An Evolutionary Approach to Fitting Constrained Degenerate Second Order Surfaces

C. Robertson, R. B. Fisher, N. Werghi, A. P. Ashbrook

Division of Informatics, University of Edinburgh,
Edinburgh, EH1 2QL, UK
craigr@dai.ed.ac.uk

Keywords: Evolutionary algorithms, surface fitting, Genocop III.

Abstract. In this work we examine the applicability of an evolutionary strategy to the problem of fitting constrained second-order surfaces to both synthetic and acquired 3D data. In particular we concentrate on the Genocop III algorithm proposed by Michalewicz [8] for the optimization of constrained functions. This is a novel application of this algorithm which has demonstrably good results when applied using parametric models. Example times for convergence are given which compare the approach to standard techniques.

1 Introduction

Shape analysis of objects from range data (captured three dimensional co-ordinates of surface points) is a key problem in computer vision with several important applications in manufacturing, such as assembly, quality control and reverse-engineering. The problem is generally formulated as a nonlinear programming problem (NLP), which tries to optimally fit the data to candidate shape descriptions. The NLP optimises a function subject to several constraining equations and inequalities. Especially with nonlinear constraints, it is notoriously difficult to optimise and there is no known method to guarantee a satisfactory solution. Traditional techniques, such as gradient descent, are unsatisfactory for the solution of NLPs, due to the local nature of their search methods and the reliance on smooth derivatives in the search-space. In previous work [7] we examined the applicability of evolutionary strategies to the problem of fitting lines and surfaces to both synthetic and acquired object range data. In this paper we effectively take the next step, which is to fit degenerate second order surfaces that have *a priori* constraints and geometric relationships. The Genocop III algorithm developed by Michalewicz [8], Ch.7 was used and extended in this paper by adding a complex evaluation function. It is an evolutionary algorithm system which is specialised to handle constrained function optimisation and particular to it is the handling of non-linear constraints. It uses real-valued genes, and includes methods to deal with linear, non-linear, domain and inequality constraints. We have used a specialised fitness function (described in section 2.3.3), applied to the problem of fitting parametric 3-dimensional surface equation chromosomes to range data while simultaneously applying several necessary geometric and domain constraints. The constraints applied are of two typical types : domain, the restriction on the parameter size; relational, relationships between surfaces that are known *a priori*.

Since this problem has a specific context it is important to illustrate it. Our group is researching the reverse engineering machined parts. These parts are often complex and possess many surfaces which may have know geometrix relationships. Segmentation and parameterisation of the captured 3-dimensional range data is a difficult multi-part task involving the following elements:

1. *Data collection.* This is performed using a moving-bed, orthogonal laser ranger which provides data at up to 0.5mm steps in the X-Y plane. Noise on the data is around $0.15\ mm$.
2. *Data registration.* This is performed using a variation on the **iterated closest point** algorithm [1].
3. *Segmentation.* There are many ways of segmenting the 3d dataset, most are based upon changes in local surface curvature followed by some form of least-squares optimisation, for example [9].
4. *Exploitation of constraints.* Constraints may be applied to exploit knowledge about surface relationships.

The formulation of constraints and the application of constraint-based correction and optimisation of surface fitting has been achieved previously [5] with notable success using the several constraint application strategies. There are, however, some associated problems with this approach: complex formulation of the constraint function; heavy reliance on the global convexity of the solution space; reliance on very accurate initial estimate of solution.

The 'processing pipeline' that is required for this approach also can lead to a build-up of problems that must be solved in the constraint application stage. In order to alleviate some of these problems a more holistic strategy is proposed where segmentation, fitting and constraint management takes place simultaneously. Previously [7] it was demonstrated that simultaneous fitting and constraint management could be achieved in a single evolutionary algorithm with careful chromosome management and good generation of starting conditions.

In this paper, the technique has been explored and improved by:

1. *Making the representation more efficient* by only applying the technique to degenerate quadric surfaces.
2. *Enhancing the evaluation* of chromosomes by applying specialised fitting functions for degenerates.
3. *A simple objective function*, the least-squares error metric, then using the constraints to define the manifold of allowable solutions.
4. *Including a naive segmentation function* as part of the evaluation function.

2 Method

2.1 Data Generation

Free Quadrics

A free quadric is a second order surface of the type:

$$a_1x^2 + a_2y^2 + a_3z^2 + a_4xy + a_5xz + a_6yz + a_7x + a_8y + a_9z + a_{10} = 0 \qquad (1)$$

This covers all second order shapes and these fall into several family groups : cylinders, cones, paraboloids, hyperboloids, ellipsoids, planes. Some of these forms are degenerate and there is sufficient variety in the shapes to cover all easily machineable surfaces. Note that higher order free surfaces also exist on machined parts, perhaps as a result of casting, but are not considered here.

Degenerate Quadrics

In this paper, the degenerate forms of these surfaces are used, this is a small subfamily of surfaces of the following types: spheres, cylinders, right circular cones. The machined surfaces that we address rarely contain whole pieces of these shapes and the surfaces are often fragmentary or partial. In order to generate synthetic versions of this subfamily, patches of the given types were generated as shown in figure 1. These fell into the following three groups:

1. *Spherical sections* generated about a given vector (spherical caps) as well as whole spheres
2. *Cylinders* of differing radii, length and wedge angle.
3. *Cones* of differing slope-angle, wedge-angle and length. Truncated cones were also used.

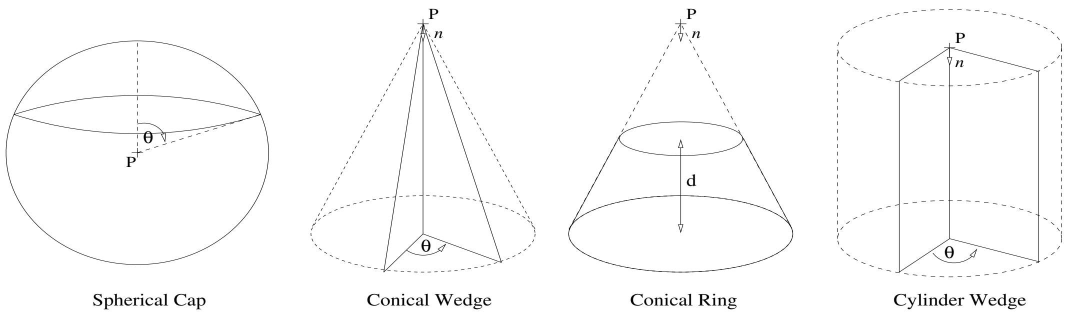


Fig. 1. Degenerate Quadric Patches

2.2 Gene and Chromosome Formulation

In standard GAs binary encoding forms the chromosomes in the solution, however in an evolution program each gene is a floating point number. Genes are then concatenated into a chromosome. In the cases we have previously investigated, typical part-chromosomes were parameter vectors ($A = a_i : i \in \{1, .., 10\}$ representing second order surfaces. In the new parametric representations, the vectors may be any length.

In the case of **planes**, we have used the 4 gene parametric representation : $A :< \hat{n}, d >$ where $\hat{n}$ is the unit normal describing the plane and d is the constant defining its minimum distance from the origin. In the case of **spheres**, we have used the 4 gene parametric representation : $A :< \mathbf{P}, r >$ where **P** is the centre point and

r is the sphere radius. In the case of **cylinders**, we have used the 7 gene parametric representation : $A :< \mathbf{P}, \hat{n}, r >$ where **P** is defined as the start point of the cylinder, $\hat{n}$ is the axis direction and r is the radius. For **cones**, the 7 gene representation used is : $A :< \mathbf{P}, \hat{n}, \alpha >$,where **P** is defined as the start point (or tip) of the cone, $\hat{n}$ is the axis direction and α is the half-angle between the axis and the slope of the cone.

A full chromosome, G, describing a given object, is a set of concatenated part-chromosomes: $G = \{A_j\}$. The parametric representation of a set of degenerate quadrics as a chromosome is much shorter than a set of general quadrics as explored in [7]. This cuts down the complexity of constraint representation and makes it amenable to straighforward manipulation without the need to employ geometrical constraints on the surfaces' form, only on pairs of forms taken as systems.

2.3 Algorithm

Traditional Methods of Generating Constrained Populations. Evolutionary methods have been shown to be useful for solving general *NLP* problems [11][6][7]. There are four main techniques for dealing with chromosomes that contravene constraints on solutions [10]: rejection, which discards infeasible solutions immediately throughout the process; repairing, which depends on methods to repair solutions back to feasible; modifying operators, which means designing crossover, mutation and other operators than only ever produce feasible offspring; penalties, which is the most common technique for optimizing constrained functions. A penalty function is one which punishes chromosomes for straying from the constraints by decreasing their fitness or removing them from the population. There are no good guidelines for designing such penalty functions however [10].

Almost all optimization problems are constrained in some way. What we required is some way of generating solutions that are both iteratively improving as well as satisfying these constraints. Most optimization problems are defined on a search space, D, as follows [1] : $D \subseteq R^q$, where $D = \prod_{k=1}^{q} < l_k, r_k >$ and each x_k is in the interval $< l_k, r_k >$. The set R^q is thus a crucial characteristic of the problem. Significant optimization theory only exists where this set is convex. In GENOCOP, this convexity is assumed, i.e we seek to optimize : $f(x_1, \ldots, x_q) \in R^q$, where $(x_1, \ldots, x_q) \in D \subseteq R^q$. D is convex and is defined by the range of variables $l_k \leq x_k \leq r_k$ for $k = 1, \ldots, q$.

Because D is convex, for each point in the search space, there exists a range where other variables remain fixed. We also assume that this range can be efficiently computed. This property is useful for performing **mutation**. If the variable x_k is to be mutated, it can be moved inside its range so any offspring produced are feasible.

Also, for any two points, $\mathbf{x}_1$ and $\mathbf{x}_2$, in the space D, the linear combination $a\mathbf{x}_1 + (1-a)\mathbf{x}_2$, (where $a \in [0, 1]$) is also in D. This is used for **crossover**.

Genocop II and III Calculus based methods assume that the objective function, $f(\mathbf{x})$, and all constraints are twice continuously differentiable functions of $\mathbf{x}$. The general approach is to transform the non-linear problem (*NLP*) into a sequence of sub-problems

[1] This brief account follows the one given by Michalewicz [8] which the reader is encouraged to read for further details.

and then solve those, requiring an explicit computation of the objective function. Some of these methods become ill-conditioned and fail.

Genocop II uses a sequential quadratic penalty function and is formulated as the optimisation of the function:

$F(\mathbf{x}, r) = f(\mathbf{x}) + \frac{1}{2r}\overline{C}^T\overline{C}$, where $r > 0$ and $\overline{C}$ is the vector of active constraints, $c_1, \ldots, c_l$

Attia has provided solutions to the instability of this approach [12]. The set of all constraints, C, is divided into the linear constraints,L, the non-linear equations,N_e and the non-linear equalities,N_i. A set of active constraints, A is then built from N_e and the violated constraints from N_i (a constraint is said to be violated if it is more than some tolerance δ from its correct value), which are called V. The structure of Genocop II is outlined in [8]. Inside its main loop, Genocop I optimizes the function $F(\mathbf{x}, r) = f(\mathbf{x}) + \frac{1}{2r}\overline{A}^T\overline{A}$. Several mutation operators take an initially identical population and introduce diversity to it. At convergence, the best individual,$\mathbf{x}^*$, is saved and the the value of the penalty parameter is decreased.

Most of the essential elements of Genocop III are the same as those of Genocop II. However, in this algorithm two populations are kept, a reference set $\overline{R}$ and a search set $\overline{S}$. The reference population is a set of fully feasible individuals which satisfy all the constraints whereas the search population may not. At each iteration, the search population are allowed to move around the solution space and are repaired back onto the constraint manifold. If the search point is $\overline{S}$ and the reference point is $\overline{R}$, then a random point $\overline{Z}$ is created from the segment between $\overline{S}$ and $\overline{R}$ by generating a value, $a \in [0, 1]$ then :

$$\overline{Z} = a\overline{S} + (1 - a)\overline{R} \tag{2}$$

Once a feasible $\overline{Z}$ is found, if it is better than $\overline{R}$, then that reference point is replaced with some probability. As the iterations progress, the set of reference points converge to the maximum or minimum on the search space.

Evaluation Function and Point Assignment for the Fitting. In our application of Genocop III the evaluation function is almost certainly more complex than was initially intended for the algorithm. For each point,x_i, the true geometric distance to the theoretical surface is computed and this is used as the least-squares error for that point relative to that surface.

$$e_i = min_p\{dist(\mathbf{x}_i, S_p)\} \tag{3}$$

where e_i is the error for the point $\mathbf{x}_i$, p is the index of M theoretical surfaces, i is the index of N points, S_p is the parameterised surface and *dist* is the distance to that surface.

The evaluation function to be minimised is then the sum of these minima :

$$E = \sum_{i=0}^{i=N} e_i \tag{4}$$

It is possible to use this as a simple segmentation scheme (especially if the chromosome population variations are small and the start conditions are close to the solution).

The point assignment is thus straightforward. In some tests, such as the real object discussed in section 3.3 the data is pre-segmented. If the assignment information is available it should be used in order that the computation time can be reduced.

Starting Conditions and Relational Constraints In virtually all cases, domain constraints on individual genes are used to narrow the search space for that gene. These are represented as one permanent part of the sequential quadratic penalty function matrix [8] used in the evaluation function. A good example of where domain constraints can reduce the search space is in the case of the three parameters describing a unit normal. Each of these parameters can never be outside the range $[-1, +1]$ so these make good domain constraints.

In-chromosome relational constraints are straightforward to formulate when a parametric form is used. For example, consider two planes,$P_1 = < x_1, x_2, x_3, x_4 >$ and $P_2 = < x_5, x_6, x_7, x_8 >$ which are known *a priori* to be orientated orthogonally. In this case, the chromosome would have the form $G = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$ and the orthogonality constraint would then appear as a non-linear inequality of the form: $(x_1 - x_5)^2 + (x_2 - x_6)^2 + (x_3 - x_7)^2 \leq \epsilon$ where ϵ is the constraint tolerance value.

In order to perform the optimisation, Genocop requires a starting position on the constrained manifold. This is the seed for the search points which are then mutated around it. It is also used to produce the set of reference points as described earlier in this section. In our case this means designing a chromosome which is both close to being a concatenation of the individual least-squares results for the part-chromosomes as well as fulfilling the domain and relational constraints. Starting conditions for increasingly complex solutions with increasingly complex constraints have previously been found to be difficult [7] for the general quadric. However, when a parametric representation is used, start conditions become very simple to generate, even when many constraints are used. For example, when a parametric representation for a right, circular cone is used the chromosome consists of seven floating-point genes with one constraint, that of normality for the axis vector. When a general quadric is used, however, the chromosome consists of ten floating-point genes with six constraints to ensure its form. When further constraints are added, a start condition for the reference population becomes difficult to find for the general representation but relatively easy for the parametric one (since the only constraint is normality for 3 genes in the whole sequence). The complexity increases as further quadrics are added. Consider a chromosome with three general quadrics representing three right, circular cones. In a general representation this would be 30 floating-point genes together with 15 shape constraints and three relationship constraints. In a parametric form it would be 21 genes and six constraints in total. This reduction is quite marked but clearly relies on knowing *a priori* the classification of the surfaces in the data.

3 Results

In total 70 single-object experiments were carried out, all of which reached successful convergence, i.e the summed error over all points in the data set stablized. The final values for chromosome parameters are used as the test of value for each of the experiments. Where there were significant convergence effects these have been noted. All of the data used for the synthetic surfaces has Gaussian noise added with standard deviation 0.1mm. This is comparable with the laser range finder and therefore represents typical data noise.

3.1 Caps, Rings and Wedges

Spherical Caps

Ten spherical cap datasets were generated for decreasing values of θ from 170^o to 30^o to test feasibility of fitting and speed of convergence. Sphere radii were kept at $10\,mm$ with centre position $(0, 0, 15)$. The number of data points per data set was 1000 so the data at 170^o is much more descriptive of the shape than the data at 30^o, illustrated in the convergence rate, fig.2(a).

One important aspect of these tests is that with $\theta = 30^o$ data the variation in position genes is much higher than with $\theta = 170^o$ (figures 2(b) and 2(c)). Intermediate positions show that the rate of these variations and rate of convergence change gradually with angle. Note that our previous fitting tool which utilises Taubin accumulation [4] fairs similarly in the tests, having almost exactly the same average error per point. After $25,000$ evaluations all radii converged to the correct value given experimental noise.

Cylinder Wedges

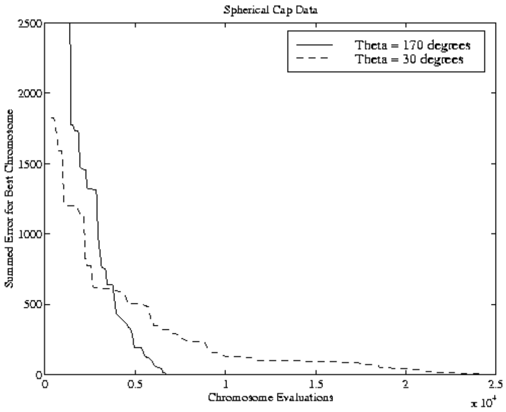
Ten cylindrical wedge datasets were generated for decreasing values of θ from 170^o to 30^o to test feasibility of fitting and speed of convergence. Cylinder radii were kept at 10mm, data was generated around the normal $(0, 0, 1)$ and the starting point was $(15, 10, 10)$. The length of the cylinder in each case was 20mm. Error graphs for the cylinder wedges tell us little about convergence except that it is dependent upon the initial reference population, which the algorithm generates internally. All of the test examples converged within $100,000$ evaluations with the axis correct to within around 0.5^o and radius correct to around 0.05mm, as shown in table 1.

θ	Normal Error/^{o}	Radius Error/mm
170	0.5733	0.0287
155	0.4196	0.0012
140	0.0000	0.0027
125	0.5754	0.0040
110	2.5648	0.1464
90	0.5500	0.0032
70	0.5751	0.0191
50	0.5747	0.0033
45	0.0000	0.1213
30	0.0000	0.1851

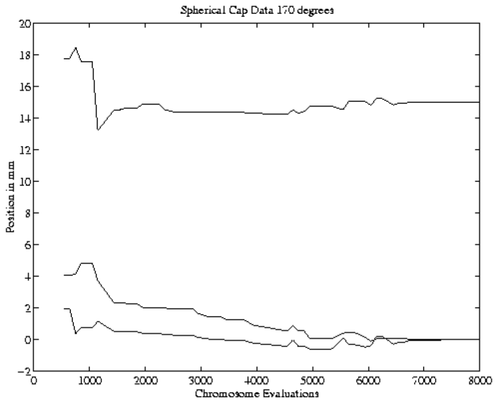
Table 1. Absolute Errors on Cylinder Wedge Parameters

Conical Wedges

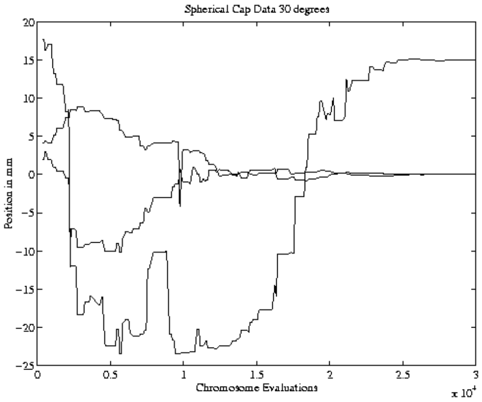
Ten conical wedge datasets were generated for decreasing values of θ from 170^o to 30^o to test feasibility of fitting and speed of convergence. Cone slope half-angle was



(a) Error Convergence for $\theta = 170^o$ and $\theta = 30^o$



(b) Position Genes During Convergence for $\theta = 170^o$ true value is $(0, 0, 15)$



(c) Position Genes During Convergence for $\theta = 30^o$ true value is $(0, 0, 15)$

Fig. 2. Error Graphs for Spherical Datasets

kept at 30^o, axis was $(0, 1, 0)$, apex position was $(0, 25, 0)$ and cone depth was 20mm. 1000 data points were used per set, with noise as before. Details of the results are shown in table 2. The error on the cone normal increases as the wedge decreases. At 30^o it is actually outside acceptable error bounds. This caveat may be explained by the error in the estimate of the apex position, which can be seen to drift as the wedge angle decreases.

θ	Normal Error/^{o}	Slope Angle Error/^{o}	Apex position estimate		
170	0.5815	0.0375	-0.002763	25.008604	0.0161751
155	0.5314	0.4210	0.001005	25.012029	0.0150246
140	0.5747	0.0460	0.005348	25.018583	-0.0153982
125	0.5808	0.0204	0.021809	25.007158	-0.0135571
110	0.5727	0.0007	0.002103	25.007488	-0.0285622
90	0.5757	0.0059	-0.014792	24.991615	-0.0065875
70	0.7000	0.3159	-0.009692	25.162830	-0.1308158
50	1.5526	1.2332	0.006790	25.636304	-0.4279015
45	1.1866	0.8757	0.006669	25.655696	-0.4368368
30	3.0208	2.7954	0.000730	26.736854	-1.0613992

Table 2. Absolute Errors on Cone Wedge Parameters and Apex Estimate

Conical Rings

Ten conical ring datasets were generated for decreasing values of length, from 100mm to 10mm from the base, from a cone of total length 110mm measured base to apex. Cone slope angle was kept at 30^o, axis was $(0, 1, 0)$, apex position was $(0, 25, 0)$. A full 180^o spread was also used and 1000 data points were used per set.

Convergence over all of the conical ring datasets was uniform. Errors are detailed in table 3. Average error for the normal axis was around 0.5^o and the error on the cone slope angle was less than 0.01^o. It can also be seen that the apex position estimates are much better that those in the cone wedge case.

Length from Base	Normal Error/^{o}	Slope Angle Error/mm	Apex Position Estimate		
100	0.5730	0.0020	-0.010790	24.981763	0.027289
90	0.2149	0.0023	-0.014435	25.012355	0.012871
80	0.5630	0.0104	-0.000247	25.026578	-0.005360
70	0.4681	0.0042	0.037935	24.979377	0.002949
60	0.5464	0.0012	0.001648	24.981952	-0.002780
50	0.5723	0.0017	0.005134	24.981803	0.000237
40	0.4681	0.0046	-0.031845	25.003873	0.090405
30	0.3621	0.0021	0.006406	24.984144	-0.004444
20	0.5630	0.0430	0.043338	24.811958	-0.014173
10	0.5835	0.0037	-0.025032	24.973985	-0.342472

Table 3. Absolute Errors on Cone Ring Parameters and Apex Estimate

3.2 Constrained Degenerate Quadric Pairs

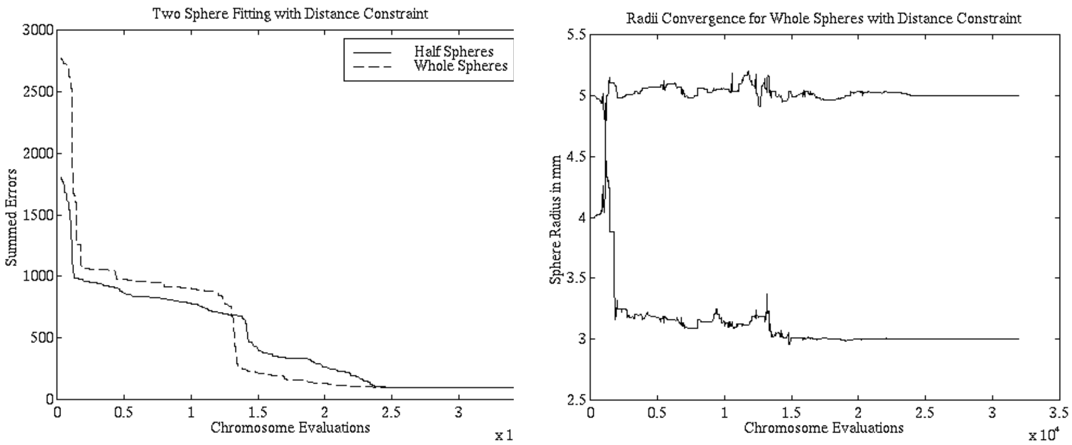
Distance Constraints

Spheres were fitted with the distance constraint applied to their centres. The sphere parameters used were as follows:

Sphere 1: Position $(0, 0, 10)$, Radius 5, complete sphere and half sphere, 1000 data points.

Sphere 2: Position $(0, 0, 0)$, Radius 3, complete sphere and half sphere, 1000 data points.

Summed errors for the fitting are shown in fig. 3(a) and the convergence of the radii genes is shown in fig, 3(b). Position for both spheres was found accurately to within 0.001mm and the radii were found to within 0.01mm. Since the results were similar for both half-spheres and spheres, only whole spheres are shown.



(a) Summed Errors for Constrained Sphere Fitting (Whole Spheres)

(b) Radii Convergence for Constrained Sphere Fitting using Whole Spheres with Distance Constraint (true values are 3mm and 5mm)

Fig. 3. Constrained Sphere Fitting Graphs

Convergence for spheres and half-spheres that were overlapping produced similar results although using slightly more chromosome evaluations. It was also noted that the quality of results obtained was as good as without constraints in both cases, i.e radii to within 0.001mm and position parameters to within 0.005mm.

Mixed Constraints

In mixed constraint experiments whole cones and cylinders were used. In the first set of experiments cones were used with four constraints : two unit normal constraints on the axes, a distance constraint between apexes and an orthogonal constraint on the axes. Data for the cones was as follows:

Cone 1: Apex position $(0, 0, 0)$, Axis $(0, 0, 1)$, Length 20, half-angle $\alpha = 30^o$ or 0.5235988 radians.

Cone 2: Apex position $(0, 0, 20)$, Axis $(0, 0, 1)$, Length 20, half-angle $\alpha = 30^o$ or 0.5235988 radians.

Cone 3: Apex position $(20, 0, 0)$, Axis $(1, 0, 0)$, Length 20, half-angle $\alpha = 30^o$ or 0.5235988 radians.

Once again, Gaussian noise is applied with a standard deviation of 0.1mm.

Two Cones with Apex Distance and Same Axis Constraints

In this experiment, cones 1 and 2 were used. The starting vector for the reference population was simple to find since the constraints on the fitting have been much simplified since earlier efforts to perform this test [7]. Previously, for two right circular cones with a distance constraint and an axis constraint the total number of inter-gene constraints would have been 16, now it is only 3. The cones are generated on the same axis but with apexes 20mm away from each other. The vector for the reference populations was as follows:

$\mathbf{P}_1 = (0, 0, 1), \hat{n}_1 = (0, 0.141, 0.9899), \alpha_1 = 0.5$ radians

$\mathbf{P}_2 = (0, 0, 21), \hat{n}_2 = (0, 0.141, 0.9899), \alpha_2 = 0.5$ radians

which is in the same order as the generation data above. Position for each of the two part-chromosomes is 1mm away from the correct position and the slope angle is 0.024 radians (1.35^o) from the true value. The normal axis is also at an angle of around 0.5^o to the correct value. These values were chosen because they are typical of values found after registration using ICP [1]. The result vector was as follows :

$\mathbf{P}_1 = (0.000216, -0.004444, -0.013367), \hat{n}_1 = (-0.000111, 0.000021, 0.999969)$, $\alpha_1 = 0.523469$ rad.

$\mathbf{P}_2 = (0.002930, -0.007804, 19.986713), \hat{n}_2 = (-0.000325, -0.000021, 0.999951)$, $\alpha_2 = 0.523393$ rad.

This shows that the convergence to the correct cone models and constraint satisfaction is remarkably good, to the noise level on the data set.

Two Cones with Apex Distance and Orthogonal Axis Constraint

In this experiment, cones 1 and 3 were used. The cones were generated on the same axis but with apexes 20mm away from each other. The vector for the reference populations was as follows:

$\mathbf{P}_1 = (0, 0, 1), \hat{n}_1 = (0, 0.141, 0.9899), \alpha_1 = 0.5$ radians

$\mathbf{P}_2 = (0, 0, 21), \hat{n}_2 = (0, 0.141, 0.9899), \alpha_2 = 0.5$ radians

The result vector was as follows :

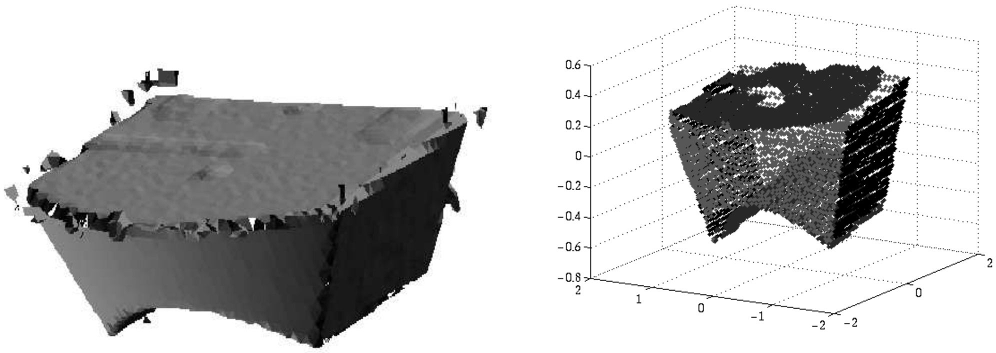
$\mathbf{P}_1 = (-0.006925, -0.001579, -0.024732), \hat{n}_1 = (0.000279, -0.000089, 0.999950)$, $\alpha_1 = 0.523178$ rad.

$\mathbf{P}_2 = (19.993097, -0.012630, -0.000523), \hat{n}_2 = (0.999949, 0.000792, -0.000378)$, $\alpha_2 = 0.523704$ rad.

This shows that the convergence to the correct cone models and constraint satisfaction.

3.3 A Real Object

In order to test the overall application of these technique a reasonably complex real part was examined. This machined part (called the UFO) is an object consisting of six surfaces, four planes and two quadrics as shown in figure 4. It is made to high tolerances but is formed from eight data sets which are then registered together.



UFO Part as Polygons UFO Part after Segmentation

Fig. 4. Real Object Rendered as Polygons and Vertices

The object was first segmented into individual surfaces using a region growing method and Taubin accumulation for the fitting [13]. Each of the different surfaces was saved as a 3d data set. There were 7274 data points in total, representing the polygon centres. This is a valid subsampling since the polygon mesh is statistically representative of the original data set (many tens of thousands of points). The chromosome representing the constrained shape consisted of 30 individual genes as follows: $G = A_1, ..., A_6$, where A_1 is the 7 parameter cylinder, A_2 is the 7 parameter cone and A_3 to A_6 are the 4 parameter planes. A total of 11 constraints were used, the first 6 were unit normal constraints and the final 5 were geometric as follows:

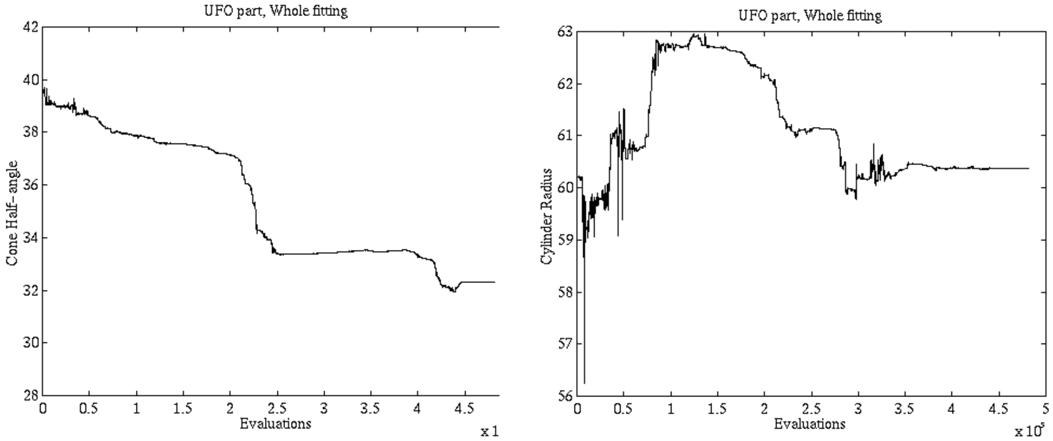
1. Cylinder axis is the same as that of the back plate normal.
2. Cone axis is the same as the bottom plate normal
3. The back plate normal is orthogonal to the bottom plate
4. The back plate normal is orthogonal to the sloping side (side 1)
5. The sloping sides are at 120^o to each other

These five constraints fully constrain the object's shape. The start conditions for this object were simply the least-squares fit for each of the surfaces which were then adjusted to fit the constraints (obviously sub-optimally). The controlling ground-truth used for this test was that the cylinder radius was known to be 60mm and the cone half-angle was known to be 30^o from the normal at its apex.

The graphs in figure 5 show the convergence of the parameters 500,000 evaluations.

The full set of estimations for the parameters is given below:

– *Cylinder :* $\mathbf{P}_1 = 154.086, -2.184, -57.711, \hat{n}_1 = 0.995561, 0.005837, 0.088583$, radius = 60.366mm.
– *Cone :*$\mathbf{P}_2 = 45.307, -5.697, -137.595, \hat{n}_2 = -0.044865, 0.039102, 0.998023$, $\alpha = 32.3^o$
– *Planes*
  1. $\hat{n}_1 = 0.995425, 0.003886, 0.091327$, d = 46.62732)
  2. $\hat{n}_2 = 0.046830, -0.885180, -0.461808$, d = 47.79109



(a) Cone Half Angle for Constrained UFO Fitting **(b)** Cylinder Radius for Constrained UFO Fitting

Fig. 5. Constrained UFO Fitting Graphs

3. $\hat{n}_3 = 0.041421, 0.845461, -0.531490$, d = 47.06203
4. $\hat{n}_4 = -0.092407, 0.041308, 0.995308$, d = 14.12157

The numbers of points used was as follows: cylinder 1896, cone 1386, back-plate 616, side1 767, side2 839, bottom plate 1770. Time to reach a stable solution was 218.75 minutes when running on a 269MHz Ultrasparc 10 at approx 50% CPU usage. These results have several important aspects. Firstly, the cone half angle is within the same margin of error as the synthetic data when only a 30^o wedge is used. This is explained by the fact that not only is this a 60^o wedge of a cone but it is truncated at only 30mm out of its 140mm height. The cylinder parameters are very good since the cylinder data describes only a 60^o wedge of the original data. It should also be noted that all of the constraints are satisfied to within the tolerance prescribed ($\epsilon = 0.001$) which represents 0.0573^o error on axis constraints. The summed least-squares error (Euclidean rather than algebraic) was 19.8367 over the whole of the six surfaces. This should also be seen in light of the fact that there were around 1% of outliers in the whole dataset and the registration process is almost certainly imperfect.

4 Conclusions

4.1 Improvements

In previous work [7] we showed that optimal surface fitting under geometric constraints was feasible with an evolutionary algorithm. In this paper we have demonstrated that the method is more accurate when parametric models are used for degenerate surfaces. We have also demonstrated that the geometric constraints for these models are much simpler to formulate. This leads to the quicker formulation of starting reference populations.

We have shown that even a naive point-assignment algorithm can perform simple segmentation when used in conjunction with geometric constraints and we have thus tentatively proposed a full processing system for range-data. The pptimization of least-squares surface fitting is comparable to methods currently employed for second-order surfaces. In some instances (for example where data is sparse) it is actually superior. This is due partially to the fact that data presentation is not ordered as it is in, say, the Taubin accumulation process [3],[4]. No claims are made for the segmentation algorithm other than in circumstances where two sphere datasets overlap if a distance constraint is applied the algorithm still convergenes in a time comparable to no-overlap. Therefore segmentation is a plausible addition to the functionality of the algorithm. The application of geometic and domain constraints has ensured that the convergence to the optimal solution (subject to tolerance) has been achieved. When geometric constraints have been applied they also have been fulfilled (subject to tolerance). These constraints have been : distance, axis normality and relative axis position. These are the only constraints applicable to the degenerate surfaces we have examined.

4.2 Caveats - New and Old

Of the problems mentioned in the first paper [7], none are now applicable. The formulation of the initial vector for the reference set is easy for parameterized chromosomes. Generating two right cirular cones with orthogonal vector normals, for example, using a parameterized model is straightforward whereas using the previous representation was difficult. When these initialisations were compounded the task was infeasible.

The problem of traversing the space different amounts in different dimensions is somewhat mitigated by this parameterizarion since the units are at least within the same order. The problem of what the slack constraint variables actually *mean* is still different for each constraint. This problem is not so pronounced here as in other methods, for example [5].

One new caveat is that the parametric versions of the objective functions are not so easy to speed up by using off-line calculations. This can mean that for objects involving many (possibly hundreds of) thousands of points the least-squares error calculations will be time-consuming. A simple sub-sampling scheme has been implemented where at each iteration a sample from such a huge dataset is taken and errors computed from this. Although initial tests have been positive, the rates of convergence are not as predictable as those found in this paper - even if the time to convergence is much lower. It is widely held that evolutionary schemes, in fact GAs as a whole, are quick to implement but slow to run. In the main this is true but with a computationally complex evaluation function it is doubly so.

4.3 Further Work

This work is a **proof of concept**, i.e that an evolutionary algorithm could solve the problem of constrained surface fitting, and as such is complete. There are many side issues that should be addressed: speeding-up the chromosome evaluation; including data registration, although how is not clear; outlier removal from the data at run time would provide modest improvements; and weighting the surface errors to skew the fit towards more important surfaces.

Acknowledgements

The work presented in this paper was funded by a UK EPSRC grant GR/H86905. The authors would also like to thank Andrew Tuson and David Corne for advice on several aspects of this work.

References

1. D. Eggert, A. W. Fitzgibbon, R. B. Fisher. "Simultaneous registration of multiple range views for use in reverse engineering", Proc. Int. Conf. on Pat. Recog., pp 243–247, Vienna, Aug. 1996.
2. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle. "Surface Reconstruction from Unorganized Points", Computer Graphics, 26(2), pp 71–78, 1992.
3. G. Taubin. "Estimating the tensor of curvature of a surface from a polyhedral approximation", Proc. 5th Int. Conf. on Computer Vision, pp 902-907, 1995.
4. G. Taubin. "Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation", Proc. IEEE PAMI, 13(11), pp1115–1138, 1991.
5. N. Werghi, R. B. Fisher, A. Ashbrook, C. Robertson, "Improving model shape acquisition by incorporating geometric constraints", Proc. British Machine Vision Conference BMVC97, Essex, pp 520–529 September 1997.
6. Z. Michalewicz, D. Dasgupta, R. G. Le Riche and M. Schoenauer, "Evolutionary algorithms for constrained engineering problems", special issue on *Genetic Algorithms and Industrial Engineering*, ed. M. Gen, G.S.Wasserman and A. E. Smith, *International Journal of Computers and Industrial Engineering*, 1996.
7. C. Robertson, D. Corne, R. B. Fisher, N.Werghi, A.Ashbrook, "Investigating Evolutionary Optimisation of Constrained Functions to Capture Shape Descriptions from Range Data", Proc. 3rd On-line World Conference on Soft Computing (WSC3) (see `http://www.cranfield.ac.uk/wsc3/` also in Advances in Soft Computing - Engineering Design and Manufacturing, eds. Roy, Furuhashi and Chawdhry, Springer-Verlag, 1998.
8. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Third Edition, Springer, 1996.
9. A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, R. Fisher, "An Experimental Comparison of Range Segmentation Algorithms", IEEE Trans. Pat. Anal. and Mach. Intel., Vol 18(7), pp673–689, July 1996
10. M. Gen and R. Cheng, "A Survey of Penalty techniques in Genetic Algorithms", In *Proceedings of the IEEE International Conference on Evolutionary Computation 1996*, 1996.
11. Z. Michalewicz and N. Attia, "Evolutionary Optimization of Constrained Problems", in , *Proceedings of the 3rd Annual Conference on Evolutionary Programming*, San Diego, CA, 1994, pages 98-108. World Scientific.
12. N. F. Attia, *New Methods of Constrained Optimization Using Penalty Functions*, Ph.D Thesis, Essex University, United Kingdom, 1985.
13. R. B. Fisher, A. W. Fitzgibbon, D. Eggert, "Extracting Surface Patches from Complete Range Descriptions", Proc. Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling, Ottawa, Canada, pp 148-155, May 1997.