3D Surface Modeling from Range Curves

Dragan Tubić, Patrick Hébert and Denis Laurendeau
Computer Vision and Systems Laboratory
University Laval, Québec, Canada
[tdragan,hebert,laurendeau]@gel.ulaval.ca

Abstract

Traditional approaches for surface reconstruction from range data require that the input data be either range images or unorganized sets of points. Since a large number of range sensors provide data along curvilinear patterns such as profiles, this paper presents an approach for reconstructing a surface from a set of unorganized curves. A strategy for updating the reconstructed surface during data acquisition is described as well. Curves are accumulated in a volumetric structure in which a vector field is built and updated. The information that is needed for efficient curve registration is also directly available in this vector field. This leads to a unified modeling approach combining surface reconstruction and curve registration. The algorithm implementing the approach is of linear complexity with respect to the number of input curves and makes it suitable for interactive modeling. Simulated data based on a set of six curvilinear patterns as well as data acquired with a range sensor are used to illustrate the various steps of the algorithm.

1. Introduction

Most surface reconstruction algorithms from range data accept either range images [3, 10, 13] or unorganized sets of points [8, 12] as input. Nevertheless, a very common type of range sensors acquire curvilinear measurements on the surface of an object. Based on optical triangulation, these sensors use a well-focused laser pattern projected on the surface and can robustly provide dense profiles in a very short time frame. Methods for surface reconstruction from arbitrary curves have not been developed yet. Instead, it is assumed that the profiles are gathered at regular intervals along a well-defined path such as to group neighboring profiles into a surface. The relative pose of each profile is assumed to be accurate. For hand-held sensors that collect profiles, it is not possible to scan a surface along a fully predefined scanning path. Thus, methods have been proposed to build local surface patches from a rigid series of profiles with known positions but only with a nearly regular scanning path [7, 2, 9, 12]. There are still two problems with these methods. First, the only type of surface curves that can be used are surface profiles. Second, although data acquisition introduces an error on the sensor position, it is not possible to adjust the pose for a single profile. This results in a loss of quality. The current alternative for low cost hand-held sensors is to collect range images [4, 11] at the expense of losing robustness of laser sensors. In this paper, an approach is proposed for reconstructing the surface of free-form objects from arbitrary curves as well as for refining each individual curve pose.

Surface curves contain more information about the measured surface than an unorganized set of points, namely surface tangents that can be exploited to improve the quality of the reconstructed surface. Actually, local estimates of the surface can be obtained using tangents of intersecting curves. Furthermore it can be achieved efficiently by using a volumetric structure where the surface is incrementally recovered as new surface curves are scanned.

Volumetric structures have been exploited for surface reconstruction from range images or an unordered set of points [3, 7, 10, 12] to reduce computational complexity and provide incremental reconstruction. These volumetric approaches use an implicit representation of the surface i.e. a scalar signed distance field computed in a volumetric grid where the reconstructed surface corresponds to the zero crossing of the field. Our first contribution is to show how to build such a volumetric representation directly from a set of measured curves without any intermediate surface representation. Furthermore, the computational complexity with respect to the number of curves is linear and the reconstruction is incremental and order independent; thus no constraints are imposed on sensor displacement. Any light pattern can be used as long as the measured curves intersect.

Very little work has been reported on the subject of surface curves pose refinement (registration) [6]. Although the stability of curve registration may have contributed to this, the main reason is the computational complexity of the registration process. The simultaneous registration based on ICP (Iterated Closest Points) algorithm [1] is of complex-

ity $O(N^2)$ with respect to the number of range images. If the same algorithm is applied to register surface curves, the complexity quickly becomes prohibitive since the number of curves required for a complete reconstruction of an object is at least an order of magnitude larger than the equivalent number of range images. It can easily reach several thousand curves.

If one not only encodes the signed distance field in the volumetric structure but also the direction toward the nearest zero crossing in each voxel, then matching for a control point can be obtained directly from the nearest voxel. We have developed this idea to provide near real-time registration of range images [14]. The second contribution of this paper is to provide the same linear complexity registration for surface curves by registering them with the reconstructed surface. Another important aspect of our approach is a novel definition of a distance measure that improves robustness of registration in the presence of noise.

The next section describes the principle of surface reconstruction from curves, followed by section 2.2 which explains a modification required to allow incremental reconstruction. Section 2.3 exposes curve registration. A novel definition of a distance aimed at improving robustness of registration is presented in section 2.4. Efficient computation of the vector field required for reconstruction is detailed in section 2.5. The results presented in section 3 illustrate each aspect of the approach from simulated and real data.

2. Modeling using Range Curves

As reported in [14], the reconstructed surface is represented implicitly as a vector field. Such a *volumetric* representation contains both the reconstructed surface and its corresponding matching information as direction and distance toward the reconstructed surface. While the surface is represented as the zero-crossing of the norm of the field, the distance and direction are represented by the field itself. In the following section, we describe how an implicit representation of the surface can be created from a set of non-parallel, intersecting surface curves.

2.1 Reconstruction from intersecting curves

The main idea behind our approach is to perform reconstruction by *approximating the surface in the neighborhood of the intersection points of the surface curves*. The reconstruction is based on a fundamental property of differential surfaces stating that all surface curves passing through some point have tangents located in a plane tangent to the surface at this point. The most important consequence of this property is that the tangent plane of a surface can be computed at the intersection point of at least two surface curves if their respective tangents are known at this point.

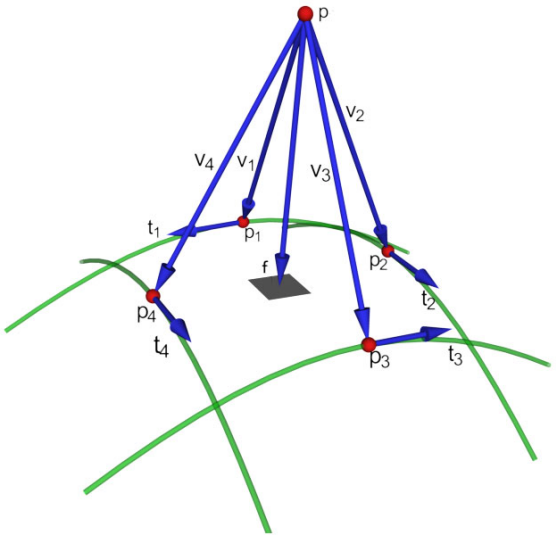


Figure 1. Reconstruction from multiple curves. The normal is obtained as a least-squares estimate of the "most perpendicular" vector to the tangents $\mathbf{t}_i$, $i = 1, 2, 3, 4$ at the closest points $\mathbf{p}_i$ of the four curves.

As mentioned earlier, the reconstructed surface $\hat{S}$ is represented as a vector field $\mathbf{f} : R^3 \rightarrow R^3$ where $\mathbf{f}(\mathbf{p})$ is the direction, and the distance from a point $\mathbf{p}$ to the closest point $\mathbf{p}_c$ on the surface $\hat{S}$ is:

$$\mathbf{p} + \mathbf{f}(\mathbf{p}) = \mathbf{p}_c \in \hat{S}, \tag{1}$$

where

$$\mathbf{p}_c = \underset{\mathbf{q} \in \hat{S}}{\operatorname{argmin}}\, d(\mathbf{p}, \mathbf{q}), \tag{2}$$

and where d denotes a distance measure. In practice, the field $\mathbf{f}(\mathbf{p})$ is computed at points (voxels) on a regular lattice (volumetric grid) in the vicinity of a surface usually referred to as its envelope. The envelope encloses lattice points which are located at a distance to the surface that is smaller than a predefined positive value ϵ. In this work the envelope is defined for each curve that contains the set of lattice points being closer than a predefined distance to the curve.

The approximation of the surface, i.e. the tangent plane at the closest point on the surface to some voxel $\mathbf{p}$ can be obtained as a least-squares estimate of the normal using tangents at the closest points of all nearby curves. This is illustrated in Figure 1. This estimate corresponds to the "most perpendicular" vector to a set of tangents.

More formally, let $\alpha_1, \ldots, \alpha_N$ be N surface curves passing within predefined distance $\epsilon > 0$ from a point (voxel) $\mathbf{p}$ and let $\mathbf{t}_1, \ldots, \mathbf{t}_N$ be their respective tangents at the closest points to the point $\mathbf{p}$. Then the normal on the surface is obtained as the vector $\mathbf{n} = [n_x, n_y, n_z]^T$ that minimizes:

$$\xi = \sum_{i=1}^{N} \langle \mathbf{t}_i, \mathbf{n} \rangle^2. \quad (3)$$

Taking the derivatives of ξ with respect to n_x, n_y and n_z and setting them equal to zero defines the following system of equations:

$$\frac{1}{N} \sum_{i=1}^{N} \mathbf{t}_i \mathbf{t}_i^T \mathbf{n} = \mathbf{C}\mathbf{n} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T. \quad (4)$$

The estimated value for $\mathbf{n}$ is the eigenvector associated with the smallest eigenvalue of matrix C which is just a covariance matrix of the tangents. The distance toward the surface is obtained as the average value of the projected distance vectors $\mathbf{v}_i = \mathbf{p}_i - \mathbf{p}$ on the estimated normal, i.e

$$d(\mathbf{p}, \hat{S}) = \sum_{i=1}^{N} \langle \mathbf{p}_i - \mathbf{p}, \mathbf{n} \rangle \quad (5)$$

where $\mathbf{p}_i$ is the closest point on the curve α_i. Finally the value of the field at point $\mathbf{p}$ is:

$$\mathbf{f}(\mathbf{p}) = d(\mathbf{p}, S)\mathbf{n}. \quad (6)$$

In order to estimate the tangent plane to a surface at some point $\mathbf{p}$, at least two non-parallel tangents are needed to compute the matrix $\mathbf{C}$ at $\mathbf{p}$ since a single tangent does not define a plane. This condition can be verified by analyzing the eigenvalues of matrix $\mathbf{C}$: if two eigenvalues are zero, then only one tangent (or two or more parallel tangents) exists and the normal estimated at that point is not used.

At the intersection points of noiseless curves, all tangents are coplanar, hence one eigenvalue is always equal to zero and the tangents span a plane. In practice, all three eigenvalues are larger than zero since the surface is approximated in the neighborhood of the intersection points. Also, when the tangents are estimated from very noisy data, the three eigenvalues may have similar values in which case the tangents span an ellipsoid and the estimate of the tangent plane is meaningless. To make sure that the estimated tangent plane is valid, an additional verification is made on the eigenvalues. Let λ_1, λ_2 and λ_3 be the three eigenvalues such that $\lambda_1 < \lambda_2 < \lambda_3$. Since matrix $\mathbf{C}$ is normalized, the sum of eigenvalues is always equal to one i.e. $\lambda_1 + \lambda_2 + \lambda_3 = 1$. The eigenvalues have to satisfy the following two conditions: $\lambda_2 > 0.05$ and $\lambda_1 < 0.5\lambda_2$. These two thresholds do not affect the shape of the reconstructed surface. Imposing them simply means that the tangents used to compute $\mathbf{C}$ must be approximately coplanar; otherwise no surface is locally reconstructed.

A single curve influences the field only at points located inside its envelope and its influence drops to zero outside this envelope. The computed field is thus discontinuous over the edges of the envelope and the reconstructed surface appears discontinuous as well. The solution to this problem is to weight the tangents using a continuous function of distance ideally dropping to zero at the edge of the envelope. Any decreasing monotonic function can be used for this purpose. In our experiments the following function proved to be useful:

$$\omega(d) = e^{-d^2/\sigma}. \quad (7)$$

The value of σ is chosen equal to $1/2\epsilon$ to make sure that the value of ω is close to zero outside of the envelope.

The tangents can also be weighted in such a way to reduce the influence of less-confident data, for example by measuring the angle between the surface (curve) normal and the measurement direction. The weighting value τ is defined as the cosine of the angle between the two directions. By taking into account τ and ω, the matrix $ideally$ defined in Eq. 4 becomes

$$\mathbf{C} = \frac{1}{\sum_{i=1}^{N} \tau_i \omega_i} \sum_{i=1}^{N} \tau_i \omega_i \mathbf{t}_i \mathbf{t}_i^T. \quad (8)$$

The reconstructed surface can be extracted from the vector field $\mathbf{f}$ using the Marching Cubes algorithm. The Marching Cubes algorithm uses a scalar field that is obtained by computing the norm of $\mathbf{f}(\mathbf{p})$. This norm is a scalar field whose sign is obtained as the sign of the scalar product between $\mathbf{f}(\mathbf{p})$ and the direction of the sensor.

2.2 Incremental reconstruction

For the reconstruction approach described above, it was assumed that all the data has been collected prior to reconstructing the surface. If the reconstruction has to be performed on-line, then the field needs to be updated incrementally by integrating a single measured curve at a time. However, the least-squares estimate of the surface normal and, consequently the vector field, cannot be computed incrementally. On the other hand, matrix $\mathbf{C}$ computed at some voxel $\mathbf{p}$ is obtained as a sum and can therefore be updated incrementally. Let $\mathbf{C}(\mathbf{p})$ be the matrix C for the voxel $\mathbf{p}$. Equation 8 can be rewritten as:

$$\mathbf{C}(\mathbf{p}) = \frac{1}{\sum_{i=1}^{N} \tau_i \omega_i} \sum_{i=1}^{N} \tau_i \omega_i \mathbf{C}_i(\mathbf{p}), \quad (9)$$

where $\mathbf{C}_i(\mathbf{p}) = \mathbf{t}_i \mathbf{t}_i^T$. During reconstruction, a matrix $C(\mathbf{p})$ is attached to each voxel and is updated after acquiring each curve α_i by summing it with $\mathbf{C}_i(\mathbf{p})$. Matrix $\mathbf{C}_i(\mathbf{p})$ depends only on the curve α_i and is computed using tangent $\mathbf{t}_i$ at the point on the curve that is closest to $\mathbf{p}$. The value of the field $\mathbf{f}(\mathbf{p})$ is computed only during the extraction of the surface using the Marching Cubes algorithm or during registration.

2.3 Pose refinement using vector fields

On the one side, unlike range images, surface curves cannot be registered with each other one pair at the time. In general, the number of intersections between two curves is insufficient to compute the rigid transformation needed to align the curves. For instance two surface profiles intersect at a single point. On the other side, registering all curves simultaneously has $O(N^2)$ complexity, with respect to the number of curves and quickly becomes limiting due to the large number of curves that is needed to reconstruct an object. An efficient way to circumvent these problems is to register curves to the *reconstructed model* instead. Since the vector field contains all the information needed for matching, the computational complexity remains linear with respect to the number of curves. Matching a single control point is of $O(1)$ complexity.

Once the vector field is computed, registering a curve becomes straightforward: for a control point $\mathbf{p}$ on a curve, the corresponding point $\mathbf{p}_c$ on the reconstructed surface is approximated by the value of the vector field at the closest voxel $\mathbf{p_v}$ to the point $\mathbf{p}$. An error introduced by field discretization can be corrected by linearly interpolating the value of the field at the voxel $\mathbf{p}_v$ (see Figure 2):

$$\mathbf{p}_c = \mathbf{p} + \mathbf{f}(\mathbf{p_v}) + \frac{\mathbf{f}(\mathbf{p_v}) < \mathbf{f}(\mathbf{p}_v), (\mathbf{p}_v - \mathbf{p}) >}{\|\mathbf{f}(\mathbf{p}_v)\|^2}. \quad (10)$$

For registration, the model is first reconstructed from all available curves. Then, each curve is registered to this model one at a time. The vector field is then recomputed, and the whole procedure is repeated until no further improvement is possible. Even though the field is represented indirectly through the matrix $\mathbf{C}$ at each voxel, we will refer to it as the vector field $\mathbf{f}$. The registration is described in the following pseudocode.

repeat
  Initialize field $\mathbf{f}$ to zero; **for** *i = 1:number of curves* **do**
    Compute field $\mathbf{f}_i$ for curve i and add it to $\mathbf{f}$;
  end
  for *i = 1:number of curves* **do**
    repeat
      Find matching points for control points of curve i;
      Compute and apply transformation on curve i;
    until *convergence*;
  end
until *convergence*;

Algorithm 1: Simultaneous Registration

In this work, all curve points are used as control points. Convergence is verified by measuring the displacement of points after applying the computed transformation; when it falls below a threshold the algorithm stops.

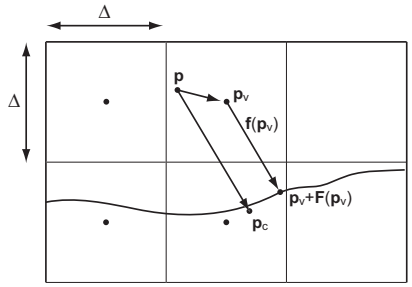


Figure 2. Matching a point $\mathbf{p}$ using the closest voxel centre $\mathbf{p}_v$. The closest point $\mathbf{p}_c$ is obtained using Eq. 10.

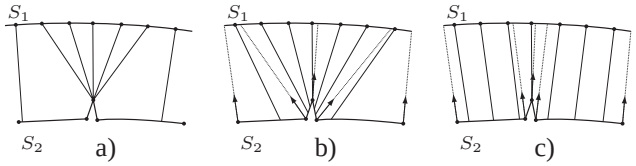


Figure 3. The effect of noise on the matching step of registration. a) Matching with noisy curve (or surface) S_2 using Euclidean distance. b) Matching using the distance defined in Eq. 14 with unfiltered normals c) Matching in the direction of filtered normals.

2.4 Defining the distance measure

As illustrated in Figure 3.a, points with high noise level tend to attract a large number of correspondences. This slows down the registration process and makes it less accurate. To improve the matching robustness, the *direction* toward the closest point on the line segment has to be corrected while taking into account two important constraints: i) the distance field has to be continuous ii) the position of measured 3D points should not be altered. The solution to this problem is to redefine the distance instead of using the Euclidean distance. For the sake of clarity, the distance computation is first illustrated in 2D, the curve being contained in a plane.

A curve α is represented as a set of line segments $\alpha = \{l_1, \ldots, l_{N-1}\}$ between the measured points $\{\mathbf{p}_1 \ldots \mathbf{p}_N\}$, with $l_i = \overline{p_i p_{i+1}}$. The tangent $\mathbf{t}_i$ at each measured point $\mathbf{p}_i$ is computed and filtered by averaging it with its neighbors.

The distance between a point $\mathbf{p}$ and the curve α is always computed with respect to the closest line segment l_c. It is therefore sufficient to provide the distance from a point to a line segment. The set of all points for which a line segment l_c is the closest is called a *fundamental cell* associated with the *generator line segment* l_c. For the purpose of surface

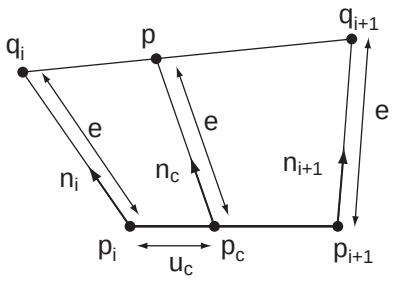


Figure 4. Distance in the direction of interpolated normals. Distance e between point p to the line segment $\overline{p_i p_{i+1}}$ is defined as the distance between $\mathbf{p}$ and the point $\mathbf{p}_c$ whose normal $\mathbf{n}_c$ passes through $\mathbf{p}$. The distance e is computed by finding iso-segment $\overline{q_i q_{i+1}}$ that contains the point $\mathbf{p}$.

modeling, the field needs to be calculated only within a relatively small distance $\epsilon > 0$ from the measured curve, thus limiting the size of the cell to the set of points which are closer than ϵ. If d is the Euclidean distance then the fundamental cells correspond to the cells of the Voronoi diagram for a set of line segments.

To compute the distance, a line segment $\overline{p_i p_{i+1}}$ is parameterized as:

$$\mathbf{l}(u) = \mathbf{p}_i + u(\mathbf{p}_{i+1} - \mathbf{p}_i), \quad 0 \leq u \leq 1. \tag{11}$$

The tangents at the two end points are then interpolated as:

$$\mathbf{t}(u) = \mathbf{t}_i + u(\mathbf{t}_{i+1} - \mathbf{t}_i), \quad 0 \leq u \leq 1. \tag{12}$$

The normal $\mathbf{n}(u)$ at each point of a line segment is defined as the vector being perpendicular to the tangent and can be also interpolated:

$$\mathbf{n}(u) = \mathbf{n}_i + u(\mathbf{n}_{i+1} - \mathbf{n}_i), \quad 0 \leq u \leq 1. \tag{13}$$

Finally, the distance d between point $\mathbf{p}$ and the line segment $\overline{p_i p_{i+1}}$ is defined as the distance between $\mathbf{p}$ and the point $\mathbf{p}_c$ whose normal $\mathbf{n}_c$ passes through $\mathbf{p}$. This is illustrated in Figure 4. More formally:

$$d(\mathbf{p}, \overline{p_i p_{i+1}}) = d(\mathbf{p}, \mathbf{l}(u_c)) = e, \qquad 0 \leq u_c \leq 1, \tag{14}$$

such that

$$\mathbf{p} = \mathbf{l}(u_c) + e \cdot \mathbf{n}(u_c). \tag{15}$$

To obtain a closed-form solution for the distance between a point $\mathbf{p}$ and a line segment $\overline{p_i p_{i+1}}$ according to the above definition, we note that, if the distance is d, then the point lies on the line segment whose end points are $\mathbf{q}_i = \mathbf{p}_i + e \cdot \mathbf{n}_i$ and $\mathbf{q}_{i+1} = \mathbf{p}_2 + e \cdot \mathbf{n}_2$, as illustrated in Figure 4. This line segment is an *iso-segment* whose points are all located at a distance d from the generator line segment $\overline{p_i p_{i+1}}$. The distance is computed by making the area of the triangle $\mathbf{q}_i \mathbf{q}_{i+1} \mathbf{p}$ equal to zero, i.e. the cross-product of $\mathbf{q}_i - \mathbf{p}$ and $\mathbf{q}_{i+1} - \mathbf{p}$ is zero. This leads to a system of three quadratic equations with a single unknown e. Any of these equations can be used to solve for e after making sure that the chosen parameters do not vanish altogether.

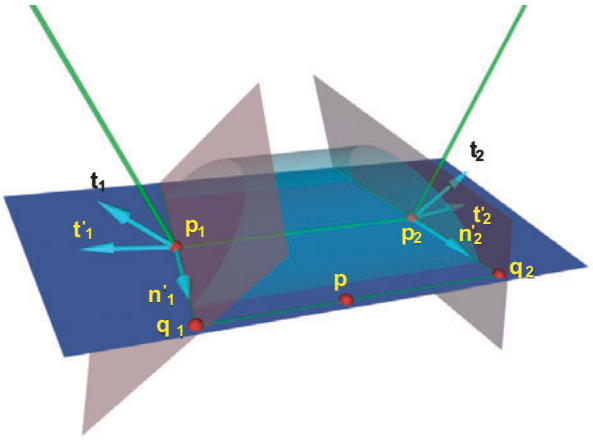


Figure 5. Computing the distance in direction of interpolated normals. Before computing the distance, the tangents $\mathbf{t}_1$ and $\mathbf{t}_2$ are projected on the plane $\overline{\mathbf{p}_1 \mathbf{p}_2 \mathbf{p}}$.

For the 3D case, it is sufficient to project the tangents and normals on the plane containing $\mathbf{p}_1, \mathbf{p}_2$ and $\mathbf{p}$ as illustrated in Figure 5. Projected tangents $\mathbf{t}'_1$ and $\mathbf{t}'_2$ are then used in equation 12 instead of $\mathbf{t}_1$ and $\mathbf{t}_2$.

The effect of choosing the distance defined in Eq. 14 is illustrated in Figure 3.b and c. Filtering the tangents, and consequently the normals, affects the direction toward the closest point, and therefore the matching directions. This makes the matched points more evenly distributed over the reconstructed surface. It is important to note that only tangents on the curves are filtered, the measured 3D data remaining unchanged.

2.5 Efficient computation of the vector field

Equation 14 is valid only inside the fundamental cell of a line segment. Prior to computing the distance to the curve at some point $\mathbf{p}$, one has find to which cell the point $\mathbf{p}$ belongs. This is a tedious task that can be avoided by inverting the process and by rather finding all voxels falling inside each cell separately. By doing so, the complexity of field computation is linear with respect to the number of line segments (measured points). Computing the field for a cell is therefore a two-step process. First, the bounding box of a cell is computed. The two defining corners of the bounding box can be obtained as the minimum and maximum values of $\mathbf{p}_1 + \epsilon, \mathbf{p}_1 - \epsilon, \mathbf{p}_2 + \epsilon, \mathbf{p}_2 - \epsilon$. Then, for all grid points located inside the bounding box it is verified whether they are

between the two delimiting planes of the cell. Finally, the distance is computed and accepted when it is smaller than the maximum allowed distance ϵ.

3 Results

The quality of the modeling algorithm has been assessed by experimenting under varying levels of noise, registration errors, and different laser patterns using both synthetic and real range data. In the following the results obtained in these experiments are presented.

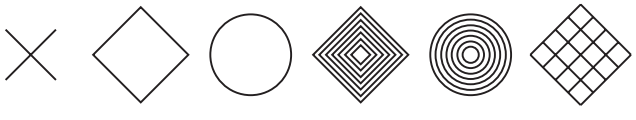
Figure 7a illustrates the most simple case where a surface is reconstructed in the neighborhood of two intersecting curves. Since the surface is approximated, the error of the reconstructed surface increases with distance from the intersection. However, it should be noted that the reconstructed surface still follows the shape of the original surface faithfully. This cannot be accomplished by considering curves as unorganized sets of points, i.e. by fitting a plane in the neighborhood of a point.

An example of reconstruction from multiple curves, using real range data is shown in Figure 7b. The data for this model has been acquired using the 3D sensor described in [5] which acquires surface curves by measuring distances to a projected cross-hair laser pattern (Fig. 6a).

Figure 8 depicts an example of incremental reconstruction using a set of circular light patterns. The figure also illustrates another important aspect of reconstruction: surface filtering by integrating redundant data. When the curves are well registered, adding more curves reduces the variance of noise while preserving fine details.

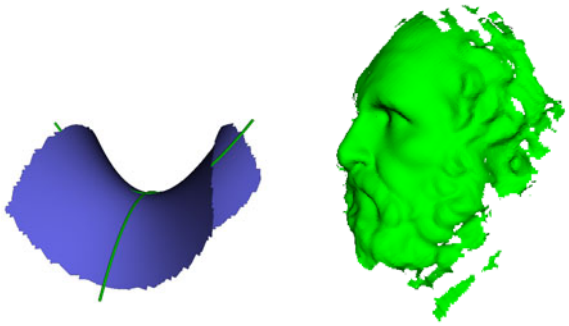
The most important parameter for the reconstruction is the size of the envelope (ϵ). On the one hand, since the reconstruction performs averaging in a neighborhood whose size is equal to ϵ, the envelope should be as small as possible to prevent a loss of fine details. Furthermore, increasing the size of the envelope slows down the algorithm since the number of points for which the field is computed grows as a power of two with the size of the envelope. On the other hand, too small an envelope results in poor performance in presence of noise and registration errors. In particular, if the envelope is smaller than the level of noise (especially outliers), small isolated patches appear around the reconstructed surface, as illustrated in Figure 9b. A solution to this problem is to increase the size of the envelope (Figure 9c), but with the aforementioned performance penalty and loss of details. Preferably, the small patches can be removed since they are not connected to the main surface.

As shown in Figure 10, the registration part of the algorithm converges quickly and remains stable over a large number of iterations. Convergence has been tested by measuring the average displacement of circular curves (6c) as a function of the number of iterations. Other patterns behave similarly. The inner loop converges on average in less



(a) (b) (c) (d) (e) (f)

Figure 6. Light patterns used in experiments.



(a) (b)

Figure 7. Reconstruction examples. a) Reconstruction from two intersecting curves. b) Reconstruction from real data.

than 30 iterations. To visualize the effect of the field update in the outer loop, the convergence curves in a) are drawn for three iterations of the outer loop. One may observe the decrease in the initial state (Figure 10b).

The quality of registration has been assessed by registering synthetic curves with varying levels of noise, outliers and registration errors. For that purpose, six noisy synthetic data sets have been generated using the model of a head and for six curvilinear patterns. Registration errors were introduced by perturbing the pose rotation angles randomly within $[-\delta_r, \delta_r]$ and the translation within $[-\delta_t, \delta_t]$. Noise was simulated by translating each point in the direction of laser projector for random distance chosen within $[-\delta_n, \delta_n]$ while the outliers were generated by randomly choosing 1% of the total number of points and displacing them for a random distance chosen within $[-\delta_o, \delta_o]$. Random variables follow uniform distributions. Values for the curve displacement and noise levels were chosen smaller than the size of the envelope. Otherwise, some curves are outside of the envelope and cannot be registered. The size of the model is approximately $20cm$ while the size of a voxel is $1mm$.

The results summarized in Figure 11 show the performance of the registration algorithm as a function of the initial registration errors and the level of noise. Residual registration error has been obtained as the distance between the registered points and their corresponding closest points on the reference model. Prior to computing residual errors, all curves as a whole were registered to the reference model using an ICP algorithm.

The average residual error as well as the variance slowly

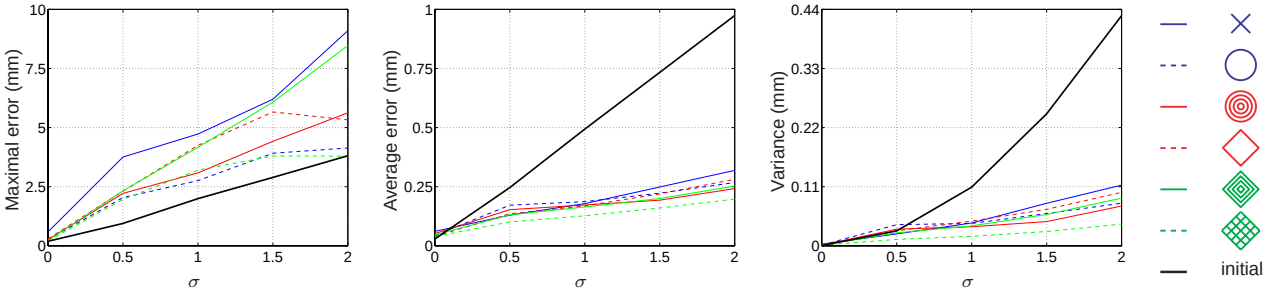


Figure 11. Residual registration errors for noiseless curves as a function of initial displacements of curves and the level of noise. The values δ_r, δ_t, δ_n and δ_o (see the text for their meaning) are related to X-axis index σ as : $\delta_r = \sigma(deg)$, $\delta_t = \sigma(mm)$, $\delta_n = \sigma(mm)$ and $\delta_o = 5\sigma(mm)$. The thick black line indicates initial registration error.

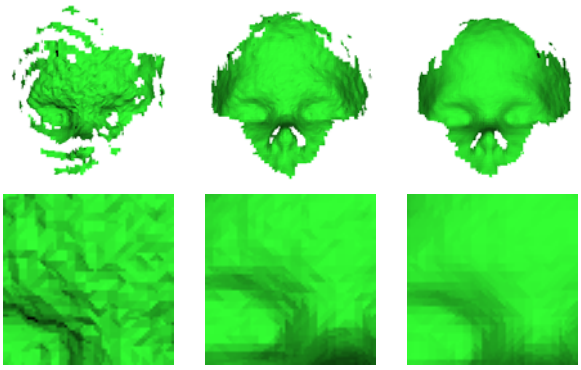
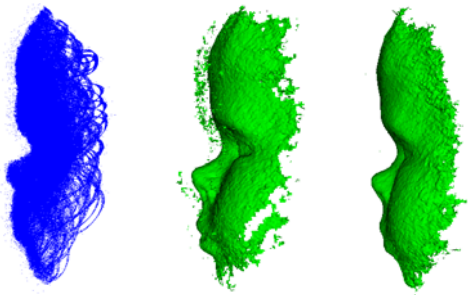
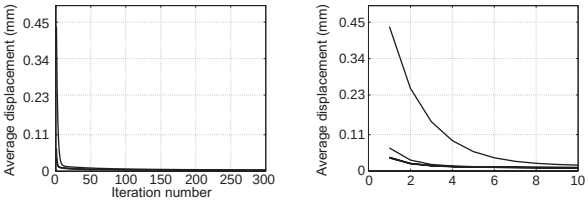


Figure 8. Filtering by averaging redundant data. From left to right: reconstruction from 10, 120 and 240 curves. Bottom row: left eye detail of the reconstructed model.



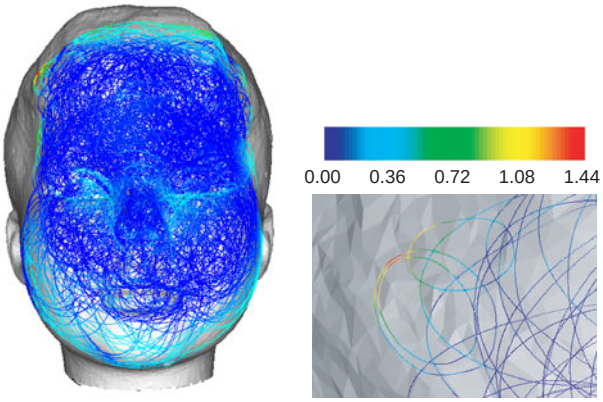
(a) (b) (c)

Figure 9. Influence of high level of noise and outliers on the reconstructed surface (simulation). a) Noisy range data with 1% outliers. b) If the level of noise is larger than the size of the envelope, small isolated patches appear around the surface. c) Increasing the size of the envelope removes the isolated patches.



(a) (b)

Figure 10. Average displacement of points as a function of the number of iterations. a) Inner loop of the registration algorithm. b) Evolution of the inner loop convergence during 5 iterations of the outer loop.



(a) (b)

Figure 12. Color encoded distribution of residual registration errors (mm). a) Curves on the border of the scanned area are only partially matched and therefore unreliably registered. b) Zoom in the upper-left region of the forehead. Only three curves have a registration error larger than 1mm.

increase with noise and initial registration errors. An average error of $0.25mm$ or below is sufficiently small to yield a reconstructed surface without visible artifacts. More importantly the diagrams show that for errors smaller than the envelope size, the performance of the algorithm degrades gracefully as the level of noise increases rather than to collapse after a certain level of noise. Finally, it should be noted that all laser patterns gave similar results. As shown in Figures 11b and c, patterns having the most uniform and dense distribution of points perform slightly better.

The high maximum residual error in Figure 11a indicates that some curves were not well registered. On the other hand, small variance and small average error indicate that the number of curves having this large error is small. As illustrated in Figure 12 the curves with large errors are located on the boundary of the scanned area where the density of curves is low. In these regions the field could not be computed so that curves are only partially matched against the reconstructed surface, thus giving rise to an unreliable registration. Among 450 curves, there were only three curves with an error larger than $1mm$. This problem can be solved by rejecting curves that are partially matched.

Registering real range data of the head model obtained with the sensor described in [5] leads to similar results. Initial average registration errors were $0.35mm$ and with a variance of $0.09mm^2$. After registration, these errors were reduced to $0.26mm$ (average) and $0.07mm^2$ (variance), which is in conformity with results obtained with synthetic data. The residual registration error was evaluated using a highly accurate model of the head (the same model used to generate synthetic range data) measured with a range sensors whose precision is $25\mu m$.

The last experiment shows the performance of the implemented algorithm. The incremental reconstruction from curves having 250 points each is performed in real-time at the frame rate of the sensor (30fps). Thereafter, the registration algorithm is run. Execution time of the inner loop of the registration algorithm for 450 curves is on average less than 2 seconds. Field recomputation takes more time depending on the size of the envelope: 9 sec for $\epsilon = 3mm$ and 23 sec for $\epsilon = 5mm$. Total registration and reconstruction time is 55 sec for $\epsilon = 3mm$ and 140 sec for $\epsilon = 5mm$. It should also be noted that software implementation was not optimized. The execution times were obtained using a PC with 1.2GHz AMD Athlon processor.

4 Conclusion

A volumetric approach for modeling from curves measured on the surface of a free-form object is proposed. Algorithms for both registration and reconstruction are of linear complexity with respect to the number of curves. The reconstruction is incremental thus allowing integration of a single curve at a time. In addition, the reconstruction is order independent. No intermediate surface representation is required: an implicit volumetric representation of the surface is created directly from the curves. Any curvilinear light pattern can be used for this purpose as long as the resulting curves intersect to provide redundant data for registration and reconstruction.

The major drawback of the volumetric approaches is a very inefficient use of computer memory. Even though proposed compression schemes such as run length encoding [3] can be used to reduce memory requirements, it is still not as efficient as surface based representations since these methods encode only occupancy of voxels regardless of the object geometry. We do believe that compression of volumetric data based on the geometry of the object is possible and it will be the subject of our future work.

References

[1] P. Besl and N. McKay. A method for registration of 3-d shapes. *IEEE Transactions PAMI*, 14(2):239–256, 1992.

[2] S. J. Cunnington and A. J. Stoddart. Self-calibrating surface reconstruction for the modelmaker. In *BMVC'98 proc.*, volume 2, pages 790–799, 1998.

[3] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH '96 Proceedings*, pages 303–312, 1996.

[4] O. Hall-Holt and S. Rusinkiewicz. Stripe boundary codes for real-time structured-light range scanning of moving objects. In *Proceedings of ICCV 2001*, 2001.

[5] P. Hébert. A self-referenced hand-held range sensor. In *Proceedings of 3DIM*, pages 5–11, May 2001.

[6] P. Hébert and M. Rioux. A self-referenced hand-held range sensor. In *Proceedings of SPIE*, volume 3313, pages 2–13, January 1998.

[7] A. Hilton and J. Illingworth. Geometric fusion for a hand-held 3d sensor. *Mach. vis. and applications*, 12:44–51, 2000.

[8] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *SIGGRAPH '92 Proceedings*, volume 26, pages 71–78, July 1992.

[9] J. Kofman and G. K. Knopf. Registration and integration of narrow and spatiotemporally-dense range views. In *Proceedings of SPIE, Vision Geometry VII*, volume 3454, pages 99–109, July 1998.

[10] T. Masuda. Object shape modeling from multiple range images by matching signed distance fields. In *Proceedings of 3DPVT*, pages 439–448, June 2002.

[11] M. Proesmans, L. V. Gool, and F. Defoort. Reading between the lines - a method for extracting dynamic 3d with texture. In *Proceedings of ICCV 1998*, 1998.

[12] G. Roth and E. Wibowo. An efficient volumetric method for building closed triangular meshes from 3-d image and point data. In *Graphics Interface*, pages 173–180, May 1997.

[13] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *IEEE Transactions PAMI*, 17(4):344–358, 1995.

[14] D. Tubić, P. Hébert, and D. Laurendeau. A volumetric approach for interactive 3d modeling. In *Proceedings of 3DPVT*, volume 1, pages 150–158, June 2002.