Volume xx (200y), Number z, pp. 1–14

Effective Annotations over 3D Models

F. Ponchio[1], M. Callieri[1], M. Dellepiane[1] and R. Scopigno[1]

[1]Consiglio Nazionale delle Ricerche, Istituto di Scienza e Tecnologie dell'Informazione (CNR-ISTI), Pisa, Italy

Abstract

Annotation support in interactive systems is often considered a simple task by the CG community, since it entails the apparently easy selection of a region and to connect it with some information. The reality appears more complex. The scope of this paper is twofold: first, to review the status of this domain, discussing and characterizing several approaches proposed in literature to manage annotations over geometric models; second, to present in detail an innovative solution proposed and assessed in the framework of Cultural Heritage (CH) applications, called ClippingVolumes. At the annotation definition stage ClippingVolumes uses 3D data to characterize the annotation region; subsequently, annotations are visualized by adopting a two-pass rendering solution which uses stencil buffers, thus without introducing new geometric elements, changing the topology or duplicating geometry elements. It solves most of the issues that afflict the current state of the art, such as: fragmentation; annotation transfer to multiple representations; and enabling to annotate multi-resolution data encoding. The latter is a mandatory requirement to produce efficient web-based systems. We implemented and we fully tested this approach in the framework of a complex system that supports the documentation of CH restoration projects.

CCS Concepts

•***Computing methodologies*** → ***Mesh geometry models;*** *Shape analysis;*

1. Introduction

The technical term *annotation* refers here to a mechanism that links a sub-portion of the geometrical representation of an object to some related information. This concept has a long story and was managed in the past by the use of 2D technical drawings or thematic maps with an associated legend. Nowadays, digital instruments open much wider capabilities.

The term *annotation* describes a twofold action: (a) the selection of a location/region over the surface of a 3D model and (b) the creation of an explicit link between that spatial element and *structured*, *semi-structured* or *unstructured* data. The data associated with an annotation usually provides a *characterization* of the selected location/region (e.g. what is the constituent material, the presence of a certain sign/marking, the name of a structural/functional subcomponent, any insight produced by a visual analysis process, etc).

Two different concepts of annotation exist

1. *Semantic segmentation* (see Figure 1): where a functional subset or a structural subdivision in parts is defined over a given shape (e.g. the hand or the head of a human model [KHS10]; the handle of a pot model; the windows of a car model or the wings of a plane model [CFG*15]). Many methods have been presented in literature for the (semi-)automatic shape characterization and segmentation. These algorithms provide a partitioning of the 3D model in parts based on *shape-related* criteria and focus on how to produce the segmentation [ARSF07b, GA08, ARSF07a, AAB*14], rather than the task of associating a tag or semantic data with those parts. This segmentation, often conceived as a structural decomposition, depends mostly from inherent shape characteristics of the mesh, or from functional interpretations of the parts or components, rather than a user-driven process (see the next class). Working with multi-part models (mostly CAD) is a similar case and metadata can be defined on each of their sub component (e.g. weight, material, part ID, function, links to external data). Finally automatic segmentation may focus on

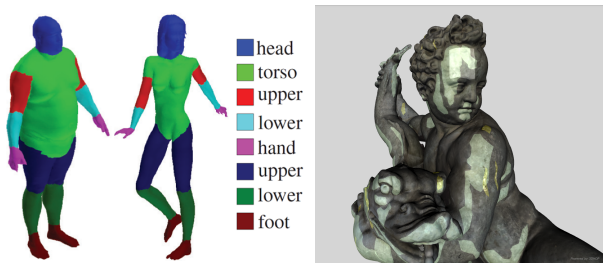

Figure 1: *Two examples of the different classes of annotations. On the left, an example of* ***semantic segmentation*** *over a human shape, classifying the different body parts [KHS10]; on the right, an example of* ***user-driven characterization*** *of the surface which delimits some degraded regions and assigns them some textual annotations [ABC*18].*

structural components as well as much finer features, e.g. recognizing chisel marks [PCCS11].
2. *User-driven characterization* (see Figure 1), where each annotated region is selected according to a specific user request or insight (e.g. a small region affected by some degradation process; an area with a bump related to a damage occurred to the shape of the object; etc.) and is linked to some metadata or to some information. In this case, most of the applications enable a user-driven selection mechanism, since often the identification of those regions and the associated data heavily depend on the knowledge of a human expert and/or are the results of a visual analysis process [STLL13, MVL16, ABC*18]. Some methods propose to use both semantic and user-dependant selection [YH13] to link portions of objects.

A detailed review of the *semantic segmentation* methods is beyond the scope of this paper and we also do not focus here on other kind of 3D mapping that are sometimes addressed as annotations, such as:

1. *Value maps*: a scalar or vector value is associated with each point of the surface and is often displayed with a false colour (e.g. the surface reflectance value or the normal map).
2. *Volume characterization*: annotations that cover a certain volume area *inside*, *outside* or *around* the 3D surface; e.g. the range of motion of an articulated arm, the volume illuminated by a light, the extent of the electrostatic field around a molecular surface.

Implementing an annotation module might be considered a simple component of an interactive system or of a data management tool, whereas some of the most important issues are quite challenging. Designing the GUI, providing efficient as well as effective solutions for converting the user input into a robust geometric characterization and associating the selected knowledge with the annotated region are some of those major issues currently under study. In many applications defining an annotation does not only mean specifying an *id* or a small text. Annotations could also mean linking spatial regions to any multimedia document or endorsing semantic data encoding approaches. Therefore, the technical scope of this paper appears twofold:

- to present the *issues* related to the selection of annotation regions and to review the solutions proposed in the literature by proposing a characterization. We will focus here on the problem of how to define each selected region and how to encode it.
- to present *ClippingVolumes*, an innovative solution to the problem of annotating complex triangle meshes. Implementing annotation becomes a challenging task, especially when we manage very complex or multiple descriptions of the objects of interest (e.g. web-based systems usually adopting multi-resolution representation schemes). The ClippingVolumes solution has been implemented to solve these issues, as a component of a restoration documentation system and assessed in the framework of a complex restoration testbed [ABC*18].

The paper is organized as follows. Section 2 introduces very briefly some application domains requiring the support of annotations. Section 3 presents a characterization of the three basic annotation primitives (points, lines or regions). Section 4 characterizes the domain and presents in detail the different approaches devised to translate the user input (annotation selection) into a specific geometric parcel defined on top of the specific data representation scheme, thus focusing on the geometric sub-task. Section 5 treats the issues related to the management of complex data representations and annotation transfer. Section 6 presents in detail our innovative solution, *Clipping Volumes*, which solves the annotation transfer issue and enables the use of multiresolution data representations. Once the geometric counterpart of an annotation has been defined, the next action to accomplish is to link the semantic data to the geometric parcel and store that data in a proper format; this latter phase is the focus of Section 7. Finally, we produce an overall evaluation and some concluding remarks in Section 8.

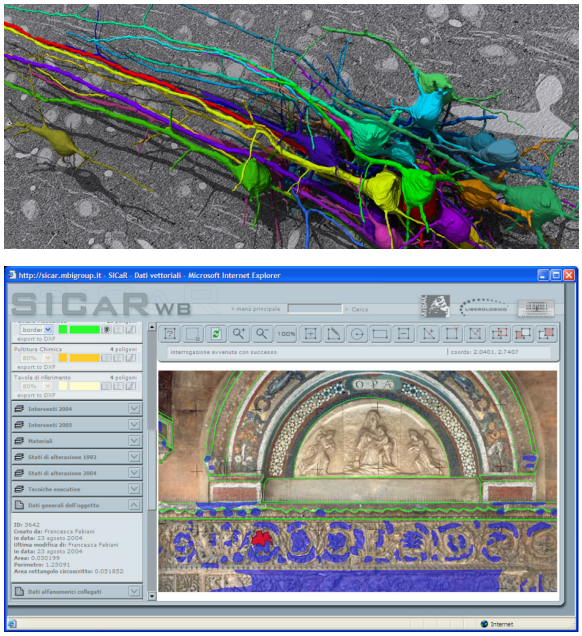

Figure 2: *Top: an example of annotation over biological imaging data. Bottom: an example of annotation created to document the conservation status of a marble surface (restoration of CH artworks).*

2. Applications requiring annotation support

Annotations are a crucial instrument to document the reasoning process and its results; this is a crucial feature for all those applications requiring the visual analysis of the (virtual) object of interest. Therefore, annotations are a key ingredient in many different application domains:

- *Medical and biology:* different inspection devices (TC, MRI, electronic microscopes, etc.) create digital images or stacks of images. Therefore, a basic step in the analysis of those results is the detection of *regions of interest* over the 2D or 3D digital imaging results (Figure 2 top). Highlighting and preserving this information is necessary to support the data analysis process, to facilitate collaborative work and to document the diagnostic process [BCPS16].

- *Cultural Heritage (CH) and Digital Humanities (DH):* images and 3D models are now consolidated working tools. Archaeology is a pioneering domain where the use of digital clones is now a common practice. The focus is now to evolve from the first steps of observation (creating digital representations of the artworks under study) to the design and use of tools to support the investigation of scholars working over digital representations [SCC*11]. Therefore, annotations remains a common feature of many CH tools or systems [SSRS12, HG10, AAB*14]. Semi-automatic methods have been proposed to annotate collections of artworks, adding semantic metadata by means of an automatic, suggestion-driven process [FKW17] or by adopting harvesting methods able to collect and integrate community annotations [HG10].
  In the domain of artwork restoration the analysis of the conservation status, the planning of the restoration action, or its post-execution documentation are usually implemented by the production of complex maps, where the surface of the artwork is characterized and described in detail [ABC*18], leading to very complex annotation patterns (see Figure 2 bottom).
- *Archives of 3D models:* with the advent of easy-to-use 3D digitization and digital fabrication technologies, a number of web-based archives for 3D models appeared. Sketchfab [Ske14] or the ShapeNet system [CFG*15] are good examples of the latter. Many of these systems allow the owner to add annotations over the 3D model, to better qualify it or to add metadata. Moreover, identification of parts by annotating the model is also instrumental to the implementation of enhanced search and retrieval features [ARSF07b].
- *3D Content created for interactive applications and computer games:* the creation of assets for the entertainment industry, such as the content representing the different levels in a computer game, requires huge modelling work, including the careful tagging of parts of the scene according to their intended behaviours (e.g. event triggering zones, forbidden areas, etc.).
- *CAD and Building Information Models (BIM)*: many of those systems support annotations, to add either generic metadata (e.g. part catalogue numbers) or numerical data specific to selected points (e.g. to store some measures computed on specific locations of a mechanical piece). BIM tools, now widely used in engineering and architecture, can be conceived as a direct derivation of the concept of annotated geometry.

Therefore, the support of annotations has been included in many 3D systems; it is a feature offered also on recent web-based applications oriented to professionals [YH13, AAB*14, CFG*15, ABC*18].

3. Annotations - Selecting the spatial reference primitives

An annotation can be associated with different subsets of the object of interest. We can characterize those subsets in three classes by considering the *dimensionality* of the reference, thus producing different geometric primitives as the result of an annotation:

- ***Point***: a single point-wise position in the object 3D space is used to locate and to link the annotation;
- ***Linear***: a (poly-)line is defined over the surface for the identification of linear structures (e.g. fracture lines, discontinuity lines);
- ***Region***: an area is usually specified by an irregular polygon. It is used to define a subset of the surface which would be associated with a given annotation or class.

A fourth type of annotation, mainly used in the medical field where volumetric data is available, is based on the selection of (sub-)**volumes** [PLM09]. But in this paper we concentrate on surface 3D models and on the typical annotations used with this type of data. Even when the surface data is treated in a volumetric way (see next section) and the annotation is extracted by means of a volume, it is still representing the output of a region selection.

Most applications would explicitly require to support all those three annotation types, usually assigning them a different meaning or a different purpose. In some systems (especially the ones based on a discrete representation, i.e. raster images or voxels spaces) for all three classes above the annotation might end up with the selection of a subset of elementary cells.

The implementation of those classes presents some similarity at least at the GUI and interaction level since the user has to select some positions in view space and transform them back in object space. In the first case, a single point is returned and stored. In the second case, the user usually selects a concatenated set of points, which can be rendered as a polyline. The third case is more complex since the initial input is a closed loop of points in view space, but different implementations will produce different results: a selected group of pixels / texels / voxels in an image / texture / volumetric space; or a selected group of triangles belonging to the original mesh; or a new set of triangles trimmed over the selected profile.

4. Implementing annotations

A central phase in the implementation of an annotation system is the procedure to select the region to be annotated. Different approaches have been proposed for the region selection and encoding phase, which strongly depend on the *working space* and on the *representation scheme* adopted for encoding the object of interest. Those different spaces are graphically depicted in Figure 3. We review and discuss those approaches in the following subsections.

Annotations are rarely a one-to-one characterization of the surface area and they would often overlap. Therefore, annotations have to support the possibility that several different annotation *id* might be associated to the same element (pixel / texel / voxel / triangle). In many applications, several different characterizations can be associated to the same parcel of the surface. Indeed CH applications might devise many different characterizations: materials, degradation or conservation status, type of restoration intervention occurring over the surface, and so on and so forth. Similarly, medical or biological applications have to characterize a dataset taking into account morphological characteristics, delimitation of organs, different level of activity of the different tissues, etc.

Finally, the management of annotations usually implies pairing some sort of data archive to the geometric representation of the shape. Such archive stores the *content* associated with the annotations (i.e. free text or any other media and content supported by the annotation system, see Section 7) and the established *links* between the geometry and the content. In some literature this is called

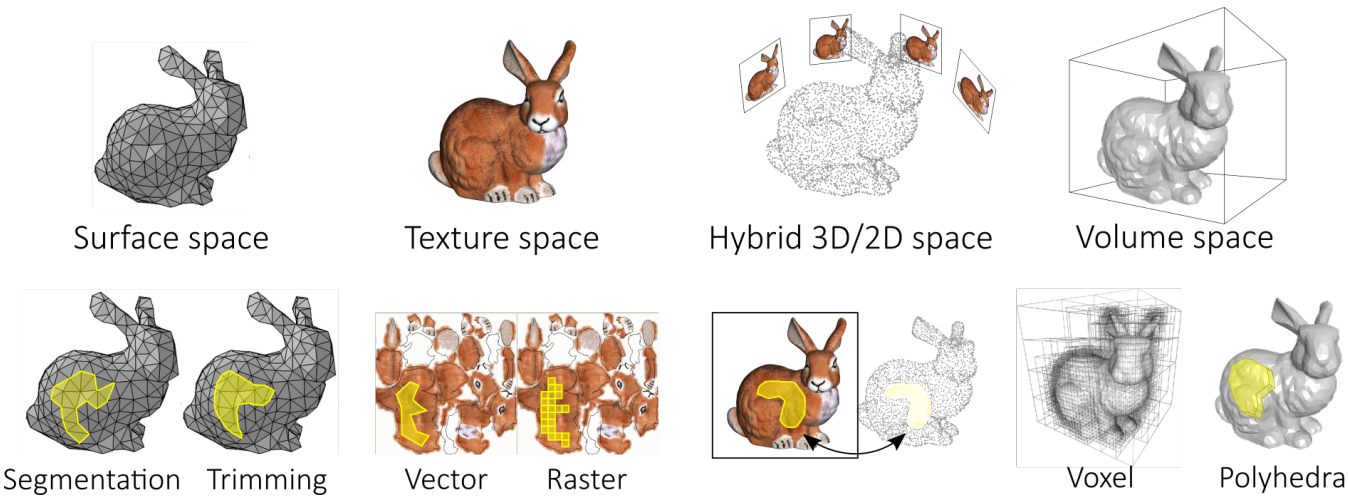

Figure 3: *A graphic representation of the different annotation approaches .*

persistent annotation in order to make the distinction between this approach and some tags or data directly encoded in the geometric model.

4.1. A taxonomy of implementation approaches for annotations

The *point* annotation is the easiest one to manage and requires a simple internal representation: only three spatial coordinates and, sometimes, a normal direction. It is usually implemented by picking a point over the surface of the digital object.

As far it concerns the other two types, *region* or *linear* annotations, we can classify the different annotation methods by first focusing on the geometric space where the action takes place. We have four different geometrical approaches:

- ***Surface space***: selection of a set of *triangles* or *points* over the surface of the 3D model. Two different ways of implementing are coexisting:
  - *Segmentation*: a subset of the elementary 3D model primitives (triangles, points) is selected and assigned to the new annotation [YH13, BCPS16, SSD*11];
  - *Trimming*: a portion of the mesh is clipped over the profile of the annotation, eventually cutting the triangles crossing the annotation border, usually storing them as a new mesh parcel.
- ***Texture space***: when a UV parametrization is defined over the 3D surface creating a link between the 3D space and a 2D texture space. We can also work in the associated texture space to delimit a portion of the surface using 2D primitives. Also in this case we have different options:
  - *Raster*: the area is represented directly using the pixels of the texture, by selecting a subset of pixels;
  - *Vectorial*: lines or regions are defined over the texture space by using 2D polylines.
- ***Hybrid 2D-3D projected image space***: annotations are drawn on photos or rendering with a known / calibrated reprojection matrix, and then propagated onto the 3D surface or point-cloud by projection [MDLV14, MVL16].
- ***Volumetric***: the requested portion of the surface is defined by intersecting the model with a *volume*. The annotation is not connected to a volume, but still restricted to the intersecting 3D surface. Again different interpretations are possible:
  - *Voxels enumeration*: we directly select a subset of voxels using for example an octree and selecting specific nodes over the hierarchical structure [STLL13]; we are thus using a *raster* approach.
  - *Simple volumetric primitives*: elementary solid primitives (i.e. spheres, cylinders or axis-aligned boxes) are used to intersect the object surface [SSRS12, REMA09]. This is one of the simplest approach to implement since the intersection of elementary primitives towards a 3D model is a basic geometric operation supported by most libraries; however, due to the lack of flexibility provided by the set of elementary primitives, this selection is often highly imprecise.
  - *Clipping polyhedra*: the annotation region is defined as the intersection of the 3D surface with a clipping volume defined by a closed manifold mesh (see Section 6).

This characterization is used in Section 8 to produce a cross-comparison of the different methods (see Table 1).

Some applications offer the capability of representing the object of interest through multiple media, also encompassing different spaces (e.g. storing both 2D images and 3D models to represent a single object). Supporting *annotation transfer* to multiple representations thus becomes a fundamental issue (see Section 5).

Annotation methods could also be characterized by focusing over the specific GUI approach offered to the user. According to the users' point of view, this is often the most evident difference. Therefore, we can introduce also the following *GUI-related characterization*, somehow orthogonal to the previous one:

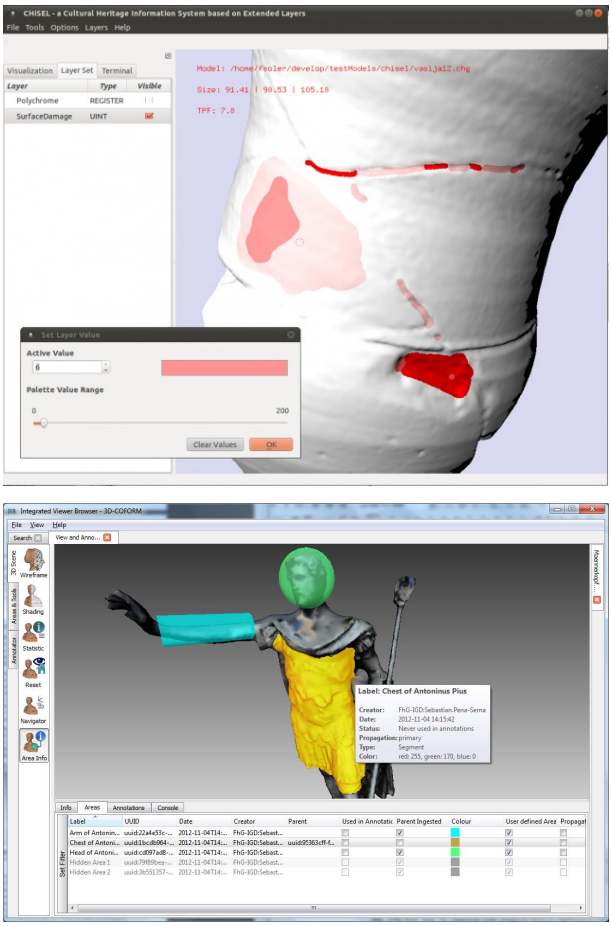

Figure 4: *The result of a painting-driven selection of annotated lines and regions in the Chisel system [STLL13] (on the top); assisted selection of a region in the IVB system by Fraunhofer-IGD [SSRS12] (on the bottom).*

- ***Vectorial***: the user draws polylines over the surface; these are converted in a list of editable 3D control points of a connected sequence of segments, used to define the region or the linear element over the supported representation space (e.g. [SBJB*14, ABC*18], see Figure 7).
- ***Painting***: by adopting a painting metaphor, the area is defined sweeping the mouse over the surface; all primitives *touched* (points, texels, triangles, voxels) become part of the selection (e.g. [STLL13], see Figure 4-top).
- ***Assisted (shape kernels)***: the GUI uses an assisted segmentation approach, usually based on some geometry-based kernels (such as discontinuity of normals, proximity, uniformity of colour, clipping volumes) helping the user in the selection of the area of interest (e.g. [SSRS12], see Figure 4-bottom).

Finally, there is a long tradition of systems supporting annotation over 2D images (e.g. microscopy images or slices from CT or MR 3D images in medical or biological applications). Existing software, consolidated approaches and user experience can be leveraged in the case of 3D applications. Therefore, taken into account again the GUI approach, the available systems can be subdivided in:

- ***Annotation interface working in 3D space***: the user works directly in 3D space to select the specific regions or elements.
- ***Annotation interface working in 2D space***: the system proposes to the user a 2D space where the annotation can be selected, to simplify the interaction. The result is then mapped automatically to the associated 3D space.

Existing approaches are presented in the next Subsections, following the four classes characterization: Surface, Texture, Hybrid and Volumetric. Each subsection contains a list of pros and cons characterizing each approach.

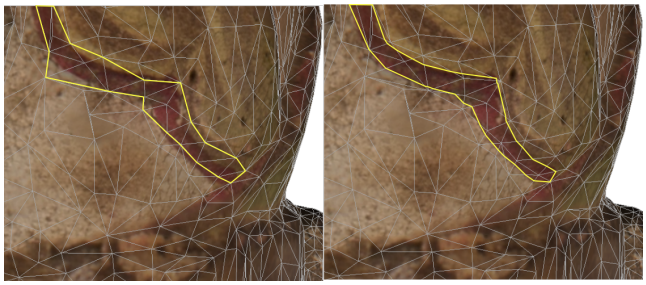

Figure 5: *An example of a portion of surface with a region to annotate with two different solutions (delimited by the yellow polylines): on the left, the smallest subset of triangles overlapping the region of interest; on the right, the yellow profile delimits more tightly the required annotation region and requires to trim the mesh.*

4.2. Surface space - Annotations over 3D meshes

Annotation over a mesh-based representation involves first selecting the region to be trimmed (following the closed chain of points defined in *view space* by the user), then transforming back those points in *object space* and finally determining the subset of triangles affected by the action. The latter task can be implemented in different manners:

- *Best overlapping subset:* we can detect the subset of the original triangles which best-fits the annotation region (see Figure 5-left). In this case, the geometry and topology of the mesh remains unaltered, we only need to add a new annotation *id* to the selected triangles. Multiple overlapping annotations are enabled by supporting the assignment of one or many annotation *id's* to each single triangle.
- *Trimming the selected region:* this solution modifies the current mesh, by trimming the triangles which overlap the annotation border. The advantage of this solution is that the profile of the annotation stored in the system fits exactly the profile defined by the user (see Figure 5-right). The disadvantage is that by splitting the geometry we introduce *fragmentation* in the mesh, new vertices and changes in the 3D model. Moreover, implementing overlapping regions, in order to produce different information layers, is more complex and leads to increased mesh fragmentation.

Several systems based on the approaches above have been proposed. An interactive semantic enrichment tool for 3D CH collections was presented in [SSRS12]; it is fully based on the CIDOC-CRM schema and supports its sophisticated annotation model. It adopts a triangle-based representation and offers features for the easy detection of the mesh subset (intersection with elementary primitives, such as spheres or cylinders) to perform more sophisticated selections of the triangulated areas to be annotated.
The ChER-Ob system [SKA*16] allows the production of annotations on 3D meshes models, by implementing selection mechanism for vertices, rectangular regions over the surface or volumetric subsets, delimited by a view frustum volume.
The 3DSA system [YH13] allows the creation of annotations on 3D surfaces. The web system works only with single resolution models, and regions are defined by selecting *best overlapping subsets*.
The surface-based approach is also quite common in medical applications [BCPS16].

Surface / Mesh-based methods
Pros:

+ Easy to implement.

Cons:

− The geometric resolution of the model, in terms of density and uniformity, limits the precision of the annotation, especially when triangles are not trimmed. This limitation becomes very notable with the low-poly representations very common in web visualization: simplified geometry with high quality textures;
− Annotation storage can be expensive if implemented naively, and conversely optimized storage of annotation data is complex;
− Transferring an annotation to a different representation, as for multiresolution encoding or for the use of multiple representations, is not straightforward (see Subsection 5.1).

4.2.1. Surface Space - Annotations over point clouds

A region-based annotation over a point-cloud can be simply defined by selecting a subset of the points. The lack of a defined surface makes it a bit complicated to translate the user input into a set of points, especially for models with uneven and / or low sampling density or for an arbitrary point of view. To overcome the lack of a proper surface, point-splatting as well as other interpolation and rendering techniques may be used to trace annotations using a more continuous surface defined by the point cloud. One of the most diffused point cloud web viewer, POTREE [Sch16], allows the creation of annotation by offering features for the selection of vectorial elements (points + polylines) that snaps to the closest points and for volume-based selection. None of these annotations are transferred on the input point cloud, conversely they are created as independent 3D elements (data replication).

4.3. Hybrid 2D-3D projection space

A different approach is to work on 2D images that are registered (i.e. calibrated and oriented) to the 3D model. In this way, annotations are created using a simple 2D input and then transferred into the 3D space through perspective projection. This principle is quite flexible, as the target 3D space may be a point cloud, a 3D model surface or even the UV texture space.

This solution comes naturally when the 3D mesh or point cloud is the result of a photogrammetric / structure from motion (SfM) reconstruction. In this case, the digital model can be stored by coupling the 3D point cloud with the related set of photographic images, with an implicit perspective projection linking each photograph to the 3D point cloud.
This approach has been recently proposed by the Aioli system [MVL16], which works with point-based 3D models and implements annotations by working in image space. Users select the annotation regions in image space [MDLV14, MVL16], starting from any of the photographs available. For each photograph, Aioli stores the camera parameters needed to project the image on the 3D space and vice versa. The user can, therefore, select the best image documenting the area of interest and then he draws the annotation regions directly on this images (e.g. by drawing a polyline in 2D space). Once defined, this region can be first projected back to the 3D representation and then automatically propagated to all other relevant images. The key element is being able to detect all the 3D points contained in the annotated region. Once we have this subset of points, we can reproject them in any other images to propagate the annotation. In this way, a single annotation action performed on a single image is automatically propagated to both the 3D model and to all images that "see" the same portion of the surface.

The focus of the AIOLI system, *trans-media annotation*, is an important objective, usually neglected in most systems. Another tool, the CHER-OB visualization and analysis platform [SKA*16] also allows the management of annotations over 3D models and images in the same tool, but here annotations made on one media are not directly connected with (or translated into) a corresponding annotation on the other media.

Surface and Point-clouds 2D-3D projection
Pros:

+ Adopts an easy-to-use image-based selection approach; well suited for 3D models produced with photogrammetry/SfM;
+ Implements a powerful *annotation propagation* feature, able to propagate from one image to the 3D model and then to all other images containing the same region. This feature greatly reduces the manual efforts of annotation (since every single action is propagated to all media);
+ Allows a web-friendly implementation since a 3D interface is not even necessary.

Cons:

− Fitting precision depends on the density of the mesh or of the point cloud, it depends also on the density of the photographic coverage. In the case of unevenly sampled models, reprojection will produce imprecise results over the other images, thus making propagated regions imprecise;
− Reliable projection of the 2D selection on the 3D data and reprojection can be complicated (i.e. handling self-occlusions) and rely on good quality models with dense and even sampling;
− In some cases, forcing the user to make the selection on a single image could be inconvenient (e.g. in the case of regions covering a highly curved portion of the surface, not visible in their entirety in a single photograph);
− Only a single type of dataset can be used: a set of calibrated images and a very dense point cloud.

4.4. Annotations over UV texture space

An annotation system often manages datasets in which the 3D model encodes both geometry and appearance attributes. This is usually obtained by exploiting textured meshes: a 3D mesh, an UV parametrization and the associated texture map. Therefore, we could also take advantage of the available parametrization and create our annotation in the texture UV space, back-propagating it in 3D space or simply encoding it in the texture and using the annotated texture at rendering time. This type of approach is common with GIS systems, which often support the annotations over textured space. Conversely, a number of issues make this approach quite complex to implement on standard textured 3D models.

Actually, directly working on the 2D texture image is often impractical. The texture image could not convey an easily intelligible representation of the 3D object, due to texture distortions and since the texture space is usually fragmented (we will return on this issue later). Obviously, we can implement a 3D input interface for defining our annotations and transfer those selections to the UV space, creating 2D vectorial data.

Working through a paint interface is another possible solution, where areas can be selected using a brush-stroke paradigm. Paint interfaces are quite flexible, intuitive, and make the selection of complex areas easier. Various 3D painting tools exist on the market (e.g. ZBrush, 3D-Coat, Mudbox, Substance Painter), supporting painting over the texture space. However, they mostly provide instruments to "mask-select" the surface and do not include vectorial elements; moreover, using them exclusively for data input and then store and visualize the annotations in a different tool would be difficult, as they lack a direct and exploitable way to output the selected areas. Since some of these tools (e.g. 3D-Coat) implement also the concept of *layers*, implementing the entire annotation workflow (i.e. creation, data storage, visualization) becomes possible inside one of these tools, even if this would be stretching out the original software purpose.

From the data management perspective, working in a 2D space has several advantages: 2D layered data is compact and pretty standard; GIS-style operations (i.e. intersections, unions, geometric processing) become possible; implementation of web-based systems is facilitated, since transmitting textured data usually requires less bandwidth than transmitting geometry.

The possible fragmentation of the texture appears as the major disadvantage of this method. Only simple shapes can be parametrized over a continuous and undistorted texture space. Using a single projection to cover the whole object is easy only on 2.5D objcts (bas-reliefs, walls with frescoes), common in some archaeological settings (e.g. [KTKK08]). Conversely, more complex shapes produce complex textures containing seams, lack of continuity, distortion, etc., which make extremely difficult to manually edit the texture file. This introduces a set of strong requirements on the parametrization especially when working with vector data.
Architectural models have been proved to be a well-fitting case for this texture-based approach since it is possible to split the model into mostly flat surfaces (e.g. the ruins of Pompeii ruins in [CLDT16]).
In more complex cases, the goal is to split the model into pieces, such that a contiguous parametrization with low distortion can be found on each piece [SBJB*14], as shown in Fig. 6.

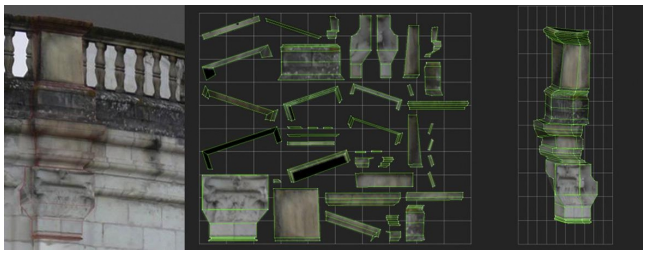

Figure 6: *The figure shows the non-trivial job of turning a fragmented parametrization (texture in the middle) into a contiguous one (texture on the right) [SBJB*14].*

Therefore, the main obstacle in the *UV texture approach* is that the vast majority of available 3D models are not suitable for this method: models with complex topology usually producing highly fragmented and distorted texture spaces; point clouds, and in general 3D models created from images (SfM/photogrammetric pipelines), again present extremely fragmented parametrizations; and, obviously, we cannot manage models which do not have an associated texture, such as models adopting a per-vertex encoding of the colour attributes.

UV Texture space
Pro:

+ Easy to implement, both on desktop applications or on the web;
+ Leveraging of 2D tools and related data structures.

Cons:

− Limited to meshes with well defined and clean UV parametrization (quite rare);
− Introduce heavy constraints on the quality of the UV parametrization
− Selection of regions becomes very complex and imprecise in the case of fragmented or distorted textures.

4.5. Annotations in voxel space

Voxels spaces are widely used in many application contexts, namely all those where the input data is discrete volumetric samples of the 3D space (e.g. CT or MR data). As briefly mentioned before, the most straightforward approach is to work on 2D image sections and to apply algorithms able to propagate the segmentation/annotation regions to adjacent image slices.
Features for the direct selection of 3D voxel regions are usually not provided, the interactive selection of complex shapes in a volumetric space being a difficult task.

Volumetric representations are used also to represent complex shapes, not only in the medical and in the biological domains but also for CH applications. An example of a system based on a volumetric representation is the CHiSEL system, supporting the documentation of artistic and restoration knowledge [STLL13]. CHiSEL starts from 3D scanned data (triangle meshes) and converts this data into a volumetric representation based on a classic

octree structure. CHiSEL also allows the creation of layers of raster information (i.e. single numeric values or labels) over the surface of the artwork, created by binding a set of voxels to a common value, which is stored in a relational database. Each single voxel can therefore be associated with several attributes either assigning one or more numerical values or the *id* of some categories. The selection of a region is performed in CHiSEL by using a painting-style approach using a graphics tablet (Figure 4.top).
Some of the disadvantages of the CHiSEL approach are: linear annotations are discretized and a user-defined polyline would be rasterized in voxel space; storage occupancy could become easily an issue, when we need a very high resolution. We have to store a huge voxel space if we need a half-millimetre resolution for an object of size 5*5*5 meters, i.e. 10K*10K*10K voxel space; even if represented with a compact octree, the space occupancy easily becomes significant. Thus, the voxel-based approach introduces efficiency issues and, at the same time, makes a web implementation quite hard.

The Agata system [SML17] has been designed as a follow-up of the CHiSEL system [STLL13]. The Agata system adopts a hybrid representation to solve some shortcomings of the previous approach: data representation is mesh-based, namely a multiresolution encoding built over a triangle-mesh input, by adopting the Nexus library [PD16], which encodes the surface by a network of surface patches; this mesh-based representation is also spatially indexed by means of an octree. Features for producing line- and region-based annotations are provided.

Volumetric - CHiSEL
Pro:

+ Since the representation is a 3D raster space, and, as such, discrete, transforming user input in a characterization of the surface becomes quite easy to implement.

Cons:

− Since CHiSEL use a discretized approach, the resolution used has an impact on the accuracy;
− On complex application testbeds, the space occupancy requirement could become excessive, introducing efficiency problems;
− Quite complicated to be implemented on the web, due to the large requirement in data transmission.

5. Coping with complex or multiple 3D representations

Sophisticated applications often require to process complex, very high-resolution models or to manage a multiplicity of data representations. The necessity of implementing modelling system having a web-based interface further complicates the problem, since space- and transmission-efficient data representations have to be used. Therefore, two emerging (and interrelated) problems are:

1. The availability of *multiple representations* of the same object. Two clear examples are: the multiple LOD representation that can be produced with geometry simplification codes [Hop96] or by cutting portions of an original model and pasting them to create derived models (e.g. the head of a statue produced from the full body model). In those cases, we need technology able to *propagate* annotations over the entire set of related models archived in the application repository.
2. The necessity of using *multiresolution encoding* schemes [Lue03] to comply with the efficiency and accuracy requirements of several technical applications. The use of a multiresolution encoding complicates the definition, computation and management of annotations. This was noted in a the Agata system mentioned above in Subsect. 4.5 [SML17]. Indeed, a multiplicity of representations require the propagation of the annotation. At annotation definition time the adoption of a view-dependent rendering mode introduces some additional issues (e.g. is the mesh rendered in this instant the more adequate to select and clip the annotated region?).

5.1. The Transfer Annotation approach

Scalas et al. [SMS17] proposed the *Transfer Annotation* approach as a solution to the first problem. A selected region is defined by a set of triangles T_1^k which are associated to the annotation k over model M_1. The authors propose an approach to propagate it to any other model M_i, assuming the latter is derived by M_1 and defined in a space congruent with the one of M_1. The approach proposed is based on the selection of the minimal set of triangles T_i^k in M_i that best approximates the region defined by T_1^k. According to the initial hypothesis of this approach, the encoding of the annotation region should not imply a topological change of the triangulated surface (e.g. no triangles trimming to prevent fragmentation of the mesh); conversely, the annotation is encoded by assigning an annotation id code to each corresponding triangle in mesh M_i.
Propagation can then be executed as a batch process, given a known set of meshes which represents the same object. The implementation has to consider a small number of special cases which complicates a bit the implementation of the correspondence algorithm.

Transfer Annotation solution
Pros:

+ It allows the propagation of annotations over different triangulated models of the same object;
+ The proposed solution is sufficiently easy to implement, even if special cases occur and have to be treated.

Cons:

- It produces an approximate fit, with different potential shapes of the annotations over different models since the definition of the matching triangle patch is strictly dependent on the specific resolution and topology of each model;
- Degenerate cases can be easily introduced when small annotation regions are defined and when the impact of the different level of coarseness of the meshes may lead to triangles degenerating to lines of edges.

5.2. Annotations over multiresolution representations

If the 3D object is represented with a *multiresolution model* [CGG*05, PD16] tracing annotation areas becomes a challenging task. Let us take into account an application which uses a multiresolution encoding and, moreover, works on the web using the classical client-server approach (i.e. 3D data stored on the server; on-demand transmission to the client of a view-dependent representation fulfilling the specific view requirements). The adoption of a

multiresolution approach is a mandatory solution in all those applications which require both high fidelity of the data and efficiency in the rendering process as well as on-demand and selective transmission of geometry data, as it is for web applications. Multiresolution approaches allow the extraction of view-dependent representations with a very efficient traversal of the base representation; each view-dependent representation is optimized with respect to the current real-time view specifications [PD16].
Unfortunately, these highly effective solutions become quite complex to manage when we have to define a region of interest over the rendered surface. Following the mouse clicks and selecting a set of points in the current view space is quite easy, thus producing a concatenated set of points in view space that can be converted back into view rays in object space. However, if we want to precisely compute the intersection in object 3D space of those rays with the object surface, we need to have access to the full resolution mesh.

6. The ClippingVolumes approach

We have introduced in Section 4 that regions associated to an annotation can be selected and represented as the part of the surface intersecting a given volume. Following this approach, the Cher-Ob system uses view frustums to clip the subset of the surface to be associated with an annotation [SKA*16].
Clipping volumes have an obvious advantage: the definition of the annotation region is somehow *independent of the model*, offering a simple solution to the problem of annotation transfer and management of level of detail or multiresolution schemes.

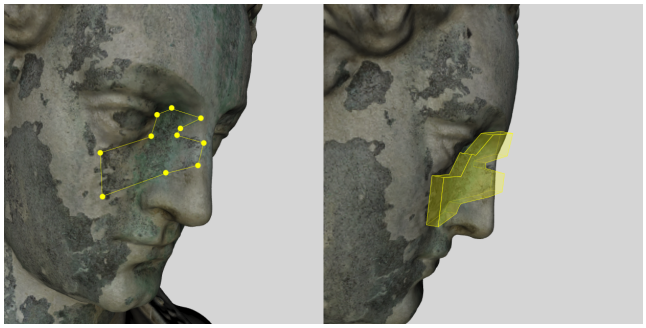

Figure 7: *The ClippingVolumes approach starts from a closed polyline drawn by the user and then creates a clipping volume (presented visually in the image on top-right only for illustration purposes); the latter is used at rendering time to highlight the selected surface chunk with a specific colour.*

We present here an elegant and efficient solution to the problem of managing annotations over multiresolution representations as well as to transfer annotations over multiple models. This solution emerged while designing a system for documenting the restoration of CH artworks [ABC*18]; this system provides extensive instruments for annotating and mapping data over the 3D representation of the artwork (Figure 9). We call this innovative solution *annotation trimming with Clipping Volumes*, in brief **ClippingVolumes**, and describe it in detail in this paper.
The *ClippingVolumes* solution identifies the annotation region as the intersection of the triangle mesh with a 3D volume defined by a closed, watertight triangle mesh. Rendering is performed by using the original mesh and performing an ad hoc rendering of all the annotation regions, where the triangle portions inside any annotated region are rendered with a different colour. This allows rendering of several potentially overlapping characterizations in real-time (Figure 9), tightly fitting the annotation profiles and without introducing mesh fragmentation issues.
The solution has been designed for a *web-based client-server* architecture. Consequently, in each instant of time, the local client holds a 3D representation defined at a variable level of detail which depends on the current view specifications and the hardware capabilities of the local client computer [PD16].

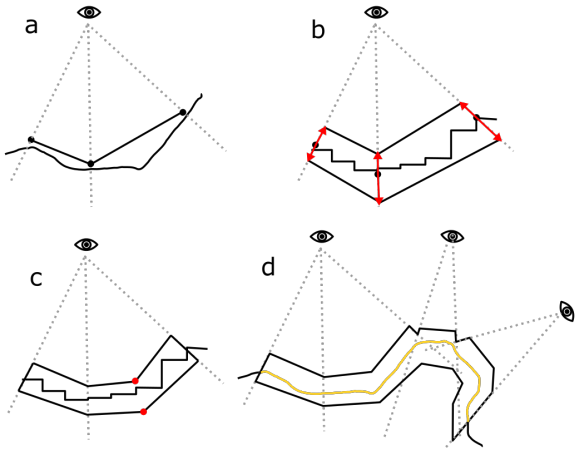

Figure 8: *ClippingVolumes: the drawings represent graphically the process: a) triangulation of the polyline, b) rasterization of the model and creation of an offset, c) refinement. Finally, the last image (d) shows how we manage the problem of regions which cannot be seen from a single view point, by performing a union of several clipping volumes.*

Clipping volume definition. As usual, the user is asked to trace a polyline delimiting the requested region; this polyline is traced in the currently rendered view space (Figure 7.left). Given this input data, the ClippingVolumes computes an associated bounding volume and uses this volume to define and then visualize the selected surface region (Figure 7.right). Most important the ClippingVolumes solution decouples creation and the archival of the annotation from its rendering.

Clipping volume computation. The remote client cannot perform the computation of the clipping volume since it usually holds only view-dependent data and not the full resolution mesh. Therefore, a *server-side service* gets in input: (a) the set of points (polyline vertices in 3D coordinates) which bounds the region and (b) the current view point (at polyline definition time). The view point allows to project those polyline vertices over the surface S and to triangulate the region defined by the polyline, resulting in a triangulated patch A_t (Figure 8.a). To compute A_t we use a planar triangulation algorithm, which works in view space. Our plan is to extrude A_t along

the viewpoint direction to create a truncated prism which should follow the surface and avoid intersecting other parts of the model (Figure 8.b). To create this prism we should fix a depth associated to each vertex of A_t (see the red arrows in Figure 8.b). In order to define these depth values we rasterize in view space the real surface S to compute the min/max offsets which define the minimal width of the prism.
The triangulation A_t can be further refined to better follow the surface with the extruded clipping polyhedra (Figure 8.c). This further refinement has been introduced to cope with very thin walls to avoid including, in the interior of the clipping volume, also the back surface of a thin wall.

Users could need to define an annotation region not visible in its entirety from a single view (i.e. the selection of the entire finger in a hand). The ClippingVolumes solution manages those cases by simply enabling the possibility of merging nearby annotations. A complex region can therefore be identified by two (or more) view points (Figure 8.d), producing multiple clipping volumes which are unified with a simple Boolean union operation.

Therefore, the computed clipping prism implicitly *defines* the selected region of the surface at full accuracy since it has been computed on the full resolution model. The prism is also used to compute some related numeric data (e.g. surface area, perimeter). Those values are stored in the annotations data base, together with the specification of the clipping volume, stored as a small mesh describing the boundary of the truncated clipping prism.

The description of a clipping volume is short enough to be effectively stored in the element that describes each annotation in the associated annotation database. Its space complexity is linear with the number of vertices selected in the border polyline and can be easily sent back to a remote visualization client (in the case of a web-based system). Therefore, the approach proposed here and implemented in the Neptune Fountain's documentation system [ABC*18] is to ingest dynamically the data of all annotations into a database, rather than encoding this data in the 3D mesh representation (see Figure 9).

Using clipping volume at visualization time. According to the visualization needs (i.e. during interactive navigation over the mesh or to visualize the results of specific queries to a documentation system) the system will be requested to visually render (a subset of) the available annotations.
At rendering time we use a two-pass rendering process for each annotated region. Using a *stencil buffer* rendering approach, all the surface portions laying inside each clipping volume are rendered with a specific colour overlay. We took inspiration from the *shadow clip volumes* approach [Eve03]: instead of producing region under shadow as in the original paper, this approach is used to colour each annotation region with the tint associated to the given annotation. To be more clear: at rendering time we do not use trimmed chunks of geometry (which is NOT stored explicitly in the database), but just use the defined clipping volumes to render in a different manner the triangles (or portions of triangles) intersecting those clipping volumes.
The overhead is only limited to the second pass rendering of the geometry of the clipping volumes, thus quite small.

By definition, we can easily render annotation regions on any representation, since we are not using triangles patches but a specific rendering approach for the (portions of) triangles contained in the clipping volume. Therefore, the ClippingVolumes approach is able to support applications using discrete LOD representations or multiresolution schemes (enabling view-dependent rendering modes), as well as the *propagation* of annotations on any congruent representation of the same object.
In addition the ClippingVolumes solution could also be easily extended to manage the case of point-cloud representations or to produce precisely fitting annotations on coarse-textured models.

The proposed solution is also inherently *web-friendly* since it works on multiresolution models (triangle-meshes or point-clouds), requires little computations at the remote client level, rendering can be efficiently demanded to the GPU, the annotations data structures have a very small footprint, and result quite efficient as thousands of annotations can be managed on the same model.

Two short video clips showing the tracing of annotations and the navigation of annotated data in real-time are accessible at: `https://youtu.be/ZroSWSywx2s` and `https://youtu.be/HDYwE85Iips`; those examples are related to the Neptune Information System testbed [ABC*18].

Annotation with ClippingVolumes
Pros:

+ The final annotation profile is exactly fitting the profile selected by the user (exact mapping);
+ It allows to encode and manage easily the case of overlapping annotations;
+ It allows the propagatation of annotations on an entire multiresolution representation or on different models of the same object, since it works at the rendering level;
+ Mesh fragmentation and duplication of geometry are avoided;
+ It provides a web-friendly solution.

Cons:

− The implementation is a bit more complex. It requires working on the geometry characterization and processing side as well as on the rendering side;
− It cannot support a painting-based GUI;
− The integration of annotations coming from other tools (i.e. automatic segmentation) can be complex since they have to be transformed in a ClippingVolumes specification.

7. Connecting and encoding content

The next question to discuss is *what* do we link to a given geometric annotation element.
Annotation over *textual documents* is a consolidated subject, with many works produced by an active community (e.g. see the standardization effort reported in http://www.openannotation.org/).
Annotating more interactive *visual documents* (such as 3D models) is a bit more complex. There are a number of different approaches, from the simple ones to the more complex: we can store just an *id* (simply a numerical identifier pointing to a list of classes or concepts defined in an archive); or some keywords [ARSF07a]

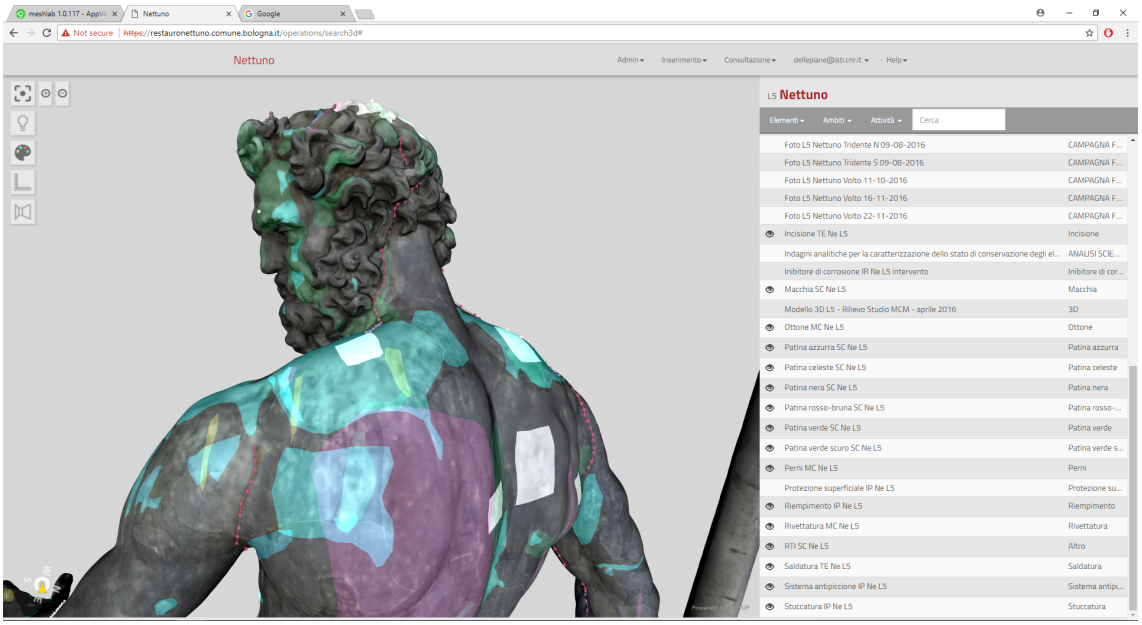

Figure 9: *This snapshot presents a complex surface characterization, created by annotating the surface with the ClippingVolumes method; this result is part of the work produced in the framework of the Neptune Fountain restoration (Bologna, Italy).*

or categories [CFG*15], maybe supported by domain ontologies or taxonomies [MVHL18]; or some unstructured text (this is the case of the ChER-Ob system [SKA*16]). In some cases, the annotation structure simply follows a consolidated approach applied in paper documentation [ABC*18].

A more structured approach, such as a *semantic data representation*, is also possible. The idea of structuring *semantically* the information and creating links among those data arose already several years ago. There is vast literature on this subject [DOS07, ETG*12, RNH12]. Several annotation systems followed this semantic approach. Among the early solutions, Attene et al. proposed a system where manual or automatic annotations where tagged by means of a knowledge base, with concepts built over specific domain ontologies [ARSF07b, ARSF09]. Havemann et al [HSB*09] adopted a semantic approach based on CIDOC-CRM to encode semantic data over a small collection of 3D models. Serna et al. [SSD*11, SSRS12] presented the first interactive semantic enrichment tool for 3D CH collections that is fully based on the CIDOC-CRM schema [Doe03] and that fully supports its sophisticated annotation model. A similar approach has been adopted also by Rodriguez et al. [REMA09] to structure the knowledge over a group of statues offered to the public by means of a web-based system. The more recent Agata system [SML17] also adopts an XML schema and the CIDOC-CRM standard to encode and query the content of annotations. A semantic approach based on ontologies and CIDOC-CRM is also endorsed in [MVHL18] to structure and enable annotation over heritage buildings.

Annotating large collections of 3D models is a costly and time-consuming process, especially when we endorse a user-driven characterization approach. Hunter and Gerber have discussed the potential and issues in endorsing community-based approaches, where annotation and tagging are out-sourced to a community of users [HG10]. But there is some resistance of the professional CH community in adopting crowdsourcing approaches.

8. Comparative remarks and conclusions

This work presented and characterized the different approaches proposed in literature for the management of annotations on 3D models. We introduced several factors which come into play when evaluating the different solutions. We would resume and underline here some critical evaluation criteria or issues in the implementation of the different approaches:

- *3D vs 2D interface:* an annotation can be created directly on the 3D model, while manipulating the digital object, or it can be defined on a 2D image space.
- *Painting vs Polyline Interface:* we described the two options, i.e. selecting the annotation geometry by *painting* over the surface or by precisely individuating a *chain of points*. Pros and cons of these two approaches concern the usability (for professional applications the second one is usually preferred) and the complexity of implementation.
- *Efficient geometric data management:* when datasets are very complex, efficient data management policies are required to minimize data transfer. The required bandwidth for data visualization or for mesh modification is a major problem, for web-based systems and not only. Textured low-poly models help in having a low geometric resolution and in maintaining high visual quality, but annotation might be quite imprecise in this case. The adoption of annotation methods which work well with adaptive or *multiresolution* data management is usually preferred. When we have to manage a large number of annotations over each sin-

	Painting interface	Polylines interface	Fitting precision	Transfer / propagation	Space occupancy	Web friendliness	Multiresolution
Surface - Segmentation	○	○	●	◉	◉	◉	●
Surface - Trimming	●	◉	○	◉	●	◉	●
Texture - Raster	○	○	○	●	◉	○	●
Texture - Vector	◉	○	○	●	○	○	●
Hybrid 2D-3D projection	○	○	◉	◉	◉	○	○
Volumetric - Voxel	○	◉	●	◉	●	●	◉
Volumetric - ClippingVolumes	●	●	○	○	○	○	○

Where: ● : more complex to support or implement; ◉ : neutral; ○ : easy to support or implement.

Table 1: *The different classes introduced in Section 3 are evaluated here according to several functional or implementation issues.*

gle model, such as the case of complex restoration projects, the space required to encode and store the annotations also plays an important role.

- *Fitting precision:* this depends on the type of representation used and the quality of the data. For example, the selection of a region *tightly fitting* a specific area over the surface requires a dense geometry (in case of a triangle-based or point-based representation) and eventually a good quality texture, defined at high resolution and not fragmented. Different annotation management approaches require by definition a specific type and quality of the input data; some of them are more flexible or robust towards variable data types and data quality.
- *Overlapping annotations:* the capability of creating and rendering annotations with part of the surface selected in common is required by most applications.
- *Implementation complexity:* this criterion takes into account the ease of implementation of the different methods. For example, some approaches require the use of complex geometric algorithms which could be demanded to server-based processing in a web context, also to avoid to implement geometry processing kernels in JavaScript; in some cases, this would be affected by bandwidth issues.
- *Web-friendliness:* it becomes more and more common to implement visual media systems running on the web. The necessity of providing annotation support in a web context introduces some issues; some methods are easy to implement in a web-context (e.g. all methods based on texture or the ClippingVolumes method) while others stays complicated to implement (e.g. the ones based on discrete volumetric representations since they require a lot of bandwidth and/or processing).
- *Support propagation:* we should consider the problem of propagating the annotation over multiple representations of the same object.
- *Unprojectable regions:* finally, most of the available systems perform the selection of an annotation region in 2D viewing space, picking a closed chain of points over the surface. This is true for both 2D methods, working on a single image and 3D methods, selecting a view and then selecting the region border. In the case of complex shapes, a user could need to select a region that is not visible in its entirety in a single view such as the neck of a statue or a portion laying over a folded area of the surface. In that case a feature able to merge multiple regions into a a singla annotation allows an iterative selection process on different views. As an example, this feature was included in the documentation and annotation system designed for the Neptune Fountain restoration [ABC*18].

Table 1 lists the major approaches presented in Section 4 and presents a qualitative evaluation with respect to the key elements above.

In conclusion, we reviewed the state of the art related to the solution proposed for the geometric identification of annotations and their related management, as well as presented in detail the new *ClippingVolumes* algorithm, proposed for the management of annotations in a complex documentation system supporting CH restoration. Supporting annotations is quite a complex feature to be added to a documentation or visual analysis system, but it is a very important component. Annotations are a basic component for all those systems aiming at going beyond simple visualization since they allow to document the reasoning process or to characterize in a sophisticated manner the digital surfaces subject of study. We have presented several solutions which allow us to implement annotations. Nevertheless, some of the current needs for the usability of tools (web friendliness, support of multiresolution) make some of the classic approaches less practical. Hence, the new solution proposed here, ClippingVolumes, was defined by starting from a set of clear constraints which characterize the design of a modern documentation system (necessity of a web-based interface, adopting a server-client approach, having to manage highly complex and high-resolution meshes and thus requiring the adoption of a multiresolution data representation approach). Given those constraints, the proposed ClippingVolumes solution manages annotations in an efficient and precise manner. The use of all these new technologies (i.e. HTML5, multiresolution encoding, compression methods, view-dependent rendering, annotation support) allows us to develop quite complex and sophisticated tools, running on low-profile devices connected to the web, which should become common working instruments in many domains.

Acknowledgements

...

References

[AAB*14] AUER M., AGUGIARO G., BILLEN N., LOOS L., ZIPF A.: Web-based Visualization and Query of semantically segmented multiresolution 3D Models in the Field of Cultural Heritage. *ISPRS Annals of Photogrammetry, Remote Sensing and Spa-*

tial Information Sciences (May 2014), 33–39. doi:10.5194/isprsannals-II-5-33-2014. 1, 3

[ABC*18] APOLLONIO F. I., BASILISSI V., CALLIERI M., DELLEPIANE M., GAIANI M., PONCHIO F., RIZZO F., RUBINO A. R., SCOPIGNO R., SOBRA G.: A 3d-centered information system for the documentation of a complex restoration intervention. *J. of Cult. Herit. 29* (2018), 89–99. 1, 2, 3, 5, 9, 10, 11, 12

[ARSF07a] ATTENE M., ROBBIANO F., SPAGNUOLO M., FALCIDIENO B.: Part-based annotation of virtual 3d shapes. In *Cyberworlds, 2007. CW'07. International Conference on* (2007), IEEE, pp. 427–436. 1, 10

[ARSF07b] ATTENE M., ROBBIANO F., SPAGNUOLO M., FALCIDIENO B.: Semantic annotation of 3d surface meshes based on feature characterization. In *Semantic Multimedia (Second Int. Conf. on Semantic and Digital Media Technologies, SAMT 2007)* (2007), Springer LNCS 4816, pp. 126–139. 1, 3, 11

[ARSF09] ATTENE M., ROBBIANO F., SPAGNUOLO M., FALCIDIENO B.: Characterization of 3d shape parts for semantic annotation. *Computer-Aided Design 41*, 10 (2009), 756 – 763. Selected Papers from the 2007 New Advances in Shape Analysis and Geometric Modeling Workshop. doi:http://dx.doi.org/10.1016/j.cad.2009.01.003. 11

[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 (May 2016), 707–720. doi:10.1007/s11548-015-1327-6. 2, 4, 6

[CFG*15] CHANG A. X., FUNKHOUSER T. A., GUIBAS L. J., HANRAHAN P., HUANG Q.-X., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., XIAO J., YI L., YU F.: Shapenet: An information-rich 3d model repository. *CoRR abs/1512.03012* (2015). 1, 3, 11

[CGG*05] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Batched multi triangulation. In *Proceedings IEEE Visualization* (Conference held in Minneapolis, MI, USA, October 2005), IEEE Computer Society Press, pp. 207–214. 8

[CLDT16] CAMPANARO D. M., LANDESCHI G., DELLâĂŹUNTO N., TOUATI A.-M. L.: 3d gis for cultural heritage restoration: A âĂŸwhite boxâĂŹ workflow. *Journal of Cultural Heritage 18*, Supplement C (2016), 321 – 332. URL: http://www.sciencedirect.com/science/article/pii/S1296207415001582, doi:https://doi.org/10.1016/j.culher.2015.09.006. 7

[Doe03] DOERR M.: The cidoc conceptual reference module: An ontological approach to semantic interoperability of metadata. *AI Mag. 24*, 3 (Sept. 2003), 75–92. 11

[DOS07] DOERR M., ORE C.-E., STEAD S.: The CIDOC Conceptual Reference Model: A new standard for knowledge sharing. In *Tutorials, Posters, Panels and Industrial Contributions at the 26th International Conference on Conceptual Modeling - Volume 83* (Darlinghurst, Australia, Australia, 2007), ER '07, Australian Computer Society, Inc., pp. 51–56. URL: http://dl.acm.org/citation.cfm?id=1386957.1386963. 11

[ETG*12] ECHAVARRIA K. R., THEODORIDOU M., GEORGIS C., ARNOLD D., DOERR M., STORK A., SERNA S. P.: Semantically Rich 3D Documentation for the Preservation of Tangible Heritage. In *VAST: International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage* (2012), Arnold D., Kaminski J., Niccolucci F., Stork A., (Eds.), The Eurographics Association. doi:10.2312/VAST/VAST12/041-048. 11

[Eve03] EVERITT, CASS AND KILGARD, MARK J. : Robust Stenciled Shadow Volumes (Graphics/Visualization Seminar to the Center for Computational Visualization, University of Texas). URL http://www.cs.dartmouth.edu/~spl/Academic/ComputerGraphics/docs/shadow-volumes-mjk.pdf, 2003. 10

[FKW17] FOLEY J., KWAN P., WELCH M.: A web-based infrastructure for the assisted annotation of heritage collections. *J. Comput. Cult. Herit. 10*, 3 (July 2017), 14:1–14:25. doi:10.1145/3012287. 3

[GA08] GOLDFEDER C., ALLEN P.: Autotagging to improve text search for 3d models. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries* (New York, NY, USA, 2008), JCDL '08, ACM, pp. 355–358. doi:10.1145/1378889.1378950. 1

[HG10] HUNTER J., GERBER A.: Harvesting community annotations on 3d models of museum artefacts to enhance knowledge, discovery and reuse. *Journal of Cultural Heritage 11*, 1 (2010), 81 – 90. doi:http://dx.doi.org/10.1016/j.culher.2009.04.004. 3, 11

[Hop96] HOPPE H.: Progressive meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), ACM, pp. 99–108. 8

[HSB*09] HAVEMANN S., SETTGAST V., BERNDT R., EIDE O., FELLNER D. W.: The arrigo showcase reloaded—towards a sustainable link between 3d and semantics. *ACM J. Computing and Cultural Heritage 2*, 1 (July 2009), 4:1–4:13. URL: http://doi.acm.org/10.1145/1551676.1551680, doi:10.1145/1551676.1551680. 11

[KHS10] KALOGERAKIS E., HERTZMANN A., SINGH K.: Learning 3d mesh segmentation and labeling. *ACM Trans. Graph. 29*, 4 (July 2010), 102:1–102:12. doi:10.1145/1778765.1778839. 1

[KTKK08] KATSIANIS M., TSIPIDIS S., KOTSAKIS K., KOUSOULAKOU A.: A 3d digital workflow for archaeological intra-site research using gis. *Journal of Archaeological Science 35*, 3 (2008), 655 – 667. URL: http://www.sciencedirect.com/science/article/pii/S0305440307001069, doi:https://doi.org/10.1016/j.jas.2007.06.002. 7

[Lue03] LUEBKE D. P.: *Level of detail for 3D graphics*. Morgan Kaufmann, 2003. 8

[MDLV14] MANUEL A., DE LUCA L., VÉRON P.: A hybrid approach for the semantic annotation of spatially oriented images. *International Journal of Heritage in the Digital Era 3*, 2 (2014), 305–320. 4, 6

[MVHL18] MESSAOUDI T., VÃL'RON P., HALIN G., LUCA L. D.: An ontological model for the reality-based 3d annotation of heritage building conservation state. *Journal of Cultural Heritage 29* (2018), 100 – 112. URL: http://www.sciencedirect.com/science/article/pii/S1296207417304508, doi:https://doi.org/10.1016/j.culher.2017.05.017. 11

[MVL16] MANUEL A., VÃL'RON P., LUCA L. D.: 2D/3D Semantic Annotation of Spatialized Images for the Documentation and Analysis of Cultural Heritage. In *Eurographics Workshop on Graphics and Cultural Heritage* (2016), Catalano C. E., Luca L. D., (Eds.), The Eurographics Association. doi:10.2312/gch.20161391. 2, 4, 6

[PCCS11] PIETRONI N., CORSINI M., CIGNONI P., SCOPIGNO R.: An interactive local flattening operator to support digital investigations on artwork surfaces. *IEEE Trans. on Visualization and Computer Graphics, 17*, 12 (2011), 1989–1996. 2

[PD16] PONCHIO F., DELLEPIANE M.: Multiresolution and fast decompression for optimal web-based rendering. *Graphical Models 88* (2016), 1 – 11. doi:http://dx.doi.org/10.1016/j.gmod.2016.09.002. 8, 9

[PLM09] PENG H., LONG F., MYERS E. W.: Vano: a volume-object image annotation system. *Bioinformatics 25*, 5 (2009), 695–697. URL: http://dx.doi.org/10.1093/bioinformatics/btp046, arXiv:/oup/backfile/content_public/journal/bioinformatics/25/5/10.1093_bioinformatics_btp046/1/btp046.pdf, doi:10.1093/bioinformatics/btp046. 3

[REMA09] RODRIGUEZ-ECHAVARRIA K., MORRIS D., ARNOLD D.: Web based presentation of semantically tagged 3d content for public sculptures and monuments in the uk. In *Proceedings of the 14th International Conference on 3D Web Technology* (New York, NY, USA, 2009), Web3D '09, ACM, pp. 119–126. URL: http://doi.acm.org/10.1145/1559764.1559783, doi:10.1145/1559764.1559783. 4, 11

[RNH12] RONZINO P., NICCOLUCCI F., HERMON S.: A Metadata Schema for Cultural Heritage Documentation. In *Electronic Imaging & the Visual Arts - EVA 2012 Florence* (2012), Cappellini V., (Ed.), Firenze University Press. URL: http://digital.casalini.it/10.1400/187333, doi:10.1400/187333. 11

[SBJB*14] STEFANI C., BRUNETAUD X., JANVIER-BADOSA S., BECK K., LUCA L. D., AL-MUKHTAR M.: Developing a toolkit for mapping and displaying stone alteration on a web-based documentation platform. *Journal of Cultural Heritage 15*, 1 (2014), 1 – 9. URL: http://www.sciencedirect.com/science/article/pii/S1296207413000630, doi:https://doi.org/10.1016/j.culher.2013.01.011. 5, 7

[SCC*11] SCOPIGNO R., CALLIERI M., CIGNONI P., CORSINI M., DELLEPIANE M., PONCHIO F., RANZUGLIA G.: 3d models for cultural heritage: Beyond plain visualization. *Computer 44*, 7 (2011), 48–55. 3

[Sch16] SCHÜTZ M.: Potree: Rendering large point clouds in web browsers. *Technische Universität Wien, Wiedeń (Master Thesis)* (2016). 6

[SKA*16] SHI W., KOTOULA E., AKOGLU K., YANG Y., RUSHMEIER H.: Cher-ob: A tool for shared analysis in cultural heritage. In *Proceedings of the 14th Eurographics Workshop on Graphics and Cultural Heritage* (Goslar Germany, Germany, 2016), GCH '16, Eurographics Association, pp. 187–190. doi:10.2312/gch.20161404. 6, 9, 11

[Ske14] SKETCHFAB: Publish, share and discover 3D content over the web. URL https://sketchfab.com, 2014. 3

[SML17] SOLER F., MELERO F. J., LUZÓN M. V.: A complete 3d information system for cultural heritage documentation. *Journal of Cultural Heritage 23*, Supplement C (2017), 49 – 57. doi:https://doi.org/10.1016/j.culher.2016.09.008. 8, 11

[SMS17] SCALAS A., MORTARA M., SPAGNUOLO M.: 3D Annotation Transfer. In *Eurographics Workshop on Graphics and Cultural Heritage* (2017), Schreck T., Weyrich T., Sablatnig R., Stular B., (Eds.), The Eurographics Association. doi:10.2312/gch.20171295. 8

[SSD*11] SERNA S. P., SCOPIGNO R., DOERR M., THEODORIDOU M., GEORGIS C., PONCHIO F., STORK A.: 3d-centered media linking and semantic enrichment through integrated searching, browsing, viewing and annotating. In *Proceedings of the 12th International Conference on Virtual Reality, Archaeology and Cultural Heritage* (2011), VAST'11, Eurographics Association, pp. 89–96. URL: http://dx.doi.org/10.2312/VAST/VAST11/089-096, doi:10.2312/VAST/VAST11/089-096. 4, 11

[SSRS12] SERNA S. P., SCHMEDT H., RITZ M., STORK A.: Interactive semantic enrichment of 3d cultural heritage collections. In *EG VAST 2012 Symp.* (2012), Eurographics, pp. 33–40. 3, 4, 5, 6, 11

[STLL13] SOLER F., TORRES J. C., LEÓN A. J., LUZÓN M. V.: Design of cultural heritage information systems based on information layers. *J. Comput. Cult. Herit. 6*, 4 (Dec. 2013), 15:1–15:17. doi:10.1145/2532630.2532631. 2, 4, 5, 7, 8

[YH13] YU C.-H., HUNTER J.: Documenting and sharing comparative analyses of 3d digital museum artifacts through semantic web annotations. *J. Comput. Cult. Herit. 6*, 4 (Dec. 2013), 18:1–18:20. URL: http://doi.acm.org/10.1145/2532630.2532634, doi:10.1145/2532630.2532634. 2, 3, 4, 6