To appear at British Machine Vision Conference, Edinburgh, UK. (1996).

Free-Form Surface Matching using Mean Field Theory

A. J. Stoddart, K. Brunnström
Dept. of Electronic & Electrical Engineering
University of Surrey, Guildford, Surrey GU2 5XH, UK
S & T Datakonsulter AB, Box 24183, S-104 51, Stockholm, Sweden
email: `a.stoddart@ee.surrey.ac.uk, kjell.brunnstrom@st.se`

Abstract

A new method for free-form surface matching is presented. The method provides an initial guess of the transformation, which may subsequently be refined by the Iterated Closest Point method.

The algorithm is based on Mean Field Theory (MFT). MFT has recently been shown to be a powerful optimization technique. It has a very similar architecture to a Hopfield neural network, but is considerably more effective. We also show how the computational complexity can be substantially improved by a hashing scheme. This combination of geometric hashing with MFT is likely to be applicable to many other recognition and labelling problems.

1 Introduction

The free-form surface matching problem remains a difficult problem. Several factors make it difficult. We may not assume that simple features such as spheres, cylinders or planes can be extracted. Also, the points on a surface cannot be ordered as they can in the case of a curve.

The task of finding an accurate registration is usually handled by the iterative closest point (ICP) algorithm (Besl and McKay, 1992). This method finds the nearest local minimum of a cost function formulated in *pose space*. This means that the user must supply an initial guess reasonably close to the global minimum. This paper presents a method that could automatically supply this guess. We refer to the process of pose refinement as *registration* and finding initial guesses as *matching*.

We cast the problem of matching two free-form surfaces as a search or optimisation problem in *correspondence space*. This leads to a problem with many local optima, where we are interested in the global optimum. Recently Mean Field theory has attracted attention as a powerful tool for optimisation problems. Because it is an annealed method it has a better chance of finding global optima than simple gradient descent methods.

The problem we address is as follows. Suppose there is some detailed surface model of an object and this object is imaged using a range sensor giving 3D information, then we are seeking the transformation that will bring the model and the data into alignment. In a single view the object is only partly imaged and the image may contain measurements of other objects. Of course the "model" may be the data from another view.

For a discussion on free-form surface matching see Besl's survey article (1992). Recent work on matching free-form surfaces includes (Bergevin et al., 1995) and (Hebert et al., 1995). We should note that although our method does not make restrictive assumptions such as convexity or the absence of clutter it is not well suited to surfaces consisting largely of a few planes, cylinders or spheres. Other methods exist in this case. We do not rely on reliable extraction of features such as zero crossings of curvature. There are no restrictions on the topology. There may be holes, discontinuities or clutter. We do require some 'reasonable' amount of overlap between scene and model.

In related work we have explored use of Genetic Algorithms in place of MFT (Brunnström and Stoddart, 1995, 1996)

2 Free-Form Surface Matching

Peterson and Söderberg (1989) have introduced a new and powerful type of neural net based on Mean Field theory (MFT). Due to limitations of space we refer the reader to the original paper for further details, and simply note that MFT is a optimiser suited to labelling problems, and is similar in scope to the Hopfield neural net and Probabilistic Relaxation.

To apply Mean Field theory to free-form surface matching, the problem must be formulated as a discrete labelling problem. This is done by randomly sampling the model surface to a 'sufficient' density, and randomly sampling the scene surface to a similar density. We now pose the problem as a point matching problem. We wish to label (associate) each point in the scene with a point from the model, or a null label. Because the points are randomly chosen we don't expect an exact match. However, when sampled to a sufficient density we hope to match to a relatively close point.

Based on this sampling we need to formulate an objective function that measures the quality of the match. This objective function will be described in detail below. It is based on a function that counts the number of good matches, by using the translational and rotational invariants such as relative orientation of surface normals and distances between points.

2.1 The objective function

The objective function must measure match quality in correspondence space, so it is designed to be invariant to translation or rotation. Because of the random nature of the sampling we can never achieve an exact match but there will usually be reasonably good approximate matches. The quality of the best match will be a function of the average spacing between the randomly selected points on the surface.

We suppose that the M model points are labelled by $\alpha = 1..M$ and the N scene points are labelled by i or $j = 1..N$. If scene point i takes on label α this is denoted by α_i. We denote a joint labelling of all objects by the set $\{\alpha_1, \alpha_2, ...\alpha_N\}$. As a shorthand we define the multi-index $\vec{\alpha} = \{\alpha_1, \alpha_2, ...\alpha_N\}$.

Now suppose that we choose 2 points i, j from the scene surface. They will contain 4 vectors (2 positions and 2 normal vectors) of information between them, i.e. $\vec{r}_i, \hat{n}_i, \vec{r}_j, \hat{n}_j$. There are 10 independent parameters and 4 invariants under rigid translation and rotation. We are interested in features that remain invariant to rotation and translation. The most obvious is the length of the vector $\vec{v}_{ij}$ from point i to point j, i.e. $\vec{v}_{ij} \equiv \vec{r}_j - \vec{r}_i$.

Pairwise relative orientation is also an invariant. Suppose we define the angle θ_{ij} as

$$\cos(\theta_{ij}) = \hat{n}_j \cdot \vec{v}_{ij} \tag{1}$$

The angles θ_{ij} and θ_{ji} form two additional invariants.

Finally there will be a twist angle between the two normals representing the extent to which they are out of plane. We define this angle as β_{ij} given by

$$\cos(\beta_{ij}) = (\widehat{\hat{n}_j \times \vec{v}_{ij}}) \cdot (\widehat{\hat{n}_i \times \vec{v}_{ij}}) \tag{2}$$

Now consider two model points and two scene points. We wish to define a pairwise match quality $E(\alpha_i, \alpha_j)$ which will be a product over four terms. The first term $e_d(\alpha_i, \alpha_j)$ can then be expressed as

$$e_d(\alpha_i, \alpha_j) = \exp\left\{-\frac{[|\vec{v}_{\alpha_i\alpha_j}| - |\vec{v}_{ij}|]^2}{2\sigma^2}\right\} \tag{3}$$

For a perfect match this returns 1, and decays as a Gaussian with standard deviation σ. Clearly σ must be related to measurement noise and sampling density. We expect the sampling density to dominate here, and we use a simple probabilistic model to determine σ. We do not tune σ.

Suppose we have M points randomly distributed on a model surface with area A. The density will be $\rho = N/A$. For an infinite plane the distance to the nearest point will be given by a Poisson distribution and the expected value of the distance to the closest point will be $\Gamma(\frac{3}{2})\sqrt{\frac{\rho}{\pi}}$. This is used as the basis for selecting σ. For simplicity we always set the sampling density to be the same on the scene and the model.

The angular quality measure is defined as

$$e_n(\alpha_i, \alpha_j) = \exp\left\{-\frac{(\theta_{\alpha_i,\alpha_j} - \theta_{ij})^2 + (\theta_{\alpha_j,\alpha_i} - \theta_{ji})^2 + (\beta_{\alpha_i,\alpha_j} - \beta_{ij})^2}{2\mu^2}\right\} \tag{4}$$

As the normal is a derivative quantity we assume that the error will be dominated by the measurement error and not the random sampling process. This is true on gently varying surfaces, but not those with many small details. We select the parameter μ from an estimate of the typical error in a surface normal. We use a value for $\mu = 20^\circ$ throughout this paper. Finally we obtain the pairwise quality measure we require

$$E(\alpha_i, \alpha_j) = -e_d(\alpha_i, \alpha_j)e_n(\alpha_i, \alpha_j) \tag{5}$$

This quantity is equal to -1 for a perfect match and degrades to zero. The minus sign is inserted because in MFT we *minimise* energy. From this expression we can find the energy $E(\alpha_i)$ of a single object given all the labels $\vec{\alpha}$,

$$E(\alpha_i) = \sum_{j \neq i} E(\alpha_i, \alpha_j) \tag{6}$$

This measure will range between 0 to $-(N-1)$. We can also find the global objective function for a particular labelling $\vec{\alpha}$ by summing up all the pairwise relations i.e.

$$E(\vec{\alpha}) = 2 \sum_i \sum_{j<i} E(\alpha_i, \alpha_j) = \sum_i E(\alpha_i) \tag{7}$$

This measure will range between 0 to $-N(N-1)$.

2.2 Mean Field theory

The detailed derivation of the MFT algorithm is too long to present here, so we restrict ourselves to a short summary, and refer the reader to the Peterson and Söderberg paper (1989) for more details.

The first step is to introduce a temperature parameter T. It is assumed that there is some probability distribution

$$p(\vec{\alpha}) = \frac{1}{Z} \exp\left\{-\frac{E(\vec{\alpha})}{T}\right\} \tag{8}$$

Z is a normalising factor and the probability distribution is called the Gibbs or Boltzmann distribution. The detailed probability is of space complexity M^N so instead a mean field approximation is sought, which we denote $r(\alpha_i)$. This is the "probability" that object i takes on label α. There are NM such label weightings, sometimes referred to as a soft labelling assignment. The full complexity of the joint probability distribution has been discarded.

For a fixed temperature we can iterate the following equations which converge to the 'true' value of $r(\alpha_i)$. We compute an intermediate 'support' $q(\alpha_i)$ given by

$$q^{(n+1)}(\alpha_i) = \sum_j \sum_{\alpha_j} E(\alpha_i, \alpha_j) r^{(n)}(\alpha_j) \tag{9}$$

The $r(\alpha_i)$ are then updated by

$$r^{(n+1)}(\alpha_i) = \frac{\exp\{q^{(n+1)}(\alpha_i)/T\}}{\sum_{\alpha_i'} \exp\{q^{(n+1)}(\alpha_i')/T\}} \tag{10}$$

This system of equations is then annealed, that is the temperature T is reduced from a high value to a low value. This aspect resembles simulated annealing.

The resemblance to a Hopfield neural network may now be seen. The support in equation (9) is identical to the way inputs are combined in an ordinary neuron. The r update equation (10) is a straightforward generalisation of the neuron activation function. When there are only 2 labels this reduces to the conventional neuron activation function, and one input/output wire may be disregarded. The temperature corresponds to the gain parameter of a neuron.

How is the cooling schedule chosen? In simple optimisation problems like this one the energy defined by

$$E[r] = \sum_{ij} \sum_{\alpha_i, \alpha_j} r(\alpha_i) E(\alpha_i, \alpha_j) r(\alpha_j) \tag{11}$$

usually undergoes a sigmoidal transition from a high value to a low value over a distinct range of temperatures. This corresponds to the specific heat of some spin systems in thermodynamics. A crude cooling schedule can be chosen by starting above this transition region and following a geometric progression until below the region. In our case this means starting at $T = 5$ and decreasing T by $T \to 0.8T$ until $T < 0.2$

Once again we note that our method is not suited to problems with continuous symmetries such as planes or cylinders. The reason is that the objective function will begin to have valley like structures and the method will degrade.

Finally the labelling $\vec{\alpha}$ needs to be processed further to produce the required pose estimate. Our procedure is simply to choose the k objects with lowest energy. These give us k point correspondences and we then use a conventional least-squares fit to obtain the pose (Kanatani, 1994).

2.3 Complexity

What is the complexity of this basic method? If the label weights r are updated over i iterations then the complexity will be $O(iN^2M^2)$. To accurately represent a complicated surface by points we need many of them - this means our complexity is potentially very high.

If we knew only that we should optimise the energy as given by equation (11) this may be about the best we could expect of any method to find the global optimum. However we do have additional knowledge of the problem, and in almost any optimisation problem using some additional knowledge can enhance performance.

How can we express and use this knowledge? The knowledge we have is that the data come from some 3 dimensional space - the $E(\alpha_i, \alpha_j)$ follow a pattern, and are not simply random numbers. In particular many of them are zero, and it turns out that this knowledge is the easiest to exploit.

One of the most efficient schemes for matching is geometric hashing (Lamdan and Wolfson, 1988). For a problem with binary invariants it has complexity $O(kN^2 + kM^2)$ where k is a constant depending on the density of the hash table. In the next section we borrow some ideas from geometric hashing to dramatically reduce the computational complexity.

3 Hash-based support

The bulk of the computational complexity comes from the computation of the support, as given by equation (9). This involves a loop over i, α_i, j, α_j which is of complexity N^2M^2.

The energies $E(\alpha_i, \alpha_j)$ are only significantly non-zero when all 4 scene invariants are within a few σ or μ of the respective model invariants. We may exploit this by use of a hash table. We firstly select a few invariants to hash on, for example the distance and one angle.

A discrete hash table is set up with bins of size about σ by μ. Each of the $N(N-1)$ model invariants are computed and (α, α') is stored in the hash table. This step is of order M^2.

This pre-stored hash table by now be used to compute the support. The invariants corresponding to all scene pairs are computed and used as keys to hash into the table. The model pairs within some preset threshold are retrieved. Then the support $q^{(n+1)}(\alpha_i)$ is increased by adding a term

$$E(\alpha_i, \alpha_j) r^{(n)}(\alpha_j) \tag{12}$$

Providing the threshold is chosen suitably this method of computing the support will give an almost identical result to the previous method. However the complexity reduces to $O(kN^2)$ per iteration where k is the hit rate.

In the worst case the hit rate could be $k = M^2$ and no advantage would accrue. In the best case the hit rate is $k = 1$ and traditional geometric hashing alone would provide a good answer. Geometric hashing breaks down when the hit rate rises to the point where no clear overall match is selected or the cost of verifying false matches becomes too high.

The MFT version continues to work well beyond this since it is capable of finding the globally optimum labelling in terms of the objective function. The case of free form surface matching is within this 'hashing breakdown zone', since almost every object-label match will receive many votes. It is the annealing steps that allow a globally best match to emerge.

The overall complexity of this method is $O(ikN^2 + M^2)$. Since i is about 10, it is slower than pure geometric hashing but compared to the original $O(iN^2M^2)$ of MFT it is a vast improvement.

4 Results

We will here present a few experiments illustrating the strengths and weaknesses of the presented approach to free-form surface matching.

The first data set we will show is the 'foot' surface. This model consists of 3000 polygons generated from a range data set by a deformable surface. In figure 4(a) we see the actual model. We attempt to register the foot with itself as the first experiment. We choose 40 points on the scene and 40 points on the data. The effects of this can be seen in figure 4(b) where we align the scene and model and show the 40 scene and 40 model points spread randomly on the model. The shortest point-to-point distances are typical of the best match we can expect. As we have mentioned, the matcher is stochastic because it samples the surface at random. The performance then should be judged by how often it gets the relative pose close enough to start an iterative closest point algorithm. Of the 2 parameters the translation is usually less open to question, so we use the rotation to judge the success of the algorithm. We regard a rotation of less than 25 degrees (around any axis) to be a success. By this criterion we get a 5 out of 10 success rate from sampling 20 points each from the scene and model. With 40 points each we get a

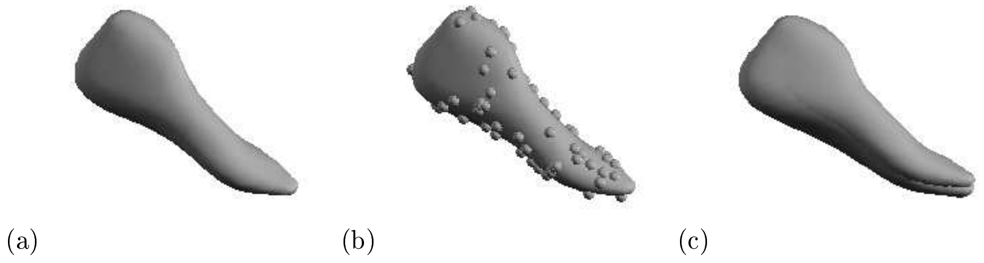

Figure 1: The foot dataset. (a) shows the scene and model, (b) shows 80 random points on the foot (c) shows the resulting registration

10 out of 10 success rate. For half of these runs the rotation is accurate to within 5 degrees. In figure (c) we see a typical result.

The next example we consider is a bust of Beethoven. This consists of 5000 triangles. We match the object to itself. In figure 2(a) we show the bust and in figure 2(b) we show a typical result.

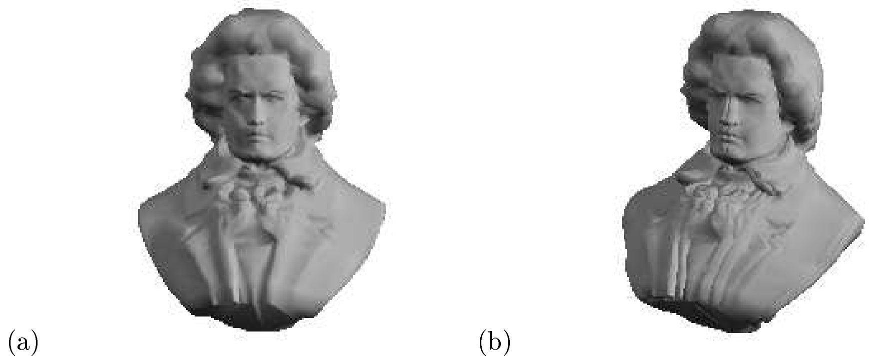

Figure 2: The Beethoven dataset. (a) shows the scene(=model) (b) shows the resulting registration

We now move on to consider data sets with clutter and noise, and where the scene and model differ. The first data set is the bunny data set. This is range data, subsampled and converted to a triangular mesh of 5000 triangles from each view. The previous examples are in cases where the normal is corrupted by the random sampling process, but in this case there is additional sensor noise. The sensor noise will primarily affect the normals computed from the triangles because they depend on a derivative.

Figure 4(a) and 4(b) show the scene and model that we match. In figure 4(c) we show the match achieved. The success rate is 9 out of 10.

In the next examples we consider rather more complicated objects. In figure 4 (a) we present the cow model consisting of 6000 triangles. In figure 4 (b) we show a successful match. We used 100 points on the scene and model. The success rate is 4 out of 10. The main problem here is that there is an approximate symmetry

Data Set	N	M	Success Rate
foot	20	20	5/10
foot	40	40	10/10
Beethoven	40	40	7/10
Bunny	40	40	9/10
Cow	100	100	4/10
Soldier	120	120	7/10

Table 1: Summary of results

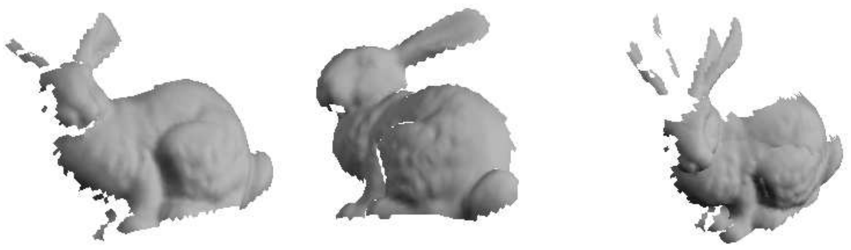

Figure 3: The bunny dataset. (a) shows the scene and (b) shows the model, (c) shows the resulting registration

when the cow faces backwards. The largest piece of surface is the stomach, and this matches quite well between these two positions. We believe that this problem can be overcome by focusing more points on the areas of the surface with high curvature but small area. When it does achieve a match the accuracy of the alignment is quite good and in the example shown the accuracy is better than 2 degrees. The worst error is 5 degrees. The final dataset that we will consider is the soldier. This consists of two overlapping views of a toy soldier each consisting of about 20000 triangles. In figure 4(a) and (b) we show a reconstruction of the entire soldier performed from several views by (Hilton et al., 1996). In figures (c) and (d) we show two overlapping views of the soldier that we will try and match. In figure (e) we show one of the matches with accuracy of about 3 degrees. The mismatch is visible when comparing the helmet with figure (f). When run with 80 points in both scene and model we achieved a 4 out of 10 success rate, using a threshold of 14 degrees to define success. The rate rose to 7 out of 10 when $N = M = 120$. All seven lie within 14 degrees of the correct result and 4 lie within 5 degrees.

The algorithm was coded in C++ and run on a Sun workstation. The execution time for a 40x40 point match is 20 seconds and for a 80x80 run it is 90 seconds. The approximate dependence on N^2 may be noted.

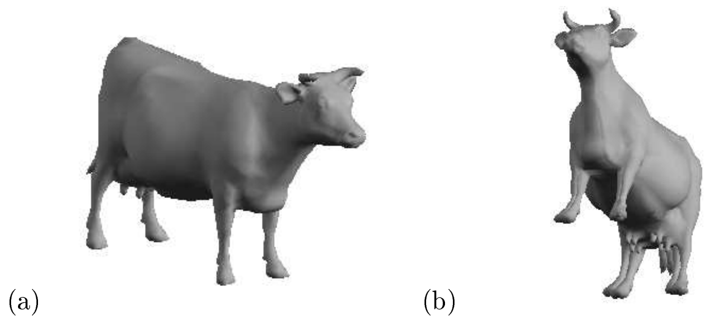

Figure 4: The cow dataset. (a) shows the scene and (b) shows the resulting registration

5 Discussion

We have in this paper presented a simple approach for finding the initial guess for the free-form matching problem. We presented promising results for several data sets. The algorithm is based on a powerful optimisation technique from mean field theory.

The basic steps of the algorithm are as follows. The scene and model surfaces are randomly sampled. Using binary invariants this is formulated as a point matching problem and optimised using mean field theory. High scoring points are used to compute the required transformation. The basic algorithm has the same architecture as a recurrent neural network. Since mean field theory has a annealing schedule it is an approximate global optimiser, unlike gradient descent type methods.

The basic algorithm is of complexity A^2 where A is the area of the scene and model. We present an algorithmic shortcut that can reduce the complexity to a similar complexity to Geometric Hashing, i.e. of order nearly A. There need be no drop in performance relative to the full MFT theory.

This algorithm is unusual in that it does not use any second derivative information. This gives it wide applicability and could be seen as a major advantage.

On the other hand we we believe that there is considerable scope for refinement of the algorithm. In particular if we begin to incorporate curvature information, either as a unary cost or as a selective surface sampling technique we expect major improvements in performance. This is because the use of curvature would supplement an already viable method, rather than being a intrinsic component of the method.

Finally we would like to acknowledge the sources of the data. The foot model is based on Cyberware data supplied by Tim McInerney, using the Slime package. The Beethoven and the Cow models were produced by Viewpoint Animation using a mechanical digitiser. The bunny dataset was used by Turk and Levoy in their zipper paper. The soldier data set was used by Soucy and Laurendau and collected by M. Rioux of the National Research Council of Canada.

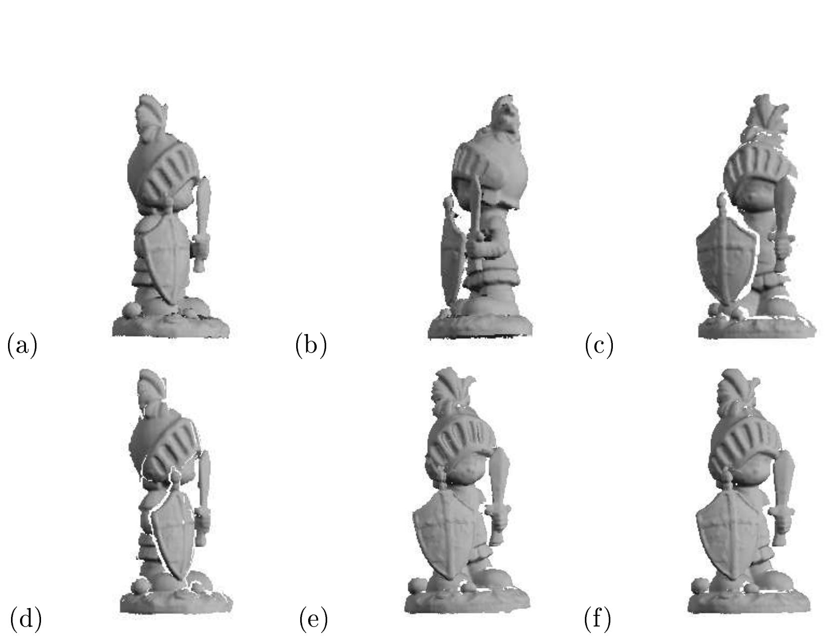

Figure 5: The soldier dataset. (a) and (b) show views of the reconstructed soldier. (c) and (d) show two views that we use as scene and model. (e) shows the resulting registration and (f) shows the exact registration

References

Bergevin, R., Laurendeau, D., and Poussart, D. (1995). Registering range view of multipart objects. *Computer Vision and Image Understanding*, *61*, no. 1, 1–16.

Besl, P.J. and McKay, N.D. (1992). A method for registration of 3-d shapes. *IEEE Trans. Pattern Analysis and Machine Intell.*, *14*, no. 2, 239–256.

Brunnström, K. and Stoddart, A. J. (1995). Genetic algorithms for free-form surface matching. CVAP181, Tech. Rep., Oct. 1995.

Brunnström, K. and Stoddart, A. J. (1996). Genetic algorithms for free-form surface matching. In *13th Int. Conference on Pattern Recognition*, Vienna, Austria.

Hebert, M., Ikeuchi, K., and Delinguette, H. (1995). A spherical representation for recognition of free form surfaces. *IEEE Trans. Pattern Analysis and Machine Intell.*, *17*, no. 7, 681–690.

Hilton, A., Stoddart, A. J., Illingworth, J., and Windeatt, T. (1996). Reliable surface reconstruction from multiple range images. In *Fourth European Conference on Computer Vision*, pp. 117–126, Cambridge, U.K.

Kanatani, K. (1994). Analysis of 3-d rotation fitting. *IEEE Trans. Pattern Analysis and Machine Intell.*, *16*, no. 5, 543–549.

Lamdan, Y. and Wolfson, H.J. (1988). Geometric hashing: a general and efficient model-based recognition scheme. In *1st Int. Conference on Computer Vision*, pp. 238–249, London, U.K.

Peterson, C. and Söderberg, B. (1989). A new method for mapping optimization problems onto neural networks. *Int. Journal of Neural Systems*, *1*, no. 1, 3–22.