A VOLUMETRIC APPROACH FOR 3D SURFACE RECONSTRUCTION

A. Guarnieri, A. Vettore, M. Pontin
CIRGEO– Interdept. Research Center of Geomatics, University of Padova, Italy - cirgeo@unipd.it

KEYWORDS: laser scanning, 3D modeling, surface reconstruction, octree, marching cubes

ABSTRACT

In the last years, measuring sensors like the TOF or optical-based terrestrial laser scanners have been more and more increasingly used, given their capability to acquire the 3D geometry of the surveyed object. Applications for these systems span among different fields, such as industry, medicine, land management, heritage and VR environments. Regardless long-range or close-range laser scanner was employed, two follo wing steps need to be performed in order to reconstruct the object shape: range image registration and integration. The former allow for "tailoring" together the acquired point clouds, representing a "view" of the object surface as sampled by the laser sensor. Then, after all views have been aligned each other, a unique representation of the object surface is generated through the integration of those 3D views. At this stage several factors prevent from building the descriptive surface by simple connection of the range images: non-uniform density sampling, measuring and registration errors. To solve for such modeling issues, a volumetric approach has been devised for the generation of a mesh, i.e to create a surface representation from range data acquired by an optical triangulation laser scanner. The developed method is based on the "consensus surface" concept introduced by Wheeler, Sato and Ikeuchi, by which some kind of errors of the range images can be better identified and corrected. Then it has been refined by integration with the so-called "Marching cubes" algorithm, a well used surface generation procedure in the field of Computer Graphics. Finally, the proposed method has been completed with the development of a tool for hole-filling , though its application is limited to little holes with enough regular edges. Pros and cons along with the results of our meshing algorithm, applied to a little statue, will be presented as well.

1. INTRODUCTION

Nowadays, terrestrial laser scanners are playing a more and more growing role in the field of survey systems, given their capability to acquire in relative short time the 3D geometry of a real object. Applications for these measuring sensors span among different fields, such as industry, medicine, land management, heritage and VR environments. For example, in reverse engineering and quality control 3D models are employed to assess the matching of the product with the original design, virtual museums are created and filled with 3D models, whose real counterparts could be physically located even at far distances each other. Parts of the human body, like teeth, feet or the bust, are often surveyed by means of optical laser scanners in order to get a detailed 3D model to support and optimize the application of correcting devices. Lately, virtual environments filled up with 3D representations of statues, bas-relieves and archaeological findings constitute an interesting way for remote users to get a closer interaction with Cultural Heritage objects located all around the world.
Sensors suited for 3D data acquisition can be classified in two main groups: contact and non-contact. The latter class, composed by laser scanner devices, can be further subdivided according the employed measuring principle and the operating range: optical scanners, like triangulation or pattern-based, and Time of Flight (TOF) sensors. Non-contact sensors allow for object surveying without the need to get in touch with it, what it is often the case when dealing with ancient and valuable sculptures or fragile pieces.
After data acquisition, regardless the kind of laser sensor has been used, two following steps need to be performed, in order to reconstruct the object shape: range image registration and integration. The former allow for "tailoring" together the various point clouds, representing a "view" of the object surface as sampled by the laser sensor. Then, after all views have been aligned each other, a unique representation of the object surface is generated through the integration of those views. At this stage several factors prevent from building the descriptive surface by simple connection of the range images: non-uniform density sampling, measuring errors and registration errors. For instance, during data acquisition, due to the shape complexity and sensing device capabilities, some object portions are described with more range images than other (though a little overlap between view pairs is required for the registration stage). This leads to different point density sampling along the whole object, that should be taken into account by the meshing algorithm in order to provide an optimized surface representation of the object. Moreover, registration errors allow for residual distances between overlapping regions, which have to be eliminated to correctly model such parts of the object as a continuous surface.
In this work a volumetric approach has been devised for the generation of a mesh, i.e. to create a surface representation from range data acquired by an optical triangulation laser scanner.
Volumetric algorithms represent a relative new method to the mesh generation problem and seem to be very promising tools given their capability to solve for some issues affecting "classical" methods, like the Delaunay Triangulation or Soucy & Laurendeau approach [Soucy et al, 1996] or the "zippering" procedure developed by Turk & Levoy, as well [Turk et al., 1994].
The developed method is based on the "consensus surface" concept introduced by Wheeler, Sato and Ikeuchi [Wheeler et al., 1998], by which some kind of errors of the range images can be better identified and corrected. Then it has been refined by integration with the so-called "Marching cubes" algorithm, a well used surface generation procedure in the field of Computer Graphics [Cline et al., 1987]. Finally, the proposed method has been completed with the development of a tool for hole-filling , though its application is limited to little holes with enough regular edges. Pros and cons along with the results of our meshing algorithm, applied to a little statue, will be presented as well.

2. THE CONSENSUS SURFACE ALGORITHM

Let's suppose that m range views (Mk , k=1,2,…,m) have been acquired by an optical laser scanner (range camera) in order to build a 3D model of the surveyed object. Such range views cover the whole object's surface and present enough overlap each other in order to be successfully aligned so that a unique point cloud out of the whole measured volume can be obtained.

Assuming that the set of acquired *m* point clouds have been already aligned and triangulated, the next step to be addressed in the 3D modeling pipeline is the view integration, i.e. the building of an object's surface description (mesh) starting from the set of registered range views. Regardless the nature of the adopted method, a meshing algorithm should provide following basic features: flexibility, noise robustness and data averaging. The former requirement means that the algorithm should be able to work with any kind of input data, regardless their geometric configuration. Measured data are always affected by different forms of noise, in this case a meshing algorithm is required to detect and discard too noisy data, which likely don't belong to the actual object's surface. Then, the algorithm should exploit the data redundancy provided by the overlap between range view pairs, by performing a sort of measurements averaging in order to reduce the noise effect due to the unavoidable acquisition and registration error.
Taking many triangulated surfaces in 3D space and converting them to a triangle patch surface model is however a difficult task. The problem is that it is difficult to determine how to connect triangles from different surfaces without knowing the surface beforehand. Not only the determination of connectedness becomes more difficult, but the algorithm must also consider how to eliminate the noise and small alignment errors from the resulting model.
However, it seems that this issue can be solved for by resorting to volumetric methods, which make the surface-merging problem more tractable, as demonstrated by several researchers.

2.1 The Volumetric modeling

In the field of surface generation methods, occupancy grids are the earliest form of volumetric representation. An occupancy grid is formed by discretizing a volume into many *voxels* and noting which voxels intersect the object. The result is usually a coarse model that appears to be created by sticking a set of cubes together to form the object shape. Of course, using small enough cubes, the shape will look fine, but this becomes a problem since the amount of memory required will be $O(n^3)$ where the volume is discretized into *n* slices along each dimension. Fortunately, an algorithm developed for graphics modeling applications has made volumetric modeling a bit more useful by virtually eliminating the blocky nature of occupancy grids. This algorithm is called the *marching-cubes* algorithm [Cline et al., 1987]. The representation is slightly more complicated than the occupancy grid representation. Instead of storing a binary value in each voxel to indicate if the cube is empty or filled, the marching-cubes algorithm requires the data in the volume grid to be samples of an implicit surface. In each voxel, the value D(x) of the signed distance from the center point of the voxel, x, to the closest point on the object's surface is stored. The sign indicates whether the point is outside, $f(x) > 0$, or inside, $D(x) < 0$, the object's surface, while $D(x) = 0$ indicates that x lies on the surface of the object. The marching-cubes algorithm constructs a surface mesh by "marching" around the cubes while following the zero crossings of the implicit surface $D(x) = 0$. The signed distance allows the marching-cubes algorithm to interpolate the location of the surface with higher accuracy than the resolution of the volume grid. Figure 1 shows an example of the interpolation.

2.2 Building the Consensus surface

Given a number of triangle sets (surface meshes) which are aligned wit h respect to the desired coordinate system, the problem now is taking such triangulated surfaces and converting them to a triangle patch surface model. This task is made difficult by the fact that many surfaces are available, and some elements of those surfaces do not belong to the object of interest but rather are artifacts of the image acquisition process or background surfaces.

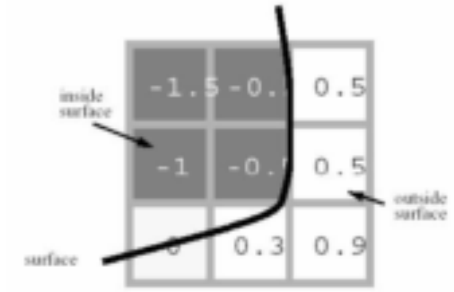
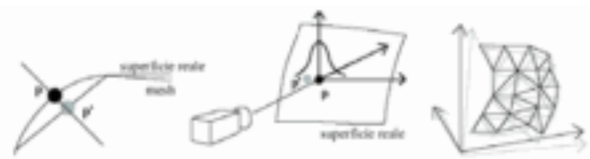


Figure 1: Example of zero-crossing interpolation from the grid sampling of an implicit surface.

The method applied in this work to solve for the range view integration resorts to the *consensus-surface* algorithm, developed by M. D. Wheeler, which is based on the computation of the signed distance function f(x) for arbitrary points x from given N triangulated surface patches of various views of the object surface. In this section the main features of this algorithm are discussed, however more details can be found in [Wheeler, 1996] and [Wheeler et al., 1998]. As described above, the positive value of D(x) indicates the point x is outside the object surface, a negative value indicates that x is inside, and a value of zero indicates that x lies on the surface of the object. Therefore, once distance values D(x) have been assigned to each voxel, the surface representation can be extracted computing the isosurface implicitly defined by $D(x)=0$ with the marching-cubes method. The computation of D(x) can be subdivided into two following steps:

1. Compute the magnitude: compute the distance $|D(x)|$ to the nearest object surface from x.
2. Compute the sign: determine whether the point is inside or outside of the object

However, this simple approach cannot be successfully applied to real data given the unavoidable presence of noise and extraneous data. For example, it is not uncommon to see triangles sticking out of a surface or other triangles that do not belong to the object. This can occur due to sensor noise, quantization, specularities and other possibly systematic problems of range imaging. Generally, three main kind of errors affect the quality of the resulting 3D model, as shown in figure 2: sampling, measuring and alignment error.



(a) (b) (c)

Figure 2: Example of data error due to sampling (a); sensor measurement (b); alignment (c).

During data acquisition the object is surveyed from different point of views in order to capture its whole shape. Moreover, corresponding range images should present a certain level of overlap for the point cloud alignment to be successfully. Despite overlap means that the same portion of an object's surface is surveyed at least twice, it is unlikely that the measuring sensor will capture exactly the same points as in the previous adjacent scan. Correspondingly, two triangles of two different meshes representing the same surface wont be never coincident. A further error source is introduced by the laser sensor itself: range measurements can be modeled as gaussian distributions along the thre sensor axes (X, Y, Z). The higher error variance will be along the depth (usually the Z component), being this one the weakest measuring direction of an optical laser scanner. Finally, the registration step contributes with a residual alignment error which tends to grow with the number of range views to be pairwise registered. This issues make it very easy to infer the incorrect distance and more critically the incorrect sign, which will result in very undesirable artifacts in the final surface. For example, figure 3 shows how a single noisy bump from one view can result in a bump on the final model. Moreover, a badly oriented triangle can create an implicit distance with the incorrect sign. This results in a hole rising out of the surface.

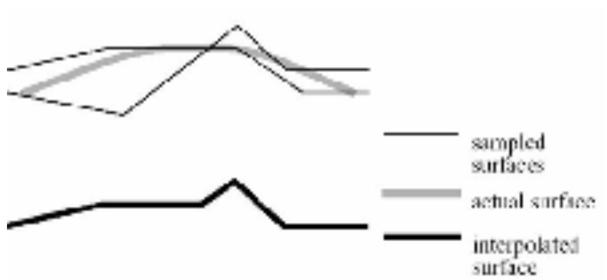


Figure 3: Example of the effect on the final model of a noisy bump in the range data

As proposed by Wheeler, the solution to these problems is to estimate the surface locally by averaging the observations of the same surface. The trick is to specify a method for identifying and collecting all observations of the same surface.
Nearby observations are compared using their location and surface normal. If the location and normal are within a predefined error tolerance (determined empirically), they can be considered as observations of the same surface. Given a point on one of the observed triangle surfaces, other nearby observations from other views, which are potentially observations of the same surface, can be searched for in the 3D space. This task can be accomplished efficiently using k-d trees [Friedman et al., 1977] which is a structure for storing data of arbitrary dimensions for optimal nearest neighbors search. If an insufficient number of observations are found, then these observations can be discarded as isolated/untrusted and the search can continue. Thus, such approach requires to define a quorum of observations before using them to build the surface representation. The quorum of observations can then be averaged to produce a *consensus surface*. As an improvement over using an equally weighted voting scheme, a confidence value is assigned to each input surface triangle: higher values mean that corresponding vertices are less noisy. In this work, the surface points/triangles from a range image have been weighted by the cosine of the angle between the viewing direction and the surface (triangle) normal. This is simply computed by following formula:

$$\omega = \hat{v} \cdot \hat{n} \tag{1}$$

where $\hat{v}$ and $\hat{n}$ are the viewing direction and normal, respectively, of the given triangle.
An example describing the computation of the Consensus surface is shown in figure 4. Here, assuming that three meshes are available and denoted with x the voxel center, in the first step the closest point P1 on mesh 1 is found. Then P2 and P3, respectively on mesh 2 and mesh 3, are searched for as closest points to P1. Computing the weighted average of these three points results in the consensus surface point PC1. In the same way points PC2 and PC3 are determined and if all of them have a weight higher than the threshold *quorum* they are considered valid and point closets to x is kept (here PC1), while the others are discarded. In case none of the points Pi has a weight higher than the quorum, then the point with the highest weight among the three is chosen as consensus surface point, in order to reduce the influence of the noise.

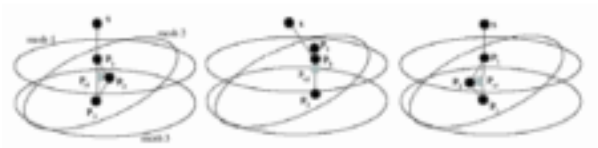
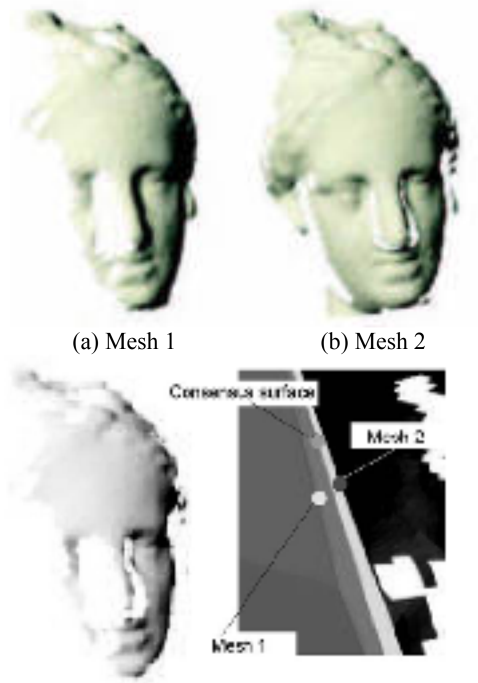


Figure 4: Example of the computation of the points belonging to the consensus surface.

Basically the consensus surface algorithm allows to build a surface representation in terms of an implicit distance function D(x) = s, considering as surface points the average of the points belonging to overlapping meshes. Of course, in case where only one mesh is present (i.e. no overlap is present), then the consensus surface will be described by the current mesh.
An example of the computation of such consensus surface for two meshes is shown in figure 5. Here the points of the resulting surface computed using only one mesh (i.e. outside of the overlapping area) are not displayed.

2.3. The Octree representation

In order to assign the samples of the distance function D(x) to the voxels, a bounding box enclosing the object is firstly established and then the corresponding volume is subdivided in voxels of convenient size.



(a) Mesh 1 (b) Mesh 2

(c) Consensus surface (d) Detail

Figure 5: Example of computation of the consensus surface from 2 meshes.

The bounding volume should be scanned step by step: for each voxel close enough to the surface the distance D(x) from the current voxel center is calculated and assigned to it. Of course, to achieve desired accuracy a dense sampling of the volume has to be used . Since the memory requirements of a volume grid is cubic with respect to the density of the sampling for volumetric modeling, the first thing that gets sacrificed is accuracy.

Indeed, a first problem with a voxel grid representation is that the number of voxels is n^3 where each axis of the volume is discretized into n elements. This affects the achievable accuracy since the dimension should be chosen to be small enough that the grid can fit in memory: it is easy to reach memory limits with less powerful computers. In addition to storage cost, one should remember that for each voxel the signed distance must be computed; thus, the number of computations of the signed distance function D(x) will be cubic as well. Specifically, computation resources are wasted by computing signed distances in parts of the volume that are distant from the surface. On the other hand, the only voxels that need to be examined are those near the surface, a small fraction of the entire volume grid.

Therefore to optimize the procedure in terms of memory requirements and execution time the octree data structure has been employed. Octrees were developed as an efficient way for representing 3D occupancy grids for computer graphics and CAD modeling. Basically, an octree is a hierarchical tree data structure where each leaf represents a volume in 3D space and each of them can have eight childs, what corresponds to divide a given volume into eight octants. This scheme can be repeated, if necessary, to any level of subdivision desired. An example of the octree structure is shown in figure 6.

Octrees can be used to efficiently represent the object's surface since the sampling resolution can be adjusted to the level of detail necessary at each region of the volume. Indeed with octrees it is possible to sample finely near the surface and coarsely away from it (figure 7).

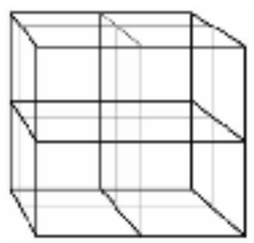


Figure 6: Octree-based subdivision of a volume and resulting octants

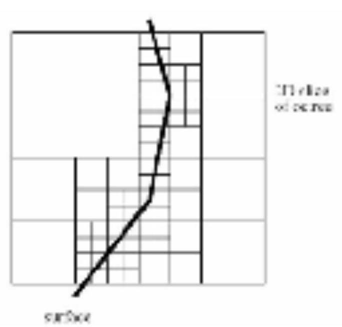
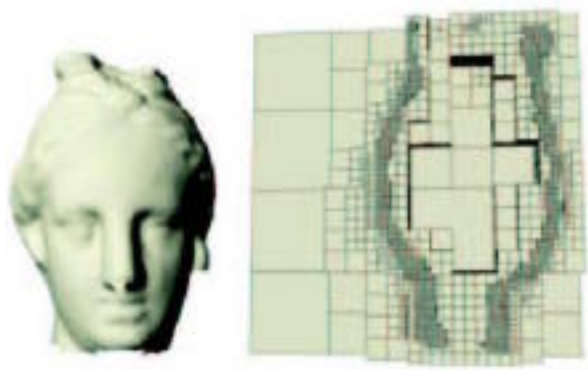


Figure 7: 2D slice of an octree representation of a simple surface.

The octree representation solves both the accuracy and the efficiency problems while keeping the algorithm implementation simple. Instead of iterating over all elements of the voxel grid, a recursive algorithm is applied on an octree that samples more finely in octants only when necessary. Basically each node (i.e. the cube) of the hierarchical tree is assigned a value specifing one out of three possible states: inside, outside or intersecting the object's surface. The tree is recursively visited in such a way that if a node is labeled as intersecting, then each of his eight childs is examinated. This search continues until the maximum resolution allowed for the volume subdivision is reached. The intersection state is determined by evaluating if a node of the mesh of the range view exists inside the voxel. In order to take into account even the case where all the vertices of an intersecting triangle lie outside of the voxel, the check is performed using a cube whose original lateral dimension d is slightly expandend to $d+f$. The value of this parameter is set by comparing the mesh resolution rm with d. If rm< d the voxel size is increased by f = 2□d, while if rm > d, then a value of f = 2□ rm is set to be sure that triangle vertices fall inside the voxel. To interpolate the zero crossings properly, the implicit distance for the voxel containing the surface (the zero crossing) and for all voxels neighboring this voxel are needed. However, since in the octree structure it is not possible to detect a neighbor "leaf" cube, close to the one currently being processed, the distance D(x) is computed from all the eight vertices of the cube instead from the voxel center only. Therefore, the octree-based subdivision of the volume enclosed by the bounding box is examined and the distance function D(x) is computed only for the eight vertices of the voxels belonging to the last resolution level, as they will likely be the closest ones to the surface.

The octree in practice reduces the $O(n^3)$ storage and computation requirement to $O(n^2)$, where n denotes the number of voxel used for the volume subdivision. This is because the surfaces of 3D objects are, in general, 2D manifolds in a 3D space. Figure 8 shows an example of the application of the octree subdivision method to a little statue, while in table 1 the results obtained accordingly to various voxel sizes are reported. It should be noted that if the number of voxel per side is doubled, then the total number of voxel grows by eight times while the number of voxels actually examined in the last resolution level grows by four times only. This demonstrate that the computation requirements reduce to $O(n^2)$ employing the octrees.



(a) (b)

Figure 8: Cross section (b) of the statue (a) showing the octree-based volume scanning.

Table 1: Relationship between visited voxels and voxel resolution

N. of voxels/side	N. of total voxels	N. of examined voxels	Percentage	Fraction
64	262.144	36.873	14.07%	—
128	2.097.152	161.143	7.68%	4,37
256	16.777.216	673.049	4.01%	4,18
512	137.388.096	2.734.867	1,99%	4,06

3. THE MARCHING CUBES ALGORITHM

Once the volume enclosed by the bounding box has been sectioned with the octrees and he distance function D(x) has been sampled at the eight vertices of each octant, a triangulated mesh representation of the object's surface can be easily generated through the marching cubes algorithm, developed by Lorensen and Cline [Lorensen et al., 1987]. Basically, each vertex of an octant will be classified as, "outside" or "inside", by comparing the previously computed value of the function D(x) with a threshold *s*. Denoting with V a vertex and assuming *s = 0*, the rule of this classification becomes:

$$\begin{aligned} &\text{if } D_V(x) > 0 \rightarrow State(V) = 1 \rightarrow \text{"outside"} \\ &\text{if } D_V(x) < 0 \rightarrow State(V) = 0 \rightarrow \text{"inside"} \end{aligned} \tag{2}$$

In this way it is straightforward to code how each triangulated range view intersects a cube. Then, a global mesh is built by placing a triangle vertex on each side of the cube whose two vertices have different states, 1 and 0. This means indeed that a triangle of a range view is intersecting such cube of the octree structure. The positon of the triangle veretx is computed by interpolation of the distance values assigned to the cube vertices of the cube. Afterthat, those new vertices are joined together to form triangles in such a way that side of a cube connecting two vertices having state 1 and state 0 should intersect a triangle. Therefore "inside" and "outside" vertices of the octrees are always separated by a surface (triangle). Though the possible configurations for the combinations of the 2 states for each cube are $2^8 = 256$, taking into account rototranslations and simmetries of a cube the actual number is reduced to 15 different cases, which are partly displayed in figure 9.

The method proposed by Lorensen and Cline presents however some ambiguities when different triangulating options are allowed. For example in the case of a voxel facet having two inside and two outside vertices lying on diagonally opposite sides(figure 10), the triangulation can be ccarried out in different ways. Such ambiguities can be however easily solved for by introducing eight extra cases, as suggested by Shoeb [Shoeb, 1998], where the configurations with inverted vertices are not considered equivalent.

4. TEST AND RESULTS

The method described in previous sections aimed to build a triangulated mesh from a set of registered range views has been applied to a set of 12 scans acquired with an optical laser scanner (figure 11). In order to perform the triangulation with the marching cubes algorithm, a table containing all the 256 possible configurations was set up. Given a voxel and the values of the distance function D(x) assigned to its eight vertices the generation of the triangles is straightforward. The configuration corresponding to the actual values is searched for along the table and the vertices of the new triangles are placed along the side of the cube according to the strategy described in the previous section. For the test, an average scan resolution of 0.2 mm was chosen. Figure 12 shows the results of the application of the consensus surface and of the marching cubes for three different choices of the mesh resolution, i.e. 0.93 mm, 0.45 mm and 0.22 mm respectively in (a), (b) and (c).

5. CONCLUSIONS

In this paper a volumetric method for range data integration has been presented. Through the use of consensus surface, octree representation and marching cubes a set of well aligned 3D views can be successfully integrated in order to build a triangulated mesh that best approximate the actual object's surface, sampled by an optical laser scanner. The method is able to take into account all the input data (points of the range views), while reducing the effect of the noise, tipically due to surface sampling, sensor measurements and registration errors. Results of test performed on a real object (head of a little statue) revealed that the algorithm is robust against noise and quite flexible: by varying a few critical parameters it is possible to adapt it to the input data configuration. For example, setting up the voxel size allows for the choice of the most suited mesh resolution; changing the threshold values for the angle and distance between two neighbor range view points, according to the input data, allows to better discriminate if two points of two different 3D views belong to the same surface portion. Similarly, the value of the weights can be set accordingly with the input data in order to produce a surface without discontinuities or spurious elements, whic h can be present in the original scans.

ACKNOWLEDGMENTS

This work has been accomplished in the context of the National Research Project titled "The Cultural Heritage risk map: survey, georeferencing, monitoring and multiscale modeling". National coordinator Prof. C. Monti, local coordinator Prof. A. Vettore.

REFERENCES

[1] Cline H. E., Lorensen W.E., 1987. Marching cubes: a high resolution 3d surface construction algorithm. Computer Graphics, 21(4):163–168.

[2] Friedman J. H., Bentley J., Finkel R., 1977. An algorithm

for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software, 3(3), pp. 209-226.

[3] Ikeuchi K., Wheeler D., Sato Y., 1998. Consensus surfaces for modeling 3d objects from multiple range images. Proc. of ICCV 9'98, pp. 917–924.

[4] Shoeb, 1998. Improved marching cubes, http://enuxsa.eas.asu.edu/~shoeb/graphics/improved.html.

[5] Soucy M, Laurendeau D., 1992. Multi-resolution surface modelling from multiple range views. Computer Graphics, Proc. of IEEE CVPR'92, pp. 348–0353.

[6] TurkG., Levoy M, 1994. Zippered polygon meshes from range images. Computer Graphics, SIGGRAPH 94 Conference Proceedings, pp. 311–318.

[7] Wheeler M. D., 1996. Automatic Modeling and Localization for Object Recognition, PhD Thesis, Carnagie Mellon University.

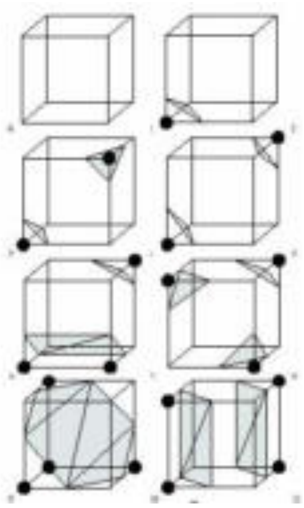
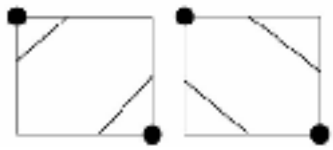
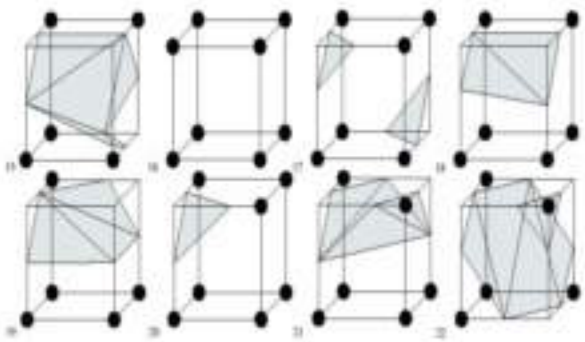


Figure 9: Example of possible triangle/cube configurations



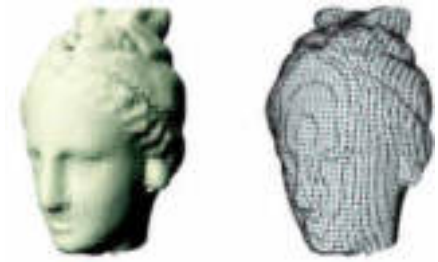
(a)



(b)

Figure 10: Ambigous (a) and extra marching cube configurations (b)

Figure 11: A set of 6 out of the 12 range images employed



(a)

(b)

(c)

Figure12: Examples of meshes generated at different resolutions; 0.93 mm (a), 0.45 mm (b), 0.22 mm (c)