in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2003, vol. 1, pp. 858-865.

3D Modeling Using a Statistical Sensor Model and Stochastic Search

Daniel F. Huber Martial Hebert

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

Accurate and robust registration of multiple three-dimensional (3D) views is crucial for creating digital 3D models of real-world scenes. In this paper, we present a framework for evaluating the quality of model hypotheses during the registration phase. We use maximum likelihood estimation to learn a probabilistic model of registration success. This method provides a principled way to combine multiple measures of registration accuracy. Also, we describe a stochastic algorithm for robustly searching the large space of possible models for the best model hypothesis. This new approach can detect situations in which no solution exists, outputting a set of model parts if a single model using all the views cannot be found. We show results for a large collection of automatically modeled scenes and demonstrate that our algorithm works independently of scene size and the type of range sensor. This work is part of a system we have developed to automate the 3D modeling process for a set of 3D views obtained from unknown sensor viewpoints.

1 Introduction

Modeling-from-reality is the process of creating digital three-dimensional (3D) models of real-world scenes from 3D views as obtained, for example, from range sensors or stereo camera systems [9][12][2][1][15]. Such sensors typically capture the 3D structure of a scene from a single viewpoint, so multiple views must be combined to obtain a complete model. Accurate and robust automatic registration of multiple 3D views is crucial in our goal of automating the modeling-from-reality process. Formally, we want to solve the following problem: Given an unordered set of overlapping 3D views of a static scene, recover the original sensor poses[1], thereby aligning the data in a common coordinate system. We call this problem *multi-view surface matching* because it can viewed as an extension of pair-wise surface matching to more than two views [8]. We do not assume any prior knowledge of the sensor viewpoints or even which views contain overlapping scene regions. Furthermore, the views are unordered, meaning that consecutive views are not necessarily close together spatially. Multi-view surface matching is analogous to assembling a 3D jigsaw puzzle. The views are the puzzle pieces, and the problem is to correctly assemble the pieces without even knowing what the puzzle is supposed to look like.

In [8], we introduced the multi-view surface matching problem, developed a framework for solving the problem, and demonstrated a system that used the framework to automate the modeling from reality process. In this paper, we build upon that work, adding two significant contributions. First, we develop a framework for evaluating the quality of model hypotheses and pair-wise matches. We use maximum likelihood estimation to learn a probabilistic model of registration success. This method offers several advantages over traditional registration metrics, including superior discriminability and a principled way to combine multiple, disparate measures of registration accuracy. Second, we develop a stochastic algorithm to robustly search the large space of possible models for the best model hypothesis. This new approach is more expressive than our previous method. It can detect situations in which no single model using all the views exists, outputting a set of correct model parts rather than an incorrect single-part solution. Additionally, we describe quantitative evaluation methods for validating our algorithms and show results for a large set of automatically modeled scenes. We also demonstrate that our algorithm works independently of the scene size and type of range sensor.

1.1 Previous work

Except for [8], existing modeling from reality systems are not automated or are automated under restrictive condi-

[1]The original sensor poses can be determined only up to a rigid body transform. In practice, we express the poses with respect to an arbitrarily selected input view.

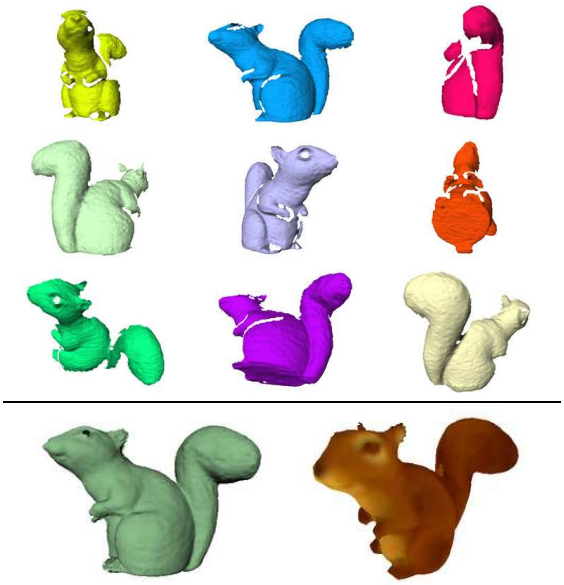

Figure 1. Top: Input views for the squirrel test object (odd-numbered views from a set of 18 views). Bottom: Automatically modeled object using the 9 input views (left) and using all 18 views and adding texture mapping (right).

tions. Such systems register the input views either through calibrated pose measurement (e.g., the scene is rotated on a turn-table) or manual registration. Manual registration can be partially automated by using a surface matching engine to register user-specified view pairs with no initial relative pose estimate and then manually verifying the results [10][1]. If the view ordering is known and the motion between views is sufficiently small, registration algorithms that rely on local search may be used instead [14][16].

These methods have significant limitations. Pose measurement systems limit the scale of scenes that can be modeled; a full-scale building exterior cannot be modeled with a turn-table system. Manual registration is tedious for models with large numbers of views, which limits the scalability and commercial viability of modeling from reality. Our system eliminates these restrictions, allowing fully automated 3D modeling of scenes at a variety of scales without special-purpose hardware.

1.2 Overview of approach

Our automatic modeling from reality system takes as input a set of N views ($V_i, i \in 1 \ldots N$) and outputs a reconstructed model of the original scene, or a set of model parts if a single model using all the views cannot be found (figure 1). The surface S_i for view V_i is represented as a triangular surface mesh in the sensor's local coordinate frame. Our multi-view surface matching algorithm registers the views, producing a model hypothesis that partitions the input views into one or more sets (*parts*) and outputting the absolute poses for the views within each part. We use Neugebauer's multi-view registration algorithm to simultaneously register the views of each part in the output hypothesis [13], and we use Curless' VRIP system to perform surface reconstruction on each registered part [5].

Our multi-view surface matching algorithm begins by registering all pairs of views using a pair-wise surface matching algorithm. Pair-wise surface matching aligns two surfaces when the relative pose between them is unknown. We use a modified version of Johnson's algorithm for this purpose [11][10]. The matches produced by surface matching are improved using a pair-wise registration algorithm. Pair-wise registration aligns two surfaces assuming a good initial estimate of the relative pose is known (e.g., the iterative closest point (ICP) algorithm and more recent variations [3][6]). We use Neugebauer's multi-view registration algorithm for pair-wise registration [13]. Hereafter, we assume that pair-wise surface matching incorporates pair-wise registration as a final step.

If a pair of views contains overlapping scene regions, pair-wise surface matching often finds the correct relative pose, but it may fail for a number of data-dependent reasons (e.g., not enough overlap or insufficient complexity of the surfaces). Even if the views do not overlap, surface matching may find one or more plausible, but incorrect, matches. Multi-view surface matching would be greatly simplified if we could determine with certainty which pair-wise matches were correct. However, this cannot be accomplished just by looking at pairs of views. Two views could have zero registration error but still be an incorrect match, and the mistake may be detectable only indirectly through a sequence of other matches. To solve this problem, we must examine a network of views. We call such a network a *model graph*, and it contains a node for each input view. We insert the matches from pair-wise surface matching as edges into a model graph G_{LR} (LR stands for local registration). If we can find a connected sub-graph of G_{LR} that contains only "correct" matches, it is straightforward to convert the relative poses associated with the matches into absolute poses for each view. Unfortunately, such a connected sub-graph may not exist within G_{LR}. This could happen if one of the input views is corrupted or if some subset of the input views does not overlap sufficiently with any of the remaining views (e.g., front views and back views of an object, but no side views). Our algorithm handles these situations gracefully by searching the space of *all* sub-graphs of G_{LR} for the best model (according to our model quality measure) rather than only searching for connected sub-graphs.

We have therefore reduced the multi-view surface match-

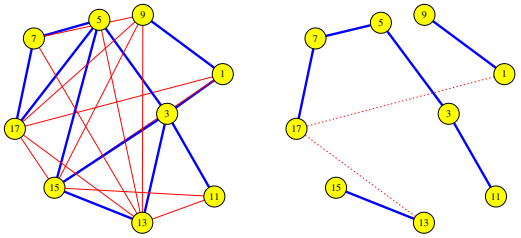

Figure 2. Left: An example model graph. G_{LR} for the views in Figure 1. Matches are labeled as correct (thick/blue lines) or incorrect (thin/red lines) for illustration. Right: An example model hypothesis with three parts. Hidden edges (dashed lines) indicate divisions between parts, which are connected by visible edges (solid lines).

ing problem to two fundamental questions: 1) What constitutes a "good" model? and 2) How do we robustly and efficiently search for the best model in an extremely large hypothesis space? We answer the first question in section 2, learning a probabilistic model of pair-wise registration quality (Q_{L}) from training data. We use this local quality measure to define a global quality measure (Q_{G}) for entire models. In section 3, we answer the second question, using an iterative edge swapping algorithm to maximize Q_{G}. But first, we must cover some background definitions.

Overlap – A point $\boldsymbol{p}$ overlaps surface S if three conditions hold: 1) the closest point distance D_{CP} ($D_{\mathrm{CP}}(\boldsymbol{p}, S) = \min_{\boldsymbol{q} \in S} \|\boldsymbol{p} - \boldsymbol{q}\|$) is less than a threshold t_{D}; 2) the point $\boldsymbol{q}$ on S closest to $\boldsymbol{p}$ is an interior (non-boundary) point of S; and 3) the angle between the surface normals at $\boldsymbol{p}$ and $\boldsymbol{q}$ is less than a threshold t_θ. The region $\mathcal{R}$ on surface S_i that overlaps surface S_j is called the *overlapping region* of S_i.

Model graph – A model graph is an attributed undirected graph $G = (N, E, A, B)$ that encodes the topological relationship between overlapping views (figure 2, left). G contains a node N_i for each input view V_i. The attributes A_i for node N_i consist of the absolute pose T_i for view V_i. The attributes $B_{i,j}$ for edge $E_{i,j}$ include the relative pose, $T_{i,j}$, between V_i and V_j, and the registration quality Q_{L} for the view pair. An edge E_{ij} in G indicates that S_j, transformed by $T_{i,j}$, overlaps S_i (i.e., they have a non-null overlapping region). The relative pose between two connected, non-adjacent, views V_i and V_j can be computed by composing the relative poses along any path from N_i to N_j in G.

Model hypothesis – A model hypothesis is a model graph that is a spanning tree of G_{LR} with an additional edge attribute called visibility (figure 2, right). If an edge is hidden (visibility = false), it separates the model into two parts at that point. Visible edges connect the views within each part. This hypothesis representation covers the space of all acyclic sub-graphs of G_{LR}. Restricting our representation to be acyclic simplifies our algorithm without eliminating any correct solutions. This is because we implicitly re-establish omitted correct matches when computing the global quality of a hypothesis.

2 Evaluating model quality

Given a model hypothesis H, we want to define a function $Q_{\mathrm{G}}(H)$ such that the hypothesis that maximizes Q_{G} corresponds to the correct solution to the multi-view surface matching problem. First, we look at the simpler problem of computing the local quality (Q_{L}) of a pair of registered views.

2.1 Maximum likelihood local quality model

The concept of local quality is related to the verification process present in most 3D object recognition systems. At some point, such a system must decide whether the hypothesized match is "good enough." Typically, this is done by thresholding some statistic of the data or of features derived from the data. The question is what statistic/features should be used and how to set the threshold. The answers depend on the sensor and the scene characteristics. For example, two surfaces that are ten centimeters apart might indicate a good registration for terrain observed from twenty meters away but not for a desktop object seen from one meter.

Rather than relying on fixed, user-defined thresholds, which tend to be brittle, we explicitly model the behavior of the system for a given sensor and scene type, using supervised learning to determine a statistical model of local quality from training data. The benefit of this approach is that we do not need detailed knowledge of the sensor or even of the surface matching and pair-wise registration algorithms. Although we present this method in the context of multi-view surface matching, the idea has broad applicability. For example, a 3D recognition algorithm could employ an analogous model of registration quality to reduce its false positive rate.

We have derived several local quality measures that differ in the details but share the same high-level framework based on statistical pattern recognition [4]. For now, we assume that the pair of views under consideration is the result of pair-wise surface matching. Our approach is to estimate the probability that a match is correct based on a vector of features derived from the registered surfaces. We denote the event that the match is correct by M^+, the event that it is incorrect by M^-, and the vector of derived features by $\boldsymbol{x}$. With this terminology, $P(M^+|\boldsymbol{x})$ is the posterior probability of a correct match given the observed features, and

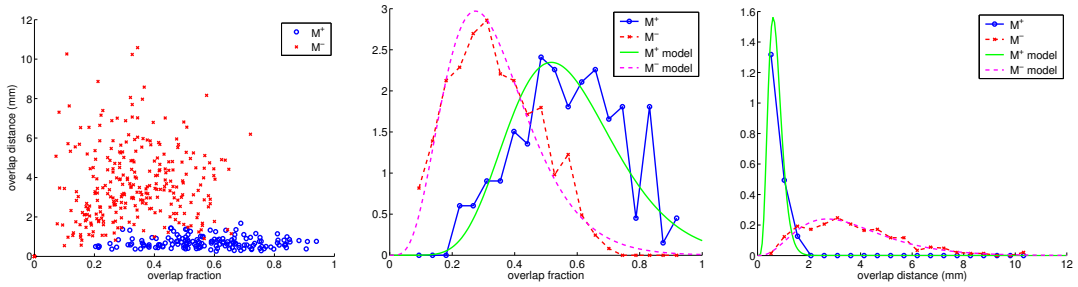

Figure 3. The joint distribution for F_{OV} and D_{OV} (left) and the marginal distributions for F_{OV} (center) and D_{OV} (right) for a set of 443 matches (153 correct and 290 incorrect) obtained by exhaustive pair-wise surface matching on three real objects using the Minolta Vivid 700 range sensor. The PDFs for the maximum likelihood class models are shown overlaid.

$P(M^-|\boldsymbol{x})$ is that of an incorrect match. We define the quality of a match between views V_i and V_j with relative pose $T_{i,j}$ to be the log odds ratio of the posterior probabilities:

$$Q_{\mathrm{L}}(V_i, V_j, T_{i,j}) = \log\left(\frac{P(M^+|\boldsymbol{x})}{1 - P(M^+|\boldsymbol{x})}\right)$$
$$= \log\left(\frac{P(M^+|\boldsymbol{x})}{P(M^-|\boldsymbol{x})}\right) = \log\left(\frac{P(\boldsymbol{x}|M^+)P(M^+)}{P(\boldsymbol{x}|M^-)P(M^-)}\right) \quad (1)$$

Thus, Q_{L} is an unthresholded Bayes classifier for matches.

We estimate the distributions $P(\boldsymbol{x}|M^+)$ and $P(\boldsymbol{x}|M^-)$ and the prior probabilities $P(M^+)$ and $P(M^-)$ from labeled training data. The priors are estimated directly from the frequency of M^+ and M^- in the data. We model the conditional distributions using parametric density estimation, first choosing a parametric form for the distributions and then computing the maximum likelihood parameters. We model our features using a Gamma distribution, which has a flexibly-shaped PDF that encompasses the exponential distribution and can approximate a Gaussian. Our motivation for using parametric methods is primarily the compactness of the representation. Non-parametric methods, such as kernel density estimation, could be used instead.

Associated with Q_{L} is a Bayesian classifier C_{L}, which is a thresholded version of Q_{L} (decide M^+ if $Q_{\mathrm{L}} > \lambda$, else decide M^-). We use C_{L} to remove the worst matches from G_{LR}, conservatively choosing the threshold λ based on training data to avoid removing any correct matches.

2.2 The overlap local quality measure

Using this framework, we have derived several local quality measures. We illustrate the concept with the overlap local quality measure; additional quality measures are described in [7]. The overlap quality measure is based on two features: overlap fraction and overlap distance. Intuitively, surfaces that overlap significantly and are close wherever they overlap should have high quality.

Overlap fraction, F_{OV}, is the maximum proportion of each surface that lies in the overlapping region. Let R_i be the overlapping region of surface S_i with $T_{i,j}S_j$ and R_j be the overlapping region of $T_{i,j}S_j$ with S_i. The overlap fraction is

$$F_{\mathrm{OV}} = \max\left(\frac{A(R_i)}{A(S_i)}, \frac{A(R_j)}{A(S_j)}\right), \quad (2)$$

where $A(S)$ denotes the surface area of S. Using the maximum ensures that if one surface is a subset of the other, F_{OV} will be 1, which coincides with our intended meaning of overlap fraction. Overlap distance, D_{OV}, is the RMS distance between a set of closest point pairs in the overlapping region. This is essentially the same error measure used in the ICP algorithm [3].

To compute F_{OV} and D_{OV}, we first sample K points, $\boldsymbol{p}_{ik}$, from S_i and an additional K points, $\boldsymbol{p}_{jk}$, from S_j. We then determine whether each sample point overlaps the other surface using the overlap definition (section 1.2). Let u_{ik} be the overlap indicator variable for the points $\boldsymbol{p}_{ik}$ and likewise u_{jk} for the points $\boldsymbol{p}_{jk}$. Let N_i and N_j be the number of overlapping points from $\boldsymbol{p}_{ik}$ and $\boldsymbol{p}_{jk}$ respectively (i.e., $N_i = \sum_{k=1}^{K} u_{ik}$ and $N_j = \sum_{k=1}^{K} u_{jk}$). Then equation 2 is approximated by

$$F_{\mathrm{OV}} \approx \max\left(\frac{N_i}{K}, \frac{N_j}{K}\right) \quad (3)$$

If we define $d_{ik} = u_{ik} D_{\mathrm{CP}}(T_{j,i}\boldsymbol{p}_{ik}, S_j)$ and $d_{jk} = u_{jk} D_{\mathrm{CP}}(T_{i,j}\boldsymbol{p}_{jk}, S_i)$, then the overlap distance can be computed by

$$D_{\mathrm{OV}} = \sqrt{\frac{\sum_{k=1}^{K}(d_{ik}^2 + d_{jk}^2)}{N_i + N_j}} \quad (4)$$

If F_{OV} is zero or if there are insufficient overlapping sample points to reliably compute D_{OV}, we leave Q_{L} undefined.

Finally, we estimate the joint distributions $P(D_{\mathrm{OV}}, F_{\mathrm{OV}}|M^+)$ and $P(D_{\mathrm{OV}}, F_{\mathrm{OV}}|M^-)$. We assume that F_{OV} and D_{OV} are conditionally independent given the class label M and factor the joint distributions (figure 3). Substituting into equation 1, we obtain the overlap local quality measure:

$$Q_{\mathrm{L}} = \log\left(\frac{P(D_{\mathrm{OV}}|M^+)P(F_{\mathrm{OV}}|M^+)P(M^+)}{P(D_{\mathrm{OV}}|M^-)P(F_{\mathrm{OV}}|M^-)P(M^-)}\right) \quad (5)$$

With this probabilistic formulation, it is easy to combine several independent features into a single quality measure, since the probabilities are just multiplied. Moreover, disparate features with different units – such as color or texture similarity – can also be integrated using this framework.

2.3 From local quality to global quality

Now that we have a method for computing local quality for a pair of views, we can extend this method to handle an entire model, G. Assuming the edges in G arise from surface matching and that the local quality measures of the matches are independent, we could define model quality as the sum of the local quality measures. However, this ignores the information encoded in non-adjacent views. Instead, we compute global quality by summing the local quality measured between all connected views:

$$Q_{\mathrm{G}}(G) = \sum_{(i,j)\in V_c} Q_{\mathrm{L}}(V_i, V_j, T_{i,j}) \quad (6)$$

where V_c the set of connected (not necessarily adjacent) view pairs in G. When computing Q_{G} for a model hypothesis, the hidden edges are first deleted (i.e., quality is computed on a part by part basis).

Unfortunately, the statistical model that we learned for computing Q_{L} for pair-wise matches does not apply to the computation of Q_{G}, since we compute local quality between non-adjacent views, and the distributions of the feature values used in the computation change as a function of path length. For example, accumulating registration error will cause D_{OV} for correct matches to increase with longer path lengths. To solve this problem, we learn a separate statistical model for each path length ($l = 2 \ldots L$). In this case, we generate random paths of a given length from G_{LR} of the training models. The model for path length L is used to compute the quality for any path longer than L. The prior probabilities $P(M^+)$ and $P(M^-)$ can no longer be estimated from data, since the multi-view surface matching algorithm actively chooses which matches to include in the model hypothesis. Therefore, we assume uniform priors.

3 Searching for the optimal model

Now that we have a measure of model quality, we turn our attention to the problem of finding the model hypothesis that maximizes Q_{G}. Even for a small number of views, combinatorics prevent us from exhaustively searching the acyclic sub-graphs of G_{LR}. Instead, we use an iterative stochastic algorithm that repeatedly applies edge update operations to a model hypothesis. In its simplest form, the algorithm performs randomized hill-climbing, but the Q_{L} values for the edges can be used to guide the update steps. Finally, we show how the algorithm can incorporate simulated annealing to avoid being trapped in local minima.

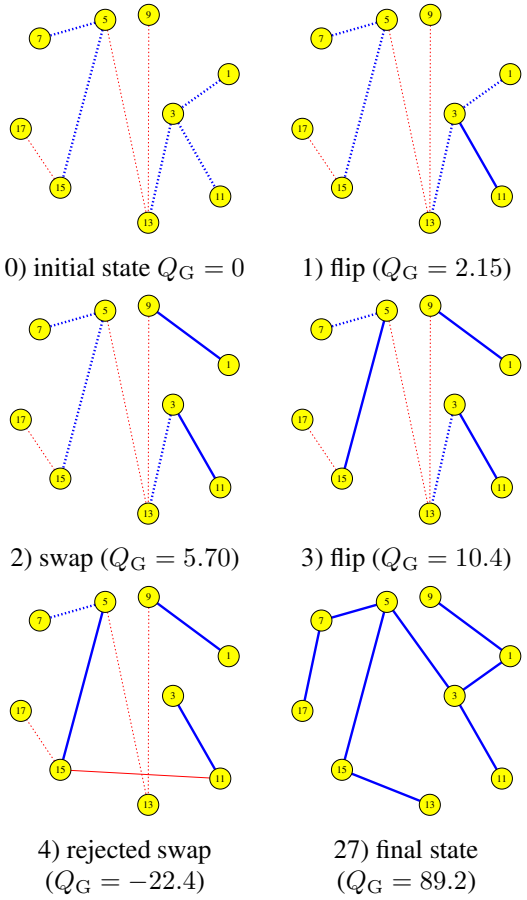

Figure 4. The first four iterations of the iterative edge swapping algorithm for the squirrel test object illustrate the swapping and flipping operations. The final hypothesis (bottom right) corresponds to the model shown in Fig. 1.

The algorithm begins with a hypothesis randomly chosen from the spanning trees of G_{LR} with the visibility of all edges set to false (figure 4). This corresponds to a model with N parts. Alternatively, we can use one of the iterative addition algorithms described in [8] to generate an initial hypothesis that may be quite close to the correct solution. The algorithm then iterates until termination (maximum it-

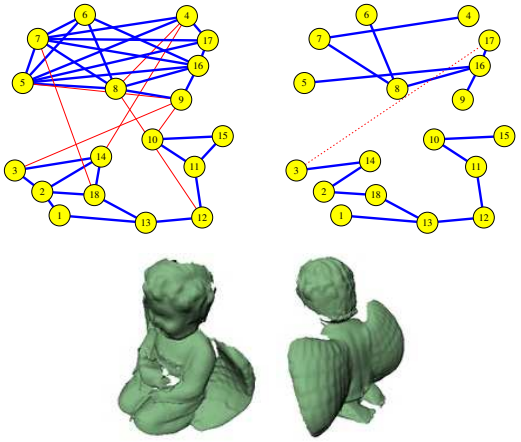

Figure 5. G_{LR} **for the angel3 test object (top left). The hypothesis found by our algorithm (top right) correctly consists of two parts – front views and the back (bottom).**

eration count is reached or a given number of iterations passes without updating the hypothesis). In each iteration, a new hypothesis H' is proposed based on the current hypothesis H and is evaluated using Q_{G}. If the quality improves, the proposed hypothesis is accepted ($H \Leftarrow H'$).

H' is generated from H by applying one of two types of update operations: an edge flip or an edge swap (figure 4). An edge flip involves choosing an edge from H and toggling its visibility state. Changing an edge from hidden to visible joins two model parts, while changing from visible to hidden splits a model part into two. For an edge swap, H' is generated by choosing an edge from H, removing it, and then inserting an edge, with visibility set to true, chosen from the edges of G_{LR} that re-establish the spanning tree property.

We could generate proposed hypotheses entirely at random, but the local registration quality of the matches can be used to guide the algorithm. Intuitively, for edge flip operations, we want to make low-quality edges hidden and make high-quality edges visible. Likewise, for edge swap operations, we want to replace low-quality edges (visible or hidden) with high-quality ones. We use the following weighting scheme: First, an update operation (flip or swap) is chosen at random. For edge flips, we then choose randomly between flipping an edge from visible to hidden or vice versa. This gives us a set of viable edges and their corresponding local quality values. When flipping an edge from hidden to visible, we want to bias the selection towards higher quality edges while ensuring that even the lowest quality edges are selected with reasonable frequency. We therefore compute weights for the viable edges by adding an offset to the quality values such that the largest weight is a constant C larger than the smallest one. The target edge is then chosen by sampling from this weighted distribution. Conversely, when flipping an edge from visible to invisible, the low-quality edges should receive higher weights. In this case, we negate the quality values and use a different offset to obtain a factor of C spread in the weight values. The same scheme is used for edge swaps. The latter weighting method is used to pick a low quality edge from H for removal, and the former weighting method is used to pick a high-quality edge from G_{LR} that re-establishes the spanning tree property. Figure 4 shows several iterations of this algorithm for the squirrel test data. Figure 5 shows an example where the algorithm correctly determines that the best hypothesis from G_{LR} is a model with two parts.

Even with stochastic updates, the edge swapping algorithm can be susceptible to local minima. One strategy for avoiding local minima in high-dimensional optimization problems is to use simulated annealing. Our algorithm is easily extended to incorporate simulated annealing. With simulated annealing, an update may be accepted even if the Q_{G} decreases. The probability of accepting such a hypothesis is a function of the size of the change in quality ($\Delta Q = Q_{\mathrm{G}}(H') - Q_{\mathrm{G}}(H)$) as well as "temperature" parameter T:

$$\begin{array}{lll} \text{if} \quad \Delta Q > 0, & \text{accept } H' & \\ \text{else,} & \text{accept } H' \text{ with prob. } e^{\Delta Q/T} & \end{array} \tag{7}$$

An initially large T enables frequent quality-lowering updates, which can allow the algorithm to pass through local minima to alternate modes of Q_{G}. T is reduced over time according to a "cooling schedule." As $T \to 0$, quality-lowering updates are increasingly unlikely, and the algorithm reduces to hill-climbing.

4 Experiments

4.1 Local and global quality measures

Our first experiment shows that the overlap local quality measure provides improved discriminating power over the traditional registration error metric, RMS overlap distance (D_{OV}). We computed Q_{L} and D_{OV} for the 436 matches from pair-wise registration of three test objects using the Q_{L} model shown in figure 3 (which was learned from different objects). We then classify the matches by thresholding the two measures and record the performance as an ROC curve, which shows the trade-off between false positives and true positives as the threshold is varied (figure 6). The results show that our local quality measure outperforms the RMS overlap distance metric.

We also performed an experiment to verify that our global quality measure is actually minimized by the correct

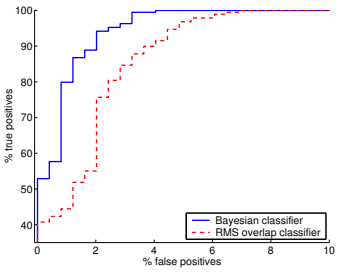

Figure 6. ROC curve comparison of the Bayesian classifier versus RMS distance, the most common conventional registration error measure.

solution. To test this, we computed Q_{G} for 1000 random model hypotheses from each of several test scenes, comparing against the Q_{G} for the ground truth solution. We also verified that hypotheses containing several correct model parts did not produce a Q_{G} higher than that of the ground truth single-part solution by repeating the above experiment using only correct matches from the corresponding G_{LR}.

4.2 Performance on a large set of test models

We tested our algorithm on a set of 18 real objects and 18 synthetic ones. For each real object, 15 to 20 views were obtained with the Vivid 700 scanner using the hand-held modeling technique described in [8]. For each synthetic object, 32 synthetic range images were created from viewpoints that tessellate a sphere surrounding the object. The range images were corrupted with Gaussian noise ($\sigma = 1$mm) to simulate worst case operation of the sensor.

We use two metrics for evaluating the correctness of the multi-view surface matching output and the reconstructed models. The first metric, maximum correspondence error (E_{MC}), measures the maximum displacement of any point on a surface S_i from its ground truth position [17]. This approach is better than reporting rotation and translation errors. We make E_{MC} scale independent by normalizing it by the size of the S_i (i.e., the diagonal length of the bounding box of S_i). Using E_{MC}, we can define the term "correct match," which is used in classifying training data and for labeling model graphs for illustration. We define a match to be correct iff $E_{\mathrm{MC}} < 0.05$ for both surfaces. Since our ultimate goal is to create geometrically accurate 3D models, we define a second error metric, the maximum reconstruction error (E_{MR}), which is the Hausdorff distance between the reconstructed surface and the original surface (i.e., the maximum distance between any point on the reconstructed surface and its closest point on the original surface).

We say a model hypothesis is correct if every match in the hypothesis is correct and if it contains the minimum

Figure 7. Example modeled scenes with number of views, scene size, and sensor used. Bottom right: The algorithm failed due to symmetry in the scene.

number of parts possible for the given G_{LR}, partially correct if it contains only correct matches but the number of parts is not minimal, and incorrect if it contains any incorrect matches. For the 36 test objects, our algorithm constructed 32 (89%) correct models , 1 (3%) partially correct model, and 3 incorrect models (8%). Figure 7 shows one of the failures. G_{LR} for 6 of the test models did not contain a single-part solution, and our algorithm found the correct multi-part model in 4 (67%) of these cases. Table 1 shows statistics for several of the test objects. Our algorithm failed only on CAD modeled objects that were highly symmetric (e.g., the saucer section of the enterprise). Some of these failures can be overcome with more sophisticated quality measures [7], but clearly an explicit treatment of symmetry could be beneficial.

Finally, we verified that our algorithm works across different sensors and scene sizes. We have automatically constructed models using three different real sensors – the Minolta Vivid 700, the Zoller and Fröhlich (Z+F) LARA 25200, and a stereo camera system, and we have modeled scenes varying in size from 200mm to 200m (figure 7).

object	views	iters (time)	E_{MC}	E_{MR}
gnome	27	69 (106)	n/a	n/a
squirrel	18	48 (87)	n/a	n/a
angel1	17	45 (34)	n/a	n/a
Buddha	32	61 (194)	0.04%	0.30%
teeth	32	61 (215)	0.037%	0.20%
enterprise	32	n/a	109%	10%

Table 1. Statistics for a representative sample of real (top) and synthetic (bottom) test scenes. Iters is the number of iterations before the correct solution was found, with time in seconds. E_{MC} and E_{MR} are normalized by the size of the scene, which is 200mm.

5 Summary and future work

We have answered the two questions set forth in the introduction: "what is a good model?" and "how do we search for the best model?" For the first question, we developed a probabilistic framework for computing the registration quality of a pair of views. This framework is quite general and can be applied beyond the specific 3D modeling application described here. We also showed how the local quality measure can be extended to evaluate the quality of an entire model. For the second question, we presented a stochastic algorithm which searches the hypothesis space using simple edge update operations on a model graph and showed how the basic algorithm can be improved using edge weighting and simulated annealing.

In the future, we plan to focus on three issues: view selection, sequential algorithms, and symmetry. While the strategy of performing exhaustive pair-wise surface matching on the input views is reasonable for small scenes (≈ 50 views or less), the operation is $O(N^2)$. To scale to larger scenes, we propose to selectively register view pairs, exploiting information inherent in each view to sort the views based on the likelihood of a successful match or to partition them into groups that are likely to match with each other. We will study the case where some information, such as approximate sensor position, is known in advance. Secondly, we plan to explore the real-time modeling scenario in which views arrive one at a time, rather than as a batch. Finally, we want to analyze the effects of symmetry in 3D modeling. If we can explicitly incorporate the concept of symmetry into our algorithm, we may be able to overcome the errors that it currently makes, possibly outputting a set of solutions based on the scene's symmetry group.

Acknowledgments

Daniel Huber was supported in part by the Eastman Kodak Company through the Kodak Fellows Program.

References

[1] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. AVENUE: Automated site modeling in urban environments. In *Proceedings of the International Conference on 3D Digital Imaging and Modeling (3DIM)*, pages 357–364, May 2001.

[2] F. Bernardini and H. Rushmeier. Strategies for registering range images from unknown camera positions. In *Proceedings of Three-Dimensional Image Capture and Applications III*, pages 200–206, Jan. 2000.

[3] P. Besl and N. McKay. A method of registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2):239–256, Feb. 1992.

[4] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

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

[6] A. Heyden, G. Sparr, M. Nielsen, and P. Johansen. Multi-scale EM-ICP: a fast and robust approach for surface registration. In *Proceedings of the 7th European Conference on Computer Vision (ECCV '02)*, pages 418–32, May 2002.

[7] D. F. Huber. *Automatic Three-dimensional Modeling from Reality*. PhD thesis, Carnegie Mellon University Robotics Institute, Pittsburgh, PA, Dec. 2002.

[8] D. F. Huber and M. Hebert. Fully automatic registration of multiple 3D data sets. *Img. and Vis. Comp. (in press)*.

[9] K. Ikeuchi and Y. Sato, editors. *Modeling from Reality*. Kluwer Academic Publishers, 2001.

[10] A. Johnson, O. Carmichael, D. F. Huber, and M. Hebert. Toward a general 3D matching engine: Multiple models, complex scenes, and efficient data filtering. In *Proceedings of the 1998 Image Understanding Workshop (IUW)*, pages 1097–1107, Nov. 1998.

[11] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 21(5):433–49, May 1999.

[12] M. Levoy et al. The Digital Michelangelo Project: 3D scanning of large statues. In *Proceedings of ACM SIGGRAPH*, pages 131–144, 2000.

[13] P. Neugebauer. Reconstruction of real-world objects via simultaneous registration and robust combination of multiple range images. *International Journal of Shape Modeling*, 3(1 & 2):71–90, 1997.

[14] K. Nishino and K. Ikeuchi. Robust simultaneous registration of multiple range images. In *Proceedings of the Fifth Asian Conference on Computer Vision*, pages 454–461, Jan. 2002.

[15] P. Rander, P. Narayanan, and T. Kanade. Virtualized reality: Constructing time-varying virtual worlds from real events. In *Proceedings of IEEE Visualization '97*, pages 277–283, October 1997.

[16] S. Rusinkiewicz. *Real-time Acquisition and Rendering of Large 3D Models*. PhD thesis, Stanford University, Stanford, California, Aug. 2001.

[17] M. Wheeler. *Automatic modeling and localization for object recognition*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, Oct. 1996.