Computer Science

Building 3-D Models from Unregistered Range Images

K. Higuchi, M. Hebert, K. Ikeuchi

November 1993
CMU-CS-93-214

Carnegie Mellon

2

Building 3-D Models from Unregistered Range Images

K. Higuchi, M. Hebert, K. Ikeuchi

November 1993
CMU-CS-93-214

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

This work was sponsored in part by the National Science Foundation under IRI-9224521 and in part by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-90-C-1465, ARPA Order No. 7597 and Contract F33615-93-1-1282.

Keywords: Model building, 3-D surfaces, object recognition, range data, spherical images, surface registration.

Abstract

In this paper, we describe a new approach for building a three-dimensional model from a set of range images. The approach is able to build models of free-form surfaces obtained from arbitrary viewing directions, with no initial estimate of the relative viewing directions. The approach is based on building discrete meshes representing the surfaces observed in each of the range images, to map each of the meshes to a spherical image, and to compute the transformations between the views by matching the spherical images. The meshes are built using an iterative fitting algorithm previously developed; the spherical images are built by mapping the nodes of the surface meshes to the nodes of a reference mesh on the unit sphere and by storing a measure of curvature at every node. We describe the algorithms used for building such models from range images and for matching them. We show results obtained using range images of complex objects.

Contents

Introduction 1

Spherical Attribute Images 3

Semi-Regular Tessellation 3

Discrete Curvature Measure 3

Spherical Mapping 4

Extracting the SAI from a Range Image 5

Registering Multiple Views 7

Finding the Best Rotation Between SAIs 7

Efficient Matching 8

Computing the Full Transformation 9

Matching Partial Views 10

Performance 11

Building a Complete Model 13

Conclusion 17

References 17

1. Introduction

Most computer vision systems require accurate three-dimensional models. The problem of building such models from observations consists in taking multiple range image of the object from different viewing positions and orientations, referred to as "viewing poses", to match the data in the different images in order to recover the relative poses, and to merge the data into a single model using the estimated poses. The approaches proposed so far suffer from two major limitations. First, they require accurate knowledge of the relative viewing poses. Second, they either require a complicated feature extraction algorithm to be applied to the range image or they restrict the class of shapes that can be modelled. Our goal in this paper is to eliminate these two restrictions in order to allow modelling of natural, free-form objects from arbitrary unknown viewpoints.

Examples of feature-based model building include the work of Parvin and Medioni [8] in which they segment range data into regions and represent one view as a graph of visible regions. By matching two graphs from two arbitrary viewing directions, they determine the transformation between the graphs. This method limits the class of shapes to which it can be applied since it requires stable segmentation results. Other techniques, such as Kamgar-Parsi's [6] avoid the need for real geometrical features by defining virtual features from, for example, the iso-range contours of the object. Another example is Stein's approach [9] in which the virtual features are groups of surface normals.

Other techniques eliminate feature matching by formulating the registration problem as a non-linear minimization problem in which the objective function is the sum of the distances between the data points in one view and the transformed data points from the other view. For example, Champleboux [2] uses the Levenberg-Marquart algorithm to perform the minimization. This type of approach requires an initial estimate of the relative viewing poses.

Besl [1] proposed an algorithm for matching between free-form surfaces. The algorithm is based on iterative projection of one surface on the other. A similar approach was suggested by Chen and Medioni [3] and by Zhang [10]. Besl's approach has the advantage that it does not require extracting features or establishing correspondences between features. However, because it is an iterative algorithm, it is very sensitive to the initial transformation.

In this paper, we propose a different approach to the model building problem. Our approach is based on the representation of free-form surfaces developed in [4][5]. Figure 1 illustrates our approach: A mesh of points is fit to an input set of data points from each view, a curvature measure is computed at every node of the meshes and map to a spherical image, the Spherical Attribute Image (SAI). The transformation between views is computed by comparing their SAIs. Finally, the data points from all the range images are merged into a single set using the estimated poses and a complete surface model is computed.

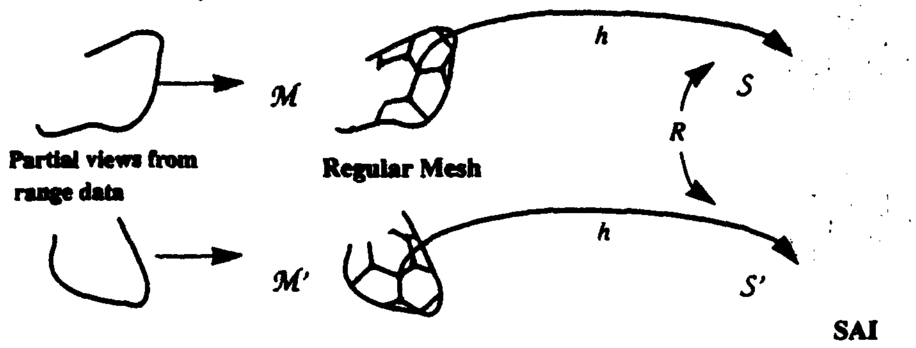


Figure 1: Surface matching using discrete meshes and spherical images.

We describe the algorithms used for SAIs from range images in Section 2.. We first describe the concept of semi-regular meshes (Section 2.1.) and the measure curvature (Section 2.2.) which are the basis for the surface representation. Then we introduce the mapping between surface mesh and spherical mesh in Section 2.3. Finally we describe the algorithm used for extracting representation from range data in Section 2.4.. This discussion will show that there is no underlying assumption about the surface except that it is without topological holes, thus supporting our claim that our approach is suitable for *free-form surfaces*. In Section 3., we describe how two partial representations of the same object from two different poses can be registered. We first show how to compute a rotation of the spherical image in Sections 3.1. and 3.2.. We show in Section 3.2. that the search for the optimal rotation can be made very efficient, provided that some tables are pre-computed. The algorithm of Section 3.2. will validate our claim that the matching algorithm requires *no initial estimates* of the transformation and that it is guarantee to find the *best transformation* up to the resolution of the mesh. We show how to convert this rotation into a full 3-D transformation between surfaces in Section 3.3.. Since no assumption is made on the transformation and since no prior estimate is needed, we will show that the algorithm is able to match surfaces from *arbitrary poses*. We discuss the issue of matching partial views in Section 3.4.. Finally, we show how to build complete models in Section 4.

2. Spherical Attribute Images

In this section, we briefly introduce the concept of SAI. First, we explain how to tessellate an arbitrary surface into a semi-regular mesh, and how to calculate the simplex angle, a variation of curvature, at the nodes of the mesh, and how to map the mesh to a spherical image. Finally, we discuss how to handle partial views of 3-D objects.

2.1. Semi-Regular Tessellation

A natural discrete representation of a surface is a graph of points, or tessellation, such that each node is connected to each of its closest neighbors by an arc of the graph. We use a type of mesh such that each node has exactly three neighbors. Such a mesh can be constructed as the dual of a triangulation of the surface. Let us first consider tessellations of the unit sphere. We use a standard semi-regular triangulation of the unit sphere constructed by subdividing each triangular face of a 20-face icosahedron into N^2 smaller triangles. The final tessellation is built by taking the dual of the 20 N^2-face triangulation, yielding a tessellation with the same number of nodes.

In order to obtain a mesh of an arbitrary surface, we deform a tessellated surface until it is as close as possible to the object surface (Section 2.4.). We need to add another constraint in order to build meshes suitable for matching. In particular, we need to make sure that the distribution of mesh nodes on the surface is invariant by rotation, translation and scale. We introduced in [5] the following regularity constraint: Let $\boldsymbol{P}$ be a node of the tessellation, $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, $\boldsymbol{P}_3$ be its three neighbors, $\boldsymbol{G}$ be the centroid of the three points, and $\boldsymbol{Q}$ be the projection of $\boldsymbol{P}$ on the plane defined by $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, and $\boldsymbol{P}_3$ (Figure 2). The local regularity condition simply states that $\boldsymbol{Q}$ coincides with $\boldsymbol{G}$. This local regularity constraint is the generalization to three dimensions of the regularity condition on two dimensional discrete curves which simply states that all segments are of equal lengths.

2.2. Discrete Curvature Measure

The next step in building a discrete surface representation is to define a measure of curvature that can be computed from a tessellation. Instead of estimating surface curvature by locally fitting a surface or by estimating first and second derivatives, we proposed in [5] a measure of curvature computed at every node from the relative positions of its three neighbors. We called this measure of curvature the simplex angle and we denote its value at node $\boldsymbol{P}$ by $g(\boldsymbol{P})$. Although $g(\boldsymbol{P})$ is not the curvature at $\boldsymbol{P}$, it behaves as a qualitative measure of curvature which is sufficient for matching purposes. Figure 3 illustrates the behavior of $g(\boldsymbol{P})$: The simplex angle varies between $-\pi$ and π. The absolute value of $g(\boldsymbol{P})$ is large in absolute value if $\boldsymbol{P}$ is far from the plane of its three neighbors and vanishes as $\boldsymbol{P}$ and its three neighbors are in the same plane. Finally, $g(\boldsymbol{P})$ is negative if the surface is locally concave, positive if it is convex. $g(\boldsymbol{P})$ is invariant by rotation, translation, and scaling[1].

1. Although the simplex angle is not formally the mean or Gaussian curvature of the surface, we frequently refer to it as the "curvature" of the surface at a node of a discrete mesh.

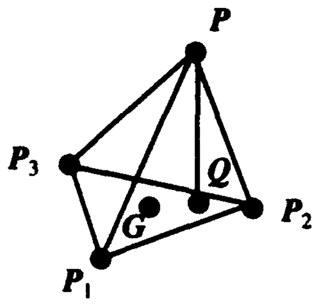
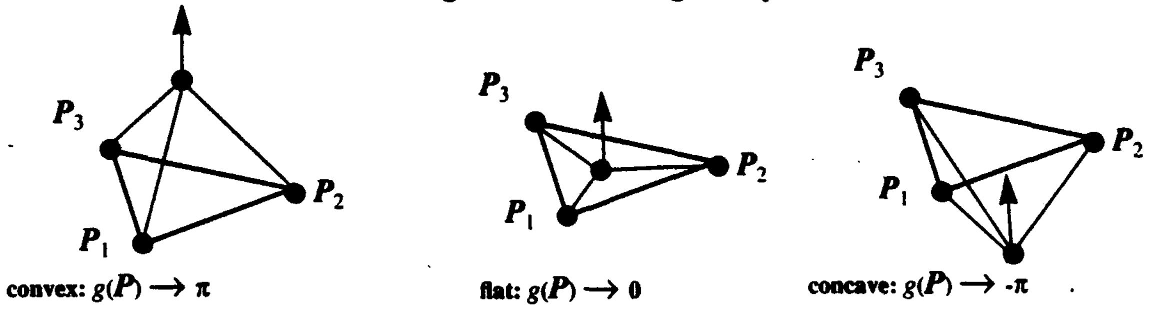


Figure 2: Local Regularity



P_3
P_2
P_1
convex: $g(P) \longrightarrow \pi$
P_3
P_2
P_1
flat: $g(P) \longrightarrow 0$
P_3
P_2
P_1
concave: $g(P) \longrightarrow -\pi$

Figure 3: Definition of the Simplex Angle

2.3. Spherical Mapping

A regular mesh drawn on a closed surface can be mapped to a spherical mesh in a natural way. For a given number of nodes K, we can associate to each node a unique index which depends only on the topology of the mesh and which is independent of the shape of the underlying surface. This numbering of the nodes defines a natural mapping h between any mesh $\mathcal{M}$ and a reference mesh $\mathcal{S}$ on the unit sphere with the same number of nodes: $h(P)$ is the node of $\mathcal{S}$ with the same index as P.

Given h, we can store at each node P of $\mathcal{S}$ the simplex angle of the corresponding node on the surface $g(h(P))$. The resulting structure is a spherical image, that is, a tessellation on the unit sphere, each node being associated with the simplex angle of a point on the original surface. We call this representation the Spherical Attribute Image (SAI)[1]. In the remainder of the paper, we will denote by $g(Q)$ instead of $g(h^{-1}(Q))$ the simplex angle associated with the sphere node Q.

If the original mesh $\mathcal{M}$ satisfies the local regularity constraint, then the corresponding SAI has several invariance properties. First, for a given number of nodes, the SAI is invariant by translation and scaling of the original object. Second, the SAI represents an object unambiguously up to a rotation. More precisely, if $\mathcal{M}$ and $\mathcal{M}'$ are two tessellations of the same object

1. In previous papers on this subject, we used to refer to the SAI as the "Simplex Angle Image". The new name reflects the fact that this representation may be used to store any attribute computed or measured on the surface, not just the simplex angle.

with the same number of nodes, then the corresponding SAIs S and S' are identical up to a rotation of the unit sphere. One consequence of this property is that two SAIs represent the same object if one is the rotated version of the other. It is this property which will allow us to match surfaces that differ by arbitrary rigid transformations.

Another important consequence of the definition of h is that it preserves connectivity. More precisely, a connected patch of the surface maps to a connected patch of the spherical image. It is this property that allows us to work with non-convex objects and to manipulate models of partial surface, neither of which are possible with conventional spherical representations.

In order to build complete models from partial views, we need to represent partial surface models using the SAI. In practice, we always build a complete closed mesh even when only a part of the surface is visible and we mark the nodes of the mesh that are in visible regions of the range image. A node is marked as visible if its distance to the closest data point is below a threshold.

2.4. Extracting the SAI from a Range Image

In the previous sections, we have described the basic approach to representing a mesh of points as a spherical image. The remaining problem is to compute the 3-D mesh from a set of 3-D points from a range image. We use directly the algorithm based on deformable surfaces introduced in [4].

The general approach is to first define an initial mesh near the object and to slowly deform it by moving its nodes until the mesh satisfies two conditions: It must be close to the input object and it must satisfy the local regularity condition. The first condition ensures that the resulting mesh is a good approximation of the object, while the second condition ensures that a valid SAI can be derived from the mesh. These two conditions can be expressed as a set of forces acting between mesh nodes and data points and between the mesh nodes and their neighbors. Once a locally regular mesh is created from the input data, a reference tessellation with the same number of nodes is created on the unit sphere. We refer the reader to [4] for the details of the algorithm.

Figure 4(a) and (b) show an intensity image and the corresponding set of points from the range image. In this example, we use the dual of the 9th subdivision of a 20-face icosahedron, (1620 faces) as shown in Figure 5(a). This initial mesh is deformed and reaches the stable state shown in Figure 5(b). The corresponding SAI data is shown in Figure 5(c). In the SAI display, the distance from each vertex to the origin is proportional to the simplex angle.

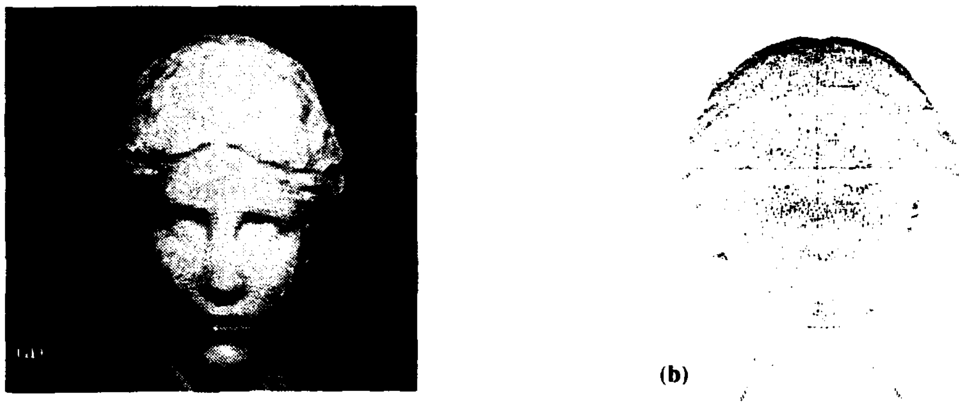


Figure 4: Input data; (a) Intensity image, (b) Range data.

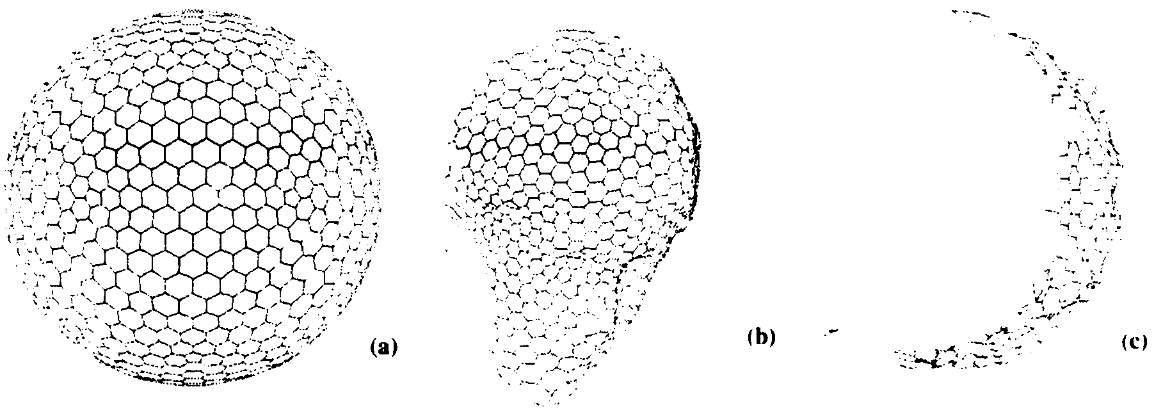


Figure 5: (a) Initial mesh; (b) Deformed mesh; (c) SAI represented on the unit sphere.

3. Registering Multiple Views

We now address the registration problem: Given two SAIs, determine the rotation between them, and then find the rigid transformation between the two original sets of points. The representations of a single object with respect to two different viewing directions are related by a rotation of the underlying sphere. Therefore, the most straightforward approach is to compute a distance measure between two SAIs. Once the rotation yielding minimum distance between spherical images is determined, the full 3-D transformation can be determined.

3.1. Finding the Best Rotation Between SAIs

In the following discussion, we will consider only the vertices of the SAIs that correspond to visible parts of the surface. Let $\mathcal{S}$ and $\mathcal{S}'$ be the SAIs of two views. $\mathcal{S}$ and $\mathcal{S}'$ are representations of the same area of the object if there exists a rotation $\boldsymbol{R}$ such that $g(\boldsymbol{P}) = g(\boldsymbol{RP})$ for every point $\boldsymbol{P}$ of $\mathcal{S}$. Since the SAI is discrete, $g(\boldsymbol{RP})$ is not defined because in general $\boldsymbol{RP}$ falls between nodes of $\mathcal{S}'$. We define a discrete approximation of $g(\boldsymbol{RP})$, $G(\boldsymbol{RP})$, by interpolating the values of g at the four nodes of $\mathcal{S}'$ nearest to $\boldsymbol{RP}$, $\boldsymbol{P}_1$ to $\boldsymbol{P}_4$. Formally, $G(\boldsymbol{RP})$ is a weighted sum of $g(\boldsymbol{P}_i)$. This interpolation is introduced only temporarily because we will see in Section 3.2. that we only need to test a small number of rotations for which the interpolation is not necessary.

The problem now is to find this rotation using the discrete representation of $\mathcal{S}$ and $\mathcal{S}'$. This is done by defining a distance $D(\mathcal{S}, \mathcal{S}', \boldsymbol{R})$ between SAIs as the sum of squared differences between the simplex angles at the nodes of one of the sphere and at the nodes of the rotated sphere. Formally, the distance is defined as:

$$D(S, S', R) = \sum (g(P) - G(RP))^2$$

The minimum of D corresponds to the best rotation that brings $\mathcal{S}$ and $\mathcal{S}'$ in correspondence.

Figure 6 shows the result of matching two views of a head. Figure 6(a) shows the intensity images of the two views of the object. Figure 6(b) shows the corresponding SAIs. Figure 6(c) shows the distribution of D as a function of two of the rotation angles, φ and θ. The graph exhibits a sharp minimum corresponding to the best rotation between the two spherical maps.

The rotation of the SAIs is not the same as the rotation of the original objects; it is the rotation of the spherical representations. An additional step is needed to compute the actual transformation between objects as described in Section 3.3. below.

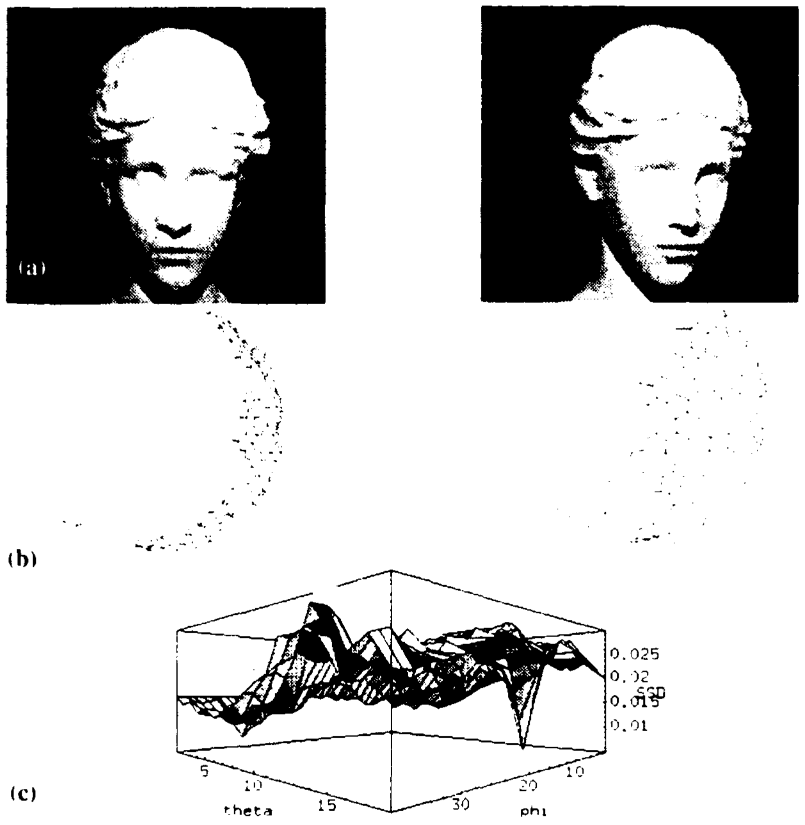


Figure 6: Matching two SAIs

3.2. Efficient Matching

The graph of Figure 6 was obtained by sampling the space of all possible rotations, represented by three angles (θ, φ, ψ), and by evaluating D for every sample value (θ_i, φ_i, ψ_i). Although it is the approach that we used initially, it would be too expensive in practice to compute the distance for all possible rotations.

We developed an efficient SAI matching algorithm based on the observation that the only rotations for which $D(\mathcal{S}, \mathcal{S}', \boldsymbol{R})$ should be evaluated are the ones that correspond to a valid list of correspondences $\{(\boldsymbol{P}_i, \boldsymbol{P'}_j)\}$ between the noes $\boldsymbol{P}_i$ of $\mathcal{S}$ and the nodes $\boldsymbol{P'}_j$ of $\mathcal{S}'$. Figure 7(a) illustrates the idea of correspondences between nodes: Node $\boldsymbol{P}_1$ of the first SAI is put in correspondence with node $\boldsymbol{P'}_{i1}$ of $\mathcal{S}'$ and its two neighbors, $\boldsymbol{P}_2$ and $\boldsymbol{P}_3$, are put in correspondence with two neighbors of $\boldsymbol{P'}_{i1}$, $\boldsymbol{P'}_{i2}$ and $\boldsymbol{P'}_{i3}$, respectively. This set of three correspondences defines a unique rotation of the spherical image. It also defines a unique assignment for the other nodes, that is, there is a unique node $\boldsymbol{P'}_{ij}$ corresponding to a node $\boldsymbol{P}_i$ of $\mathcal{S}$, given the initial correspondences. Moreover, there is only a small number of such initial correspondences, or, equivalently, there is a small number of distinct valid rotations of the unit sphere. In fact, the number of rotations is $3K$ if K is the number of nodes.

Based on this observation, we developed an SAI matching algorithm decomposed into two stages: a pre-processing phase and a run-time phase. During pre-processing, we generate the data structure shown in Figure 7(b). The data structure is a two dimensional array in which each row corresponds to a possible rotation of the SAI and in which column j of row i is the index of the node $\boldsymbol{P}_{ij}$ corresponding to node $\boldsymbol{P}_j$ and correspondence number i. At run-time, the distance is evaluated for each row of the array:

$$D_i(S, S', R) = \sum (g(P_j) - g(P_{ij}))^2$$

The row that produces the minimum D_i gives the best correspondence between nodes of the mesh, $\{(\boldsymbol{P}_j, \boldsymbol{P'}_{ij})\}$, which is used for computing the full transformation between the object meshes as described in the next section. It is important to note that this algorithm tries all possible rotations of the SAIs up to the resolution of the mesh. Consequently, it is guaranteed to find the global optimum of D and it does not require an initial estimate of the transformation. This validates our initial claims of global optimality and pose-independence of the algorithm. This is an efficient algorithm because all that is required at run time is to look up the correspondence table, to compute the sum of square differences of the corresponding nodes and to add them up. In our current implementation, the computation time is 7 sec. for $K = 980$.

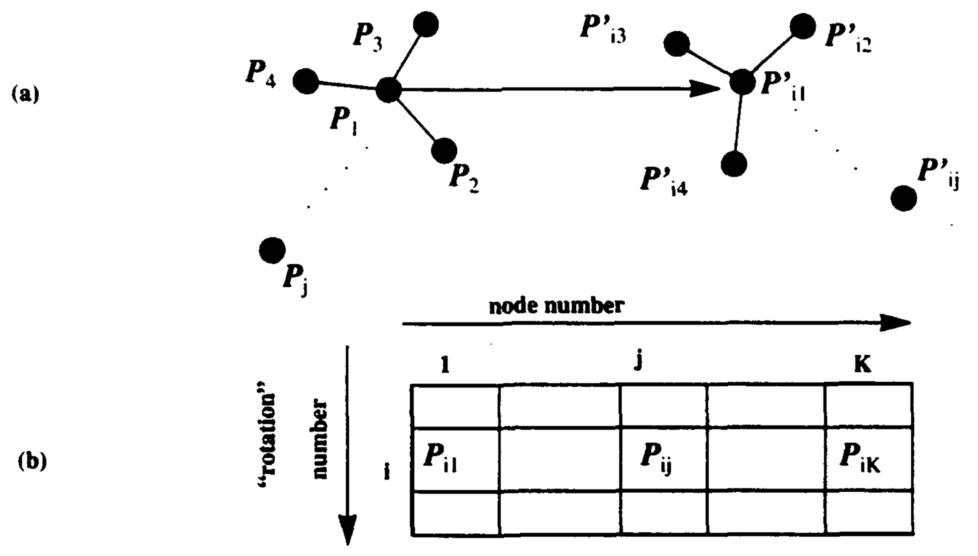


Figure 7: Efficient matching algorithm; (a) Valid correspondence between nodes; (b) Table of correspondences

3.3. Computing the Full Transformation

The last step in matching objects is to derive the transformation between the actual objects, given the rotation between their SAIs. The rotational part of the transformation is denoted by $\boldsymbol{R}_0$, the translational part by $\boldsymbol{T}_0$. Given a SAI rotation $\boldsymbol{R}$, we know the corresponding node

P' of each node ***P*** of $\mathcal{S}$. Let ***M***, resp. ***M'***, be the point on the object corresponding to the node ***P*** of $\mathcal{S}$, resp. ***P'***. A first estimate of the transformation is computed by minimizing the sum of the distances between the points ***M*** of the first object and the corresponding points $R_0M'+T_0$ of the second object. Formally, the expression to minimize is:

$$E(R_o, T_o) = \sum \| M - (R_oM' + T_o) \|^2$$

The sum in this expression is taken over the set of all the nodes of the mesh. The optimum transformation for E can be computed in a non-iterative manner by using the standard quaternion-based techniques. The resulting transformation is only an approximation because it assumes that the nodes from the two meshes correspond exactly. We use an additional step to refine the transformation by looking for the node ***M*** closest to ***M'*** for every node of the mesh and by computing again the minimum of $E(\boldsymbol{R},\boldsymbol{T})$ [5].

Figure 8 shows the final result of computing the transformation between the two views of Figure 6. Figure 8(a) shows the superimposition of the data points from the two range images before computing the transformation. Figure 8(b) shows the same combined data set using the transformation computed using the algorithm above. This display shows that the two views are registered correctly. In this experiment, no prior knowledge of the transformation was used.

3.4. Matching Partial Views

In order to compare SAIs computed from different views, we need to adjust the number of nodes because the relative sizes of the visible and hidden areas vary depending on the viewing direction.

Let us consider the problem of merging two views, $\mathcal{V}_1$ and $\mathcal{V}_2$. Let S_1 and S_2 be the number of nodes that would be visible from $\mathcal{V}_1$ and $\mathcal{V}_2$ if we had a complete model of the object. Let the visible areas of the object surface be A_1 and A_2 for $\mathcal{V}_1$ and $\mathcal{V}_2$, respectively. The ratio of the number of visible SAI nodes to the total number of SAI nodes, S_0 is equal to the ratio of the visible area to the entire object area, A_0:

$$\frac{S_1}{S_o} = \frac{A_1}{A_o} \qquad \frac{S_2}{S_o} = \frac{A_2}{A_o}$$

However, we do not know A_0 since we have only partial views of the object, but we can estimate A_1 and A_2 from each of the views. Eliminating S_0 from these equations, we obtain $S_2 = S_1.A_2/A_1$.

This equation enables us to modify the SAIs from different views so that the distribution of nodes in the visible area is consistent between views. More precisely, we compute the scale factor A_2/A_1 from the estimated visible areas from each of the images, and move the nodes of the SAI from $\mathcal{V}_2$ so that the equation is satisfied.

The key in this procedure is the connectivity conservation property of the SAI. Specifically, if a connected patch of the surface is visible, then its corresponding image on the SAI is also a connected patch on the sphere. This property allows us to bring the two connected patches into correspondence using a simple spherical scaling.

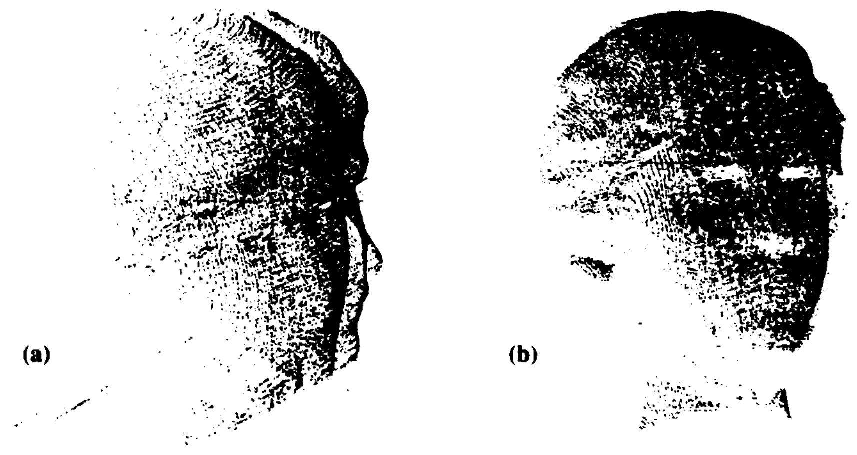


Figure 8: Merging two views; (a) Overlaid views before registration; (b) Overlaid views after registration.

3.5. Performance

Figure 9 shows an example of the error distribution after matching. This figure shows views of the mesh used for performing the registration of Figure 8 in three different orientations. The error at each node of the mesh, that is, the distance between the node and the closest point of the data set, is displayed as a needle, the length of which is proportional to the error. This display shows that the error is reasonably uniform across the mesh. The largest values of the error occur at the edge of the mesh. This is because there is poor overlap between mesh and data at those points.

For a more quantitative evaluation of the quality of the registration, Figure 10 lists error statistics computed on the example of Figure 9. The table lists the minimum, maximum, average, and standard deviation of the registration error at the nodes of the mesh. The registration error is defined as the distance between a mesh node and the closest data point after registration. The errors are listed in millimeters in the table. In both examples, the mean error is on the order of 0.1mm which is also the maximum resolution of the range sensor. The standard deviation is on the order of 0.2mm, reflecting the fact that the error is distributed in a relatively uniform manner. The large maximum error is due to "border effects". Specifically, a node at the edge of the visible part of the mesh may not overlap exactly with a region of the data set, thus causing a large error to be reported. This occurs only at a few isolated nodes at the border. This effect is more noticeable in the case of the face because only partial views are used, in which there is a larger number of border points. Finally, the minimum error is very small, on the order of 0.01mm, but this is really meaningless because it occurs only at a very few isolated points and is the result of accidental alignment between mesh nodes and data points.

These numbers show that the overall behavior of the registration error is on the order of the resolution of the sensor, in this case 0.1mm. This shows, in particular, that the node correspondences found through SAI matching are correct and the estimation of the pose based on the correspondences is basically as accurate as it can be given the finite sensor resolution.

Only the nodes of the mesh that are visible, as determined by the geometry of the sensor, are actually used in the error computation. The errors at the other nodes is meaningless since they are interpolated and not fit to the data. The errors were computed from 998 nodes out of a total of 1620 nodes in this example. The ratio of number useful of useful nodes to total number of nodes is lower in the case of the face because only partial views are used.

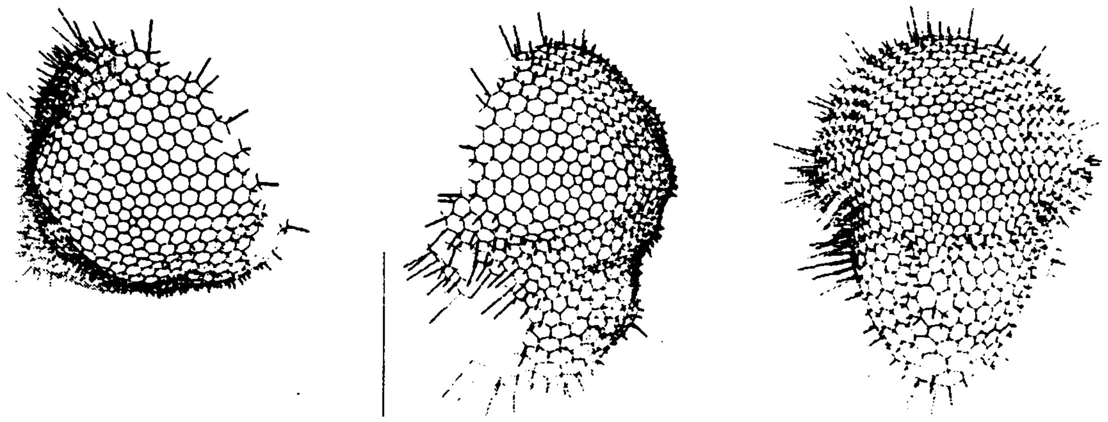


Figure 9: Distribution of errors in the registration example of Figure 8 displayed as a needle map.

Min	0.006
Max	2.46
Mean	0.167
Standard deviation	0.215
Number of points	998

Figure 10: Matching and pose estimation error statistics for the examples of Figure 9. The error values are expressed in millimeters.

4. Building a Complete Model

We have described so far an algorithm for matching two pieces of the surface of an object computed from two unregistered range images. We now consider the case of merging multiple images into a single model. The basic approach is to match the surfaces from the images in a pairwise manner, to combine the transformations obtained from the matching into transformations between each image and a single model reference frame, to convert all the data points from all the range image into this common coordinate system, and to fit a deformable mesh to this data set in order to obtain a smooth surface model.

Figure 11 shows an example of model building from three views. In this experiment, 3-D range data is obtained using a commercial light-stripe range-finder [9] which can acquire registered range and intensity images. Figure 11(a) shows the intensity images of a human hand from three different arbitrary views. Figure 11(b) and (c) show the tessellation of the visible part of the hand and the corresponding SAI for each view, respectively. We use the dual of the 7th subdivided icosahedron containing 980 faces as initial mesh. In each image, only about 30% of the object is visible; the remaining 70% of the representation is interpolated and is ignored in matching the SAIs.

Figure 11(d) shows the result of pairwise image registration. Each of the two displays shows a 3-D view of the set of data points obtained by combining the points from two views using the transformation computed from the matching. The errors of the registration of these models are 0.86 and 0.97 mm RMS distance, respectively. Figure 11(e) shows several views of the set of points obtained by combining the points from all three images using the transformations computed by the SAI matching algorithm. Finally, Figure 11(f) shows the complete surface model obtained by applying the deformable surface algorithm [4] to the entire data set.

This experiment highlights some of the characteristics of the SAI matching approach. First, the viewpoints are arbitrary in that the transformations between them are not restricted to a single-axis rotation as is often the case in modeling systems. Furthermore, the transformations between the viewpoints are not known a priori but are recovered accurately by the algorithms. Second, the matching algorithm does not require any feature extraction or feature matching. This is an important characteristic that enables us to handle arbitrary curved objects for which reliable feature extraction is difficult, such as the hand in Figure 11.

In the example of Figure 11, there is good surface overlap between all the views and there is no ambiguity as to which transformations should be used to generate the final model. Figure 12 shows a different situation that is typical of a model building application in which we have a larger number of image. From considerations of surface coverage, views 1,5 and 9 are sufficient to reconstruct the model (Figure 13). In fact, it would be preferable to use only those views instead of the 12 views to speed up reconstruction and to minimize errors. However, the transformations between these three views, indicated by the thick vertical white arrows, cannot be computed directly because there is very little overlap between the corresponding surfaces. Therefore, the only way to compute the transformations is to compute the intermediate transformations, indicated by the curved arrows, using SAI matching between consecutive views. These "elementary" transformations are compounded to form the two desired transformations. Data points from images 1,5, and 9 are converted to a common coordinate system as shown in Figure 14(a). A 980-node surface model is then computed by fitting a deformable surface as shown in Figure 14(b).

It is clear that we could have matched different views to recover the relative transformations and that this particular selection may not be optimal. Finding the optimal combination of views is still an open issue. What this example shows, however, is that the matching algorithm provides us with the basic tool for performing registration between surfaces in a general manner. It also shows that the individual transformations computed by the matching algorithm are accurate enough that they can be compounded over long sequences into transformations which accurate enough for building a complete model.

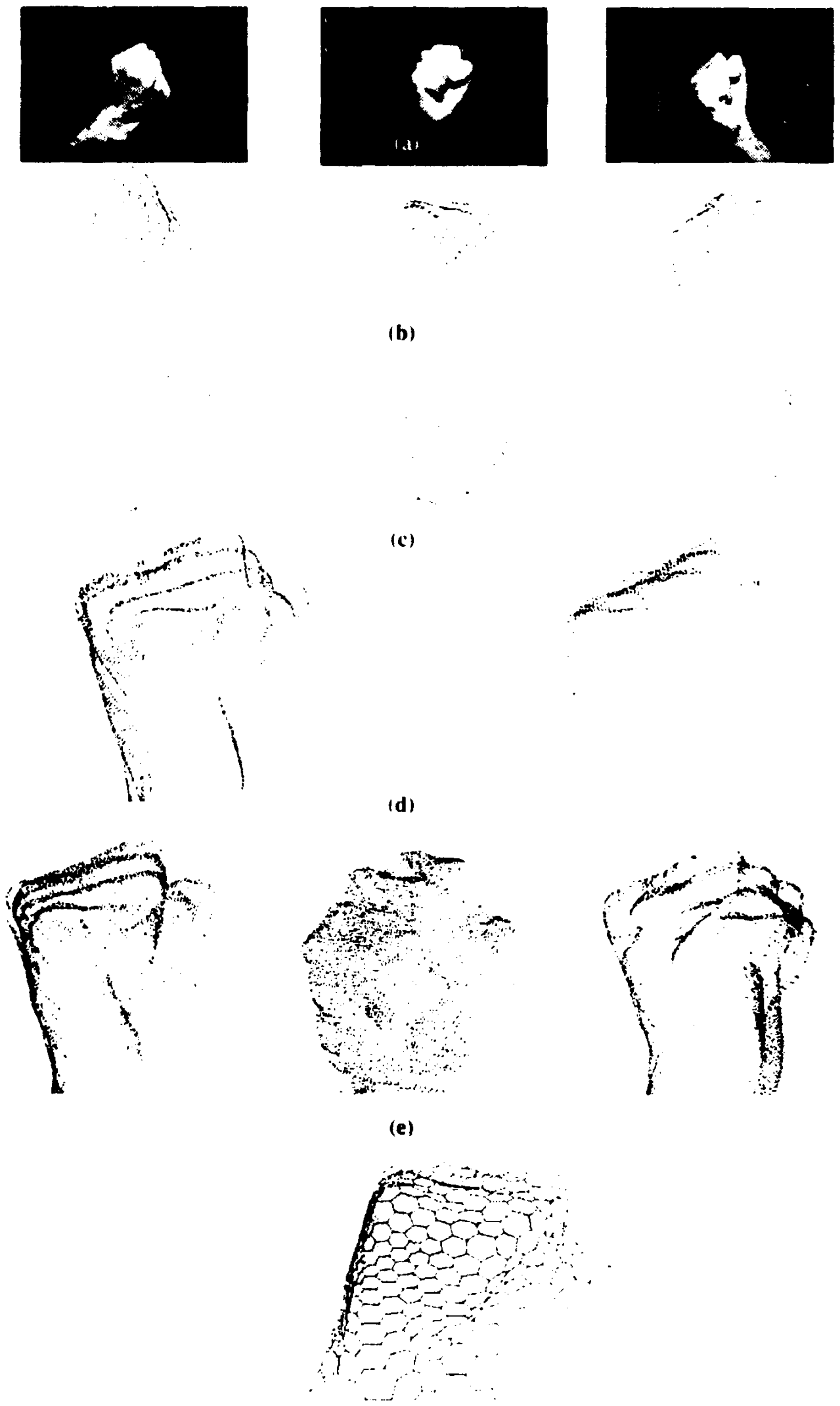


Figure 11: Building a complete model of a human hand; (a) Intensity images; (b) Deformed mesh; (c) SAIs; (d) Data points after pairwise registration; (e) Data points after full registration; (f) Complete model.

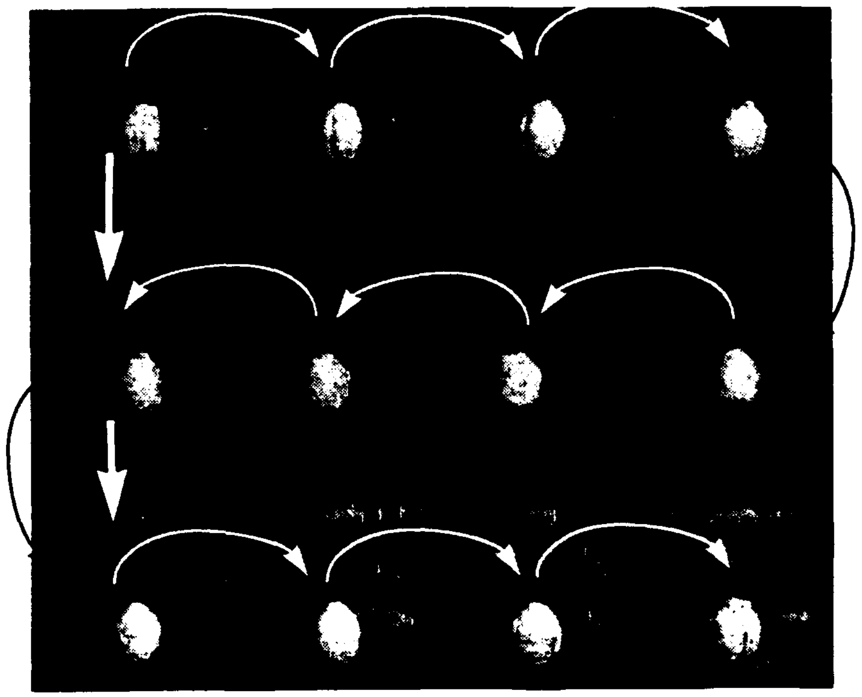


Figure 12: Twelve views of an object and computed poses.

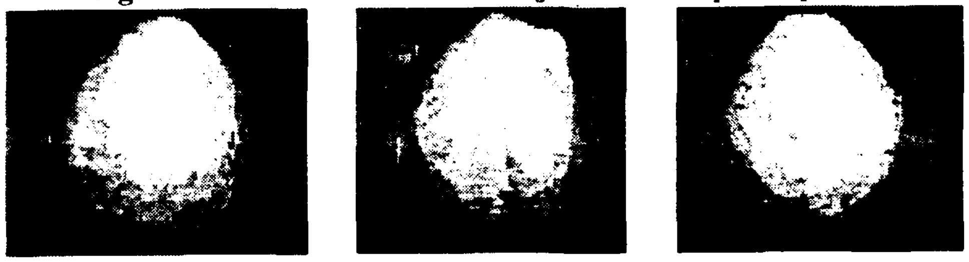


Figure 13: Three views with sufficient overlap.

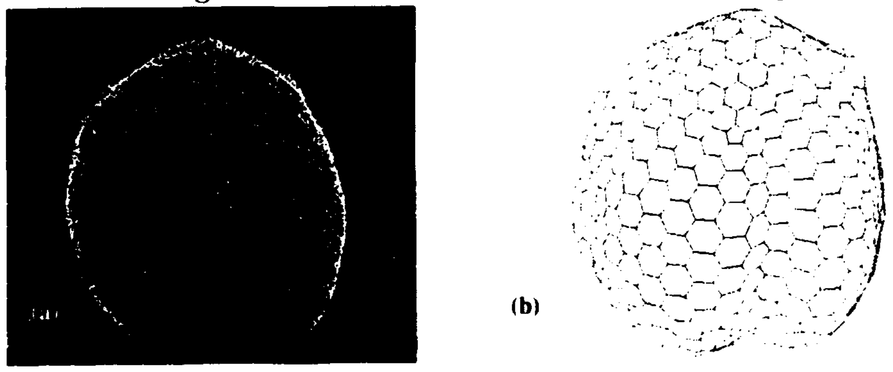


Figure 14: Complete 3-D model; (a) Combined set of data points from registered range data; (b) Surface model.

5. Conclusion

We introduced a new approach for building models of curved objects from multiple arbitrary views. The basic representation is a mesh of nodes on the surface that satisfies certain regularity constraints. We introduced the notion of simplex angle as a curvature indicator stored at each node of the mesh. We showed how a mesh can be mapped into a spherical representation in canonical manner, and how object models can be generated by merging multiple views by computing the transformations among the views.

This approach eliminates two major limitations of conventional model building systems. First, it enables us to convert the matching problem to a straightforward correlation of spherical images. As a result, the approach is able to deal with arbitrary transformations between views and to operate without requiring an initial estimate of the transformation. Second, it does not require any feature extraction or feature matching. As a result, the SAI matching approach can handle general curved surfaces. We have used the SAI as a way to store curvature. However, the concept of SAI is more general because other pieces of information may be stored at each node of the spherical image. For example, albedo or texture could be stored. This appearance information can be used to augment the definition of the distance between SAIs by adding additional terms Adding appearance information will make the approach more effective by providing a more discriminating measure of distance between shapes. Another research direction is in the determination of the best sequence of views to be used for a particular model. In the examples presented in this paper, the number of images was small enough and the overlap between them was large enough that it did not matter in which order the images are processed. In general, however, it is important to compare the pairs of images that are most likely to yield the most accurate matching results.

References

[1]Besl, P., Kay, N.D., A Method for Registration of 3-D Shapes, PAMI-14(2), 1992.

[2]Champleboux, G., Lavallee, S., Szeliski, R., Brunie, L., From Accurate Range Imaging Sensor Calibration to Accurate Model-Based 3-D Object Localization, Proc. CVPR-92, 1992.

[3]Chen, Y., and Medioni, G., Object Modelling by Registration of Multiple Range Images, Image and Vision Computing, 10(3), April 1992.

[4]Delingette, H., Hebert, M., Ikeuchi, K., Shape Representation and Image Segmentation Using Deformable Surfaces, Image and Vision Computing, 10(3), April 1992.

[5]Delingette, H., Hebert, M., Ikeuchi, K., A Spherical Representation for the Recognition of Curved Objects, ICCV'93, Berlin, 1993.

[6]Kamgar-Parsi, B., Jones, L.J., Rosenfeld, A., Registration of Muliple Overlapping Range Images: Scenes Without Distinctive Features, PAMI-13(9), 1991.

[7]Martin, W.N., Aggarwal, J.K., Volumetric Descriptions of Objects from Multiple Views,

PAMI-5(2), 1983.

[8]Parvin, B., Medioni, G., B-rep from Unregistered Multiple Range Images, Proc. IEEE Robotics and Automation, 1992.

[9]Stein, F., Medioni, G., TOSS-A System for Efficient Three Dimensional Object Recognition, Proc. DARPA Image Understanding Workshop, 1990.

[10]Zhang, Z., *Iterative Point Matching for Registration of Free-Form Curves*, Research Report 1658, INRIA, March 1992.