IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 October 2023, accepted 1 November 2023, date of publication 7 November 2023,
date of current version 14 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3330916

RESEARCH ARTICLE

New Method for the Preservation of UV Coordinates in the Remeshing Process: Application to Footwear Design

EDUARDO CALABUIG-BARBERO[1], JUAN-CARLOS TOLEDO-GARCÍA[1], ANTONIO JIMENO-MORENILLA[2], AND HIGINIO MORA-MORA[2]

[1]INESCOP—Footwear Technology Center, 03600 Elda, Spain
[2]Department of Computer Technology, University of Alicante, 03690 San Vicente del Raspeig, Spain

Corresponding author: Eduardo Calabuig-Barbero (ecalabuig@inescop.es)

This work was supported by the Spanish Research Agency (AEI) [DOI: 10.13039/501100011033] under project HPC4Industry PID2020-120213RB-I00.

ABSTRACT Polygon meshes are a key element in computer-aided design (CAD). When working with this type of geometry, the simplification or reduction of polygons in the meshes is a very common operation for working with complex objects, but with a low density of polygons. These techniques are fundamental to be able to use CAD systems in an industrial environment that offer quality and optimisation of processes to make them viable for production and market times. However, these operations cause that some information of the original mesh is not maintained in the refined ones or is altered reflecting artifacts in the working mesh. This paper presents a new method for the preservation of texture coordinate information (also called UV coordinates or UV parameterization, so they are usually stored in a 2D matrix, U for rows and V for columns) between different refined meshes based on the original mesh information. It is a method that manages to imitate with a high degree of accuracy the original texturing information, event on meshes with a high level of simplification, regardless of the initial texture, which makes it especially fast even with high-definition textures. We demonstrate the performance of the approach on real elements used in the footwear industry.

INDEX TERMS UV-coordinates, detail preservation, mesh simplification, footwear, texture mapping.

I. INTRODUCTION

The preservation of the information inherent in polygonal meshes when they are subjected to operations such as remeshing or simplification is a crucial aspect in computer graphics. These remeshing and simplification operations, among many others, are very common, both in the field of computer graphics and in the field of video games, animated films, and CAD systems. The latter are becoming increasingly relevant in industry, even in sectors such as fashion and footwear, where 3D modelling is used to design and conceive the product (in addition to all the technical production information), thanks to the agility achieved in these systems and the realism of the geometric models with which they work.

Previously, to work with CAD models with a high level of detail and realism, it was necessary to use geometries with a large number of polygons, with the associated performance problems and processing times. However, thanks to the simplification techniques of the working meshes [1], [2], [3], [4], [5] and also with the remeshing techniques developed in [6], [7], [8], [9], and [10] it is possible to work with much less dense meshes while retaining the detail and realism of the original meshes. In addition, many of these techniques already address the issues of texturing and texture continuity, as well as the loss of information in this simplification and remeshing processes, [1], [3], [11], [12], [13], [14], [15]. Thanks to these advances, the use of simplification and remeshing techniques is becoming more and more integrated in the industry, facilitating the exploitation of CAD geometries in new technologies based on Web environments, Augmented Reality/Virtual Reality environments

The associate editor coordinating the review of this manuscript and approving it for publication was Walter Didimo.

or new paradigms such as the Metaverse, which require highly optimised information, but maximising the level of realism [16], [17], [18], [19], [20]. However, even in state-of-the-art Quad remeshing techniques [9], [10], [21] there are still problems derived from the process of generating new meshes, caused by loss of information, lack of accuracy in data migration or very high computational requirements, which makes their use at an industrial level more limited.

One of the main problems is the preservation of the texture coordinates of the original mesh in the newly generated mesh. This is especially important in CAD systems used in industrial environments, as the texture coordinates can take a very specific calculation of the working object, such as in the process of surface flattening [22], [23], [24].

The flattening process is crucial when it is used for the manufacture of an object and not only for texture visualisation purposes. This is the case in the footwear industry, as it is used to extract the 2D patterns from the 3D surface of the last itself, which will be used to cut the different pieces that will make up the shoe. It requires, therefore, a very complex calculation with a high level of precision. Thus, this relationship between the 3D mesh and the plan allows the pieces to fit perfectly on the manufacturing last.

Another important application of this technique is the texturing of digital CAD models, since the UV parameterisation calculated by the specific footwear CAD systems is ideal for the working geometries of the model and a level of appearance close to reality is achieved, without distortions or defects in the applied texture.

Thanks to the correct texture coordinates adjusted to the pieces of the footwear models, a totally real appearance of the material and the texture that composes it is achieved, without unnatural deformations that can detract from the realism of the CAD model. This is very relevant for the use of these CAD environments in the footwear industry, as it allows them to be used as a digital asset. On the one hand, it is useful for the initial phases of product definition and conceptualisation, in which designers and decision-making team define the collections to be manufactured in a totally digital way, drastically reducing the manufacture of samples, with the consequent economic benefits, time savings and, therefore, reducing the environmental impact. On the other hand, in the industrialisation and manufacture process of footwear, it helps notably in the correct manufacture of the product based on highly precise technical information, both numerical and visual, which will be used in the different footwear production processes (technical production sheets, CAD files for automatic cutting and processing, marketing, etc.).

This paper presents a new method for the preservation of UV coordinates, specifically in the Quad remeshing process of an initial geometry. Thanks to the proposed method, the result is a mesh with a quad structure, highly optimised and suitable for editing and modelling, but preserving the original UV parameterisation, whose technical and precise information will be used in subsequent design and manufacturing processes.

The paper is organised as follows: Section II shows an analysis of the current problems in different remeshing techniques in relation to the preservation of information during this process. In section III, the method is subjected to different experiments to show its quality and efficiency. Finally, the results obtained, the advantages of the proposed method and possible future studies of the work carried out are analysed.

II. BACKGROUND

3D mesh transformation methods such as simplification and remeshing face the problem that, in most cases, information from the original mesh needs to be available in the new transformed or generated mesh. Information such as colour, texture, details, or other attributes of the original mesh may not be available, or if available, may be incorrect in the new mesh, due to the creation of new information, modification or deletion of existing information. In addition, maintaining or accommodating this information at different mesh resolution levels may require computational costs that are too high for a real application.

However, in approaches such as [1] already provided a generic solution to simplification studies that do not offer any alternative in this context. Basically, these approaches regenerate the new information as a function of the Euclidean distance of the simplified mesh from the original one, so that they have a point-to-point relationship between the two meshes, regardless of the simplification approach used. Whether it is texture information, scalar or vector values or surface detail, these techniques approximate the original and the simplified mesh information, so the definition of the original data (texture definition, for example) is crucial to obtain good results. In addition, there is the problem that there may be aliasing on adjacent edges between different textured faces.

On the other hand, in [3] in addition to simplifying the geometry, a correspondence map is created between the input mesh and the simplified mesh, and the texture is optimally regenerated to reduce distortion and solve problems in the texture joints. Although it offers very good results, the processing time is quite high and depends on the texture resolution, as well as consuming more memory because of the need to maintain the correspondence map. Continuing with this type of approach, in [13] a method that adapts the texture mapping during the simplification process is presented. It provides very good results, with low distortion for meshes with high simplification. However, the method focuses on edge collapse, so it has the problem of calculating a new texture coordinate for a different vertex than the existing ones. The real challenge of our study is precisely that, to obtain the same texture representation in a mesh whose topology and distribution of vertices, edges and faces is totally different from the original and, moreover, with very high requirements of precision and similarity to the original,

so that possible defects that may appear are practically unnoticeable.

There are studies, like [12] and [14], that focus on the regeneration of texture maps from a set of textures (texture atlas) usually coming 3D digitisations, where they do address the problem of maintaining the additional information of the CAD model, which in these cases is commonly the digitised texture together with the object. Moreover, there is an added problem since, by the very nature of the digitiser, the texture is usually divided into parts and, in addition to respecting the original texture coordinates, it is essential that exists continuity between each of the parts of the texture once applied to the optimised mesh. These methods, although they have good results, cannot reflect with high accuracy and in real time the texture on the CAD object as it is shown in the real world. On the other hand, there are also approaches such as [11] which perform a texture mapping process according to the characteristics of the geometry to try to reduce texture distortions. These approaches are totally different from the proposal of this research as they do not address the problem of imitating the position and coordinates of a texture from its base object, but focus on how to adjust or modify the texture coordinates to match as closely as possible the geometry of the object, which is closely related to surface flattening approaches, [22], [23], [24].

Other approaches such as [6] and [10] do not fully address the real problem of accurately maintaining the original grid information but rely instead on the Integer-Grid Maps (IGM) introduced in [25] for generating quad meshes efficiently and with consistent results. In these cases, an important step is the parameterisation of the mesh and the transition functions between elements (usually computationally expensive) from a lower definition mesh to the higher definition mesh and vice versa. They are based on the principle of IGMs so that their structure is maintained as far as possible. However, to ensure a correct parameterisation of texture maps, they are often combined with specific solutions that address the parameterisation problem globally and even tackle the problem of patches or cuts, ensuring seamless texturing [26], [27].

However, this type of solutions only calculates an optimal flattening and parametrisation of the input mesh, so it will always be an approximation and not the initial parametrisation of the working mesh. This means that the result is not the same, which is a problem for many of the applications of this technology.

On the other hand, other methods of quad remeshing can be found as in [21] which present a method for the quadrangulation of a mesh, taking into account its main lines or features, thus providing a solution to a very complex problem that previous studies fail to address completely. As a main feature that differentiates it from other approaches, it can work with patches that are not rectangular, providing a greater flexibility in adjusting the remeshing according to the characteristic lines of the object. However, it does not preserve attributes or other additional information that the original mesh may have beyond the purely geometric data, therefore, texture coordinates, UV mappings or any added information are not maintained, so it would be necessary to apply a subsequent method for the recalculation of this information.

If we look at other quad meshing methods such as [9], it can be observed again that although they tackle and optimally solve the problem of quad remeshing of an input surface or mesh, they do not address the problem of maintaining other additional information that might exist in the mesh. Since [9] is implemented using a private library, it has not been possible to obtain information about the approach on which it is based. However, it has been possible to test and validate it with real examples, more focused on geometries and CAD models used in the footwear industry. As shown in [28] the different tests and outcomes of remeshing provide correct results in terms of the mesh geometry, however, in the different tests carried out on CAD objects of footwear models, the texture coordinates of the original object are lost. FIGURE 1 shows the result of the method presented in [9] where the good quality of the quad mesh obtained can be appreciated, but as the last piece shows, it lacks texture coordinates.

After the analysis of different existing methods, it can be concluded that it is necessary to implement a new method for the preservation of UV coordinates. This method must ensure that these coordinates are correctly obtained during the different processes of modification and refinement of the working geometry, such as those usually carried out in the footwear industry, when computer modelling the shoe and its components.

III. METHOD OVERVIEW

The main goal of the method is to compute the most accurate UV coordinates based on the information provided by the initial mesh M and the simplified one S. To perform this task, we use both the 3D geometric data of M and S and the original UV parametric information in M. In ALGORITHM 1, a pseudocode of the proposed method is shown.

This solution analyses each of the vertices of the simplified mesh S (mesh that has lost its texture coordinates). Depending on the neighbourhood of each vertex and its adjacent faces, the final texture coordinates are determined by a process of interpolation of the original texture coordinates. Thanks to this method, we obtain a texture coordinate that is based both on the texture coordinates of the original mesh and on the geometric position of these original vertices and faces, which means that the texturing on the simplified mesh is respected with a high degree of precision.

First, we need to find the closest face of M, F_j, to each vertex of S. To do that, we locate first the closest vertex of M, called V_j, to the vertex V_i of study (function FindNeighbours in line 2 of ALGORITHM 1). Then the 1-ring neighbourhood of V_j is obtained and the faces that belong to this vertex's neighbourhood are processed to find which of them is the closest to V_i (functions GetFaces and GetClosestFace in

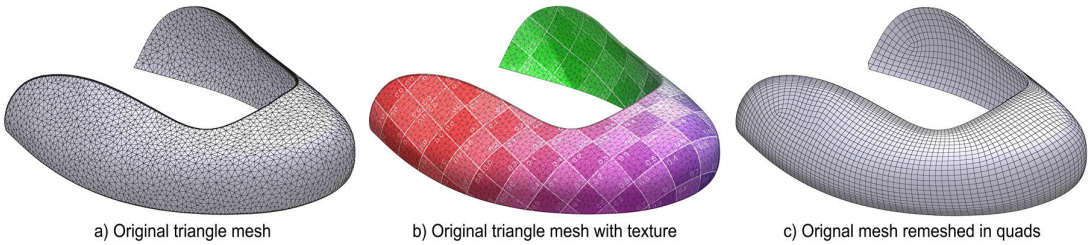

FIGURE 1. Quad remeshing result for an input triangular mesh. a) Original triangular mesh, b) original triangular mesh with texture applied, and c) quad mesh reconstructed with the Quad Remeshing method from [9]. The resulting textured quad mesh is not shown as the Quad Remesher method does not maintain the texture coordinates of the input mesh.

Algorithm 1 UV Coordinates Calculation for Each Vertex of S

```
1  for each V_i ∈ S do
2        VNeighbours = FindNeighbours(V_i, M)
3        FNeighbours = GetFaces(VNeighbours)
4        F_j = GetClosestFace(V_i, FNeighbours)
5
6        GetVertices(F_j, a, b, c)
7        V_pi = ProjectVertex(V_i, F_j)
8
9        P_ab = IntersectPoint(Parallel(Edge(a,
     c), V_pi), Edge(a, b))
10       P_ac = IntersectPoint(Parallel(Edge(a,
     b), V_pi), Edge(a, c))
11
12       Pct_1 = (a - P_ab) / (a - b)
13       Pct_2 = (a - P_ac) / (a - c)
14
15       Line_1 = Parallel(Edge(a, c), Pct_1)
16       Line_2 = Parallel(Edge(a, b), Pct_2)
17
18       t_uv = IntersectPoint(Line_1, Line_2)
```

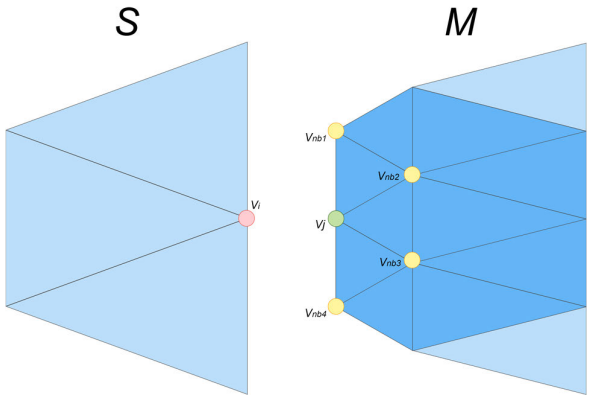

FIGURE 2. Closest faces of M from vertex V_i of S. Where $V_i \in S$, is the vertex of study, $V_j \in M$ is the closest vertex to V_i, set of V_{nb} are the neighbourhood vertices of V_j and the dark faces of M are the faces that belong to neighbourhood vertices.

lines 3 and 4 respectively of ALGORITHM 1). FIGURE 2 shows the 2D representation of this step.

Once we have the face F_j of M where the ideal data to transfer to V_i is found, we then work to locate and calculate the correct information from the UV data of the vertices that compose F_j. To do this, we rely on percentage terms, as this is an accurate way to correlate the 3D information of a vertex with UV parameterisation, since geometrically there is no relationship between these two spaces.

FIGURE 3 shows part of the process of obtaining the key points in 3D space to calculate the percentages and their use in 2D space to get the resulting UV coordinate. In this process we work always with two edges of the triangles, so that we only need two edges to intersect with the projected vertex. Thanks that we have the correlation between the 3D space and the 2D space of the face vertices and edges, the intersection applied in both cases is also correlative and provides a unique t_u for each V_{pi}.

Following the process of the algorithm, the vertex V_i is projected onto the face F_j, obtaining V_{pi} (function ProjectVertex in line 7 of ALGORITHM 1). Subsequently, we choose an edge of F_j, e_{ab} in the example, and we draw a parallel to e_{ab} so that it passes through V_{pi}. Intersecting this parallel line with the edge e_{ab} we obtain the point P_{ab} (line 9 of ALGORITHM 1). We repeat the same process with the adjacent edge, e_{ac} in the example, obtaining P_{ac} (line 10 of ALGORITHM 1). These will be the points that allow us to calculate the percentages of the edges of the 3D triangle (lines 12 and 13 of ALGORITHM 1), which will look for their corresponding position in UV space.

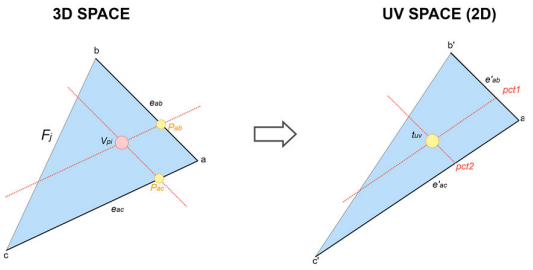

FIGURE 3. Left, intersected points used for percentages calculation in 3D space based on projected vertex over face F_j. Right, using those percentages over the UV space for the final UV calculation from the original M data.

Therefore, the next step, once we have the percentages calculated in the 3D space, will be to apply them on the UV space of F_j (face of the original mesh M). Starting from

the UV coordinates of the edges e'_{ab} and e'_{ac}, we calculate their parallel lines that intersect in each of the percentages respectively (lines 15 and 16 of ALGORITHM 1). If we then calculate the point of intersection of both parallels, we will get the UV value corresponding to the vertex V_i of study (line 18 of ALGORITHM 1). This averaged value will be the best-fit UV value for that new vertex of the simplified grid.

The time complexity of the algorithm is $O(n)*O(m)$, where 'n' is the number of vertices of S and 'm' is the number of vertices of M. Among the functions and operations internal to the general loop (for each V_i and S do), only the "Find-Neighbours" method is an iterative operation depending on the number of vertices of the simplified mesh M (usually with a reduced number of vertices). The rest of the auxiliary functions and operations are linear, so they do not affect the overall time complexity.

The advantage of the time complexity of the method is mainly that it does not depend on the dimensions of the texture used. This avoids the computational complexities of processing two-dimensional image matrices or vectors twice as large, where, in addition, it is also necessary to work with each one of the vertices of the geometry (at least with those of the simplified mesh) to save the calculated texture coordinate value. This aspect is very relevant and differentiating for a problem that seeks to work with meshes with as few vertices as possible, but with high-definition textures of high quality to achieve virtual representations of physical models that are visually almost real.

As can be seen, the method performs operations that are computationally efficient, which makes its computation times very short even for complex geometries, compared to the literature analysed. The calculation of these percentages and the way they are applied on the UV coordinates is a novel approach that allows a very accurate calculation of the UV coordinates themselves for the simplified model, with variations that are not visible to the naked eye. As will be seen in the following section, this advance achieves a notable improvement in the optimisation and topological restructuring of the work meshes in CAD environments for footwear modelling, without losing visualisation quality, something fundamental in these industries to continue implementing and evolving CAD environments in business network.

IV. RESULTS

For the experiments carried out, simplified polygonal meshes have been used, resulting from applying the algorithm of [9] on originally more complex triangular meshes. The tests have been carried out with different mesh resolutions, specifically three resolutions (depending on the density of the original object): a high-resolution mesh, a medium-resolution mesh, and a low-resolution mesh, seeking the lowest possible resolution, but without producing errors in the simplified mesh (although achieving densities of 6% of the total of the original mesh). For all the tests shown below, we used an 11th generation Intel® Core™ i7-2.50Ghz, 16GB of RAM, with an NVIDIA GeForce GTX 1660 graphics card, under the Windows 11 Pro 64-bit operating system.

A. EXPERIMENTS TO EVALUATE THE EFFICIENCY OF THE METHOD

To evaluate the efficiency of the developed method, example polygonal meshes have been used in their different resolutions. TABLE 1 shows the set of geometries used for the efficiency tests together with the dimensions of the texture used in each case.

TABLE 1. Set of test geometries with the size of the texture used.

Geometry	Triangles	Image size
Shoe last	100.000	3356 x 1514
Toe shoe piece	14.000	626 x 626
Front shoe piece	40.000	1181 x 1181
Buckle	10.000	1205 x 1205

To have a more complete validation, this variety of geometry and its textures has also been combined with different mesh resolutions in each of the test examples. The results obtained are detailed below:

TABLE 2. Processing time (in seconds) of each test geometry comparing between different resolutions.

Shoe last	120.000 quads	45.000 quads	10.000 quads
Execution time	3.0 s	1.2 s	0.35 s
Toe shoe piece	6.000 quads	3.000 quads	350 quads
Execution time	0.125 s	0.062 s	0.005 s
Front shoe piece	15.000 quads	7.000 quads	1.400 quads
Execution time	0.936 s	0.437 s	0.094 s
Buckle	10.000 quads	6.000 quads	1.000 quads
Execution time	0.270 s	0.156 s	0.016 s

Processing times of the proposed method for obtaining the new texture coordinates are very short, depending largely on the polygon density of the original mesh and the resulting mesh. However, for an original mesh of 100,000 polygons and a modified mesh of 120,000 polygons (the method is valid both, for meshes with a lower number of polygons and in the case of increasing the definition of the resulting object) the execution times are 3 seconds. TABLE 2 shows a comparison with meshes of different resolutions, showing that the times are not relevant, especially if we focus on meshes of lower resolution, as this is one of the main applications of this type of solution. Another characteristic to highlight, given that we only work with UV texture coordinates, is that the method is totally independent of the resolution and dimensions of the object's texture, so the calculation speed will be the same, even if we use high-definition textures.

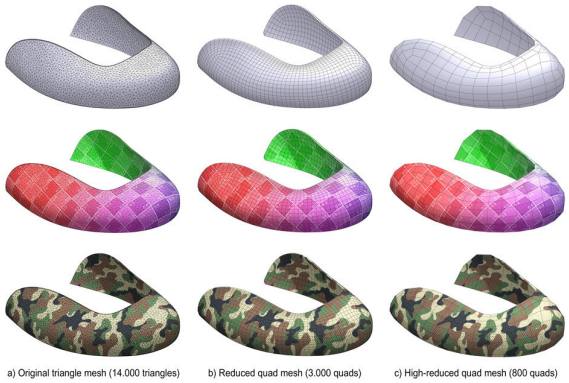

FIGURE 4. Toe shoe piece results with different meshes resolutions. From top to bottom, the original mesh, the mesh with a UV-validated texture and the texture. From left to right, a) the original mesh with 14.000 triangles, b) the reduced quad mesh with 3.000 quads (21% of the density of the original mesh) and c) the quad mesh with a high reduction to 800 quads (5.7% density). In all cases, the texture is shown exactly the same as on the original object.

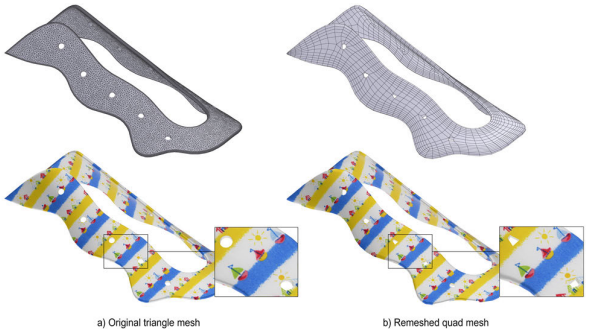

FIGURE 5. Our method applied over a front piece of a shoe with small geometry details like the holes for the laces. Despite of the high reduction of polygons and the loss of the shape in some places (see the laces holes), the texture is perfectly applied over the remeshed geometry.

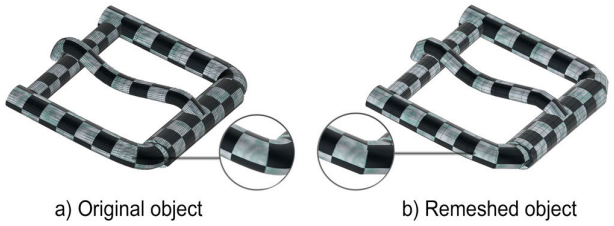

FIGURE 6. In the areas of higher curvature, if correct initial UV texture coordinates are not calculated, artefacts can occur when calculating the new coordinates for the modified mesh. On the left, a) the original object that already has texture deformations in certain areas. On the right, b) geometry remeshed in quads which, in addition to texture deformation, also has artefacts.

B. EXPERIMENTS TO ASSESS THE QUALITY OF THE METHOD

To validate the results of the method, as well as its versatility, it is necessary not only to work on meshes of different resolutions, but also with different objects and textures that may contain details that carriage a challenge or a possible problem to the method. For this reason, in this experimental phase, in addition to the different resolution meshes, we have worked with different types of geometry and textures.

FIGURE 7. Shoe last digitalization. High-quality digitized triangle mesh (100.000 triangles) together with the texture applied. Below, the texture image with the pictures of all the shoe last parts.

FIGURE 8. Texture image created by the scanning device.

FIGURE 9. Shoe last texture coordinates calculation with the proposed method. The position and features of the original texture are exactly in the position they must be. On the left, a) shoe last with 45% of the density of the original object (from 100.000 triangle to 45.000 quads). On the right, b) shoe last with 10% of the density of the original object (from 100.000 triangles to 10.000 quads).

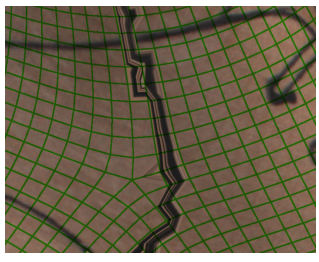

FIGURE 10. Existing problem when texture is divided in different parts inside the same image. In the middle of the image can be appreciate the interpolation of the render system between two vertices of the same quad which have widely separated texture coordinates.

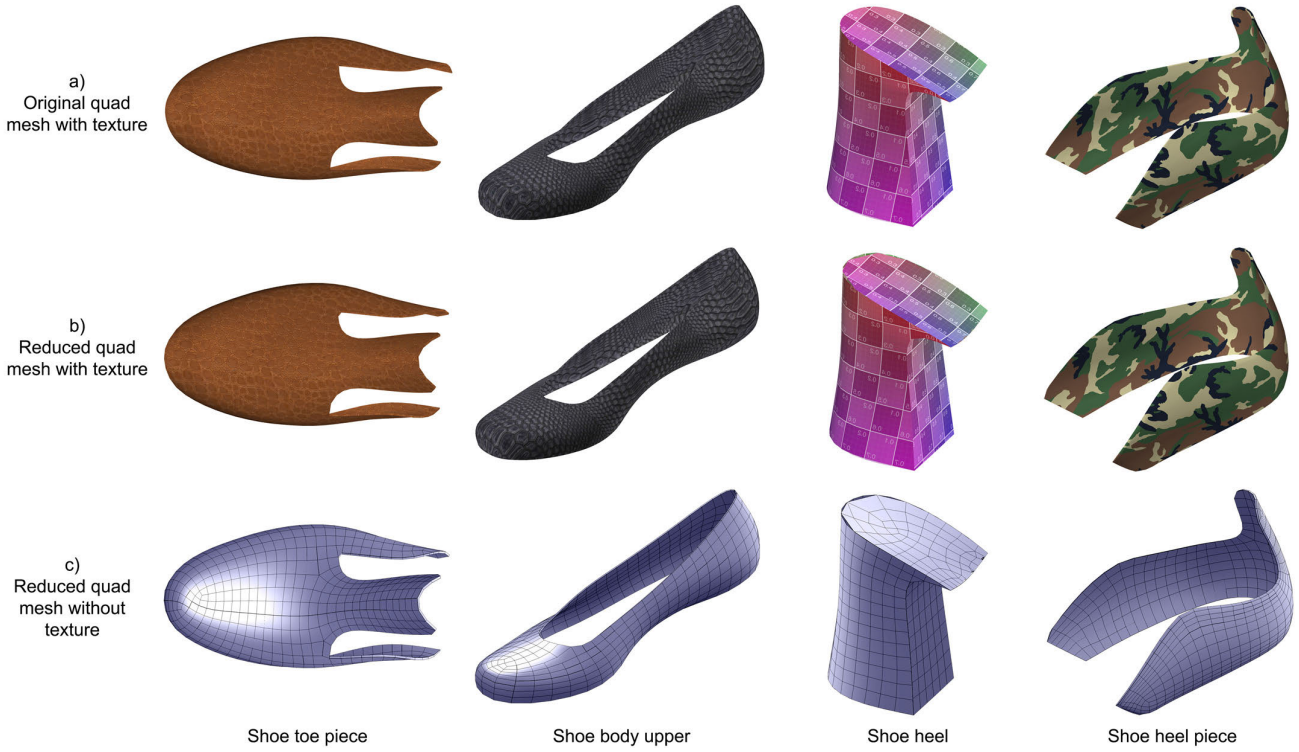

FIGURE 11. Different example geometries and textured applied to original and high-definition quad mesh and with the reduced quad mesh.

On the one hand, a very common piece or element in footwear is used, with an organic shape, but with a smooth and continuous modelling over its entire surface. It has also been tested with meshes with internal holes to check how the method solves this situation, where large changes in the geometry are produced in the simplification processes. And finally, it is shown a set of tests performed with several remeshed shoe objects together with its textures.

In the tests carried out, geometric deformations can be seen in the most reduced meshes, caused by the reduction process itself, but nevertheless the texture coordinates calculated, in all cases, the results respect the position and appearance of the texture on the original object. FIGURE 4 shows the results on a shoe toe piece, it can be seen how the texture application is the same in all cases despite the significant reduction of polygons.

FIGURE 5 shows another example of the results obtained with the proposed method applied on different geometries. Again, despite applying it on meshes with a significant reduction, even where the geometry itself loses the shape and 3D information of the original object, the applied texture does not show any deformation and appears the same as the texture on the initial object.

A very important aspect in the whole process of transferring the UV coordinates between meshes is the original calculation of the texture coordinates themselves (ones that belong to the original object), since good texture coordinates calculated accurately, thanks to the flattening of the surface, will allow fewer or even no errors during the process. In FIGURE 6 is shown an example of a geometry with texture coordinates that are valid, but not optimal as they have stresses in the areas of higher curvature. It can be seen in the image the deformation of the texture in the curved areas. Because of this, when we apply the method for obtaining the texture coordinates of the simplified quad mesh, artefacts can be seen as the rendering engine makes an interpolation between the new vertices without having any control of the process in these details.

On the other hand, the proposed method has also been used for another industry-relevant process for which no robust solution has been available so far. This involves transferring the UV coordinates after simplifying a mesh from the digitisation of a 3D object and its texture, where it is essential to maintain the original coordinates so that the texture appears in the same position and place in both objects.

FIGURE 7 shows the result of a digitalisation of a shoe last, where a mesh with high definition and therefore with a high density of triangles is obtained. It also comes with a texture of real photos of the object (FIGURE 8) that, thanks to the texture coordinates calculated by the device, fits perfectly in the 3D model. If we apply the quad remeshing process from [9] a new simplified mesh is obtained, with new vertices and no texture coordinates calculated, so it is necessary to apply some algorithm of calculation or transfer of the UV coordinates.

FIGURE 9 shows the result of applying the proposed method where we transfer (averaging) the texture coordinates of the initial model into the final model, independently of the definition of the resulting mesh.

On the one hand, it should be noted that the texture is precisely positioned with respect to the original model, so that the drawing lines of the last will be in the desired place. This, for example, is very important to synchronise the manual design work on the last and the digital work from the digitisation of the object, allowing the use of this digitisation of the texture as a reference image for the reverse engineering process.

On the other hand, the change between the parts of the texture is noticeable because there is a quite big gap in the original texture (it has been put in black to highlight the defect) and, as the remeshing method used does not consider this original texture, there can be polygons in two different parts of the texture. FIGURE 10 shows how the whole of this junction area is fitted between quads that have vertices in one part and vertices in another. The interpolation made by the rendering engine between the two vertices of the quad causes this defect. Although this problem has not been tackled in this study, a solution could be that the original texture does not have this separation between the different parts of the CAD model, which is not always possible, so that the remeshing or simplification method should consider the different parts of the texture and generate patches or areas limited to each of the parts. In the latter case, our method would generate the best possible texture coordinates without generating any defects in the applied texture.

Various results are shown in FIGURE 11, assessing the method with different geometries and textures and showing consistent results in all of them, even when the geometry suffers strong deformations due to polygon reduction. In these cases, defects or holes are produced in the reduced mesh that do not correspond to the original geometry. This can be seen in the shoe heel test object in FIGURE 11.

V. CONCLUSION AND FUTURE WORK

In this research, a method has been developed to transfer the UV coordinates of an original mesh to a simplified one or a mesh with a different topological structure. The method offers high quality results for geometries of diverse types and with completely different meshing, both in terms of geometry topology, number of polygons and resolution. It has been verified how it performs efficiently even in geometries with a high level of simplification, always offering an appearance equal to the original.

Thanks to the use of this new method, a relevant problem is solved in industries that make use of high-definition input meshes or geometries, but for their work and editing tasks they need to convert them into other types of structures with a greater simplicity in terms of volume of information. The method also preserves crucial information such as the attributes inherent to geometry, highlighting for example the UV coordinates used in the process of applying textures for the virtual representation of materials.

It is important to highlight that our method is based on the UV coordinates of the initial object, so that, on the one hand, its original calculation and precision are fundamental to obtain good results and, on the other hand, precisely thanks to this, an accurate method is achieved, capable of imitating the texturing of the original object and, furthermore, totally independent of the texture or image to be applied, making it valid and equally fast regardless of the resolution of the texture.

As future work, there are two aspects that can be improved to make the proposed method more versatile. On the one hand, the problem of objects whose texture is separated into different parts and the modified mesh contains polygons with vertices in more than one area. In addition to the possible solutions discussed in section IV that do not depend on the method presented, the method itself could be improved to consider the separation of the UV coordinates with respect to the location of the texture and apply some rectification to avoid these jumps or separations between zones of the base texture.

On the other hand, the method can present problems in meshes that have folds or areas that are geometrically very close to each other but not connected, i.e., that are not neighbouring vertices. The method, based on searching for the closest point, could determine that an undesired area is closer than the study area itself, so we would be working with the wrong triangle and the texture coordinate obtained would not be the right one.

ACKNOWLEDGMENT

The authors would like to thank INESCOP and its Computer Aid Design and Computer Aid Manufacturing (CAD/CAM) development team for their commitment and involvement.

REFERENCES

[1] P. Cignoni, C. Montani, C. Rocchini, and R. Scopigno, "A general method for preserving attribute values on simplified meshes," in *Proc. Vis.*, Oct. 1998, pp. 59–66, doi: 10.1109/visual.1998.745285.

[2] D. P. Luebke, "A developer's survey of polygonal simplification algorithms," *IEEE Comput. Graph. Appl.*, vol. 21, no. 1, pp. 24–35, May 2001, doi: 10.1109/38.920624.

[3] W. Zhang, J. Zheng, Y. Cai, and A. Ynnerman, "Seamless simplification of multi-chart textured meshes with adaptively updated correspondence," *Comput. Graph.*, vol. 106, pp. 77–87, Aug. 2022, doi: 10.1016/j.cag.2022.05.021.

[4] S. Tsuchie, "Mesh simplification accompanied by its denoising of scanned data," *Eng. With Comput.*, vol. 35, no. 3, pp. 993–1008, Jul. 2019, doi: 10.1007/s00366-018-0647-x.

[5] A. Maglo, G. Lavoué, F. Dupont, and C. Hudelot, "3D mesh compression: Survey, comparisons, and emerging trends," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 1–41, Apr. 2015, doi: 10.1145/2693443.

[6] H.-C. Ebke, P. Schmidt, M. Campen, and L. Kobbelt, "Interactively controlled quad remeshing of high resolution 3D models," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–13, Nov. 2016, doi: 10.1145/2980179.2982413.

[7] S. Melzi, R. Marin, P. Musoni, F. Bardon, M. Tarini, and U. Castellani, "Intrinsic/extrinsic embedding for functional remeshing of 3D shapes," *Comput. Graph.*, vol. 88, pp. 1–12, May 2020, doi: 10.1016/j.cag.2020.02.002.

[8] H.-C. Ebke, M. Campen, D. Bommes, and L. Kobbelt, "Level-of-detail quad meshing," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–11, Nov. 2014, doi: 10.1145/2661229.2661240.
[9] *Quad Remesher Auto Retopology*. EXOSIDE. Accessed: Feb. 11, 2023. [Online]. Available: https://exoside.com/quadremesher/
[10] H.-C. Ebke, D. Bommes, M. Campen, and L. Kobbelt, "QEx: Robust quad mesh extraction," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–10, Nov. 2013, doi: 10.1145/2508363.2508372.
[11] Y. Jin, Z. Shi, J. Sun, J. Huang, and R. Tong, "Content-aware texture mapping," *Graph. Models*, vol. 76, no. 3, pp. 152–161, May 2014, doi: 10.1016/j.gmod.2013.11.001.
[12] B. Lévy, S. Petitjean, N. Ray, and J. Maillot, "Least squares conformal maps for automatic texture atlas generation," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 362–371, Jul. 2002, doi: 10.1145/566654.566590.
[13] C. Chen and J. Chuang, "Texture adaptation for progressive meshes," *Comput. Graph. Forum*, vol. 25, no. 3, pp. 343–350, Sep. 2006, doi: 10.1111/j.1467-8659.2006.00953.x.
[14] R. Pagés, D. Berjón, F. Morán, and N. García, "Seamless, static multi-texturing of 3D meshes," *Comput. Graph. Forum*, vol. 34, no. 1, pp. 228–238, Feb. 2015, doi: 10.1111/cgf.12508.
[15] Y. Xian, Y. Fan, Y. Huang, G. Wang, C. Tu, X. Meng, and J. Peng, "Mesh simplification with appearance-driven optimizations," *IEEE Access*, vol. 8, pp. 165769–165778, 2020, doi: 10.1109/ACCESS.2020.2987939.
[16] F. E. H. Tay and A. Roy, "CyberCAD: A collaborative approach in 3D-CAD technology in a multimedia-supported environment," *Comput. Ind.*, vol. 52, no. 2, pp. 127–145, Oct. 2003, doi: 10.1016/S0166-3615(03)00100-3.
[17] E. Calabuig-Barbero, M. Davia-Aracil, H. Mora-Mora, and F. Herrero-Pérez, "Computational model for hyper-realistic image generation using uniform shaders in 3D environments," *Comput. Ind.*, vol. 123, Dec. 2020, Art. no. 103337, doi: 10.1016/j.compind.2020.103337.
[18] S. F. Qin, R. Harrison, A. A. West, and D. K. Wright, "Development of a novel 3D simulation modelling system for distributed manufacturing," *Comput. Ind.*, vol. 54, no. 1, pp. 69–81, May 2004, doi: 10.1016/j.compind.2003.07.005.
[19] A. S. M. Sayem, "Digital fashion innovations for the real world and metaverse," *Int. J. Fashion Des., Technol. Educ.*, vol. 15, no. 2, pp. 139–141, May 2022, doi: 10.1080/17543266.2022.2071139.
[20] K. Bahirat, C. Lai, R. P. McMahan, and B. Prabhakaran, "Designing and evaluating a mesh simplification algorithm for virtual reality," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 14, no. 3s, pp. 1–26, Jun. 2018, doi: 10.1145/3209661.
[21] N. Pietroni, S. Nuvoli, T. Alderighi, P. Cignoni, and M. Tarini, "Reliable feature-line driven quad-remeshing," *ACM Trans. Graph.*, vol. 40, no. 4, pp. 1–17, Aug. 2021, doi: 10.1145/3450626.3459941.
[22] C. Bennis, J.-M. Vézien, and G. Iglésias, "Piecewise surface flattening for non-distorted texture mapping," *ACM SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 237–246, Jul. 1991, doi: 10.1145/127719.122744.
[23] P. N. Azariadis and N. A. Aspragathos, "Geodesic curvature preservation in surface flattening through constrained global optimization," *Comput.-Aided Des.*, vol. 33, no. 8, pp. 581–591, Jul. 2001, doi: 10.1016/S0010-4485(00)00102-0.
[24] P. N. Azariadis, A. C. Nearchou, and N. A. Aspragathos, "An evolutionary algorithm for generating planar developments of arbitrarily curved surfaces," *Comput. Ind.*, vol. 47, no. 3, pp. 357–368, Mar. 2002, doi: 10.1016/S0166-3615(01)00155-5.
[25] D. Bommes, M. Campen, H.-C. Ebke, P. Alliez, and L. Kobbelt, "Integer-grid maps for reliable quad meshing," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 1–12, Jul. 2013, doi: 10.1145/2461912.2462014.
[26] A. Myles and D. Zorin, "Global parametrization by incremental flattening," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–11, Aug. 2012, doi: 10.1145/2185520.2185605.
[27] N. Ray, V. Nivoliers, S. Lefebvre, and B. Lévy, "Invisible seams," *Comput. Graph. Forum*, vol. 29, no. 4, pp. 1489–1496, Jun. 2010, doi: 10.1111/j.1467-8659.2010.01746.x.
[28] E. Calabuig-Barbero, G. Martinez-Martinez, J.-L. Sanchez-Romero, A. Jimeno-Morenilla, V. Lopez-Martin, and H. Mora-Mora, "Implementation of efficient surface discretisation algorithms adapted to geometric models specific to the footwear industry," *Int. J. Adv. Manuf. Technol.*, Apr. 2023, doi: 10.1007/s00170-023-11361-w.

EDUARDO CALABUIG-BARBERO received the degree in computer engineering from the University of Alicante, in 2006. He began his professional career as a Researcher in CAD/CAM software focused on footwear design and manufacturing, including computation geometry, virtual material representation, physically-based rendering, 3D modeling, and also new technologies related to industry 4.0 like cloud computing, the Internet of Things, and robotics.

JUAN-CARLOS TOLEDO-GARCÍA received the degree in computer engineering from the Polytechnic University of Valencia, in 1996. Since that year until today, he is a Researcher in CAD/CAM software for the shoe industry, including computation geometry, nesting and flattening algorithms, 3D modeling, and rendering. Besides this, he was an Associate Professor with the Computer Technology and Department, University of Alicante, Spain, for two years.

ANTONIO JIMENO-MORENILLA was born in Spain, in 1970. He received the Ph.D. degree from the University of Alicante, in 2003. He is currently a Full Professor with the Computer Technology and Department, University of Alicante, Spain. His research interests include computational geometry for design and manufacturing, rapid and virtual prototyping, and high-performance computer architectures. He also has considerable experience in the investigation of the professional skills of computer engineers.

HIGINIO MORA-MORA received the Ph.D. degree in computer science from the University of Alicante, Spain, in 2003. He is currently a Full Professor with the Computer Technology and Department, University of Alicante. His research interests include computer modeling, computer architectures, high performance computing, embedded systems, the Internet of Things, and cloud computing paradigm.