments we normalized the object in two different manners:

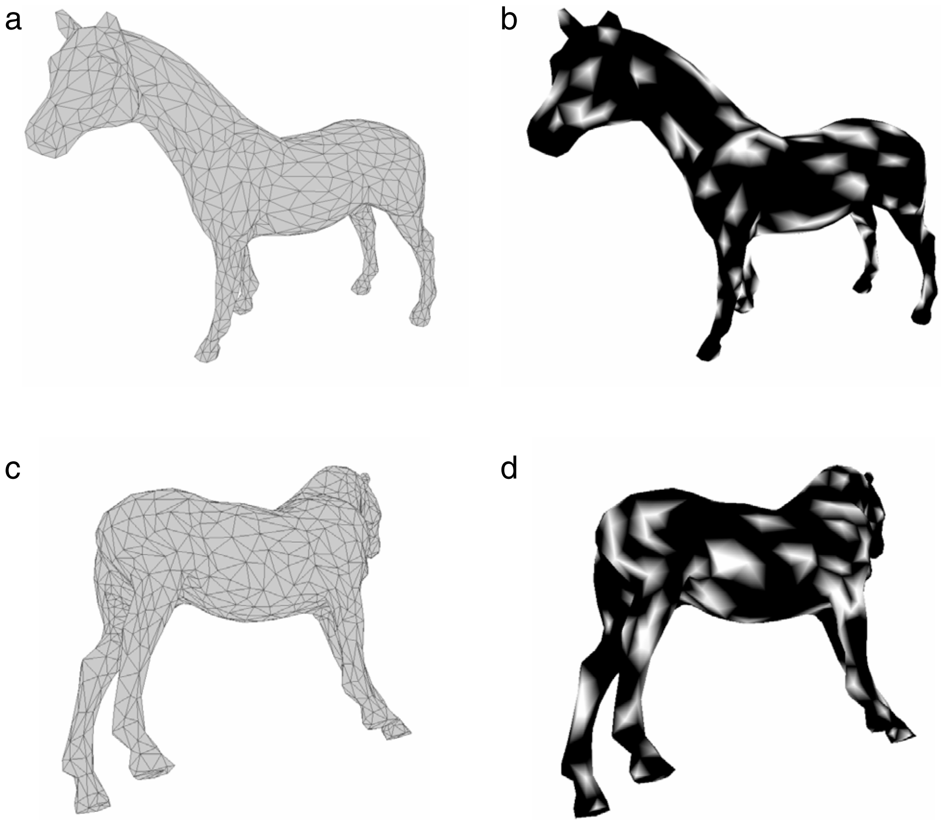

Fig. 15. Displays of the more robust areas on the 3D object *Horse*: (a)–(c) Original mesh. (b)–(d) The robust areas are shown with light colours.

1. As a function of the size of the bounding box (normalization (1)):

   $k = \max\{x_{\max} - x_{\min}; y_{\max} - y_{\min}; z_{\max} - z_{\min}\}$.

2. As a function of the average distance between two vertices in the mesh (normalization (2)):

   $$k = \frac{1}{m} \sum_{v_i, v_j \in V} \omega(v_i, v_j).$$

5.2. *Quantification of the vertex displacement*

The results of our analysis are presented for the mesh *Horse* normalized by the normalization (1) illustrated in Fig. 11(d). Fig. 12(a) and (b) respectively present the distributions of the vertices as a function of the radius values. The values are sampled with a step of 10^{-3}.

Note that there are many vertices that cannot be moved in one direction, *e.g.* $r^- \simeq 0$ or $r^+ \simeq 0$. In the case of the 3D object *Horse*, we have more than 20% for the radius r^- which is close to zero and around 30% of the radius r^+ are close to zero. As the radius value of a vertex becomes more significant, this vertex can be moved more and ultimately be more robust. However, the results illustrated in Fig. 12(a) and (c) show that a significant part of the vertices can be moved, but this also reveals the fragility of EMST structures.

For the radius r^+, the occurrences decrease very fast as the radius value increases. We observed that the maximum value for r^+ is around 1.8×10^{-2} (Fig. 12(a)) whereas as for r^- (Fig. 12(c)) the maximum value is around 4×10^{-2}. We can thus deduce from this experiment that it is easier to move a vertex v_i towards its father $f(v_i)$ than to take it away. Then, for radius r^- (Fig. 12(c)) we note an interesting peak between 1.5×10^{-2} and 2×10^{-2}. With this observation, we can consider two kinds of vertices: those which cannot be moved closer to their father and those that can be moved closer to the closest possible of their fathers. Fig. 12(b) and (d) illustrate the cumulative functions of r^+ and r^- respectively. Fig. 12(c) shows an inflection point near 1.5×10^{-2}. For the last case, in Fig. 12(e) we can see that the radius r^2 has a maximum distribution around 0.0233 which is much larger than the radius r^+. Fig. 12(f) shows the cumulative distribution which proves that most of the vertices have a radius $r^2 \in [0.02, 0.3]$. This behaviour is observed in our database, we propose to present the next result with a radius r defined as: $r = \min\{r^+, r^-\}$, since by experiments the radius r^2 has a very low influence.

After these detailed observations on the 3D object *Horse*, we compare the behaviour of criteria r^+ and r^- on seven 3D objects of our database. Fig. 13(a) and (b) illustrate, respectively, the distributions of the vertices for the 3D objects normalized as a function of the size of their bounding box (normalization (1)). Fig. 13(c) and (d) illustrate, respectively, the distribution of the same 3D objects normalized as a function of the average distance between two vertices in the mesh. The distributions of r^+ seem to be the same for the seven objects we analysed. The main part of the vertices cannot be moved, as we have previously seen for the 3D object *Horse*, but some of them can be moved. In particular, for criterion r^-, we also notice the same shape as *Horse*. Indeed, a lot of vertices cannot be moved, but we observed a peak around a particular value which corresponds to the average distance between two vertices. In Fig. 13(c), we can validate this intuition because the peak is around the same value. Indeed, this peak is a function of the average distance between two vertices of the mesh.

In order to determine how a vertex can be moved, we propose to let a criterion r quantify the possible motion of each vertex. Thus, for each vertex, we calculate:

$$r_i = \min\{r_i^+, r_i^-\}.$$

The distribution of this new criterion r is illustrated in Fig. 14, and compared to r^+ and r^-.

In Fig. 15 we visualized the more robust vertices in order to locate the most robust areas. These mesh parts could be used to synchronize hidden data during a watermarking process for example. Fig. 15(a) and (c) illustrates the original mesh of the 3D object *Horse*, while Fig. 15(b) and (d) illustrates the result of the visualization. The darker colours correspond to the most fragile areas, whereas the lighter colours correspond to the most robust areas.

As we can notice, a large number of areas, illustrated in darker colour, seem to be rather fragile. We can also notice, for the most robust areas, that the space between them and their neighbourhoods are not regular. This might be a clue for the search of geometrical criteria.

5.3. Validation of the proposed theoretical analysis

To validate our proposed theory, we need to have a relationship between the theoretical criteria as a function of r^+ and r^- presented in Section 3, and the experimental measures as a function of the percentage of the common edge rate between the EMST of the original mesh and the EMST of a noisy one. First, we select x% of the most robust vertices which are the most mobile vertices in the original EMST, where $r_i > r_{x\%}$.

Let T denote the EMST of the original mesh and T_σ the EMST of a noisy mesh such that the noise is Gaussian and its standard deviation equals σ. In fact, the analysis can be done by determining the correlation between $r_{x\%}$ and the intensity of the Gaussian noise added to the 3D mesh.

Let $\sigma_{x\%}$ be the critical standard deviation σ of the Gaussian noise such that its mean $\mu(T, T_\sigma) = x\%$. For the experiments, we take several 3D objects and we set x to compute, for each mesh, the values of $r_{x\%}$ and $\sigma_{x\%}$. We repeat this experiment 100 times for each 3D object.

In Fig. 16, for different selection rate $x_\%$, the critical standard deviation as a function of our theoretical criterion is presented. Each 3D object is linked to a point in the graph. The x-axis corresponds to the theoretical criterion $r_{x\%}$ and the y-axis to the critical standard deviation $\sigma_{x\%}$. For each selection rate, $x = 10\%$ (Fig. 16(a)) $x = 20\%$ (Fig. 16(b)) and $x = 30\%$ (Fig. 16(c)), we obtain a straight line. Indeed, there is a linear relationship between $r_{x\%}$ and $\sigma_{x\%}$ that does not depend on x:

$$\sigma_{x\%} = k \cdot r_{x\%}.$$

We estimate k, the value of the coefficient of the linear correlation by linear regression, for various vertex selection rates x. Table 1 presents the results.

We also noticed that for $x > 30\%$, we do not obtain a straight line. The behaviour of the criterion is chaotic because the areas selected are not robust. If some fragile vertices are selected (*i.e.* $r_{x\%}$ is too small), the disturbed vertex is out of its displacement cell and creates disorder in the EMST. This disorder results in a non-correlation of our criterion. In future work, it will be interesting to formalize this study in a theoretical context. This kind of technique should be interesting with various criteria that we are studying, such as estimation of the discrete curvature. The aim is to find a stable criterion in order to use the synchronization in a robust watermarking scheme.

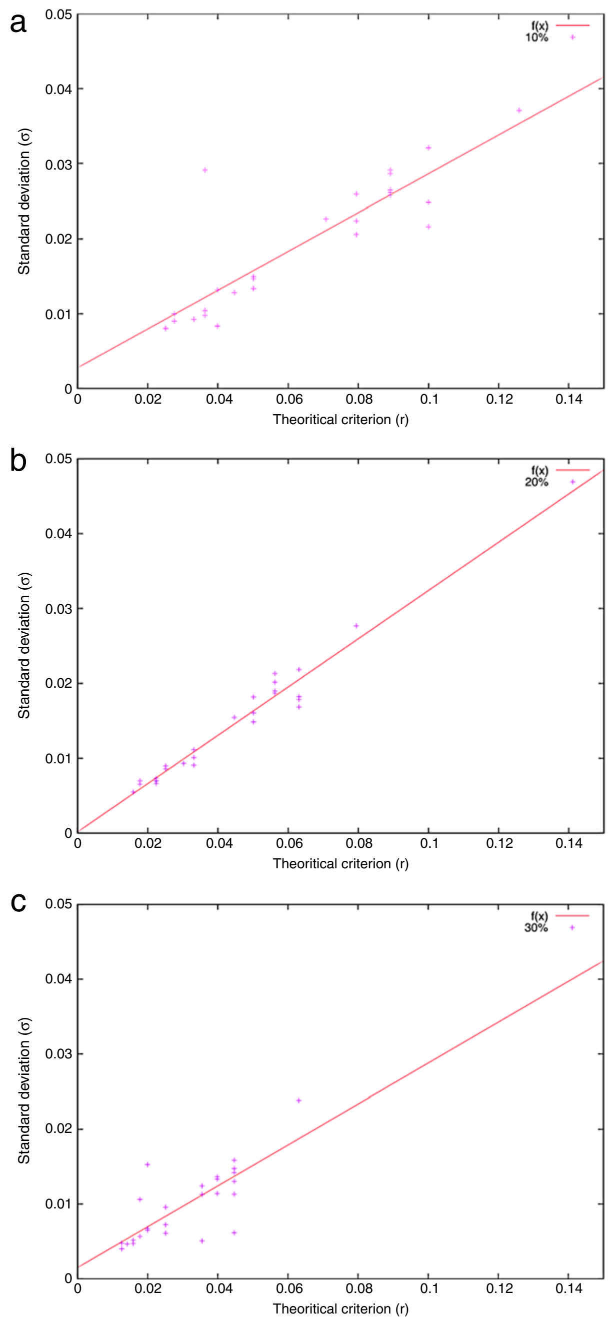

Fig. 16. Correlation between our theoretical criterion $r_{x\%}$ and $\sigma_{x\%}$ the critical standard deviation.

Table 1
Experimental results, coefficient of the linear correlation (k) as a function of the vertex selection rate (x).

Fig.	16(a)	16(b)	16(c)
x	10%	20%	30%
k	0.258	0.322	0.223

6. Conclusion and future work

In this paper, we have presented a theoretical analysis of the displacement of vertices without changing the connections in the EMST. Moreover, we have proposed a theoretical criterion in line with the Gaussian noise model in order to select the $x_\%$ most robust vertices. The proposed approach could be used to synchronize data for several 3D techniques such as watermarking or compression. In this paper, we limited the vertex displacement analysis along the half-line $]f(v_i); v_i)$, Assumption 3.2 and we assumed that this analysis can be done independently for each vertex, Assumption 3.1. These assumptions are not too much restrictive in the case of watermarking and compression applications where each vertex is processed one by one. In particular, with this analysis, we are able to locate robust areas that could be used for message embedding with the watermarking technique proposed by Amat et al. [3].

To continue this work, from a theoretical standpoint, it would be interesting to reduce the chosen constraints of our study. For a given step of Prim's algorithm, we would like to know how to move the vertex in three dimensions without changing the EMST built in the previous steps and without any *a priori* or hypotheses about these steps.

We would also like to link this study with 3D robust watermarking in order to build a new synchronization technique using EMST. This is a challenge since EMST is a fragile approach. For example, it might be interesting to build a tree with only the most robust areas for synchronization with the aim of being robust against attacks. Of course, the synchronization is strongly connected with the watermarking method and the application. For example, Su et al. proposed a very specific watermarking method for CAPD where the synchronization is a function of the flow direction [26].

References

[1] Wang K, Lavoué G, Denis F, Baskurt A. A comprehensive survey on three-dimensional mesh watermarking. IEEE Trans Multimed 2008;10(8):1513–27.

[2] Peng J, Kim C-S, Jay Kuo C-C. Technologies for 3D mesh compression: a survey. J Vis Commun Image Represent 2005;16(6):688–733.

[3] Amat P, Puech W, Druon S, Pedeboy J-P. Lossless 3D steganography based on MST and connectivity modification. Signal Process Image Commun 2010; 25(6):400–12.

[4] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. Comput Graph 1992;26:71–8.

[5] Franco P, Ogier J, Loonis P, Mullot R. A topological measure for image object recognition. Lecture notes in computer science, 2004. p. 279–90.

[6] Vlachos M, Lucchese C, Rajan D, Yu P. Ownership, protection of shape datasets with geodesic distance preservation. In: Proceedings of the 11th international conference on extending database technology: advances in database technology. New York (NY, USA): ACM; 2008.

[7] Merry B, Marais P, Gain J. Compression of dense and regular point clouds. In: Proceedings of the 4th international conference on computer graphics, virtual reality, visualisation and interaction in africa, ser. AFRIGRAPH'06. New York (NY, USA): ACM; 2006. p. 15–20.

[8] Wuhrer S, Brunton A. Segmenting animated objects into near-rigid components. Vis Comput 2010;26(2):147–55.

[9] Wang W-B, Zheng G-Q, Yong J-H, Gu H-J. A numerically stable fragile watermarking scheme for authenticating 3D models. Comput-Aided Des 2008; 40(5):634–45.

[10] Prim R. Shortest connection networks and some generalizations. Bell Syst Tech J 1957;36:1389–401.

[11] Ohbuchi R, Masuda H, Aono M. Watermaking three-dimensional polygonal models. In: Proceedings of the fifth ACM international conference on multimedia. New York (NY, USA): ACM; 1997. p. 261–72.

[12] Mao X, Shiba M, Imamiya A. Watermarking 3D geometric models through triangle subdivision. Proc SPIE 2001;4314:253–60.

[13] Cayre F, Macq B. Data hiding on 3-d triangle meshes. IEEE Trans Signal Process 2003;51(4):939–49.

[14] Tierny J, Vandeborre J-P, Daoudi M. Topology driven 3D mesh hierarchical segmentation. In: IEEE international conference on shape modeling and applications, 2007. SMI'07, 2007, p. 215–20.

[15] Luo M, Bors A. Surface-preserving robust watermarking of 3-D shapes. IEEE Trans Image Process 2011;20(10):2813–26.

[16] Wang K, Lavoué G, Denis F, Baskurt A. Robust and blind watermarking of polygonal meshes based on volume moments, LIRIS UMR 5205, tech. rep. RR-LIRIS-2009-001. CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon; 2009. Jan..

[17] Wang K, Luo M, Bors A, Denis F. Blind and robust mesh watermarking using manifold harmonics. In: IEEE international conference on image processing. IEEE; 2009. p. 3657–60.

[18] Rossignac J. Edgebreaker: connectivity compression for triangle meshes. IEEE Trans Vis Comput Graphics 1999;5(1):47–61.

[19] Bogomjakov A, Gotsman C, Isenburg M. Distortion-free steganography for polygonal meshes. Comput Graph Forum 2008;27(2):637–42. Citeseer.

[20] Wang N, Men C. Reversible fragile watermarking for 2-D vector map authentication with localization. Comput-Aided Des 2012;44(4):320–30.

[21] Kruskal Jr J. On the shortest spanning subtree of a graph and the traveling salesman problem. Proc Amer Math Soc 1956;7(1):48–50.

[22] Gordeev E. Stability analysis of the minimum spanning tree problem. Comput Math Math Phys 1999;39(5):738–46.

[23] Gordeev E. Stability analysis in optimization problems on matroids in the metric l1. Cybernet Systems Anal 2001;37:251–9.

[24] Dixon B, Rauch M, Tarjan R. Verification and sensitivity analysis of minimum spanning trees in linear time. SIAM J Comput 1992;21:1184–992.

[25] Yaman H, Karaşan O, Pınar M. The robust spanning tree problem with interval data. Oper Res Lett 2001;29(1):31–40.

[26] Su Z, Li W, Kong J, Dai Y, Tang W. Watermarking 3D CAPD models for topology verification. Comput-Aided Des 2013;45(7):1042–52.