International Journal of Shape Modeling
© World Scientific Publishing Company

A SIX-DEGREES-OF-FREEDOM PLANNING ALGORITHM FOR THE ACQUISITION OF COMPLEX SURFACES

G. IMPOCO*
Dipartimento di Elettrotecnica Elettronica e Informatica (DEEI), Università di Trieste
via A. Valerio 10, I-34127 - Trieste, ITALY.
gimpoco @ units.it

P. CIGNONI, R. SCOPIGNO
Istituto di Scienza e Tecnologie dell'Informazione (ISTI), Consiglio Nazionale delle Ricerche
via Moruzzi 1, I-56125 - Pisa, ITALY.
cignoni | scopigno @ isti.cnr.it

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Communicated by (xxxxxxxxxx)

The models produced by means of the available 3D-scanning technologies are considered accurate enough for most applications. Unfortunately, the acquisition of complex objects is still a demanding process that cannot be performed by non-specialists. Among the open problems, one of the most difficult to grasp is the *planning* of the acquisition session, i.e. choosing a set of positions of the scanner to view the whole surface of the object.

Most of the algorithms proposed in the literature either can handle few degrees-of-freedom (DOFs) or are too burdensome, as they require the optimisation of complex objective functions. Hence, they are in trouble when dealing with complex surfaces. We propose a full 6-DOF pose-planning algorithm that is simple and easy to implement. We do not search for the *next best view* (NBV) to minimise the number of acquisitions, as most previous algorithms do. Rather, we pursue the less ambitious objective of finding a (possibly small) set of views, that guarantee a complete coverage of the surface with a minimum accuracy on the sampled data. Given an incomplete model, unsampled regions are detected and simple patches are built to cover the missing surface. New views are estimated by clustering the normals to unsampled patches.

Keywords: 3D-scanning; sensor planning; acquisition planning; hole filling.

1991 Mathematics Subject Classification: 22E46, 53C35, 57S20

1. Introduction

The quality of the 3D models acquired from real objects has significantly improved in the last few years, thanks to the progress of 3D-scanning technologies. The *accuracy* (50

*This work was carried out while the first author was with the ISTI-CNR and the University of Pisa.

microns or better for triangulation-based scanners), *spatial resolution* (i.e. the mean distance between sampled points, usually in the range of 0.1-0.5 mm), and *sampling speed* (100K-300K samples per second) of modern scanning devices are considered to fulfill the requirements of most applications. In spite of all the progress made, the 3D-scanning technology is not mature for the consumer market. There are still bottlenecks that slow down the whole scanning process. Moreover, the process is not fully automatic. The intervention of expert operators is still required in some key steps, such as the integration of raw data (*registration*) and the *planning* of the acquisition campaign. Planning and optimising the acquisition of 100-200 range maps can be difficult even for expert operators. Furthermore, obtaining a complete coverage of complex surfaces is often a hard task. Among others, people involved in the *Digital Michelangelo Project* [1] reported that the progressive coverage of models comes with an effort which is inversely proportional to the fraction of the sampled surface.

We address the problem of planning the acquisition campaign, i.e. choosing a set of views that guarantees a complete and accurate sampling the surface. This topic has been extensively studied in the literature and it is usually referred to as *next best view* (NBV) selection. The goal is to determine the next *best* location and orientation of the scanner, given the data that has already been acquired. The optimality criterion is a key choice to determine the NBV. The *minimal number* of views has often been used in previous approaches. In our opinion, the number of scans is a non-critical parameter due to the acquisition speed of current scanners and the quality of post-processing software [2]. Moreover, data redundancy can be used to reduce the effect of noise and sampling inaccuracies. Therefore, the *optimality* of a scanning set is mainly related to the *completeness* (percentage of surface sampled), and the mean *glancing angle*, i.e. the angle between the scanning direction and the normal to the surface in a given point. The glancing angle is directly related to the quality of sampled data, since most scanners give high quality samples only when the acquisition direction is nearly parallel to the surface normal.

When planning the acquisition of objects, little is known about their shape. Many previous approaches make plans without any knowledge on the surface. They cannot make any assumption on the geometry of the scene, save a volume bounding the object. A more accurate understanding of the shape would be of great help. We believe that designing a preliminary acquisition plan is very easy even for modestly-skilled operators. The resulting model is usually incomplete. An automatic planning algorithm can be designed that takes advantage of, and refines this preliminary acquisition. Our approach improves this simple initial planning by detecting the surface regions that have not been covered by the initial scans. For each of these potential holes, we estimate a set of surface normal vectors using the supposed orientation of the faces of the hole. A set of views is automatically generated on the basis of a quantisation of the space of viewing directions. Experimental data show that the coverage factor is significantly improved.

This paper is an extension of a previous work [3]. The main new contribution is a fast and effective method to account for visibility during planning. Visibility information is used to discard obstructed views and to extend our algorithm to two-line-of-sight scanners, such as triangulation devices. A method to avoid collisions between the scanner and the object is also suggested. Finally, experimental data is shown to support our algorithm.

2. Related Work

Sensor planning can be defined as the problem of determining a sequence of sensing operations to view a given scene. Each sensing operation is a set of *sensor parameters* such as *viewing pose* (i.e. position and orientation of the sensor), camera focus and field of view, and *environment parameters* such as controlled lighting conditions (e.g. in BRDF acquisition). A number of constraints can be used to optimise the acquisition. A common target criterion is the minimisation of the number of sensing operations that allow the scene to be completely covered. Most of the techniques in literature do not take into account environment parameters and assume fixed most sensor parameters. The focus is mainly on finding viewing poses, thus reducing the search space to six dimensions (three for the position of the sensor and three for its orientation). We will refer to this search space as *pose space*.

Sensor planning has been addressed mainly by the computer vision community as the problem of determining the minimum set of viewing directions covering the surface of a given object. Following the approach of Maver and Bajcsy [4], planning strategies can be classified according to how much a-priori knowledge about the scene is available. If a complete model of the scene is known in advance, the plan can be computed off-line using the reference model. On the other side, if no geometrical information is available prior to the acquisition phase, the best we can do is to build the plan on-line, step by step. This approach, known as the *next best view* (NBV) problem, is more popular in the computer graphics community in the context of surface acquisition. A third class of algorithms lies midway between the first two, i.e. only partial information about the scene is accessible.

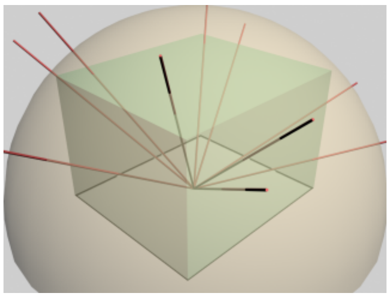


Fig. 1. A spherical bounding volume. The views (shown as lines) are constrained by the shape of the volume.

When planning the acquisition of a 3D shape no prior knowledge is given about the surface to be acquired. Sometimes the object is assumed to be bounded by a (often spherical or cylindrical) volume (see Figure 1). The scanning system is often constrained to lie onto the surface of the bounding volume, thus reducing the number of degrees-of-freedom (DOFs) of the viewing pose from six to two. This may cause some viewable surface areas to become not measurable by the system (see Figure 2). The same may happen discretising the set of viewpoints.

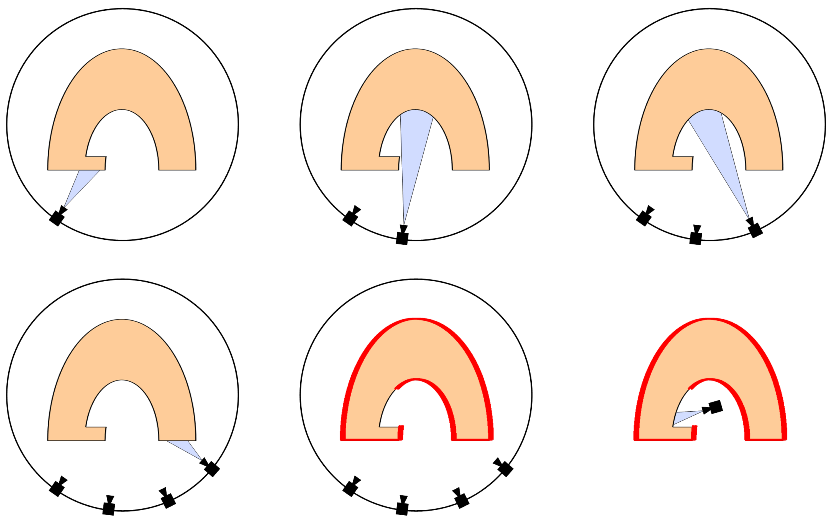


Fig. 2. Planning and acquisition of an object using a spherical bounding volume. The surface acquired (lower-middle image) is not complete, even if the acquisition system can sense it. This is due to 2D nature of the bounding surface. Conversely, if the planning is unconstrained (lower-right image) the final model is more complete.

A common situation in which a model of the scene is known a-priori is the case of industrial inspection of mechanical parts, where a CAD reference model is used to verify the compliance of manufactured objects. Tarbox and Gottschlich's algorithm [5] generates a plan based on the reference model and then verifies it using a model of the data acquisition system. Despite this off-line verification phase, when the plan is executed it can still be incomplete due to imperfections in the model of the acquisition system. A further on-line planning phase is then performed to fill the holes. Three different planning algorithms are proposed. The first one constructs the plan incrementally, choosing the sensing operation that is able to sense the largest portion of unseen surface area. A quality constraint must be satisfied in order to consider a surface point measurable from a given viewpoint. Namely, if the angle between the normal to the surface in the given point and the viewing direction (*glancing angle*) exceeds a given threshold the point is marked as non measurable from the current viewpoint. A second procedure weights each point with respect to its glancing angle in order to balance between a short plan and a high quality reconstructed model. A third algorithm employs a simulated annealing scheme to perform a local search on the space of viewpoints.

One of the first algorithms in the class of approaches with no a-priori knowledge was proposed by Connolly [6]. He partitions a spherical bounding volume using an octree representation. All octree nodes have a label initialised as *unseen*. When a portion of the surface is scanned, the nodes containing the sampled surface are labelled as *seen* and the regions in the portion of the conoid between the scanner and the surface are labelled as *empty*. Two algorithms are presented to compute the NBV. In the *planetarium* algorithm the bounding sphere surface is sampled in a number of candidate viewpoints. Visibility of the surface for each candidate position is evaluated and the NBV is selected as the viewpoint maximising the amount of unseen nodes. This is a burdensome algorithm because computing the visibility information requires ray casting through each octree node. The *normal* algorithm counts the area shared by the faces of nodes which separate empty and unseen regions. Consequently, only six directions are possible. The NBV is the direction maximising the shared area. Although this algorithm is computationally much cheaper with respect to planetarium, it is a fairly naive approach.

Maver and Bajcsy [4] proposed a planning algorithm tailored to a light stripe range sensor, constrained to translations and limited rotations in a plane above the object. Occluded regions are represented as polygons. Viewpoint visibility constraints are computed from the polygon boundaries. This algorithm cannot be easily used with other scanning configurations.

Whaite and Ferrie [7] developed a model-based approach in which a parametric model (superquadric) is fitted to the currently sensed data. At each step the model can be refined minimising the uncertainty in the model itself. Since the uncertainty is strictly tied to how well the sensed data fits the current model, the next operation is to scan the region where data fits the model worst. The main limit of this algorithm is the inability of simple parametric models to accurately represent surface detail.

A general framework for viewpoint planning is presented by Pito [8]. His algorithm assumes that the scanning volume is enclosed by a surface. The enclosing is parameterised using a representation called *positional space*, consisting of two bi-dimensional scalar fields. This representation allows to avoid the heavy memory requirements of an octree structure. The positional space encodes the visibility information for each candidate viewpoint. The viewpoint which maximises the unseen volume is chosen as the NBV. Pito's framework can account for every possible bounding volume. Nonetheless, the shape of the bounding volume must be tailored to every object, since the scanner is constrained to move on its boundary. As a consequence, in practice it is very difficult to obtain a full 6-DOF sensor planning algorithm in this framework.

All the approaches described above assume that the scanner never enters a given bounding volume to avoid collisions between the scanner and the object. Papadopoulos-Orfanos and Schmitt [9] presented an approach which incorporates a path planning algorithm for collision avoidance. They use a laser stripe scanner mounted on a robot with three translational DOFs. The acquired data is used to guide both the sensor planning and path planning algorithms. They also exploit the geometry of the laser stripe scanner to obtain a more efficient space carving strategy, based on direct and indirect shadowing (i.e. occlusions from the light source and to the camera). The main drawback of this approach is that the scanning system is not allowed to rotate around the optical centre. This constrains severely the shape of scannable objects.

Reed and Allen [10] encoded three planning constraints as operations on sets. They consider the workspace volume as a set of points. The constraints are enforced onto the workspace by applying set operators. Viewpoints are regarded as constraints. Since set operations may be computationally expensive, this approach is inefficient.

Scott et alii [11] worked out a multi-stage approach in which a coarse model acquired in a stage is used to guide scene exploration for the fine modelling of the next stage. An

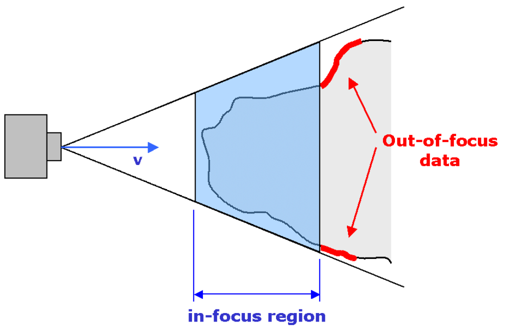


Fig. 3. A candidate NBV. Some planning algorithms do not exclude out-of-focus surface points when computing measurability from a viewing direction.

a-priori base model is needed at the first stage. It can be as simple as a bounding volume. Virtually any number of stages can be used but in practice a choice of two should suffice. The algorithm partitions the rough model into cavities, holes and planar patches and for each patch a set of candidate viewpoints is generated. A user-defined *measurability* function is computed for each patch of the rough model. A simple approximated set-covering algorithm is employed to select the views that together give the best sampling of the whole surface. This algorithm does not constrain the viewpoint space, hence full 6-DOF plans can be generated if a suitable discretisation of the viewpoint space is done. However, finding a good discretisation is not straightforward, since we cannot assume that the scanning system can sense all surface data in a given direction. Namely, a slope along the direction of optical axis may be so steep that some points would not be imaged at all. Hence, the measurability of some points from a given direction may be misrepresented. Figure 3 depicts an example in which this situation occurs. The authors do not suggest any method to cope with this problem.

Model-based object recognition and localisation is an example of planning problem with partial a-priori knowledge. Given a model of the object's shape, its pose must be determined. Here an approximation of the shape is known in advance, but not the object's pose. Most approaches in this class follow a common scheme [12,13]. A search is performed in the space of poses employing the *hypothesis-and-verify* paradigm. Hypotheses are formed regarding the identities and poses of the objects in the scene. Then, hypotheses are assessed in compliance to some metrics. New sensing operations are performed accordingly until a halting condition is met.

A good survey of view planning algorithms can be found in [14]. For a recent survey see [15] which also provides an alternative and effective classification.

3. Assumptions

Complex objects may have many protrusions and partially occluded concavities. In order to fulfil the *scanning requirement* (i.e. all the 'viewable' surface should be scanned [8]), a planning algorithm should allow the scanner to be positioned and oriented in every possible way. Planning algorithms that constrain the scanner to move on a predetermined surface (e.g. a sphere centred on the centre of mass of the object), may have forbidden positions in the sixth-order pose space (see Figure 2) that are dependent on the algorithm itself, rather than on the capabilities of the scanning system or on the features of the objects. That is why they are doomed to fail in meeting the scanning requirement.
Our algorithm, being guided by *occluding patches* (see Figure 4), has no restrictions or forbidden regions and, in principle, can sample all the surface viewable by the scanning hardware. Actually, a quantisation of the pose space is carried out. However, the surface

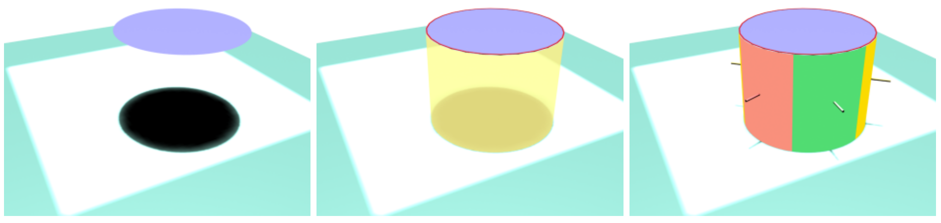


Fig. 4. *Occluding Surfaces* - A scan is taken from a direction orthogonal to the surface. In the left image an *occluding patch* casts a shadow onto the back surface. The corresponding *occluding contour* and *occluding surface* are shown in the middle. On the right the resulting clusters are displayed together with the associated normals.

patches that cannot be viewed by the sensor become smaller as the quantisation step becomes finer, since the pose space is quantised uniformly. In the limit, no missing patches will be left.
Despite many of the methods proposed in the literature keep an eye on quality and completeness of the sampled model, research in this field has put the stress mainly on the generation of short plans. As a result, surface coverage is not always satisfactory. Moreover, many proposed solutions are complex and memory-consuming. Conversely, our approach tries to maximise quality and coverage, rather than minimising the length of plans.

Like many previous algorithms, we are interested in planning the position and orientation of the scanner. Hence, we assume that the other parameters (e.g. camera focus, environment lighting, and so on) are fixed at the beginning of the scanning session. We call *viewpoint* the position in space of the center of projection of the camera. A *view*, or *viewing pose*, can be defined as a couple (C, d), where C is a viewpoint and d is the orientation of the camera.

Henceforth, we use the expressions *occluding contour* and *occluding surface* to indicate, respectively, the border of an occluding patch, and the surface that spans between an occluding contour and its projection onto the back surface (see Figure 4).

4. A Hole-Driven Planning Algorithm

Many previous approaches try to build very accurate plans in a single iteration, using complex optimisation functions or data structures. Anyway, complex shapes escape even the Wise, despite the efforts we make. This is mainly due to our limited knowledge on missing patches. When human operators plan an acquisition they have a powerful means of acquisition that automatic planners lack: the human visual system. An automatic planner can only guess the shape of missing parts.

We believe that it is not worth doing complex plans at the expenses of computational resources. Better results can be obtained using a simple iterative approach, where planning and acquisition are interleaved. Moreover, acquiring the shots that have been suggested by the planner gives a feedback to the planner itself. This resembles what humans do when they 'acquire' a representation of the world. They just look around to view parts of objects that were occluded in previous views. Then, they integrate the observed patches to their representation.

The proposed approach falls in the class of algorithms with no a-priori knowledge. We use a multi-stage refinement of the surface. A basic shape is generated using a simple sweep. Most of the basic scanning tasks can be solved by choosing one of two different simple plans.

(1) **X-Y sweep**. This is the typical setup used to scan a basrelief or a car body section. The characteristics of the scanner are assumed to be known. Given the extent of the object in a given projection plane, we can easily partition the scanning space into $n \times m$ range maps. Data is acquired by a regular sweeping of the scanner along the two directions on the projection plane. Scanner displacement depends on the inter-scan overlapping factor (usually 0.7-0.8 the scan height and width).
(2) **Cylindrical sweep**. The common approach to scan an all-round object consists in the acquisition of a set of cylindrical scans. Either the scanner is rotated around the object or a rotating platform is employed, which allows to move the object while the scanner is steady. The number of shots in a complete rotation depends on the mean diameter of the object and the inter-scan overlapping factor (usually 20-30%) and can be easily computed. The vertical sweep factor and the number of cylindrical acquisitions again depend on the inter-scan overlapping factor and the height of the object.

Both these methods are naive but require few parameters and can be easily set up. Obviously, they rarely come up with a satisfying coverage of the surface, since they do not take into account surface features to guide planning. Self-obstructing regions on the surface can produce *unseen regions* (see Figure 5). Nonetheless they give a rather well distributed set of views, that is a good starting point for further improvement.

At each step of the refinement phase, we build a plan to sample unseen surface by find-

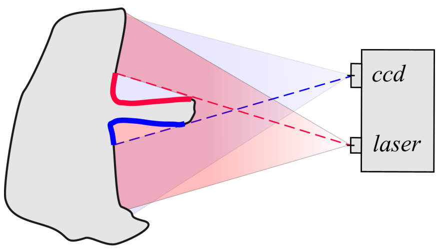


Fig. 5. An example of unseen regions originated from the presence of a self-obstructing component of the shape. Thick lines indicate unsampled surface sections.

ing a small number of views covering the occluding surface. A minimum-quality constraint is enforced setting the maximum glancing angle.

Figure 6 shows the steps of our algorithm. First, we locate all unsampled regions of the

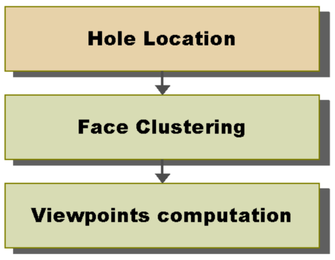


Fig. 6. Pipeline of the proposed algorithm. The *Hole Location* step differs between local and global solutions.

current model. The unsampled surface is represented as a triangle mesh. Then, the faces of this mesh are clustered with respect to their normals. The idea is that a cluster should represent a portion of an occluding surface that can be acquired in a single shot. Finally, a viewing pose is computed for each cluster, taking into account the contribution of all the normals to faces in the cluster. New surface patches are acquired using the new views. If the model is still incomplete, the algorithm in Figure 6 can be iterated. We found that one iteration often covers almost all the missing data. No more than two iterations are needed in most cases.

4.1. *Locating Holes in the Model*

The core of our approach is the *hole location* phase, since its performance is critical for the generation of good plans. Hole location relies on the following observation. When a range map is acquired, the data occluded to the sensor lies inside a conoid bounded by occluding contours. Thus, the occluding surface separates unseen volume from volume known to be empty. Since we know nothing about the unseen volume, save its boundary, our idea is to exploit this minimal knowledge to compute occluding surfaces and use them as a guide for planning (see Figure 4). We propose two different methods for locating holes. One is very simple and fast but can find only local holes. The other one can locate holes in the whole surface and is more reliable but is also more burdensome.

In the **local** method, hole location is view-dependent as it operates in $2\frac{1}{2}$D. Namely, the

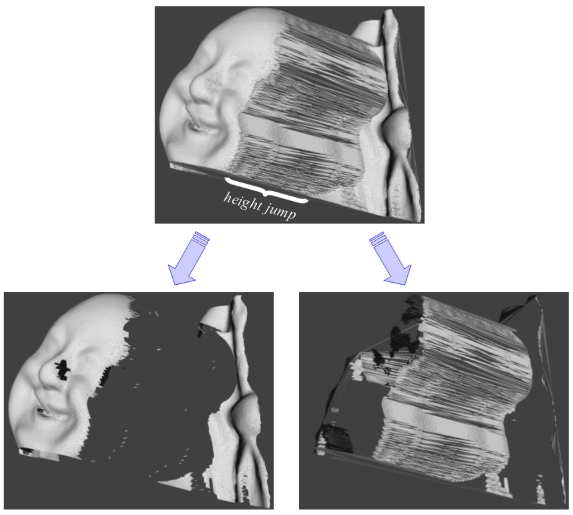


Fig. 7. A *height jump* and the corresponding *hole map*. A range map is shown in wireframe together with good (lower-left image) and (lower-right image) bad faces. Bad faces are the hole map of the current scan.

boundaries of occluding surfaces are detected by searching for strong *height jumps* in the range map (see Figure 7). The range map is triangulated and triangles associated to high-quality sampled regions are filtered out, i.e. all but very skewed triangles are removed. The resulting mesh is the dual of a 'good-looking' surface patch. In this way we obtain a sort of *hole map*, consisting of all the faces that are removed from the original mesh patch. Notice that the faces of this hole map are nearly orthogonal to the viewing direction.

The hole detection phase is preceded by a smoothing operation on the raw data, regarded as a height field. This step is fundamental in order prevent overly scattered normals in the hole surface. Since the hole surface is made up of skewed triangles the distribution of triangle normals is usually very noisy. Smoothing the raw data also attenuates noise in boundary data, which could cause artifacts in the hole surface. Smoothing is further discussed in Section 5.

The **global** method is based on volumetric diffusion [16]. Volumetric diffusion allows to close holes by expanding the distance field on unsampled regions. It also exhibits the nice property that the expanded surface is smooth. This property is useful when clustering normals. We take care of computing a *confidence* value during the volumetric diffusion. This allows to discriminate the surface parcels that are originated by field expansion from those that originate from real data. Figure 15(a) shows a model before fusion (image on the left) and the expanded mesh reconstructed at low resolution (image on the right). The colour on the reconstructed surface maps the confidence value using a colour ramp: the red indicates surface portions that were acquired with a low confidence (i.e. are badly sampled or are associated to holes), while original data is painted in blue [a]. The unsampled areas of the mesh can be detected by defining a threshold on the confidence value. Figure 15(b) shows the hole map.

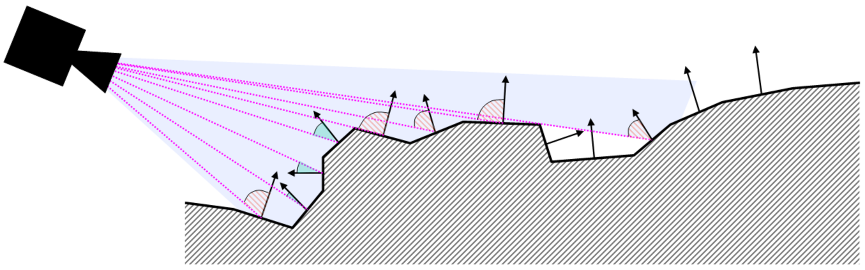
It is worth noting that we do not need a highly accurate model at this stage. A mesh reconstructed approximately suffices for the detection of un-sampled regions. Hence, we reconstruct the volume with a low sampling rate (i.e. with a voxel width much larger than the inter-sampling distance of range maps). Working with a low resolution voxel set allows a more efficient execution of the volumetric diffusion of the distance field.

4.2. *Clustering the Faces of Holes*

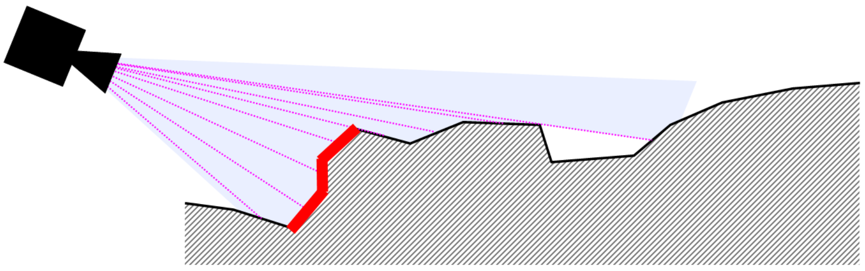
Once holes have been located and a mesh representation has been computed for them, mesh faces are grouped in clusters with respect to their normals. Each cluster represents all the faces that are measurable from a single direction. We define measurability as a function of *glancing angle*. Namely, a face is measurable if the direction of its normal is nearly parallel to the viewing direction. Figure 8(a) depicts the cross-section of a surface and the camera viewing cone. A ray is cast form the center of projection of the camera to each face of the surface. The figure shows the angles between the normals to mesh faces and the corresponding rays. The cluster associated to the current view is given by the faces whose angles do not exceed a given threshold (Figure 8(b)). A representative direction is associated to every cluster. A face belongs to a cluster if its distance from the representative direction is within a given threshold. The distance measure we employ is simply the glancing angle. The threshold value depends on a quality function of the scanning system, which is usually directly related to the glancing angle. Notice that here we assume that faces pointing in the same direction can be acquired in a single shot, i.e. they are inside the viewing cone of the scanner optics (Figure 3). This condition is met if we assume that the whole object lies inside the scanning volume. Anyway, it can be avoided if a constraint on maximum Euclidean distance is embodied in the distance measure used for clustering. A method to account for this will be presented in Section 4.3.

The clustering algorithm we employed is in the spirit of Octree Quantisation [17]. This is a simple method used for colour quantisation in palette-based colour images. Suppose you are given a colour table that can contain at most k entries. When a new colour is added to the table, if there is a free slot in the table then you are done. Otherwise you have to make room for the new entry. This can be done by merging some close neighbours into one cluster. A common colour is assigned to the cluster. An octree is used to represent the RGB space. Colour components are the coordinates within the octree. Exact colours are represented as leafs of maximum depth, while intermediate nodes represent clusters. The

[a]Due to publishing restrictions images are printed in grey levels. A colour PDF file can be downloaded from `http://vcg.isti.cnr.it/publications/papers/planning_IJSM05.pdf`



(a) A ray is cast form the center of projection of the camera to each face of the surface. The angles between the normals to the faces of the mesh and the corresponding rays are shown. The angles that exceed a given threshold are filled with a striped pattern.



(b) The faces that can be acquired at high quality in the current view.

Fig. 8. Cross-section of a surface and the camera viewing cone.

deeper the level of a node, the fewer the colours it represents. Every time the number of leaves exceeds k, some leaves are merged. In order to generate an equally distributed partition, the leaves of deepest level which represent the fewest pixels are chosen. Once every colour has been added to the octree, the colour table is given by the leaves. The mapping from colour indices to table entries can be done simply by traversing the octree structure; when a leaf is found its colour is returned. Since unit normals have only two degrees of freedom, we implemented a sort of *quadtree quantisation* in 2D, mapping azimuth and elevation angles to colours.

We chose this solution in place of more widely used clustering methods, such as k-means [18,19], because of its simplicity. Even if k-means might give better results, we do not need to be very accurate since our knowledge of the missing data (i.e. the occluding surfaces) is imprecise. Moreover, the gain in accuracy would be at the expenses of running-time. Different face clustering approaches have been proposed in the computer graphics community [20,21]. However, even if they are simple and fast as well as Octree Quantisation, they are too tied to mesh topology and can give incorrect results for our application. Figure 9 shows the comparison between two different clustering approaches.

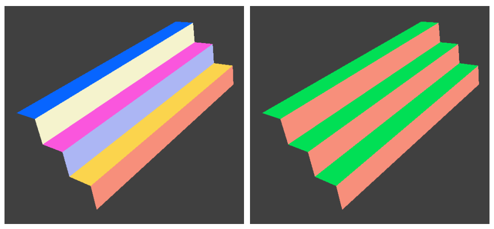


Fig. 9. Face clustering methods: *topology-driven* methods (on the left) generate a cluster for each face, while *octree quantisation* correctly gives two clusters (clusters are shown in different colours).

One is driven by the topology of the model, while the other takes into account only the normal to faces. The former stops adding new faces to the current cluster when it encounters a discontinuity. Hence, clusters are bounded by discontinuities of the surface. Conversely, the latter scheme allows the clusters to span over discontinuities. This behaviour is prefer-

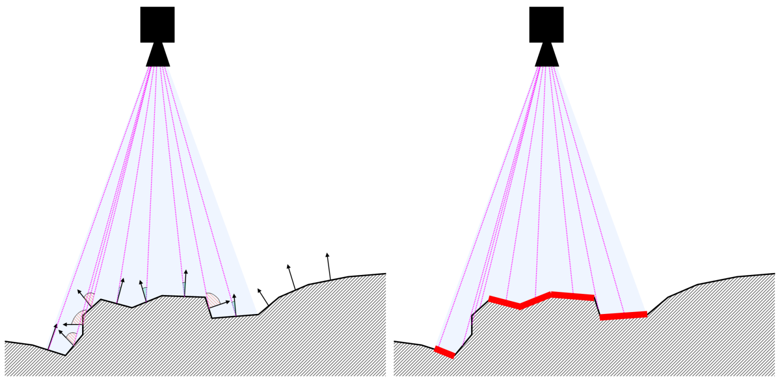


Fig. 10. Cross-section of a surface and the camera viewing cone. A cluster can contain non-contiguous faces (right image).

able for our application, as shown in Figure 10. An iterative clustering algorithms has been recently proposed by Cohen-Steiner et al. [22]. They choose some initial seed mesh faces, and grow clusters around seeds by adding faces in the boundary of current regions, until all faces are processed. Faces are added using a priority related to the distortion due to approximation. The partition is used to compute best-fitting planes, called *proxies*. In turn, proxies are used as seeds for the next iteration. The performance of this method is pretty good, and it is shown to converge in few iterations. However, it is rather slow. Moreover, the type of clustering produced is very different from what we need for our application in two respects. First, it tends to produce elongated polygons, since its objective is a clustering that closely resembles what human beings expect. Second, clusters are computed by growing regions around mesh faces. Hence, the algorithm produces only one connected component, while our clusters can contain more components (Figure 10).

In order to adapt this algorithm to our purposes, we have slightly modified the termination condition and the criterion to choose the next merging. Namely, we choose the nodes to be merged taking into account not only the number of normals represented (encoded as colours), but also the glancing angle with respect to the representative viewing direction. Since we are interested in finding the minimum number of clusters that cover the whole normal distribution, the number of leaves k must be as little as possible. The merging is thus stopped when no more merging can be carried out without violating a constraint on the maximum glancing angle.

Notice that this merging criterion assigns the same weight to all the faces. Weighting the faces with respect to their area could result in a better performance.

4.3. *Computing the New Views*

Once we have clustered face normals, the directions associated to the clusters constitute a small set of viewing directions covering the holes in the current model. To compute the viewing positions, we take the centroid of the vertices belonging to each cluster and displace it along the viewing direction. The displacement is determined in order to allow every face of the cluster to lie inside the imaging cone of the scanner (Figure 3).

Now that a small number of viewpoints has been determined, their quality can be ranked with respect to the area of unknown surface viewed. This is done by summing up the area of the faces in each cluster. The viewpoint associated with a given cluster gets a *rank* proportional to its area. A constraint on minimum area is enforced to discard low-ranked views. Even though some views are discarded using this method, some of the new scans might have wide overlapping regions. In order to avoid sampling the same surface again and again, we weight the rank of a view with respect to its neighbours. Namely, we sort the views with respect to their area and compute the angle θ_{ij} between the view v_i and the views v_j, $j = 1 \dots i-1$. The rank of the view v_i is $r(v_i) = max_j[w(\theta_{ij}) \cdot A_i]$, where $w(.)$ is a weighting function and A_i is the area associated to v_i, expressed as the ratio between the area of the cluster and the area of the current model. In our implementation, we set $w(\theta) = \theta^\gamma$, with $\gamma = 1.5$. The views whose rank is below a fixed threshold are discarded.

A method to get rid of the assumption that the object must lie inside the viewing cone of the scanner is as follows. For each vertex in the cluster the distance from the optical centre can be computed. A histogram of the distances is then built. By examining the distribution of distances the cluster is split in as many sub-clusters as needed to scan each sub-cluster in a single shot. This method is tricky to implement and may produce some unnecessary viewpoints due to its locality.

Better results can be obtained by integrating a constraint on maximum Euclidean distance in the distance measure used for clustering. We do this simply by extending the Octree Quantisation algorithm in five dimensions (two for orientation, three for position). Namely, when the algorithm tries to merge neighbouring nodes it must check not only their orientation but also their linear distance along the candidate viewing direction. This can be

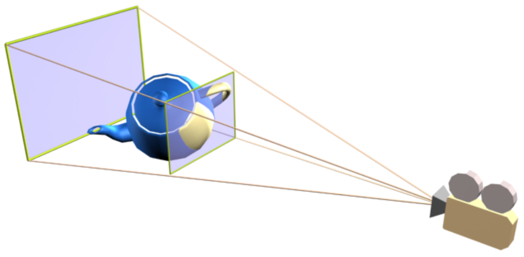


Fig. 11. Clusters are constrained to lie between the near and far planes of the sensor frustum.

thought as constraining the clusters to lie in a volume bounded by two planes orthogonal to the viewing direction (Figure 11), that is exactly what we want to do.

4.4. *Visibility and Potential Collisions*

The approach we have presented so far does not take into account any visibility issue. Namely, when clusters are computed we do not know whether they are visible or not from the associated viewing direction. This is because we do not check for visibility while clustering. Computing visibility involves casting rays along the line-of-sight of the scanner. In order to avoid burdensome computations, we use the graphics hardware as in [23]. A set of candidate viewing directions is computed as described above. For each direction, we place the camera accordingly and render the scene with flat shading. Different colours are used to encode the background, and the outer and inner side of the surface. Moreover, the outer side of each cluster has a unique colour. In order to evaluate the visibility of a candidate pose, we search the rendered image for pixels of the colour associated to the cluster. If their number is not close enough to what we expect, we discard the pose.

This method can check visibility along one line-of-sight. However, the most widely used scanning systems are based on triangulation. Hence, visibility must be computed along two directions. Our basic algorithm produces only 5-DOF plans. There is another

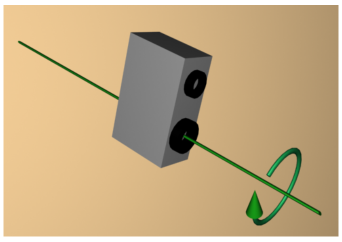


Fig. 12. Our basic algorithm allows the scanner to rotate around the viewing direction. Hence, it generates 5-DOF plans. 6-DOF plans can be generated computing visibility from both the laser beam and the sensor (see the text).

DOF left, i.e. the sensor is allowed to rotate around the viewing direction (see Figure 12.

This aspect is of no importance using one line-of-sight scanners [24]. However, it can be a limitation for scanners with two lines-of-sight (see Figure 5). In general, different orientations of the scanner around the imaging axis give different results. Figure 13 shows two

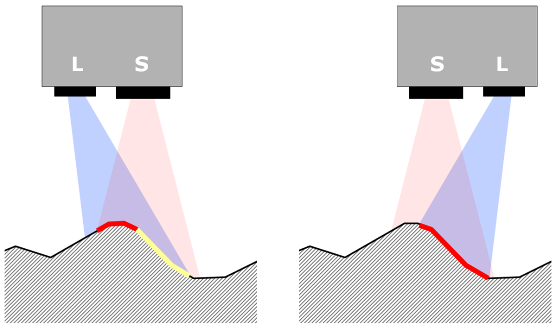


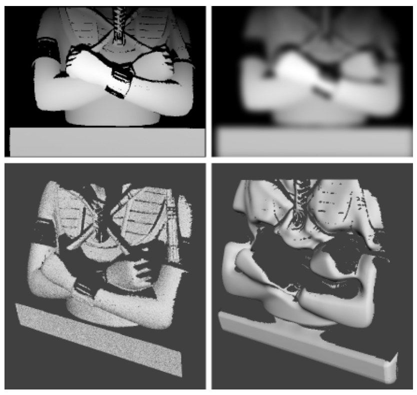
Fig. 13. Two possible orientations of the scanner around the optical axis of the sensor, S. On the left, a part of the sampled surface is almost parallel to the laser beam, L. The data acquired is thus extremely noisy and must be discarded. This does not happen placing the scanner as in the right picture. Hence, the right orientation captures a greater extent of the target surface.

possible orientations. The line-of-sight of the laser is occluded on the left, while it is free on the right. In order to cope with two lines-of-sight, we compute visibility in two steps. First, we compute visibility for the sensor as above. Then, a number of orientations around the scanning direction are checked for visibility. The orientation that gives the best visibility (i.e. the greatest number of pixels) of the current cluster is chosen. This extension allows to obtain full 6-DOF plans with little more effort. The time overhead is negligible, since few renderings are required for each candidate pose.

Avoiding collisions between the scanning hardware and the object is an important issue for an automatic acquisition system. In order to check for potential collisions, we need a tight volume enclosing the object, e.g. the visual hull of the object. If the initial model has been acquired using one of the two sweeps described in Section 4, no protrusions are missing. Hence, the model itself can be used in place of the visual hull. Moreover, we can represent the scanner using a bounding volume. Thus, we know the shape of the two objects which can potentially collide (the scanner and the object) and their relative position. Collisions can be detected using any of the standard algorithms in the literature [25].

5. Experimental Results

We have tested our algorithm both in synthetic and real environments. In our synthetic testbed, we simulate the acquisition of range scans rendering a reference model from the selected views. The new scans are simply the rendered depth buffers. In both the real and synthetic setups, the method proposed is able to detect suitable viewpoints, increasing significantly the sampling ratio of the surface. Figure 14 shows some snapshots of each step of the local algorithm. Runs of the global algorithm are depicted in Figure 15 and


(a) Range data and corresponding meshes. To the right the range data are smoothed using a gaussian filter with $\sigma = 15$. Jump edges are highly attenuated in the smoothed mesh.

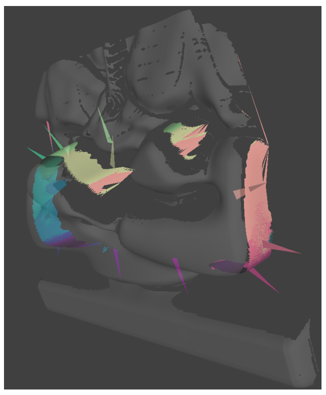
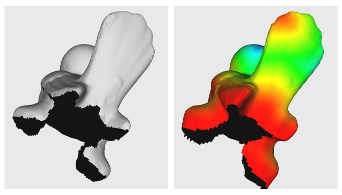

(b) Hole maps: different colours refer to different representative normals (i.e. viewpoints). A spike is shown for each viewpoint computed.

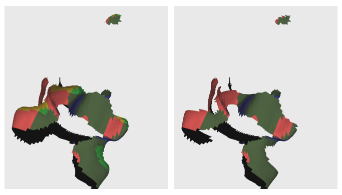
Fig. 14. Some snapshots showing each step of the local method in action on real data.

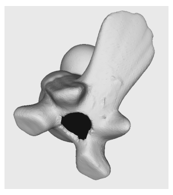
Figure 17.

As can be observed from Figure 14, some gaps were missed by the local algorithm. Some holes were not detected due to the view-dependent nature of the local hole location method (i.e. only hole surfaces nearly orthogonal to the viewing direction can be detected). Moreover, as discussed in Section 4, strong height jumps in the range map are associated to hole boundaries. Smoothing range data attenuates height jumps. In order to avoid to smooth out jump edges we tried to employ an edge-preserving smoothing filter [26]. However, it did not work well since preserving jump edges was obtained at the expenses of an unsatisfactory removal of noise in proximity of those edges, that is exactly what we are interested in filtering out. Hence the only chance to remove noise around edges, still being able to detect holes correctly, is to adjust the threshold used to detect badly-shaped triangles with respect to the smoothing parameters. In conclusion, the local hole detection needs an accurate steering of the user and is not easily executable in an unattended fashion.

This problem is not present in the global approach, as can be observed from Figure 15. Thus, it is a better choice to design a semi-automatic scanning system. On the other hand, this method is slower and more memory-demanding. The burdensome component is the diffusion of the volumetric distance field. However, since we can run it at a reduced resolution the overall timings are affordable on modern PCs. The examples reported in Figure 15 and Figure 17 require few seconds. Bigger models (e.g. the *Angel* model in Table 1) might require up to 15 minutes, which is a fairly affordable time in the framework of a complex acquisition session. The details of timings are reported in Table 1. For the results shown we used a P4 3GHz notebook, 512Mb RAM, with graphics board. Table 2 shows the number


(a) Range maps acquired using a turntable and the expanded mesh. The colour on the reconstructed surface maps the confidence value using a color ramp, where low and high confidence is mapped to red and blue, respectively.


(b) The *hole map* and the directions resulting from clustering the normals of the hole-map faces. Each colour encodes a cluster. On the right only the selected views are shown.


(c) The final model obtained after the corresponding range maps have been acquired and merged into the model.

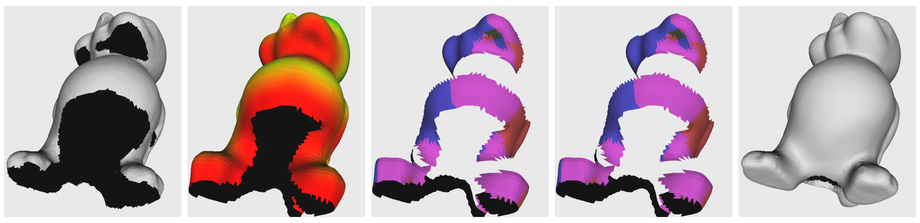
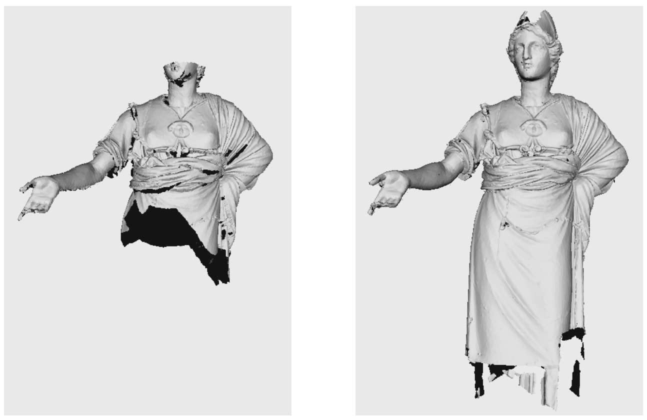

(d) Snapshots of the *bird* model shown from a different viewpoint. A large hole has been detected and split into a number of distinct clusters.

Fig. 15. Some snapshots of the *bird* model showing each step of the global method. This model was acquired by means of a (simulated) rotating platform.

of iterations and scans required. The algorithm requires few iterations, provided that the initial model is acquired as described in Section 4. Conversely, the number of iterations and time required might grow large if this assumption is violated, i.e. only a part of the object is sampled (see Figure 16(a)). Our planning algorithm is designed to sample missing surface patches, given a surface that covers all the extent of the object. Basically, we assume that no protrusions of the object are missing from the initial model. This is exactly what happens in Figure 16(a). Anyway, our algorithm is still able to sample large new areas, as Figure 16(b) demonstrates. Table 2 shows that the coverage of the *Minerva* model is significantly improved after planning.

In our test we set the voxel size to 2 mm, and the maximum glancing angle to 120 degrees. In order to cut away the views that potentially give little improvement, we retain only the clusters whose rank is above 0.05 for the first step, and above 0.025 for further iterations. Finally, we mark as *bad* those mesh faces whose normalised confidence falls below 0.4.

Evaluating the performance of our method with respect to previous approaches is hard, since the models used are often unavailable. Moreover, timings and memory requirements are rarely discussed. We believe that our approach can run faster than many previous approaches since, for example, we do not have to fit parametric models to fairly complex meshes [7] or to compute polygonal regions as in [4]. Moreover, our method is more memory-



(a) The *Minerva* model before planning. Only a part of the object has been sampled, thus violating the assumption that the initial model is acquired using one of the two sweeping methods of Section 4.

(b) The *Minerva* model after planning. The model is highly incomplete. Still, a large area of the object has been added due to planning.

Fig. 16. The *Minerva* model. The model after planning is incomplete since our algorithm is designed

efficient than other approaches as we encode the un-sampled surface using just a set of normal vectors, rather than using a volumetric model [6] or any other representation involving the storage of dense matrices [8,11]. On the other hand, we use volumetric diffusion to detect holes. This has proven to be the bottleneck of our algorithm, both in terms of computational speed and memory requirements. However, we discovered that increasing the voxel size in the early iterations does not affect the performance of our algorithm, while significantly reducing the memory burden. Hence, using a variable voxel size allows to keep low the computational requirements without sacrificing the quality of the final model.

6. Concluding Remarks

We have presented an algorithm that exploits the properties of what we called *occluding surface* to make full 6-DOF sensor plans. Since a discretisation of the viewpoint space must be carried out, a good sampling has to be found in order to fulfil the scanning requirement [8]. Constraining a-priori the viewpoint space is not a good choice, since it does not take into account the features of the surface and can fail to cover complex shapes. This can be easily managed if some a-priori knowledge of the shape of the object is available, but it is a fairly difficult problem if only incomplete or no information at all is given.

Our approach tries to guess good sensor poses with respect to the available information, i.e. a single scan or a set of already scanned range maps. We quantise the space of normals to the surface parcels, separating void and unseen volumes (*occluding surface*). The space of normals is partitioned using a quality threshold, in order to compute new views covering

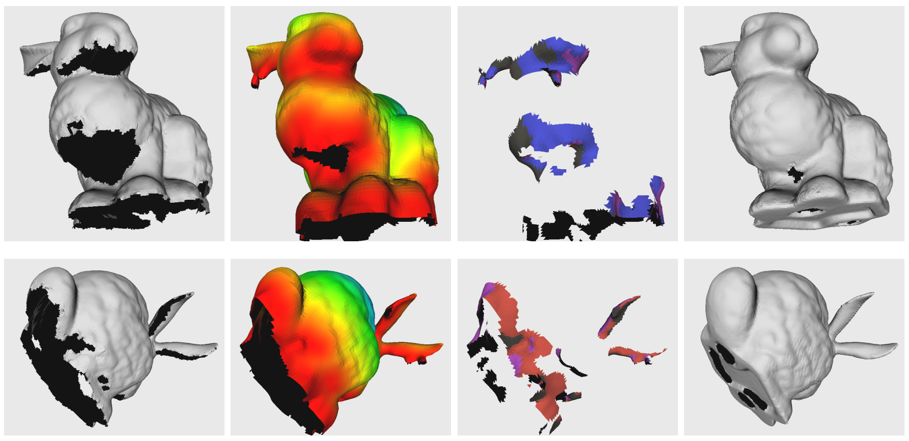


Fig. 17. Sampling the missing surface of the *Bunny* model. Most of the missing surface is detected and sampled in the first iteration.

unsampled regions. In order to check for self-intersections and line-of-sight visibility, a ray-casting algorithm is needed. We avoid direct ray-casting employing an efficient GPU-based method [23].

Both the time performance and the quality of the plan of our approach are strongly dependent on the *hole location* phase. If a faster and more reliable hole location method is found, our algorithm will be faster and more robust.

Discriminating between real holes in the mesh and unsampled parts is a problem common to all the approaches we know of. It can be solved in some special cases but no general solution exists, to our knowledge. A more subtle problem is halting when done. In principle, a completely closed model should contain no borders. However, complex models have often small holes due to the limitations of the scanning hardware. Moreover, most scanning systems cannot sample shiny or transparent surfaces. Hence, the algorithm can be stopped either when no further significative improvements are made during the last iterations, or when the size of unsampled regions falls below a fixed threshold.

The approach presented has been included in the design of an automatic system, which couples the capabilities of a laser scanner with a robot arm (*Digital Sculptor Project*, in cooperation with Scienzia Machinale s.r.l. [23]).

Acknowledgements

This research was supported by the project FIRB - “MACROGeo”.

References

1. M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk, “The Digital Michelangelo Project: 3D scanning of large statues,” in *SIGGRAPH 2000, Computer Graphics Proceedings*, Annual Conference Series, pp. 131–144, Addison Wesley, July 24-28 2000.

2. M. Callieri, P. Cignoni, F. Ganovelli, C. Montani, P. Pingi, and R. Scopigno, "VCLab's tools for 3D range data processing," in *VAST 2003* (A. C. D. Arnold and F. Niccolucci, eds.), (Bighton, UK), pp. 13–22, Eurographics, Nov. 5-7 2003.
3. G. Impoco, P. Cignoni, and R. Scopigno, "Closing gaps by clustering unseen directions," in *Proc. of Shape Modeling International (SMI'04)*, pp. 307–316, 2004.
4. J. Maver and R. Bajcsy, "Occlusions as a guide for planning the next view," *PAMI*, vol. 15, pp. 417–433, May 1993.
5. G. Tarbox and S. Gottschlich, "Planning for complete sensor coverage in inspection," *CVIU*, vol. 61, pp. 84–111, January 1995.
6. C. I. Connolly, "The determination of next best views," in *CRA85*, pp. 432–435, 1985.
7. P. Whaite and F. Ferrie, "Autonomous exploration: Driven by uncertainty," in *McGill*, 1994.
8. R. Pito, "A solution to the next best view problem for automated surface acquisition," *PAMI*, vol. 21, pp. 1016–1030, October 1999.
9. A. Papadopoulos-Orfanos and F. Schmitt, "Automatic 3D digitization using a laser rangefinder with a small field of view," in *3DIM97*, pp. 3 – View Planning, 1997.
10. M. Reed and P. Allen, "Constraint-based sensor planning for scene modeling," *PAMI*, vol. 22, pp. 1460–1467, December 2000.
11. W. Scott, G. Roth, and J.-F. Rivest, "Performance-oriented view planning for automatic model acquisition," in *Proc. of the 31st International Symposium on Robotics*, pp. 314–319, 2000.
12. W. E. L. Grimson, "Sensing strategies for disambiguating among multiple objects in known poses," in *MIT AI Memo*, 1985.
13. S. A. Hutchinson and A. C. Kak, "Planning sensing strategies in robot work cell with multi-sensor capabilities," *RA*, vol. 5, pp. 765–783, December 1989.
14. K. Tarabanis, P. Allen, and R. Tsai, "A survey of sensor planning in computer vision," *RA*, vol. 11, pp. 86–104, February 1995.
15. W. R. Scott, G. Roth, and J.-F. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," *ACM Computing Surveys (CSUR)*, vol. 35, no. 1, pp. 64–96, 2003.
16. J. Davis, S. R. Marschner, M. Garr, and M. Levoy, "Filling holes in complex surfaces using volumetric diffusion," in *First International Symposium on 3D Data Processing, Visualization, and Transmission, Padua, Italy*, June 2002.
17. M. Gervautz and W. Purgathofer, "A Simple Method for Color Quantization: Octree Quantization," in *New Trends in Computer Graphics* (N. Magnenat-Thalmann and D. Thalmann, eds.), pp. 219–231, New York, NY: Springer-Verlag, 1988.
18. S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.
19. H. Späth, *Cluster analysis algorithms*. Chichester, UK: Ellis Horwood, 1980.
20. A. D. Kalvin and R. Taylor, "Superfaces: Poligonal mesh simplification with bounded error," *IEEE C.G.&A.*, vol. 16, no. 3, pp. 64–77, 1996.
21. M. Garland, A. Willmott, and P. Heckbert, "Hierarchical face clustering on polygonal surfaces," in *Proc. of the ACM Symposium on Interactive 3D Graphics*, pp. 49–58, ACM Press, 2001.
22. D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 905–914, 2004.
23. M. Callieri, A. Fasano, G. Impoco, P. Cignoni, R. Scopigno, G. Parrini, and G. Biagini, "RoboScan: an automatic system for accurate and unattended 3D scanning.," in *3DPVT*, pp. 805–812, 2004.
24. Leica Geosystems, "http://www.exactmetrology.com/leica/leica200lr.htm."
25. C. Ericson, *Real-Time Collision Detection*. Morgan Kaufmann, 2004.
26. P. Perona and J. Malik, "Scale space and edge detection using anisotropic diffusion," *PAMI*, vol. 12, pp. 629–639, July 1990.

	# iter	iter 1			iter 2			iter 3			iter 4			iter n			total	# voxels
		field exp.	view comp.	fusion	field exp.	view comp.	fusion	field exp.	view comp.	fusion	field exp.	view comp.	fusion	field exp.	view comp.	fusion		
Lovers	4	6	2	33	6	0	9	5	1	8	6	0	0	—	—	—	76	366 K
Angel	4	136	6	42	74	2	54	100	5	54	169	4	67	—	—	—	713	4.1 M
Bowman	3	1	0	1	1	0	1	1	0	0	—	—	—	—	—	—	5	142 K
Bunny	3	4	0	2	4	0	4	4	0	5	—	—	—	—	—	—	23	371 K
Chief	2	5	2	4	5	1	5	—	—	—	—	—	—	—	—	—	22	1.6 M
Goddess	2	3	0	3	3	0	2	—	—	—	—	—	—	—	—	—	11	220 K
Laurana	2	17	1	15	31	1	29	32	4	30	—	—	—	—	—	—	160	7.6 M
Ecce Homo	3	41	7	43	34	3	48	37	4	45	—	—	—	—	—	—	262	6.4 M
Minerva	12	138	11	61	146	8	90	297	5	113	332	6	113	502	7	176	6800	89.6 M
Gargoyle	4	2	1	5	2	0	4	2	0	4	2	0	0	—	—	—	22	295 K
Thinker	3	5	1	7	5	0	5	6	0	5	—	—	—	—	—	—	34	671 K
Fist	3	2	0	3	3	0	3	3	0	0	—	—	—	—	—	—	14	152 K
Bird	2	4	2	4	2	0	2	—	—	—	—	—	—	—	—	—	14	150 K

Table 1. Time statistics of our planning algorithm. For each model, we show the number of planning iterations required and the computation time (in seconds). We report the time for the three phases of planning in each step: *field expansion*, *view computation*, and *fusion*. The total time and number of voxels processed is also shown. The processing time of all the examples is affordable, except for the *Minerva* model. Anyway, this is because we violate our initial assumption that a roughly covered model is given prior to planning (see text).

	# iter	iter 1		iter 2		iter 3		iter 4		iter n		initial cov.	% sampled	# scans
		# new views	area gain	# new views	area gain	# new views	area gain	# new views	area gain	# new views	area gain			
Lovers	4	6	1.2916	7	1.0180	8	1.0155	0	1.0000	—	—	0.6798	0.9077	27
Angel	4	2	1.2438	8	1.1735	4	1.0398	2	$\sim$1	—	—	0.6907	$\sim$1	22
Bowman	3	4	1.0574	1	1.0680	1	1.0044	—	—	—	—	0.8603	0.9541	12
Bunny	3	4	1.2152	1	1.0004	1	1.0001	—	—	—	—	0.8168	0.9931	12
Chief	2	1	1.0023	1	1.0037	—	—	—	—	—	—	0.9217	0.9703	8
Goddess	2	1	1.1436	0	1.0000	—	—	—	—	—	—	0.8626	0.9865	7
Laurana	2	1	1.0320	2	1.0114	2	1.0000	—	—	—	—	0.9697	0.9955	11
Ecce Homo	3	3	1.0854	8	1.0227	8	1.0028	—	—	—	—	0.9374	$\sim$1	25
Minerva	12	4	1.3336	8	1.1290	5	1.0450	4	1.0096	4	1.0112	0.3802	0.6604	59
Gargoyle	4	8	1.1225	8	1.0039	1	1.0025	0	1.0000	—	—	0.8741	$\sim$1	23
Thinker	3	4	1.1726	2	1.0077	1	1.0001	—	—	—	—	0.8505	$\sim$1	13
Fist	3	4	1.2355	1	1.0034	0	0.0000	—	—	—	—	0.7559	0.9370	11
Bird	2	4	1.1855	2	$\sim$1	—	—	—	—	—	—	0.8215	0.9712	12

Table 2. Planning results. For each iteration, we report the number of new views acquired and the area gained, expressed as *current area/previous area*. The total number of scans required is also reported together with the percentage of area sampled. Notice that six of the scans are acquired using the turntable before planning. Finally, we report the initial coverage given by the turntable to evaluate the sampling gain due to planning.