Three-Dimensional Model Construction from Multiple Sensor Viewpoints

Stéphane Aubry Vincent Hayward

Department of Electrical Engineering and
Research Centre for Intelligent Machines
McGill University
Montréal, Québec, H3A 2A7, Canada

Abstract

We address the problem of constructing a boundary model of an object when the input consists of a set of points that lie on its surface. We assume that the points are acquired using telemetric techniques. Such data constitute a discrete sampling of the surface: a "cloud" of points. Supplemental connectivity information between those points is necessary if one is to reconstruct an approximation to the underlying object. We use the implicit information provided by the acquisition procedure itself to achieve this goal.

1 Introduction

Many applications as diverse as computer-aided manufacturing, dentistry or archeology require digital models of actual three-dimensional objects. Manually measuring the objects and entering the data is a tedious process. Automatically building such models is thus a desirable goal.

Optical telemetric methods are among the most popular active methods of sampling an object's surfaces [4]. These methods rely on the opacity of the sensed object.

The next step is to combine the data from the several views, and to convert that information to a form suitable for a computer representation. This is the task of *data integration.* Since the data may come from several sources, they must be coalesced into a single model. This is because similar sensors positioned at different physical locations give off information that must be integrated. In general, it is necessary to position the sensing apparatus at several locations in order to completely acquire the desired scene. For example, opacity prevents optical sensors from "seeing" beyond occlusions. Hence, several optical sensors "looking" at a given scene but from different locations acquire different but complementary information. How to resolve this complementary information into a unified model is the contribution of this article.

2 Multi-view Integration: A Review

The first task of multi-view integration is to find the geometric transform between the different sensor positions. Once the inter-view transform is found, the views can be merged.

A popular method for doing so is the "silhouette" approach. This approach assumes that a clear separation between the "object" and the "background" can be found. Each image then yields an infinite solid cone whose apex is at the sensor location and whose sides are tangent to the object to be modelled. The intersection of all the cones yields a volume guaranteed to contain the object [14, 15]. Silhouette approaches inherently call for a volumetric representation such as octrees.

But since the sensing apparatus gives us a collection of surface points, the most natural representation is a boundary one. The simplest such representation approximates the desired surface with a collection of triangular facets. The collection of facets forms the surface of a simple *polyhedron.* Processing the point set is hence reformulated as defining adjacency relationships between the data points.

Formally, we wish to build the "most natural" *connectivity graph G* spanning the set V of data points, where G is a *labelled 3-connected maximal planar graph.* Clever algorithms must be designed to prune the candidate graphs since the enormous growth rate of their number $\phi(n)$, as a function of n vertices, precludes all attempts at enumeration [2]:

$$\phi(n) = \frac{(4n-11)!}{(3n-6)!}\binom{n}{2} \quad . \tag{1}$$

We now review below some of the approaches used for the construction of G. O'Rourke defines the polyhedron of minimal area as the most natural model for the set V [11]. He also gives a "greedy" algorithm to compute a good approximation for such a polyhedron. It selects the convex hull of V as an initial approximation. The hull is then incrementally "carved" by

Authorized licensed use limited to: McGill University. Downloaded on December 27, 2009 at 13:51 from IEEE Xplore. Restrictions apply.

inserting its internal points into its boundary. The selection of the internal points and how to insert them into the current shape is based on local criteria only, yielding an $O(n^3)$ algorithm.

An alternative method uses the three-dimensional Delaunay *tetrahedrization* as a seed structure[5]. Tetrahedra are then eliminated based on a local geometrical criterion that measures how "internal" to the current shape the tetrahedron is. As with O'Rourke's method, the process stops when all the internal points are on the boundary of the shape.

Hoppe reconstructs surfaces from very dense data by first constructing a *Riemannian Graph*, a special type of proximity graph derived from the Euclidean Minimum Spanning Tree [10]. Good results are given using synthetic data.

Contextual information often helps in finding the desired shape. This is the case when extra information is provided by the acquisition procedure. For example, if the data is organized into stacked contours, as with tomographic data, the contours can be used as subgraphs of the desired graph. It then remains to link the contours using a heuristic criterion [9, 16]. Alternately, the three-dimensional data may consist of stereo edges, rather than stereo points, and the surface triangulation is then constrained to include such edges [6].

The latter contribution makes a crucial observation: because each element $v \in V$ is a boundary point, v is contiguous to free space. Further, all acquisition procedures not only give off the coordinates of v, but also of a portion E of the free space around v. In the case of optical sensing, E is one or more directed line-of-sight segments joining v and the imaging system. Since these segments are known to lie entirely in free space, they convey information that the coordinate position of the surface point alone did not provide.

In [3], we presented an algorithm that assumes the data is organised in *frames*, where a frame is a set of surface points whose acquisition rays intersect at a common point P. This is the case with many range finders, where P is the range camera center for a given range image. This special structure is exploited to build G in $O(n \log^2 n)$ time. Further, the procedure exhibits the on-line property, that is G can be updated as new frames are acquired. Unfortunately, not all objects can be correctly reconstructed, unless a procedure exponential in the number of frames is used.

Alevizos *et al.*'s algorithm takes as input a *planar* set of points and their associated outward pointing sensor acquisition direction, and uniquely solves the connectivity problem [1]. Moreover, it does so in optimal worst-case asymptotic complexity, namely $O(n \log n)$. An attractive feature is that the resulting graph edges do not call for closeness measures in the Euclidean metric sense, as the Delaunay triangulation does. Indeed, the connectivity we wish to draw on the surface points is one which embeds the notion of nearest neighbour in a geodesic sense. Of course, the geodesic distance cannot be computed, for if it were, we would know the underlying surface. So using the Euclidean distance instead of the (unknown) geodesic distance is an arbitrary (albeit reasonable) choice.

The algorithm we present below solves the problem of constructing a graph G, knowing the set of surface points V and their respective directed segments, which are known to lie in free-space. As in [3], the algorithm does not use any distance measures to build the graph. Instead, it relies uniquely on the information provided by the acquisition segments. Because we assume that the object is fully opaque and that all data points are boundary points, no acquisition segment can penetrate the object. Hence the segments cannot intersect the model's triangular faces. It follows that the crucial criterion we use for the construction of G is that at every stage of its construction, no segment intersect any of its faces.

3 A Global Algorithm

The algorithm assumes no coherence or organisation for the data points. The price to be paid for this generality is the loss of the on-line property. We then say that the algorithm is *global*: all data must be acquired before the construction of G begins.

The algorithm shares several aspects with the ones we reviewed in the last section. It first constructs the convex hull of the set V of all surface data points as an initial approximation G_0 to the desired graph G. Then successive graphs G_i are iteratively constructed by local modifications of G_{i-1}. The number of data points spanned by the successive graphs is guaranteed to grow monotonically. The process stops when the set of vertices spanned by the graph is equal to V.

Assume the following notation:

- S is the set of optical, free-space segments that acquire V.
- If $v \in V$, $s(v)$ is the element of S acquiring, or associated with, v.
- $\forall \hat{V} \subset V$, $S(\hat{V})$ is the set of optical, free-space, segments associated with the elements of $\hat{V}$. Also, $S(V) = S$.
- F is the set of faces of G.
- Graph subscripts indicate iteration levels. For example, G_i is the graph at level i and S_i is the set of acquisition segments of the vertices spanned by G_i.
- Graph superscripts refer to the face of the preceding level's graph that a given subgraph is *rooted at*. For example, G_i^f is the subgraph of G_i rooted at face f, where f is a face belonging to F_{i-1}.

Authorized licensed use limited to: McGill University. Downloaded on December 27, 2009 at 13:51 from IEEE Xplore. Restrictions apply.

- We elide graph attributes as follows: $V(G_i) = V_i$, $E(G_i) = E_i$ and $F(G_i) = F_i$.
- $\overline{V_i} = V \setminus V_i$.
- $\mathcal{M}_i^f$ is the polyhedron drawn by the graph G_i^f, and $\mathcal{M}$ is the polyhedron drawn by the final graph G.

We now describe in detail one algorithm iteration i. Proofs for the claims can be found in [2].

We first construct a partition $\mathcal{P}$ of $\overline{V_{i-1}}$, and of cardinality $Card(F_{i-1})$. Each element $\overline{v}$ of a given equivalence class p^f of $\mathcal{P}$ is such that f is the first face intersected by $s(\overline{v})$.

Let $f = (v_0, v_1, v_2)$. If p^f is empty, no element of $\overline{V_{i-1}}$ crosses f. In this case, we define G_i^f as the triangle graph drawn by f. If p^f is non-empty, we define G_i^f as a maximal three-connected planar graph whose vertex set is $V_i^f = q^f \bigcup \{v_0, v_1, v_2\}$, where q^f is a non-empty subset of p^f, and whose construction we explain in Section 3.1.

Finally, we define G_i as the union[1] of G_{i-1} and of all the G_i^f subgraphs, thus completing the ith iteration: $G_i = G_{i-1} \bigcup \overset{f \in F_{i-1}}{\cup} G_i^f$.

If G_i now spans the set V, (*i.e.* if $\overline{V_i} = \emptyset$), the algorithm stops.

Claim 1 *At each level i, G_i is a three-connected maximal planar graph (3CMPG).*

An obvious corollary of the claim is that the final graph is guaranteed to be a 3CMPG, which is a proper model for the polyhedron $\mathcal{M}$.

Claim 2 *The volume enclosed by the final model is that drawn by the convex hull, less that of the union of the polyhedra drawn by the individual subgraphs of all levels. Formally, $I(\mathcal{M}) = \mathcal{M}_0 \setminus \overset{i}{\bigcup} \overset{f}{\cup} I(\mathcal{M}_i^f)$.*

Claim 2 states that each successive model monotonically "carves" the volume enclosed by the preceding one.

Claim 3 *The segments associated with the vertices spanned by G_i^f do not intersect any of the faces of G_i^f.*

Claim 3 states that each subgraph draws a figure which is *locally* consistent with the data acquisition procedure and the opacity assumption.

Proposition 1 *At any stage i of the graph construction, none of the segments associated with the vertices spanned by the current graph G_i intersects any face of F_i.*

[1]The union of two graphs has for vertex set the union of the graphs' vertex sets and for edge set the union of the graphs' edge sets.

3.1 Subgraph Construction

Recall that f is a graph face of the previous level, and that the segments associated with the elements of p^f intersect f. As in [5], we prefer to link the vertices of f to the points of p^f which are "close" to f, or those whose "penetration" into f is shallow. In this manner, we first construct subgraphs whose faces are near the original f face, and which are themselves intersected by the segments of the "deeper" data points. Again we prefer to use the acquisition segments rather than Euclidean distance measures in order to guide the carving process and to capture to notion of "closeness". This requirement can be met by a special use of *convex layers* [12].

Let $V^f(0) = V_i^f = p^f \bigcup \{v_0, v_1, v_2\}$. We construct the zeroth layer by taking the convex hull of $V^f(0)$. The next layer consists of the convex hull of the points internal to that hull, again augmented with the vertices of f. The process is repeated until a hull is found which contains no internal point. The graph of that hull is the sought subgraph G^f.

Let $G^f(i)$ be the graph drawn by the ith convex layer and let $V^f(i)$ be its vertex set. Let $p^f(i+1)$ be the subset of $V^f(i)$ with degree zero in $G^f(i)$ (See Figure 1). The elements of $p^f(i+1)$ are the data points which are internal to the ith convex layer. If $p^f(i+1)$ is empty, then $G^f = G^f(i)$. If $p^f(i+1)$ is non-empty, we then let $V^f(i+1) = p^f(i+1) \bigcup \{v_0, v_1, v_2\}$, and the $(i+1)$th convex layer is the convex hull of $V^f(i+1)$, whose graph is $G^f(i+1)$.

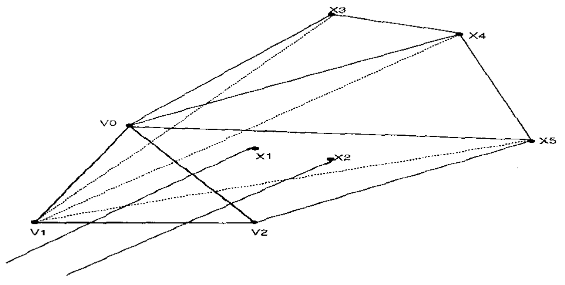

Figure 1: A convex layer at a given level i. x_1 and x_2 have degree zero in $G^f(i)$. Hence we write $p^f(0) = p^f = \{x_1, x_2, x_3, x_4, x_5\}$, $V^f(0) = \{x_1, x_2, x_3, x_4, x_5, v_0, v_1, v_2\}$, $p^f(1) = \{x_1, x_2\}$, $V^f(1) = \{x_1, x_2, v_0, v_1, v_2\}$. For clarity, the only segments shown here are those of x_1 and x_2.

Claim 4 *The convex layer algorithm always terminates.*

Authorized licensed use limited to: McGill University. Downloaded on December 27, 2009 at 13:51 from IEEE Xplore. Restrictions apply.

3.2 Algorithm's Efficiency

We now analyze the theoretical worst-case asymptotic complexity of the algorithm.

The initialization phase requires the computation of the three-dimensional convex hull of a set of N spatial points. Such a hull can be optimally computed in $O(N \log N)$ operations [12]. Each graph iteration then requires:

1. The determination of which face f in $\mathcal{M}_i$ each segment of $S(\overline{V_i})$ intersects.
2. For each face of F_i, the determination of the points of p_i^f which achieve maximum *depth*, where the depth of a point is the number of convex layers that have to be stripped from $V^f(0)$ before these points are removed.

In [8], a data structure called the *drum representation* is given for polyhedra. It can be constructed in $O(n \log n)$ for an n-vertex polyhedron. Among other uses, the representation can detect the intersection of a polygon with that polyhedron in $\log n$ time. By extension, it can also detect the intersection of the polyhedron with a segment, since a segment is a degenerate polygon.

So the intersection of all segments in $\overline{V_i}$ with the faces in F_i (Step 1 above), can be performed in $O(n_f \log n_f + n_s \log n_f)$ using the drum structure, where n_s is the number of segments and n_f is the size of the polyhedron against which to check intersection. n_s is the cardinality of $\overline{V_i}$, which is bounded above by N. Because the graph of a polyhedron is planar, the cardinality of the face set of $\mathcal{M}_i$ has the same order as that of its vertex set. But V_i is the vertex set of $\mathcal{M}_i$, and its cardinality is also bounded above by N. Hence, $O(n_f) = N$ and Step 1 requires $O(N \log N)$ operations per iteration.

Step 2 requires the determination of the maximal depth convex layer for each V_i^f set. In 2 dimensions, this can be achieved in $O(m^f \log m^f)$ [7], which is provably optimal, where m^f is the cardinality of the input set. No such result is known in three dimensions however, so we resort to a repeated application of the simple gift-wrapping technique [12, pp. 125 and 166], which requires at most $O(N^2)$ steps per layer. Since we are aggregating at least one point at each iteration, the worst-case complexity of step2, and of the algorithm, is $O(N^3)$.

The bound is achieved if and only if

1. At each step of the algorithm, only $O(1)$ faces are intersected by segments of $\overline{V_i}$.
2. At each step of the algorithm, only $O(1)$ points are aggregated into the graph. This can only happen if the above condition is satisfied *and* the cardinality of the set of maximum depth is $O(1)$.

Because of the recursive nature of the algorithm, we can reasonably hope that the partition of segments into faces is well-balanced, and that a large number of points gets aggregated at each algorithm iteration. For example, suppose that for every subgraph face $f \in G_i^f$, the number of points of p^f of maximal depth is $O(n)$, where n is the cardinality of p^f. Then the number of iterations is $O(1)$ and the algorithm requires $O(N^2)$ operations. Additionally, suppose that at every iteration, the number of faces of $\mathcal{M}_i$ that get crossed by segments is $O(N)$. Then each p^f set is at most $O(1)$, and Step 2 is performed in constant time a constant number of times. Step 1 however is incompressible, giving the algorithm total execution speed of $O(N \log N)$. The claim that the *expected* speed is smaller than $O(N^3)$ is warranted by experimental execution speeds [2].

We implemented the above algorithm on real noisy data, obtained with a triangulation-based synchronized laser range finder developed at the NRC [13]. Figures 2 and 3 show snapshots of a round object with a deep concavity (a pencil holder) and its facial representation through two levels. The subject measures approximately 20cm in each dimension. Several thousand three-dimensional surface data points were taken from three different viewpoints from a distance of about 60cm, and were smoothed with a Gaussian filter. All three views were designed to sample a substantial portion of the deficiency. The accuracy of the measurements using this setup is better than 1mm. The inter-view calibration was done with a least-squares minimization technique on a set of seven fiducial points, namely the top of the pyramids shown in Figure 2.

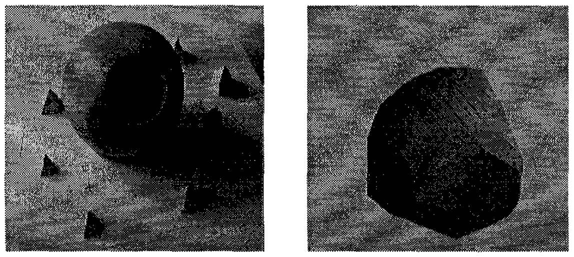

Figure 2: **Left**: A round object with a deep concavity. **Right**: The zeroth-order approximation of the object G_0 (the convex hull). The black lines are the range finder's line-of-sight rays for those points which make up the next order's approximation.

The results are somewhat disappointing from a perceptual point of view. The main reason is that the edges from the higher level always remain in the graph. Where a large deficiency is present, as is the case in the example, each higher-level face gets carved as if *it*

Authorized licensed use limited to: McGill University. Downloaded on December 27, 2009 at 13:51 from IEEE Xplore. Restrictions apply.

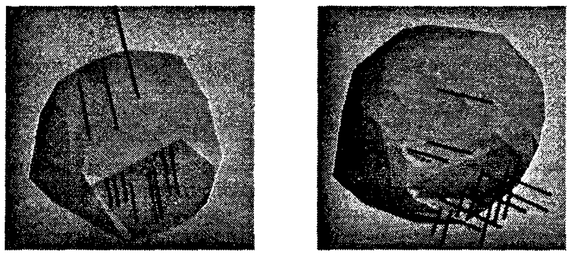

Figure 3: **Left**: The first-order approximation of the object G_1. Artifacts on the round part of the object are caused by calibration inaccuracies. The concavity is now only partially carved. **Right**: The second-order approximation of the object. The concavity is still being carved (note the many rays in the bottom part of the figure) while the round part of the object is fully modelled.

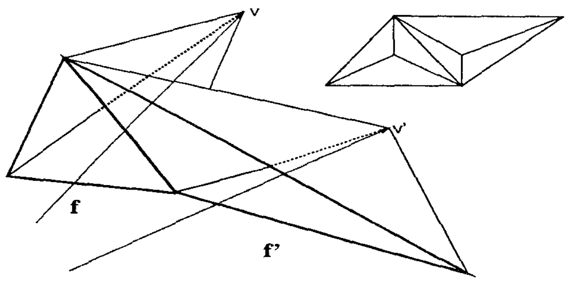

Figure 4: f and f' are two (neighbor) faces of a graph G_i. The cardinality of both p^f and $p^{f'}$ is 1. The acquisition segments are shown for the singletons $v \in p^f$ and $v' \in p^{f'}$. On the right of the figure, the corresponding graph for G^f_{i+1} and $G^{f'}_{i+1}$ is shown.

contained its own separate deficiency. As the carving proceeds, these "separate" deficiencies get larger.

A second problem pertains to the planar-facet representation itself. Claim 3 only shows that the acquisition segments of the vertices spanned by a local graph G^f do not intersect any face of G^f. However, they still *may* intersect the faces of $G^{f'}$, where $f \neq f'$ (we then say loosely that the subgraph G^f and $G^{f'}$ *intersect*). Such situations do arise in practice. One reason of course is that the data at our disposal is insufficient sampled to correctly and unambiguously reconstruct the object. As well, the sensing method introduces local spurious artifacts. But a more fundamental reason is that the simplicial representation we adopted is an arbitrary representation for the underlying surface. In practice, it tends to "carve out" too much of the object's enclosed volume.

Consider the simple case illustrated in Figure 4. f and f' are two faces, each of which is intersected by one acquisition segment. In this case, no segment-to-face intersection is present.

If we modify the position of the data points in Figure 4 to allow v' to translate towards the left of the figure, we reach a limit where v' penetrates through a face ϕ of F^f_{i+1}, as shown in Figure 5. As a result, $s(v')$ intersects ϕ.

This happens because the volume of the deficiencies carved out by the convex $\mathcal{M}_i$ objects is too large. Suppose now that we fix face ϕ at its vertices, and that we "bend" it away from v', thus making it non-planar, so as to allow v' (and hence $s(v')$) to not intersect ϕ any longer. Because we know that both s and s' entirely lie in free-space, there exists a topological mapping $\mathcal{T}$ such that the image of ϕ through $\mathcal{T}$ is a sheet with the same boundaries as ϕ, but which intersects neither $s(v)$ nor $s(v')$.

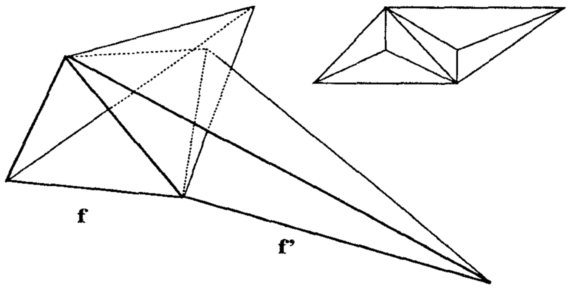

Figure 5: The faces of G^f and $G^{f'}$ intersect. For clarity, the acquisition segments are omitted from this and the following figures.

Hence, although the *geometry* of the modelled object is incorrect, the *connectivity* of its vertices is **consistent** (in the sense of [2]), with the data acquisition procedure.

Refer back to Figure 4. Faces f and f' both spawn a sub-tree, with vertex v and v' respectively. Each sub-tree models a deficiency, which is deemed to have been "discovered" by the intersecting segments $s(v)$ and $s(v')$. Thanks to the particular *geometry* of the data, the model drawn by the resulting graph is free of self-intersections.

In Figure 6, the same data configuration is shown, except that the convex layer algorithm has been performed by *merging* faces f and f'. More formally, the convex layer algorithm has been applied to the merged input set

$$V^{f,f'} = \{x_1, x_2, x_3, x_4\} \cup p^f \cup p^f ,$$

where $f = (x_1, x_2, x_3)$ and $f' = (x_1, x_2, x_4)$.

Authorized licensed use limited to: McGill University. Downloaded on December 27, 2009 at 13:51 from IEEE Xplore. Restrictions apply.

We can see that the resulting boundary and graph-theoretic interpretation of Figure 6 is as consistent with the data as that of Figure 4.

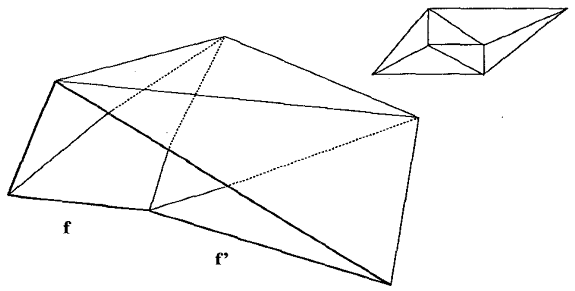

Figure 6: f and f' have been merged for the convex layer algorithm, resulting in a different graph.

Contrast the above situation with that which we illustrated in Figure 5, where the resulting *planar-facet* model violated the opacity condition. The same data geometry is shown in the companion Figure 7, but with the convex layer algorithm having been applied to the merged faces f and f'. As before, merging faces yields a different graph, but it now also eliminates the self-intersection between G^f and $G^{f'}$.

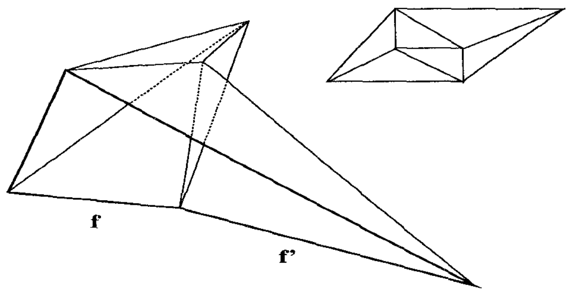

Figure 7: Modified graph for f and f'. The segments associated with the segments of vertices of $V^{f,f'}$ do not now intersect any face of $G^{f,f'}$.

This example illustrates the need for merging faces in the case where neighbouring deficiencies intersect. In [2], a heuristic modification to the algorithm based on the face-merging approach is presented.

References

[1] P.D. Alevizos, J-D. Boissonnat, and M. Yvinec. An optimal $O(n \log n)$ algorithm for contour reconstruction from rays. In *ACM Annual Symposium on Computational Geometry*, 1987.

[2] S. Aubry. *Three-Dimensional Model Construction from Multiple Sensor Viewpoints.* PhD thesis, Dept. of Electrical Engineering, McGill University, Montréal, Québec, Canada, 1994.

[3] S. Aubry and V. Hayward. Building hierarchical solid models from sensor data. In Su shing Chen, editor, *Advances in Spatial Reasoning*, pages 1–64. Ablex Publishing, Norwood, NJ, 1990.

[4] P.J. Besl. Active, optical range imaging sensors. *Machine Vision and Applications*, 1:127–152, 1988.

[5] J.D. Boissonnat. Representing 2D and 3D shapes with the delaunay triangulation. In *Int. Conf. on Computer Vision & Pattern Recognition*, pages 745–748, 1984.

[6] J.D. Boissonnat, O.D. Faugeras, and E. Le Bras-Mehlman. Representing stereo data with the delaunay triangulation. In *IEEE Int. Conf. on Robotics & Automation*, pages 1798–1803, Philadelphia, PA, April 1988.

[7] B.M. Chazelle. Optimal algorithms for computing depths and layers. *Proc. 21st Allerton Conference on Comm., Control, and Computers*, pages 427–436, October 1983.

[8] D.P. Dobkin and D.G. Kirkpatrick. Fast detection of polyhedral intersection. *Theoretical Computer Science*, 27:241–253, 1983.

[9] H. Fuchs, Z.M. Kedem, and S.P. Uselton. Optimal surface reconstruction from planar contours. *Communications of the ACM*, 20:693–702, 1977.

[10] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. Technical Report 91–12–03, University of Washington, Seattle, WA, 1991. Department of Computer Science and Engineering.

[11] J. O'Rourke. Polyhedra of minimal area as 3D object models. In *Int. Joint Conf. on Artificial Intelligence*, pages 664–666, 1981.

[12] F.P. Preparata. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.

[13] M. Rioux. Laser range finder based on synchronized scanners. *Applied Optics*, 23:3837–3844, November 1984.

[14] S.K. Srivastava and N. Ahuja. Octree generation from object silhouettes in perspective views. *Computer Vision, Graphics & Image Processing*, 49:68–84, 1990.

[15] J.R. Stenstrom and C.I. Connolly. Building wire frames from multiple range views. In *IEEE Int. Conf. on Robotics & Automation*, pages 615–620, 1986.

[16] Y.F. Wang and J.K. Aggarwal. Surface reconstruction and representation of 3-d scenes. *Pattern Recognition*, 19(3):197–206, 1986.

Authorized licensed use limited to: McGill University. Downloaded on December 27, 2009 at 13:51 from IEEE Xplore. Restrictions apply.