EUROGRAPHICS Workshop on Graphics and Cultural Heritage (2017)
R. Sablatnig and B. Štular (Editors)

3D Annotation Transfer

A. Scalas[1], M. Mortara[1] and M. Spagnuolo [1]

[1]CNR-IMATI Genoa, Italy

Abstract

In the last few years, there has been an increase in digitalization efforts within the Cultural Heritage field, which boosted the interest for new strategies to improve documentation standards. While these concepts have been largely studied for most of the CH content types, 3D data still need to be fully worked out as document types. One of the most innovative methods to glue the documentation (i.e. the semantics) of the artifacts to their geometry is to exploit the technology of the semantic web and implement the semantic annotation pipeline for 3D data. Since the 3D representation of artifacts is not a standard, and in the particular case of triangular meshes there are differences of resolutions and vertices position, there is the strong need for tools which could allow for annotation persistence between representation switch. In this paper, we present the first results in the design of an automatic algorithm for annotation transfer between triangular meshes with different resolutions, provided that they represent the same artifact.

CCS Concepts

•***Computing methodologies*** → ***Knowledge representation and reasoning;*** *Shape modeling;*

1. Introduction

One of the most innovative methods to glue the documentation (i.e. the semantics) of the artifacts to their geometry is to exploit the technology of the semantic web and implement the semantic annotation pipeline for 3D data. We speak about 3D semantic annotation because (or when) some information is linked/associated to an object, or to parts of it, and is used for understanding and storing a piece of information about the object, or its parts, which is not explicitly contained in the geometric data itself [CMSF11].

If the annotation mechanism was well-defined, the enriched geometry of digital artifacts would be ready to be fed to a whole set of services that can be built around annotated geometries. This is particularly important for research studies in which the artifacts are analyzed and compared reasoning on their features, as it happens frequently in the archaeological context. Archaeological descriptions are typically shared as texts among scientists in the area, and the text contains references to features present in the objects. Part-based annotations of digital models could allow for a novel form of archaeological descriptions, where the text could be linked directly to the geometry, opening the way to novel and more effective ways of sharing knowledge and reasoning work-flows among professionals in the field.

The use of part-based annotation for 3D models, however, poses more challenges compared to text documents or images, some of which have been nicely described in [HF07], the consistency of annotation across model resolutions being one of them.

In this paper, we present the initial results of a method for transferring annotations across digital models of the same artifact at different resolution. The method is being developed within the context of the GRAVITATE project [Gra], whose aim is to create a platform to support archaeologists and curators in the reconstruction and reunification of shattered or broken cultural objects [PWM*16]. To achieve these results, many efforts are being made trying to integrate geometry with semantics to support the experts in reasoning about fragments. Tools for automatic feature detection are being developed, to support the identification of regions of interest in fragments, and part-based annotation are important to attach information about the detected features to their geometries.

In the GRAVITATE collection, the scale of features spans from very tiny details to larger parts which represent, for instance, whole ornaments. The resolution of the acquisitions therefore has to be high enough to represent this spectrum and, as a consequence, the models are frequently quite large. On the other hand, some of the algorithms have an intrinsically high computational complexity (e.g. the space search for mating two fragments) which invokes for low resolutions models.

For these reasons, in GRAVITATE, each artifact is stored as a set of digital models of varying resolutions to be used in the different phases or for different purposes within the system. To make the selection of the resolution as transparent as possible to the users, proper mappings of the results obtained at different resolutions need to be implemented. This is particularly useful for the annotation work, as the users have to interact with the models in the process and they should not be asked to repeat the same work for different resolutions of the same object.

DOI: 10.2312/gch.20171295

In the remainder, we will describe the semantic and part-based annotation with references to related work, and show our contribution, which consists in the presentation of the first results in the design of an automatic algorithm for annotation transfer, which works between triangular meshes with different resolutions, provided that they represent the same artifact.

2. Background and related work

Generally speaking, the purpose of annotation is to create correspondences between objects, or parts of them, and conceptual tags. To focus on our interest field, we can say that the 3D annotation consists in selecting the so-called ROI (Regions Of Interest) on a 3D mesh and associating them meaningful tags.

3D part-based annotation was proposed for the first time in [RASF07], where the annotation was supported by a set of segmentation tools to suggest possible ROI on the meshes. The same approach has been proposed in the product modeling field, and lately in the medical domain where the annotation is used as a basis for follow-up monitoring of rheumatic pathologies [BCPS16]. Similar approaches have been followed also by [GMT13], where semi-automatic methods are based on the user verification of the automatically annotated parts (e.g. [YKC*16]) and typically use a geometric segmentation approach.

Assuming that a reliable method for selecting or detecting a ROI in a 3D model is available, a standard way for storing the parts is needed, to tie the semantic information to 3D data. In [RASF07], a simple format for the storage of annotation was implemented: each ROI is defined as a connected set of triangles and expressed as a list of triangles' indices, and is stored as an additional file together with the geometry. The format for annotated 3D geometry has been extended in [BCPS16] to cover ROIs of different dimensions, and complying to the Open Annotation Data Model [OAD].

3. Annotation Transfer Method

The need for an automatic annotation transfer method arises from the fact that 3D representation formats do not allow for stable markups. Let us think, for example, of a triangular mesh, the same physical object can be represented with different resolutions and, even at the same resolution, the vertices can be in different positions. The ROI on the physical object are therefore represented by different geometries in the various representations. Furthermore, the reference to the ROI on the representation should survive simple editing operations such as cuttings or affine transformations [HF07]. To achieve such goals, a tool that allows for annotations transfer would be very useful.

In this section, we describe the approach we are developing in GRAVITATE, considering that each physical object is represented with different resolutions. The lower resolution meshes are obtained from the original ones with different simplification techniques, but providing that the simplified ones are composed by a subset of the vertices of the original meshes.

Before starting the method description, it is necessary to introduce some definitions. First of all, let us denote with A_s an annotated patch which lies on the surface of a source triangular mesh

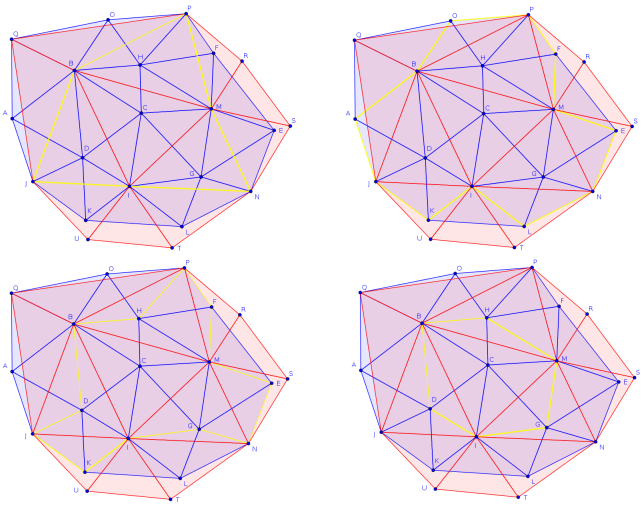

Figure 1: *Different possibilities for annotation transfer from lower to higher resolution, where M_s is the one in red, the M_t is the one in blue and the annotation is shown as a yellow outline (the figure in the top left is the one with the original annotation)*

M_s, and let us denote with A_t the annotated patch on the target mesh M_t. The patch A_s is defined as a connected set of triangles whose boundary, named the annotation *outline*, is an ordered set of vertices, connected in pairs by an edge of M_s. The outline entirely encloses a portion of M_s linked to an annotation. We will consider in the following that the outline vertices are ordered counterclockwise. Note that the transferred annotation A_t needs to be made by pairs of vertices of M_t, that is, we do not want to add new vertices on M_t to delimit the geometrical projection of A_s onto M_t, rather, we want to identify on M_t a set of connected triangles that well represent A_s on M_t.

In our study, the source and target meshes for the annotation transfer are perfectly aligned and registered. Therefore, the annotation mapping is defined by finding a suitable correspondence between the outline vertices of A_s and vertices on M_t *close* to the projection of the outline vertices of A_s onto M_t. Once the correspondence is found, the outline of the annotation A_t is reconstructed by tracing a variant of the shortest path between pairs of the outline vertices of A_s.

The correspondence between outline vertices is key to define the annotation transfer. Now, we have to think of what we want to obtain from the annotation transfer: in fact, depending on the way we map outline vertices from A_s to A_t, we can obtain a surface patch which either totally "encloses" or only partially overlaps the shape of A_s (see Figure 1). The results discussed in this paper correspond to the latter approach.

The correspondence of vertices from the outline of A_s to A_t is conceived as follows: if M_s is the lower resolution one, than there is no need for setting any correspondence, as the vertices of the outline will be in any case vertices of M_t; if M_s is the higher resolution one, the vertices of the outline are projected one at the time

onto M_t, and the correspondence is set to the vertex on M_t which is closest to the projection.

This approach may cause some problems. Indeed, it is not guaranteed that a pair of successive vertices in the outline of the annotation would map onto two different vertices of the target mesh, neither that the path between successive vertices will not intersect other parts of the outline. In practice, this issue is solved by keeping trace of the already used vertices and introducing a check on repetitions for the projection of vertices, simply rejecting the already used ones. The self-intersection issue should yet be studied in depth, but for now we tackled the problem by simply pruning all the "thorns" obtained by vertex repetition in the paths-finding in post-processing. The first results of the test phase haven't highlighted any trace of other types of self-intersection.

It remains to resolve how to obtain the desired connection between the mapped vertices. The approach we have followed tries to find the shortest path between each pair of mapped vertices on the target mesh, and includes the edges of this path in the transferred annotation outline. This shortest path is extracted with a discrete application of the Dijkstra algorithm for the shortest path on graphs, where, if we call the original vertices v and v', the weight on an arc (v_i, v_j) is defined as the distance between v_j and the segment found connecting v and v'. We have defined this type of weight for trying to reduce as much as possible the error between the original connection (which was a straight line) and the new connection (which is as a step-like line). The pseudo-code for the annotation transfer can be seen in Algorithm 1.

Algorithm 1 Transfers the annotations from a mesh M_s to another mesh M_t. It works for either the low-to-high resolution transfer and for the high-to-low one.

```
1:  procedure AnnotationTransfer(M_s, M_t)
2:      for each annotation A_s in M_s do
3:          for each vertices pair (v, v') in A_s.outline do
4:              v_1 ← findCorrespondece(v, M_t)
5:              v_2 ← findCorrespondece(v', M_t)
6:              A_t.outline ← A_t.outline ∪ shortestPath(v_1, v_2)
7:          end for
8:          M_t.addAnnotation(A_t)
9:      end for
10:     return M_t
11: end procedure
```

Since a shape can have more than one annotation, the entire process should be applied to all of them. As one can see, any new annotation (better, its outline) is found searching, for each pair of vertices in the ordered set defining the outline of the original annotation, the shortest path which links them over the objective mesh surface and connecting the various paths found.

The procedure findCorrespondence can be defined in different ways: in our case we choose to use only one definition, in which the correspondence is found projecting the vertex from M_s onto M_t following its normal and then searching the vertex on that triangle which is nearest to the starting one. In this way, we obtain the same results in the case of low-to-high resolution transfer as if we exploited the fact that the vertices are in common, but we could show a more generalized algorithm, which would work even in contexts where this is not the case. Furthermore, in this way we avoid the issue of mapping a vertex on a new one which is not connected to the rest of the annotation, but still is nearer (in an Euclidean way) to the original one.

Algorithm 2 Finds the correspondence as the mapping onto the nearest vertex on the target mesh.

```
1:  procedure findCorrespondence(v, M)
2:      t = projectVertexOnMesh(v, M)
3:      d = ∞
4:      for i ← 1...3 do
5:          d' ← ||t.v_i − v||
6:          if d' < d then
7:              v' ← t.v_i
8:              d ← d'
9:          end if
10:     end for
11:     return v'
12: end procedure
```

It is very important to highlight that once the order of the outline of the annotation is given, obtaining the annotated surface is really simple: it is sufficient to take the left triangle (in case of an counterclockwise order) of the first edge of the outline and performing a region growing using that triangle as the seed, moving in the neighborhood of the latter (the triangles adjacent to each edge of the actual triangle) until the outline is reached.

4. Conclusions and future works

The presented approach to annotation transfer has shown interesting results. First of all, the annotation transfer has been a success in a good percentage of cases (there are still few particular cases to tackle). Not only the obtained results keep the annotation alive in different results, but it is even a good quality transfer, as it can be seen in Figure 2 and in Table 1.

In future works we will focus on some limitations of the current implementation:

- *Degeneracy*: the area of an annotation, expressed as an outline composed of more than two vertices, might map to a degenerate outline (composed by two or less vertices) which encloses a null area;
- *Wrong projection*: a vertex in the higher resolution mesh can be nearer to a triangle which isn't actually connected to the annotation even if its normal points in the right direction (this is the case when the shape exhibits several nearly overlapping layers);
- *Distortion*: in cases in which some parts of an annotated area are greatly simplified, there could be an excessive distortion of the annotation itself, in terms of area and shape.

The solution to the first issue is rather trivial (it is sufficient to impose that the new annotation must have at least three vertices on the outline, obtaining them, for example, by taking all the triangles intersected by the projection of the original vertex), but the second one is not so immediate: we might consider a geodesic-distance approach for rejecting this kind of triangles. For the last case we can

Original mesh	Destination mesh	Annotation shape	Area before	Area after	Area error (%)
09_840_50k	09_840_100k	Rectangle	8.40323	8.40348	0.003
-	-	Star	2.95185	2.98175	1.013
-	-	Spiral	3.24342	3.23284	0.326
09_840_100k	09_840_50k	Rectangle	8.72794	8.71242	0.178
-	-	Star	3.21928	3.31354	2.928
-	-	Spiral	3.42321	3.27262	4.399
GR_25_1890_50k	GR_25_1890_1M	Eye	3.69926	3.69417	0.138
-	-	Mouth	2,23549	2,24027	0,214
-	-	Nose	9,24278	9,25180	0,098
GR_25_1890_1M	GR_25_1890_50k	Eye	4.51838	4.55487	0.808
-	-	Mouth	2.15199	2.12195	1.396
-	-	Nose	9.47786	9.38173	1.014
91 8-6 46_50k	91 8-6 46_1M	Flower	4.67025	4.79157	2.598
91 8-6 46_1M	91 8-6 46_50k	Flower	4.79157	4.56986	4.627

Table 1: *Some examples of the errors introduced by our method. It can be seen that the error is very little (always below 5%), even if the shape of the annotation is concave. In the first two columns the name of the meshes in the table refers to the corresponding name in the GRAVITATE repository, where the last part indicates the resolution (50k → 50,000 vertices - 1M → 1,000,000 vertices).*

Algorithm 3 Finds the shortest path between the given pair of vertices, following the weight definition already presented.

```
1:  procedure SHORTESTPATH(v1, v2)
2:      D.add((v1, 0))
3:      F.push(v1)
4:      v ← v1
5:      while v ≠ v2 do
6:          F.pop()
7:          for each v′ in v.FirstRingNeighbors() do
8:              d ← D(v) + v′.distanceFromSegment(v1, v2)
9:              if v′ is in P then
10:                 D.add((v′, d))
11:                 P.add((v′, v))
12:                 F.push(v′)
13:             else if D(v′) > d then
14:                 D(v′) ← d
15:                 P(v′) ← v
16:             end if
17:         end for
18:     end while
19:     p ← v2
20:     while p ≠ v1 do
21:         path.push(p)
22:         p ← P(p)
23:     end while
24:     return path
25: end procedure
```

define some descriptors of the properties whose keeping is desired and threshold their value to control the maximum error, inserting new vertices and edges if those properties are not met (we can start with the area ratio, of which we have already done some preliminary tests reported in Table 1).

Lastly, to speed up performance, it is our aim to find a way to exploit some spatial data structures (e.g. kd-trees) to optimize the identification of *nearest* vertices with respect to some properties (normal direction and so on).

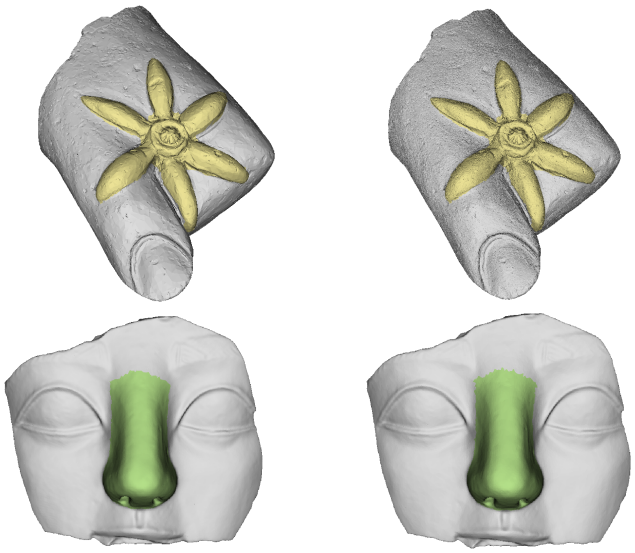

Figure 2: *The result of the annotation transfer from a lower resolution mesh to an higher resolution one can be seen in the first row (50k to 1M vertices), while the opposite transfer can be found in the second row (100k to 50k vertices).*

Acknowledgements

This paper has been supported by the H2020 EU project GRAVITATE "Geometric Reconstruction And noVel semantIc reunificaTion of culturAl heriTage objEcts", Grant Agreement n. 665155. All the 3D models belong to the Salamis collection maintained in the STARC repository of the Cyprus Institute.

References

[BCPS16] BANERJEE I., CATALANO C. E., PATANÉ G., SPAGNUOLO M.: Semantic annotation of 3d anatomical models to support diagnosis and follow-up analysis of musculoskeletal pathologies. *International journal of computer assisted radiology and surgery 11*, 5 (2016), 707–720. 2

[CMSF11] CATALANO C. E., MORTARA M., SPAGNUOLO M., FALCIDIENO B.: Semantics and 3d media: Current issues and perspectives. *Computers & Graphics 35*, 4 (2011), 869–877. 1

[GMT13] GUO J., MEI X., TANG K.: Automatic landmark annotation and dense correspondence registration for 3d human facial images. *BMC Bioinformatics 14*, 1 (2013), 232. 2

[Gra] GRAVITATE. http://gravitate-project.eu/. 1

[HF07] HAVEMANN S., FELLNER D. W.: Seven research challenges of generalized 3d documents. *IEEE Computer Graphics and Applications 27*, 3 (May 2007), 70–76. 1, 2

[OAD] Open Annotation Data Model. https://www.w3.org/TR/annotation-model/. 2

[PWM*16] PHILLIPS S. C., WALLAND P. W., MODAFFERI S., DORST L., SPAGNUOLO M., CATALANO C. E., OLDMAN D., TAL A., SHIMSHONI I., HERMON S.: GRAVITATE: Geometric and Semantic Matching for Cultural Heritage Artefacts. In *Eurographics Workshop on Graphics and Cultural Heritage* (2016), The Eurographics Association. 1

[RASF07] ROBBIANO F., ATTENE M., SPAGNUOLO M., FALCIDIENO B.: Part-based annotation of virtual 3d shapes. In *Cyberworlds, 2007. CW'07. International Conference on* (2007), IEEE, pp. 427–436. 2

[YKC*16] YI L., KIM V. G., CEYLAN D., SHEN I.-C., YAN M., SU H., LU C., HUANG Q., SHEFFER A., GUIBAS L.: A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph. 35*, 6 (Nov. 2016), 210:1–210:12. 2