University of Edinburgh
Division of Informatics

Constructing VRML models using a new simple range sensor

4th Year Project Report
Computer Science

Andreas Tsiamis

May 29, 2002

Abstract: Range Imaging offers an inexpensive and accurate method of digitizing the shape of three dimensional objects. A Low cost range sensor has been recently developed at the University of Edinburgh, which acquires range images with colour texture. In order to build build a complete VRML model of an object, multiple views of the object must be taken to achieve total coverage of the surface. The principal goal of the project is to investigate the ease with which the models of everyday objects can be developed. There are several approaches of implementation and the final model depends on the approach.

Acknowledgements

I would like to thank my project supervisor, Bob Fisher for his support both technical and psychological, throughout the year. Furthermore, i would like to thank my family for supporting me in every possible way, despite some problems that have recently occured.

CONTENTS

1. Introduction 3

2. Implementation Stages 5

3. Range Data Acquisition 6

4. Marker Location 8

5. Marker 3D Location 10

5.1 An Introduction to the RANSAC Algorithm 10

6. Marker Correspondence 13

7. Surface Repair 14

8. Initial Registration 16

9. Refined Registration Using the ICP Algorithm 18

9.1 Introduction 18
9.2 Rigid Iterative Closest Point Algorithm 18
9.3 Improvements to the Basic Algorithm 18

10. Triangulation Process 20

10.1 Integration 20
10.2 Creating Triangle Meshes from Range Images 20
10.3 Mesh Zippering 21

11. Colour Map Making 22

12. Conclusion 23

13. References 24

1. Introduction

Range imaging offers an inexpensive and accurate means for digitizing the shape of three dimensional objects. Because most objects self occlude, no single range image suffices to describe the entire object. Here a method for combining a collection of range images into a single polygonal mesh that completely describes an object to the extent that it is visible from the outside, is presented.

Currently, there is no procedure that will allow a user to easily capture a digital description of a physical object. A dream tool would give the ability to someone to have a complete digital description of an object in a few minutes. The reality however is different and much of the digitization is done by a user touching a 3D sensing probe to hundreds or thousands of positions on the object, then manually specifying the connectivity of these points. Fortunately range scanners offer promise in replacing this tedious operation.

A low-cost range sensor has been recently developed at the University of Edinburgh, which acquires range images with colour texture. By just using the range sensor enough information is available to derive a range image. Simultaneously, a colour image can be acquired to create a texture image that can be mapped on to the surface. A *range scanner* is any device that senses 3D positions on an object's surface and returns an array of distance values. A *range image* is a mxn grid of distances (range points) that describe a surface either in Cartesian coordinates (a height field) or cylindrical coordinates, with two of the coordinates being implicitly defined by the indices of the grid. Quite a number of measurement techniques can be used to create a range images, including structured light, time-of-flight lasers, radar, sonar and several methods from the computer vision literature such as depth from stereo, shading, texture motion and focus.

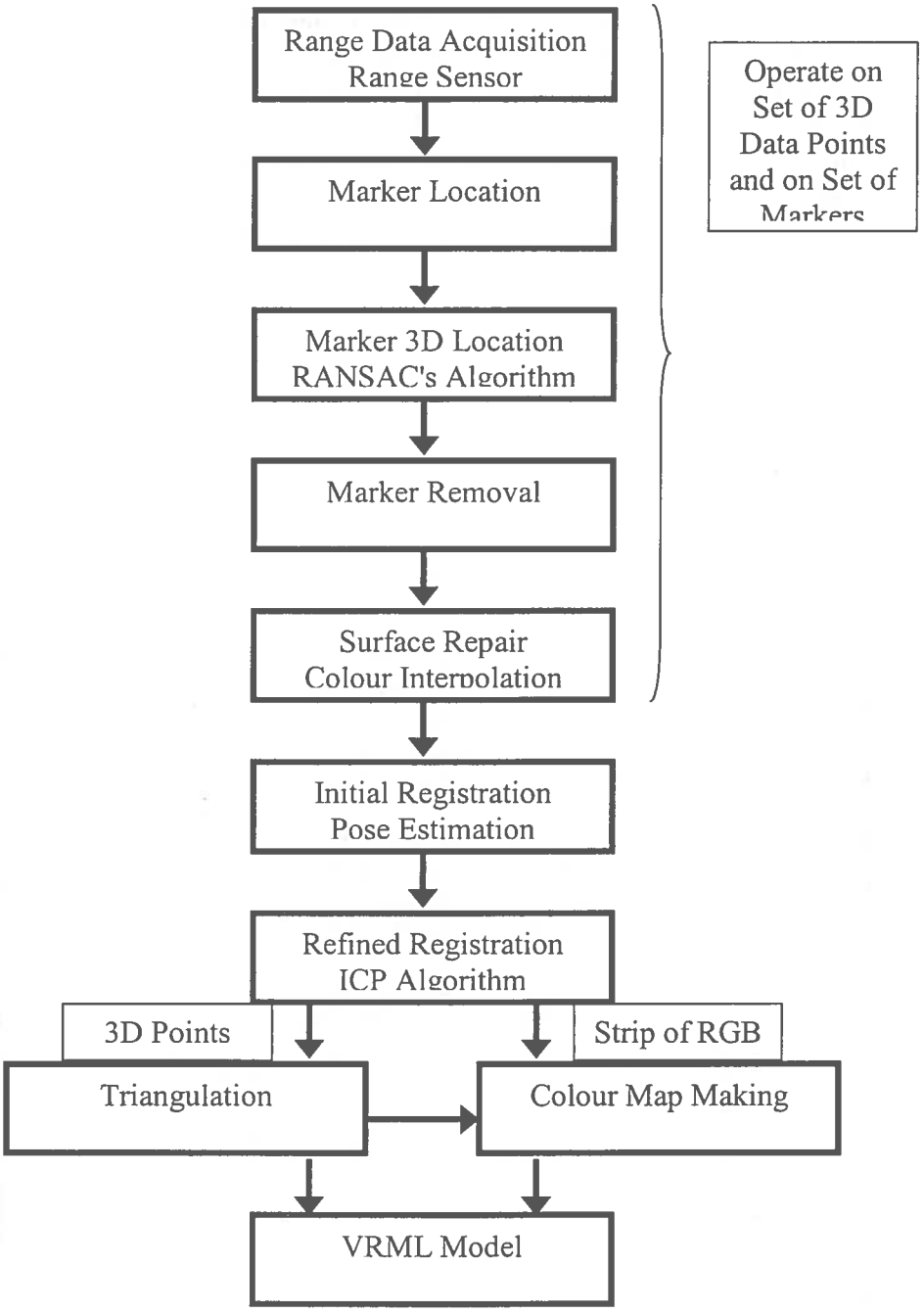
Range scanners seem like a natural solution to the problem of capturing a digital description of physical objects. Unfortunately, few objects are simple enough that can fully described a single range image. For instance a coffee cup handle will obscure a portion of the cup's surface even using a cylindrical scan. To capture the full geometry of a moderately complicated object multiple range images must be acquired in order to achieve total coverage of the surface. The accuracy of the surface shapes from the raw data depends on the implementation of the techniques applied to create the model. The two main issues in creating a single model from multiple range images are: *registration* and *integration*. Registration refers to computing a rigid transformation that brings the points of one range image into alignment with the portions of a surface that it shares with another range image. Integration is the process of creating a single surface representation from the sample points from two or more range images.

The principal goal of the project is to investigate the ease with which complete models of everyday objects can be developed. In more detail the method used here in order to build a complete model involves the implementation of the following stages:

- Acquire range data (with the aid of the range sensor)
- Locate the markers (locate position of reference points)
- 3D location of markers (locate specific markers for each range image)
- Marker correspondence (find identification of markers for each range image)

- Marker removal (remove markers from test object)
- Surface Repair (reconstruct surface area left by marker removal)
- Initial registration (compute set of transformations that aligns the data sets to a common coordinate system)
- Further alignment (further align the data sets according to techniques used)
- Triangulation process (make the texture map of the object)
- Colour map making (get the colour information for each 3D data point)

2. Implementation Stages



3. Range Data Acquisition

Consider a small hippopotamus toy as the test object being scanned by the range sensor. With the aid of the range sensor sets of 3D data points are collected. The range sensor scans the entire object and thus multiple views of the object are being scanned. Each 3D data set represents one of the multiple views of the object. By just using the range sensor enough information is available to derive a range image. Simultaneously a colour image can be acquired to create texture image that can be mapped on to the surface. Figure 1 shows the typical information acquired by the range sensor. This is only a small fraction of the existing 3D points data set. Figure 2 shows the range image corresponding to this data set.

x-coor.	y-coor.	z-coor.	RGB colour info
156.500000	-297.000000	-182.445007	132 88 82
157.000000	-297.000000	-182.455994	146 92 93
157.500000	-297.000000	-182.462006	155 94 99
158.000000	-297.000000	-182.477005	160 96 101
154.000000	-296.500000	-182.455994	118 83 69
154.500000	-296.500000	-182.279999	128 87 82
155.000000	-296.500000	-182.182999	132 91 83
155.500000	-296.500000	-182.147995	128 88 62
156.000000	-296.500000	-182.143005	136 92 84
156.500000	-296.500000	-182.147995	145 98 96
157.000000	-296.500000	-182.169998	146 96 100
157.500000	-296.500000	-182.177002	149 94 98
158.000000	-296.500000	-182.195999	162 101 110
158.500000	-296.500000	-182.235992	164 96 94
159.000000	-296.500000	-182.281006	167 101 98
159.500000	-296.500000	-182.401993	170 101 97
153.500000	-296.000000	-182.240997	119 79 64
154.000000	-296.000000	-182.270004	122 83 70
154.500000	-296.000000	-182.100998	131 90 82
155.000000	-296.000000	-182.033997	133 92 82
155.500000	-296.000000	-182.011993	133 89 70
156.000000	-296.000000	-182.024002	143 94 83
156.500000	-296.000000	-182.031998	144 92 88
157.000000	-296.000000	-182.061005	147 94 86
157.500000	-296.000000	-182.078003	151 98 87
158.000000	-296.000000	-182.085007	155 98 84
158.500000	-296.000000	-182.126007	159 96 78

Figure 1: Information extracted with range scanner. A set of 3D data points and a set of RGB colour points.

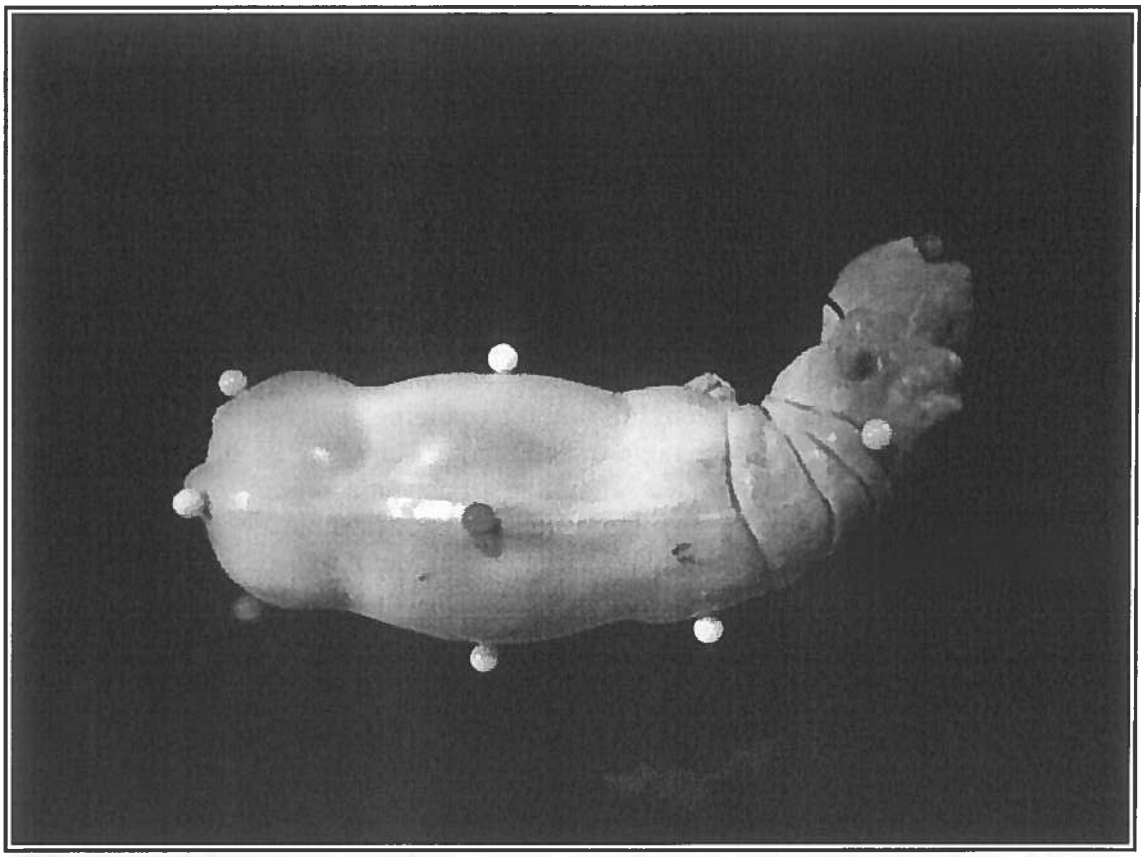


Figure 2: Range image represented by a set of 3D data points and set of RGB colour points

4. Marker Location

Each object being scanned has a set of markers placed on it. In this representation the markers are some plastic spheres. Consider Figure 3, which is one of the range images representing on the views of the hippopotamus. Starting from left to right the identifications of the markers are the following: 1,2,8,9,5,7,3,10,11,6,4. Each marker is identified with a unique number. For this test object fifteen markers (although not all of them are visible from this view) have been placed and are many enough to cover the entire surface of the object.

The center of each marker is located and thus its position on the test object. Further more, the radius of each marker is calculated. This need in order to find out which data points on the object are covered during the scanning. The radius of each marker has to estimated separately since none of the markers are exactly the same and thus can not be considered as perfectly similar. Each marker is given a unique identification in order to recognise its position on the test object. These markers as used as reference points and going to be needed at the initial registration stage.

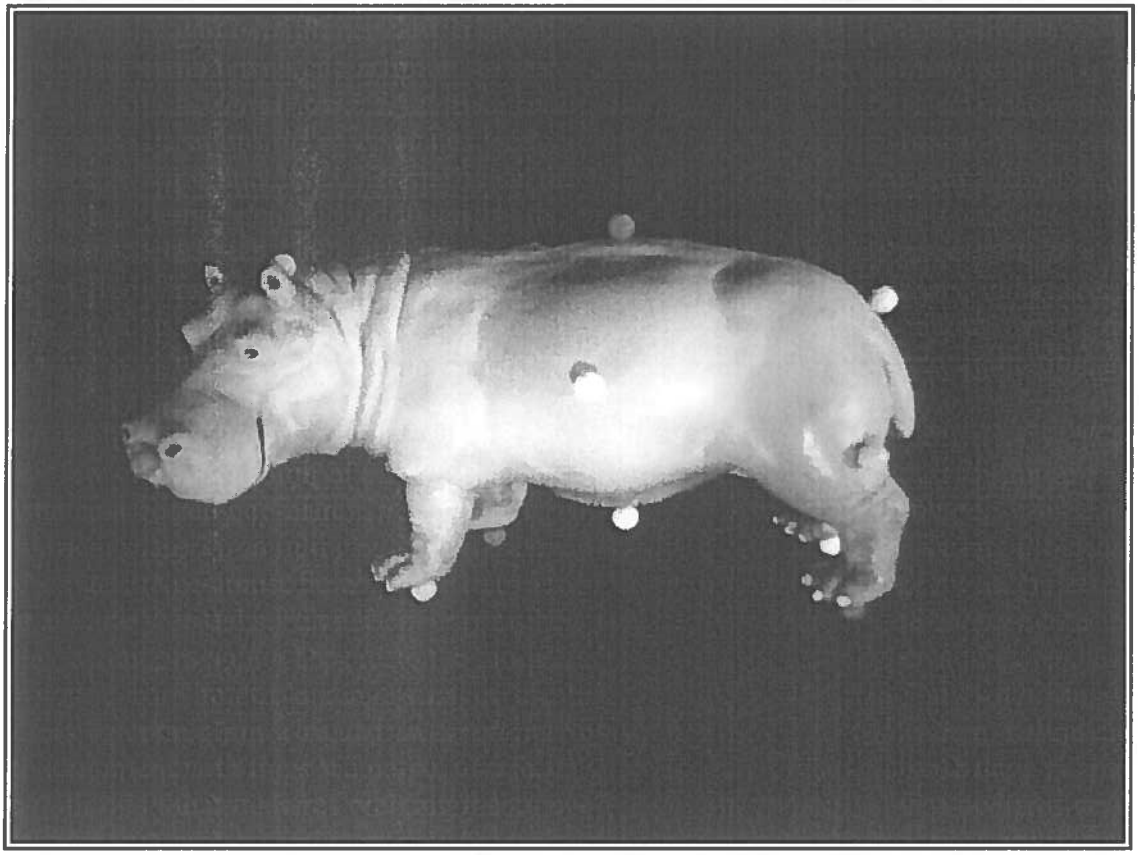


Figure 4: Image view showing marker positions

At the marker location stage the markers existing in each separate image view are identified. Thus for hippopotamus test object the markers for each of the seven different views are:

SCAN NO	MARKERS LOCATED IN CURRENT IMAGE VIEW

1	2	3	4	5	14					
2	1	4	5	6						
3	5	6	7	8	9	10	11	13	14	15
4	2	3	7	8	9	10	12	13	14	15
5	1	2	3	4	5	6				
6	4	6	10	11	12					
7	2	4	15							

5. Marker 3D Location

5.1 An Introduction to the RANSAC Algorithm

The Random Sample and Consesus (RANSAC) algorithm introduced by Fischler and Bolles [1] searches for suitable solutions directly using the data, repeatedly constructing solutions from randomly sampled minimum subsets, which are not related to any concept of an error surface and thus are not restricted to either an assumption of smoothness or the computational form of the objective function. In contrast to most optimization algorithms which attempt to maximize the quantity of data used to identify a solution, RANSAC is expected to identify solutions computed from outlier free data. To ensure this is the case the objective function used with RANSAC must be robust to outlier data and so the use of a least square metric is unsatisfactory. The scope of applicability of the algorithm is great and potentially encompasses algorithms based on optimisation with or with out the use of local derivatives. The use of RANSAC as a wrapper around closed form solutions to vision problems gives potential scope for utility to these otherwise non-robust approaches. RANSAC gives us the opportunity to evaluate any estimate of a set of parameters no matter how robust or accurate the method that generated the solution might be. The RANSAC method could thus be considered as an ideal approach to the solution of many machine vision problems. However, the random nature of the search makes direct use of RANSAC as optimisation algorithm inefficient.

RANSAC can be regarded as an optimisation algorithm since, given an objective function it can determine the parameter set which best positions the model. Provided that the data and the number of attempts are sufficiently large it is expected that model position identified by RANSAC is close to the global solution. RANSAC does not use any local information about the objective function. Instead it samples the objective function at discrete locations by randomly sampling minimum subsets of data and generating candidate solutions which are then evaluated. Consequently either the initialiasation of the algorithm (which is random) or the local smoothness of the objective function (of which RANSAC has no concept) does not restrict solutions.

Traditional RANSAC is often used to as means of identifying the inlier datapoints, which are often used in a more conventionally optimisation strategy to find the optimal minima. This is because RANSAC itself has no mechanism for searching around data-borne solution. In the RANSAC with fusion algorithm (RANSAC-f) [2] an ordered list is maintained, with the best n solutions evaluated by RANSAC. After a number of iterations this list will be populated with outlier free solutions. However, the longer the list is maintained the less likely it is that new sampled results will be competitive enough to replace the top ranked solution. Therefore, once the list is populated with good solutions the algorithm begins combining results in order to generate improved solutions. When a solution is ranked high enough to enter the list an attempt is first made to combine it with the solutions already present. Starting with the top of the list the parameters from both solutions are averaged and a new solution is found. This solution is then evaluated using the same method as the data point solutions. If the score from the fused solution is better than both the new and the listed solution then the recombined result is place in a second list and the listed element is removed. before a solution is entered into this second list an attempt is made to

combine it with any solutions already present. Again if the combined result is better than both the source data results then this solution is entered into a third list and so on until no further combination is possible. At any point, if a combination solution is no better than both source solutions it is simply placed in the current list at its ordered location.

The performance of the RANSAC-f algorithm is noticeably improved over the standard RANSAC approach, often achieving comparable results with a smaller computational cost. Although the delay in the fusion process has a small impact on the residual error, it has a significant impact on the rate of decay. RANSAC is commonly utilized to identify an outlier free subset of data, This only possible if a robust objective function is used and the algorithm is allowed to iterate for sufficient cycles. The traditional method of determining the termination point requires the proportion of outliers present in the data, information which is not commonly available. The variant of RANSAC algorithm maintains ordered lists of competitive solutions. The lists are used to uncover solutions which are not available to traditional RANSAC by combining the parameters of previous good solutions. This has been confirmed to be an effective technique in order to overcome the O(m) dependency of the traditional RANSAC algorithm.

5.2 RANSAC Algorithm in Implementation

RANSAC algorithm is used here to get the 3D location of the markers in each image view of the object. More specificaly the algorithm gives an output of the (x,y,z) coordinates of the center of the marker sphere. It addition an extra information of the radious (in mm) of each sphere is given.

The algorithm takes as input a data set of (x,y,z) coordinates for a specific image view and outputs (x,y,z) coordinates representing the centre of spheres. A typical output of an image view centre coordinates would be that of figure (3). Thus after executing RANSAC algorithm for each image view we know which 3D points from the data set correspond to a centre of a marker. Having the information of the marker radius we later on find out which 3D data points are cover by the markers and thus for which points surface information is missing.

Marker Positions for Image View No 1

x-coor.	y-coor.	z-coor.	radious
164.418915	-268.496704	-164.971390	2.704491
222.527695	-282.413208	-182.444702	2.526936
189.393631	-237.287262	-181.978455	2.546454
184.078522	-259.892670	-165.120117	2.787332
155.433258	-272.930267	-167.913925	2.743528

Marker Positions for Image View No 2

x-coor.	y-coor.	z-coor.	radious
67.673096	-266.294037	-247.876587	2.380249
21.278929	-241.597000	-214.983536	2.762125
110.363976	-273.565125	-271.993530	2.502250
122.232414	-259.391052	-247.935074	2.537578

All numbers are in mm

Figure (3): (x,y,z) coordinates representing centre of spheres and radius of sphere in mm

6. Marker Correspondence

Out of each image view on the test object we got a set of 3D points which correspond to the centers of the markers for the specific image view. This was implemented in the Marker 3D Location stage. Initially we don't know which point represents which marker center on the test object. Thus we need to give a unique identification to each 3D data point; ie we need to know which 3D data point corresponds to which marker center.

The implementation proceeds in the following way: for each view the distances between the markers on the test object are calculated. Then the distances between the 3D data points from the output files (out of the Marker 3D Location stage) were calculated. However, we already know which markers correspond for each view. Thus the need here was to compare relative distances between the set of 3D data points and between the set of markers, which belonged to the same view. Thus comparing the distances between sets of 3D data points and sets of markers gives a 3D point-to-marker correspondence. For each set of comparisons a common coordinate system was needed. Thus the 3D point being the center of the camera was considered to be the origin of the common coordinate frame. The centers of the camera for each individual scan are shown in the following table:

SCAN NO	CAMERA CENTER		
1	211.768	-277.00	54.572
2	226.768	-299.00	69.572
3	201.768	-279.00	74.572
4	201.768	-269.00	69.572
5	151.768	-293.00	74.572
6	181.768	-291.00	114.572
7	196.768	-281.00	99.572

7. Surface Repair

Once the markers have been removed from each image view there are some areas left, which contain no information. These are the areas originally covered by the markers. As we can see in figure (4) the information that existed in the black area and was represented in the 3D data set concerned information about the marker surface and not the test object surface.

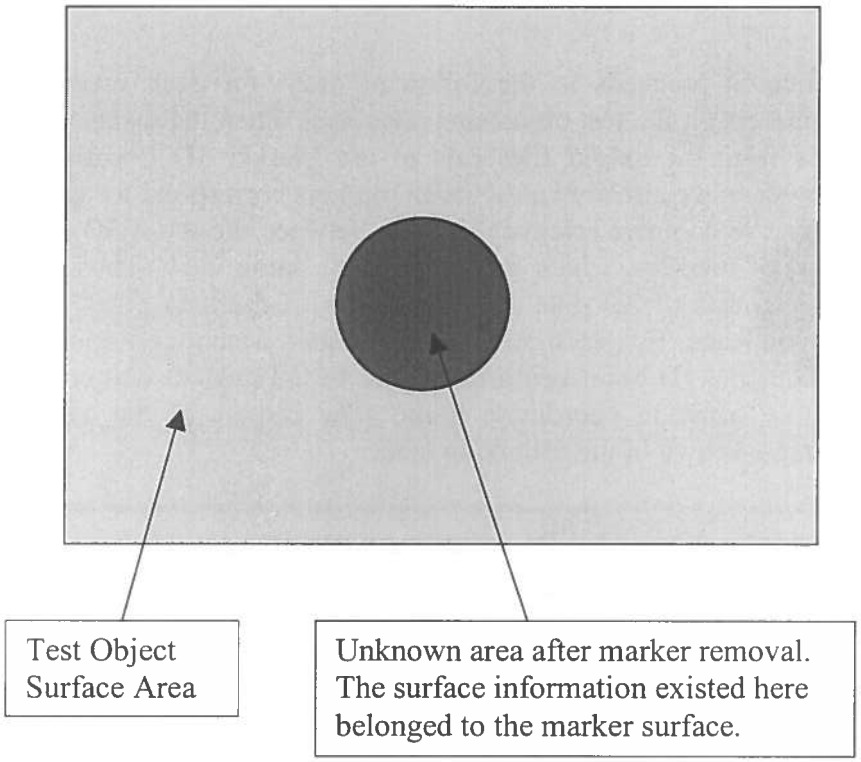


Figure (4): Unknown surface area needed to be filled with information matching the one represented by the surrounding area of the test object

The surface repair process will not give the original information of how the areas covered by the markers were originally. It will use the texture already existing on the rest of the plane and will try to match these areas with the plane. Thus the local plane needs to be estimated and the markers points need to be projected onto the plane. To fill the surface the colour form the local plane is going to be used. A method to proceed with is the following:

The idea is to fill the unknown areas with 3D points that lie on a plane.
Let $\{x_i\}$ be the set of points on the marker.
Let $\{y_i\}$ be the set of pints with in a distance D from the marker. A good distance is about 2mm.
1. Fit a plane to $\{y_i\}$.
2. Project $\{x_i\}$ onto the plane.
This will give a set of sample points $\{y_z\}$ at about the right density and lying close to where the unobserved surface ought to be.

We also need to estimate a colour for each new point.
Let $\{c_i\}$ be the RGB colours of the points $\{y_i\}$.
Let $\{Z_j\}$ be a 3D reconstructed point whose colour we are trying to estimate. The following equation:

$$\sum c_i / |z_j - y_i| \Big/ \sum 1 / |z - y_i|$$

will interpolate the colours. It weights the colours of the observed points by the distance from the reconstructed point. If the colours are blurred more that the normal squaring the distances in the equation will reduce the blur. When the distances are squared nearby colours are more emphasized. This colour reconstruction works well if the colour is smoothly varying across the area. If there is a lot of texture nearby the reconstruction will not work very well.

8. Initial Registration

This stage of the model reconstruction deals with the registration of range images into a common coordinate frame from the initial estimate of the pose. Pose estimation is the process of determining the translation and rotation of an object in one coordinate frame with respect to another coordinate frame. A common need in machine vision is to compute the 3D rigid transformation that exists between two sets of points for which corresponding pairs have been determined. The algorithm used here is considered as one of the most popular and it computes the solution to the following problem.

Let $\{m_i\}$ and $\{d_i\}$, i = 1...N be two corresponded point sets, such they are related by: $d_i = Rm_i + T + V_i$ where R is a standard 3x3 rotation matrix, T is a 3D translation vector and V_i a noise vector. Solving for the optimal transformation $[\hat{R}, \hat{T}]$ that maps the set $\{m_i\}$ onto $\{d_i\}$ typically requires the least squares error criterion:

$$\Sigma^2 = \sum_{i=1}^{N} \left\| d_i - \hat{R} - m_i - \hat{T} \right\|^2 \qquad (1)$$

The algorithm used here was developed by Arun, Huang and Blostein [3] and is based on computing the singular value decomposition (SVD) of a derived matrix. The transformation rotation is represented using a standard 3x3 orthonormal matrix, while translation is a 3D vector as in the equations above. The solution of this method is the following. After noticing that, the point sets should have the same centroid at the correct solution, the rotation component is found first by analysing point sets after their translation to the origin. A 3x3 correlation matrix given by

$$\mathrm{H} = \sum_{i=1}^{N} m_{c,i} d_{c,i}^T \qquad (2)$$

is computed based on these new centered point sets. Its singular value decomposition, $H = U \wedge V^T$, is determined. The optimal rotation matrix is then $\hat{R} = VU^T$. This computation is also known as the orthogonal Procrustes problem, the SVD solution has been known for some time [2]. The optimal translation is found as that which aligns the centroid of the set $\{d_i\}$ with the centroid of the optimal set $\{d_i\}$, that is $\hat{T} = \bar{d} - \hat{R}\bar{m}$.

The time complexity of the algorithm is of order O(N). Considering the of the linear term of the time equation, the algorithm initially spends time computing the sums for the correlation matrix. When considering the constant terms it computes the singular value decomposition of a 3x3 matrix followed by matrix multiplications and determinant operations.

A MATLAB pose estimation code in Figure (5) estimates the 3D transformation (3 rotation angles, 3 translation coordinates) from sets of matching 3D points.

```
% solves for the 3x3 rotation matrix R and 3x1 translation vector T given 3*n
% (n>2) matrices Y and X of corresponding points ideally stisfying Y = R*X + T
function [R,T] = solvepose(X,Y)

    % solve for position
    Ymean = mean(Y')';
    Xmean = mean(X')';
    K = (Y - Ymean*ones(3,1)')*(X-Xmean*ones(3,1)')';
    [V,D,U] = svd(K);
    D = eye(3,3);
    D(3,3) = det(V,*U');
    R = V*D*U';
    T = Ymean - R*Xmean;
```

Figure (4): Pose estimation code based on algorithm with SVD approach

9. Refined Registration Using the ICP Algorithm

9.1 Introduction

A popular method of refining a given registration is the iterative closest point (ICP) algorithm, firstly introduced by Besl and Mckay [5]. This iterative alignment algorithm is relatively straightforward. Given a motion transformation that initially aligns two data sets to some degree, a set of correspondences is developed between features (usually points) in one set and the next. This is done in the following way: for each point in the first data set, pick the point in the second which is closest to it under the current transformation. From this set of correspondences, an incremental motion can be computed which further aligns these points to one another. This find correspondence/compute motion process is iterated until some convergence criterion indicating proper alignment is satisfied.

Although simple the algorithm works quite effectively when given a good initial estimate. However, there are two major drawbacks. The first is that it is not obvious how the two set approach can be extended to handle multiple data sets. The second is that proper convergence only occurs if one of the data sets is subset of the other.

More precisely, the point-matching algorithm is:

9.2 Rigid Iterative Closest Point Algorithm

Let S be a set of N_S points $\{s_1,...,s_{Ns}\}$ and M the model. Let $\| s - m \|$ be the Euclidean distance between point $s \in S$ and $m \in M$. Let CP(s,M) be the closest point in M to the scene point s.

1. Let $T^{[0]}$ be an initial estimate of the rigid transformation.
2. Repeat for $k = 1..k_{max}$ or until convergence:
   (a) Compute the set of correspondences C =

$$\bigcup_{i=1}^{Ns} \{(s_i, CP(T^{[k-1]}(s_i), M))\}$$

   (b) Compute the new Euclidean transformation $T^{[k]}$ that minimizes the mean square error between point pairs in C.

9.3 Improvements to the Basic Algorithm

Many of the improved algorithms attempt to get a better correspondence between the data sets. Incorrect correspondences could lead to the generation of non-optimal transformations. The cause of an incorrect correspondence is the existence of points in a data set, which are not in the other. One attempt to improve the algorithm changes the point-to-point correspondence, to one between a point in a data set and a location in the represented surface in the other data set. Such effort was first made by Chen and Medioni [6]. The approach they followed is: the starting location was found as the data point in the second set that is closest to a line through the first point in the

direction of its estimated surface normal. The tangent plane at this intersection point is used as the surface approximation. The initial point is projected onto this plane to give the corresponding location.

Such an improvement as the above does not deal with the issue of data sets that are not subsets of another. A solution to this problem involves imposing a heuristic threshold on either the distance allowed between points in a valid pair or the deviation of the surface normals of corresponding points. Any pairs of points with distance greater than the threshold are assumed to be incorrect. The thresholds are constants related to the estimated accuracy of the initial transformations. Zhang [9] has computed an adjustable threshold based on the distribution of the distance errors at each iteration.

All the techniques discussed up to now are designed to work with two data sets. Thus if someone wants to merge multiple images he has to consider the approach of examining the sequence in pairs. This method however, accumulates any errors in the computations made, leaving the first and the last data set of the sequence poorly aligned. One method that has searched for a more globally optimal set of transformations is the following. Introduced by Turk and Levoy [8], it assumes that an additional continuous cylindrical data scan is available. Individual linear scans were registered to this, image which should have commonalities with each of them. Many global optimisation techniques have been developed, but they are still rely on pairwise correspondence computations.

In all of the techniques, computing potential correspondences is generally the most time consuming step. In brute-force approach an $O(N^2)$ number of comparisons is performed to find n pairs. A way to reduce the actual time is to subsample the original data sets. An alternative is to use the full original data sets, but organise the search using mire efficient data structures such as the k-d tree [1]. The k-d tree is even efficient O(NlogN), when higher order features of the points are incorporated in the distance metric.

10. Triangulation Process

10.1 Integration

Integration of multiple range scans can be classified in to structured and unstructured methods. Unstructured integration presumes that one has a procedure that creates a polygonal surface from an arbitrary collection of points in 3D-space. Integration in this case is performed by collecting together all the range points from multiple scans and presenting them to the polygonal reconstruction procedure. The Delaunay triangulation of a set of points in 3D-space is the basis of such a reconstruction method [Boissonnat 84]. Other candidate for such surface reconstruction is a generalisation of the convex hull of a point set, known as alpha shape [Edelsbrunner 92]

Structure integration methods make use of information about how each point was obtained, such as using error bounds on a point's position or adjacency information between points within one range image. A typical method [Soucy 92] is the following. Given n range images of an object, first partition the points into a number of sets that are called common surface sets. The range points in one set are then used to create a grid of triangles whose positions are guided by a weighted average of the points in the set. Subsets of these grids are stitched together by a constrained Delaunay triangulation in one of n projections onto a plane.

10.2 Creating Triangle Meshes from Range Images

To represent the range image data at all stages we use a mesh of triangles [8]. Each sample point in the mxn range image is a potential vertex in the triangle mesh. We need to take special care to avoid joining portions of the surface together, that are separated by depth discontinuities (Figure (6)).

To build a mesh we create zero, one or two triangles from four points of a range image that are in adjacent rows and columns. We find the shortest of the two diagonals between the points and use this to identify the two triplets of points that may become triangles. Each of these point triplets is made into a triangle of the edge lengths fall below a distance threshold. Let s be the maximum distance between adjacent range points when we flatten the range image, that is, when we don't include the depth information (Figure (6)). We take the distance threshold to be a small multiple of this sampling distance, typically 4s. Although having such a distance threshold may prevent joining some range points that should in fact be connected, we can rely on other range images (those with better view of the location in question) to give the correct adjacency information.

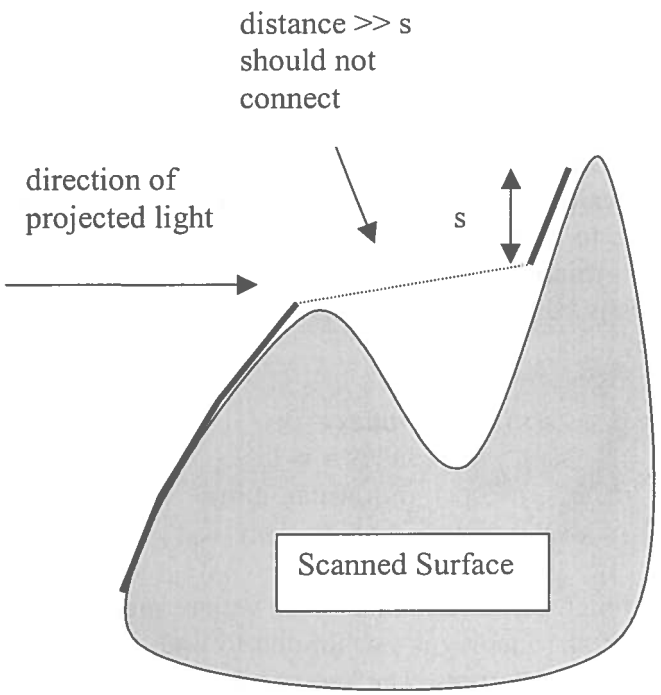


Figure (6): Buiding a triangular mesh from the range points

10.3 Mesh Zippering

The central step in combining range images is the integration of multiple views into a single object. The goal of integration is to arrive a description of the overall topology of the object being scanned. The full topology of a surface is realized by zippering new range scans one by one into the final triangle mesh Zippering two triangle meshes consists of the following steps.

- Remove overlapping portions of the meshes.
- Clip one mesh against the other.
- Remove the small triangles introduced during clipping.

11. Colour Map Making

Colour mapping is a common scalar visulalisation technique. The technique maps the scalar data value to a representative colour. The mapping is usually done via colour lookup table and the scalar values serve as indices into that table. The algorithm is the following. If the values are to be mapped to n colours, then we must determine the dynamic range (maximum - minimum) of the scalar data. If the index associated with the minimum scalar values is zero and the associated with maximum scalar value is n-1 the following will hold;

$$s(x) < \min, \text{index} = 0;$$
$$s(x) > \max, \text{index} = n-1;$$
$$\text{index} = n*(s(x) - \min)/(\max-\min);$$
$$s(x) \rightarrow \text{colour-table (index)}$$

Colour mapping is a one-dimensional technique since we are mapping a scalar value to a single colour. However, the display is not limited to one dimension and can be extended to two or even three dimensions. The key to colour mapping is the choice of the lookup table. The table should accentuate the important features, while minimising the less important differences

12. Conclusion

Range imaging is an inexpensive and accurate method for digitizing the shape of three-dimensional objects. The accuracy of the surface shapes from the raw data depends on the implementation of the techniques applied to create the model The techniques involved here were discussed at the implementation stages. Some of the techniques involved here are commonly used and approved due to their efficiency or their ease of use. According to the complexity and efficiency of the techniques one will get an accurate or less accurate surface shapes. The methods approached here give accurate surface shapes to a certain extend.

13. References

[1] Fischler MA, Bolles RC, "Random Sample Consesus: A Paradigm for Model Fitting With Application to Image Analysis and Automated Cartography", Communication Association and Computing Machine, 24(6), pp 1981, 381-395.

[2] Lacey AJ, Pinitkarn N, Thacker NA, " An Evaluation of the Performance of RANSAC Algorithms for Stereo Camera Calibration"

[3] Arun KS, Huang TS, Blostein SD, "Least-Squares Fitting of Two 3D Point Sets", IEEE Transactions of Pattern Analysis and Machine Intelligence, 9, 5, 1987, pp 698-700

[4] Schonemann P, "A Generalized solution of the Orthogonal Procrustes Problem", Psychometrica, 31, 1966, pp 1-10.

[5] Besl PJ, Mckay ND, "A Method for Registration of 3D Shapes", IEEE Transactions on Pettern Analysis and Machine Intelligence Vol 14 No 2 (1992), pp 239-256.

[6] Chen Y, Madioni G, "Object Modelling by Registration of Multiple Range Images".

[7] Zhang, Z "Iterative Point Matching for Registration of Free-form Curves and Surfaces", International Journal of Computer Vision Vol 13 No 2 (1994), pp 119-152.

[8] Turk G, Levoy M, "Zippered Polygon Meshes from Range Images", Proceedings of SIG-GRAPH 94 Orlando, FL (Jul 1994), pp311-318

[9] Preparata F, Shamos M, " Computational Geometry: An introduction Springer Verlag" 1985.