Shape Reconstruction Incorporating Multiple Non-linear Geometric Constraints

Naoufel Werghi, Robert Fisher, Anthony Ashbrook and Craig Robertson
Division of Informatics, University of Edinburgh
5 Forrest Hill, Edinburgh EH1 2QL, UK
Email: {naoufelw, rbf, anthonya, craigr}@dai.ed.ac.uk

Abstract. This paper deals with the reconstruction of 3D geometric shapes based on observed noisy 3D measurements and multiple coupled non-linear shape constraints. Here a shape could be a complete object, a portion of an object, a part of a building etc. The paper suggests a general incremental framework whereby constraints can be added and integrated in the model reconstruction process, resulting in an optimal trade-off between minimization of the shape fitting error and the constraint tolerances. After defining sets of main constraints for objects containing planar and quadric surfaces, the paper shows that our scheme is well behaved and the approach is valid through application on different real parts. This work is the first to give such a large framework for the integration of numerical geometric relationships in object modelling from range data. The technique is expected to have a great impact in reverse engineering applications and manufactured object modelling where the majority of parts are designed with intended feature relationships.

Keywords: Reverse engineering, geometric constraints, constrained shape reconstruction, shape optimization

Abbreviations: CAD – Computer Aided-design; 3D – Three-Dimensional; LS – Least squares

Table of Contents

1	Introduction	2
2	Related work	4
3	The geometric constraints	6
4	Optimization of shape satisfying the constraints	8
5	Implementation	14
6	A simple example	14
7	Experiments	17
8	Conclusion	31

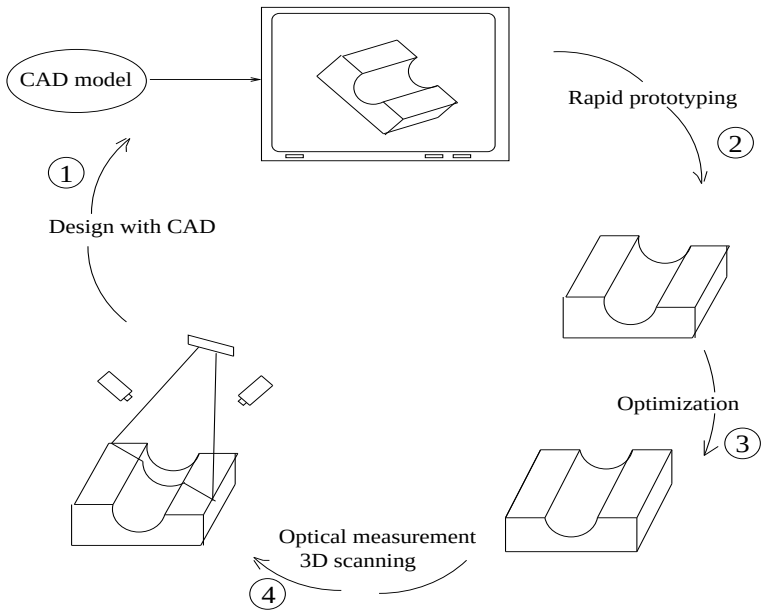


Figure 1. The production-perfection cycle of a part

1. Introduction

The framework of this work is reverse engineering. In parts manufacturing, reverse engineering is typically concerned with measuring an existing object so that a surface or solid model can be deduced in order to take advantage of CAD/CAM technologies. It is also often necessary to produce a copy of a part when no original drawings or documentation are available. In other cases we may want to re-engineer an existing part, when analysis and modifications are required to construct a new improved product. Even though it is possible to turn to a computer-aided design to fashion a new part, it is only after the real model is made and evaluated that we can see if the object fits the real world. For this reason designers rely on real 3D objects (real scale wood, clay models) as starting point. Such a procedure is particularly important to areas involving aesthetic design e.g. the automobile industry or generation of custom fits to human surfaces such as helmets, space suits or prostheses. For these reasons reverse engineering is a fundamental step of the now-standard production-perfection cycle of part (Figure.1). This process starts with the CAD stage. Next (step 2), the rapid prototyping stage converts the CAD data into a real prototype. Rapid prototyping is a technique allowing the direct production of prototypes by a computer-controlled process. Often, the shape of the produced object undergoes some improvement carried out by hand to adapt it to its real environment (step 3). The hand-improved model is

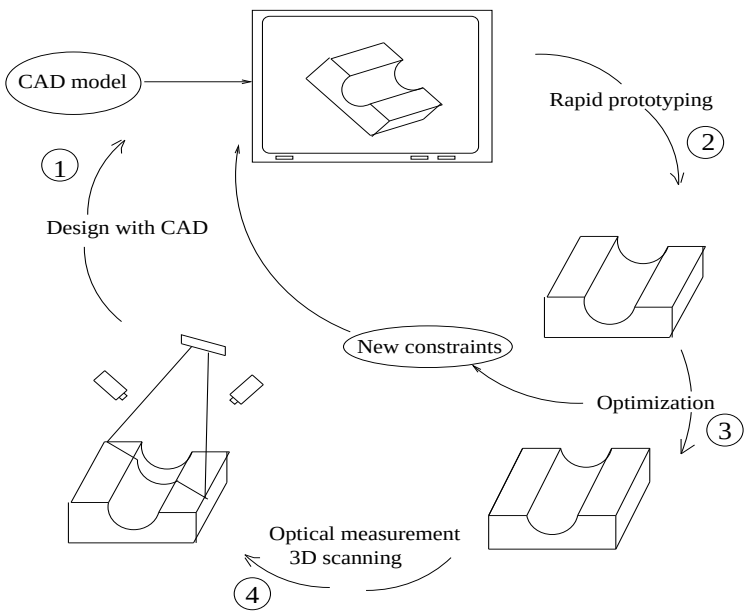


Figure 2. Many hand-worked optimization (step 3) could be replaced by establishing new constraints on the shape and incorporating them in the model design process.

back again into the digital world of CAD through 3D optical measurement techniques (step 4), for instance a 3D laser scanner.

In this process the notion of constraints is normally involved in step 1 where geometric relationships between object features together with 3D measurement data contribute in the production of the optimal object model shape.

The first motivation behind incorporating geometric constraints is that models needed by industry are generally designed with intended geometric relationships between the object features so this aspect should be exploited rather than ignored. The consideration of these relationships is actually necessary because some attributes of the object would have no sense if the object modelling scheme did not take into account these constraints. For example, take the case when we want to estimate the distance between two parallel planes: if the plane fitting results gave two planes which are not parallel, then the distance measured between them would have no significance. Furthermore exploiting the available known relationships would be useful for reducing the effects of registration errors and mis-calibration, thus improving the accuracy of the estimated part features' parameters and consequently the quality of the modelling.

The second motivation is that once the part is produced (step 2) many improvements are carried manually (step 3) to optimize the part and make it fit with the real world (e.g fit with another part, adjust the part to fit a particu-

lar customer). These improvements could be represented by new constraints on the part's shape. By integrating these constraints into the CAD design process step (Figure.2) the work piece optimization would be reduced to the minimum tasks and hence many cycles in the part production process would be saved. In other cases, such improvements could not be achieved by hand due to the complexity of the object or when we want to extend the application of the process to complex environments such as buildings or industrial plants.

Our problem is presented as follows: Given sets of 3D measurement points representing surfaces belonging to a certain object, we want to estimate the different parameters of the surfaces, taking into account the geometric relationships between these surfaces and the specific shapes of surfaces as well.

A state vector $\vec{p}$ is associated to the object, which includes all parameters related to the patches. The shape defined by the parameter vector $\vec{p}$ has to best fit the data while satisfying the constraints. Consider $F(\vec{p})$ to be an objective function defining the relationship between the set of data and the parameters and $C_k(\vec{p})$, $k = 1..M$ the set of constraint functions defining the geometric constraints. $C_k(\vec{p})$ is a vector function associated with constraint k. The problem can be then stated as follows: Find the parameter vector $\vec{p}$ minimizing the function $F(\vec{p})$ subject to the constraints

$$C_k(\vec{p}) \leq \tau_k, \quad k = 1..M \tag{1}$$

Here τ_k represents the tolerance related to the constraint C_k. Ideally the tolerances have zero values, but practically, for geometric constraints they are assigned certain values which reflect the allowed geometric inaccuracies in the relative locations and shapes of features. It is up to the designer to set the tolerances, however an appropriate definition of the tolerances for a given object can be set up by using the scheme developed by Requicha [16].

As a simple example consider the three surfaces of a tetrahedron (Fig.3). The surfaces have three orientation constraints reflecting the three angles 90^0, 90^0 and 120^0 between the three surface normals. Consider $\vec{p}$ a vector containing the parameters of the surfaces, $\vec{p}$ has then to fit the data points associated with the surfaces, minimizing a least squares error function and also satisfying the three constraint functions associated to the surface orientations.

2. Related work

A review of the main reverse engineering research in the CAD community [7, 18, 19, 22] revealed that the exploitation of geometric constraints has not been fully investigated. This lack was discussed in the survey work of Varady *et al* [20].

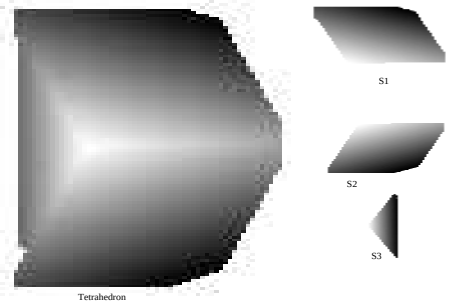


Figure 3. The tetrahedron object with the extracted surfaces

Incorporating geometric relationships in object modelling has to tackle two problems. The first is how to represent the constraints. The second is how to integrate these constraints into the shape fitting process. These two aspects are not entirely independent, the shape fitting technique imposes restrictions on the constraint representation and vice versa.

A first step in the direction of incorporating constraints for ensuring the consistency of the reconstruction was done by Porrill [15]. He linearized a set of nonlinear constraints and combined them with a Kalman filter applied to wire frame model construction. Porrill's method takes advantage of the recursive linear estimation of the Kalman filter, but guarantees satisfaction of the constraints only to linearized first order. Additional iterations are needed at each step if more accuracy is required. This last condition has been taken into account in the work of De Geeter *et al* [4] by defining a "Smoothly Constrained Kalman Filter". The key idea of their approach is to replace non-linear constraints by a set of linear constraints applied iteratively and updated by new measurements in order to reduce the linearization error. However, the characteristics of Kalman filtering makes these methods essentially adapted for iteratively acquired data and many data samples. Moreover, there was no mechanism for determining how successfully the constraints were satisfied and only lines and planes were considered in both of the above works.

The constraints considered by Bolle *et al* [2] in their approach to 3D object position covered only the shape of the surfaces. They chose a specific representation for the treated features: plane, cylinder and sphere.

Compared to Porrill's and De Geeter's work, our approach avoids the drawbacks of linearization, since the constraints are completely implemented. Moreover, our approach covers a larger category of feature shapes. Regarding the work of Bolle [2], the type of constraints which can be held by our approach go beyond the restricted set of surface shapes and cover also the geometric relationships between object features. To our knowledge the work appears to be the first to give such a large framework for the integration of geometric relationships for object reconstruction.

3. The geometric constraints

The set of constraints associated with a given object can be divided mainly into two categories. The first one is the surface intrinsic constraints covering the geometric properties which arise from the specific shapes of the surfaces. This category includes particular properties of the surface such as symmetry with respect to a point or a line. For quadric surfaces such as cones or cross-section cylinders this property is the circular shape of the surface.

The second category named, the feature extrinsic constraints, defines the geometric and topological relationships between the different object features. Table I summarizes these relationships. We notice here that points and lines in this table may be either physical features of the object like summits or vertices and edges or implicit features like centres, axes of symmetry. This list is not exhaustive and this classification may not be unique. Nevertheless it covers a large number of constraints in manufactured objects.

Table I. Relationships between features.

	point	line	plane	quadric surface
point	coincident separation	inclusion separation	inclusion separation	inclusion separation
line	-	coincident relative orientation separation	inclusion relative orientation separation	inclusion relative orientation separation
plane	-	-	coincident relative orientation separation	relative orientation separation
quadric surface	-	-	-	coincident relative orientation separation

3.1. COINCIDENCE CONSTRAINTS

Shapes commonly contain features which are associated to the same geometric entity (Figure.4.a) or which coincide at the same position (Figure.4.b). In the first case these constraints are implicitly imposed by considering the same parameters for each feature. In the second case the parameters associated to each feature are equated and the resulting equations have then to be satisfied.

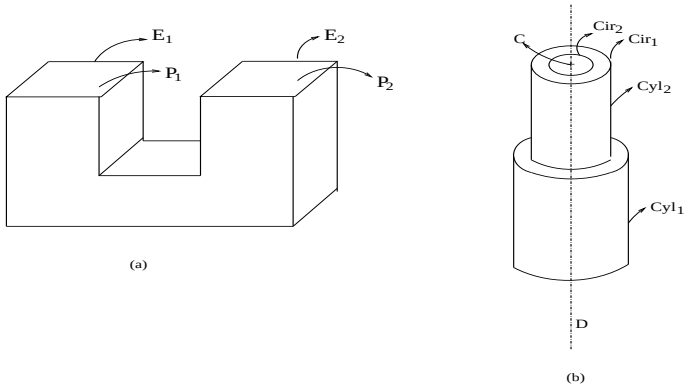


Figure 4. (a): The two edges E_1 and E_2 belong to the same line. The two faces P_1 and P_2 are associated to the same plane. (b) The centres of the circles Cir_1 and Cir_2 coincide at the same point C. The cylinders Cyl_1 and Cyl_2 have a common axis.

3.2. INCLUSION CONSTRAINTS

A particular feature point may be included in an object feature e.g. line, plane or quadric patch. Similarly a feature line may be included in a plane or a particular quadric surface (Fig.5) such as a cylinder and a cone.

3.3. RELATIVE ORIENTATION CONSTRAINT

There are many orientation relationships which can be deduced and exploited in a given part, such as the two common particular cases of parallelism and orthogonality (Fig.6.a). The presence of these two characteristics is easily detected in an object.

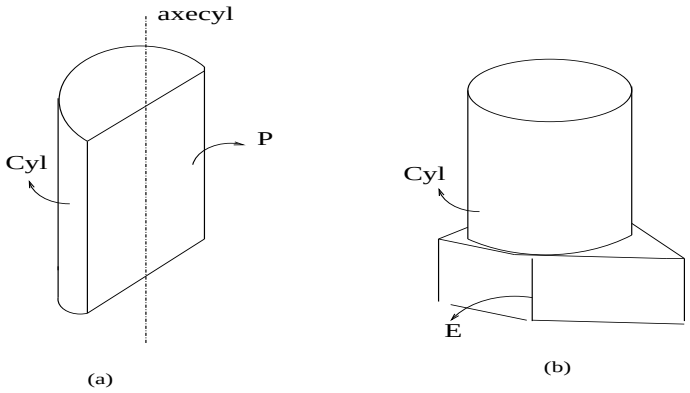


Figure 5. (a): The axis of the cylinder patch *Cyl* is included in the plane *P*. (b) The line associated to the edge *E* is included in the cylinder *Cyl*.

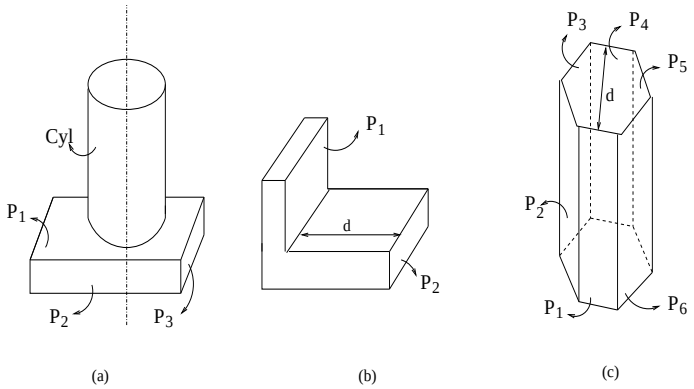


Figure 6. (a): Each pair of planes (P_1, P_2, P_3) makes an angle of 90^o, the axis of the cylinder *Cyl* is orthogonal to P_1. (b): The planes (P_1, P_2) are separated by distance d. (c): Each pair of parallel planes of the hexagonal prism are separated by the same distance.

3.4. RELATIVE SEPARATION CONSTRAINT

The relative separation between features can be exploited when the distance between parallel features (Fig.6.b) is already known or needs to be imposed or when the object has a symmetry aspect leading to some separation distance relationships (Fig.6.c).

3.5. OTHER CONSTRAINTS

There are also other types of constraints like those imposed directly on the surface parameters as a consequence of the surface representation e.g. the representation of a plane by the equation $ax+by+cz+d=0$ where $[a,b,c]$ is normal vector to the plane and d is the distance of the plane to the origin requires that the sum of the squared elements of the normal to be equal to one. Such constraints are called the unit constraints.

4. Optimization of shape satisfying the constraints

Given sets of 3D measurement points representing surfaces belonging to a certain object, we want to estimate the different surface parameters, taking into account the geometric relationships between these surfaces and the specific shapes of surfaces as well.

A state vector $\vec{p}$ is associated to the object, which includes all parameters related to the different patches. The vector $\vec{p}$ has to best fit the data while satisfying the constraints. Consider $F(\vec{p})$ to be an objective function defining the relationship between the measured data points and the parameters.

This function is generally a minimization criterion (e.g. sum of least squares residuals, maximum likelihood function, etc.).

Consider $C_k(\vec{p})$, $k = 1..M$, the set of constraint functions defining the geometric constraints where $C_k(\vec{p})$ is a vector function associated with constraint k. The problem can be then stated as follows:

$$\begin{array}{ll} \text{minimize} & F(\vec{p}) \\ \text{subject to the constraints} & C_k(\vec{p}) \leq \tau_k, \quad k = 1..M \end{array} \tag{2}$$

Thus the problem which we are dealing with is a constrained optimization problem.

4.1. THE OBJECTIVE FUNCTION

Consider $S_1,.,.S_N$ the set of surfaces and $\vec{p}_1,.,.\vec{p}_N$ the set of parameter vectors related to them. Each vector $\vec{p}_i$ has to minimize a given surface fit error criterion J_i associated with the surface S_i such as the least squares error criterion. The set of the parameter vectors has then to minimize the following object function:

$$J = J_1 + J_2 + ...J_N \tag{3}$$

By considering a polynomial description of the surfaces, each surface S_i can be represented by:

$$\vec{h_i}^T \vec{p}_i = 0 \tag{4}$$

where $\vec{h_i}$ is the measurement vector with each component of the form $x^\alpha y^\beta z^\gamma$ for some (α,β,γ). For instance a plane surface defined by the equation $ax + by + cz + d = 0$, has the measurement vector is $\vec{h} = [x,y,z,1]^T$. For a sphere defined by $a(x^2+y^2+z^2) + 2ux + 2vy + 2wz + d = 0$, it is $\vec{h} = [x^2+y^2+z^2,x,y,z,1]$

This formulation has the advantage to lead to a compact quadric expression of the objective function because of its linearity with respect to the parameters. Indeed, given m_i measurements, the least squares criterion related to the equation (4) is

$$J_i = \sum_{l=1}^{m_i} (\vec{h_l}^T \vec{p}_i)^2 = \vec{p_i}^T H_i \vec{p}_i \tag{5}$$

where $H_i = \sum_{l=1}^{m_i} (\vec{h_l}\vec{h_l}^T)$ represents the sample covariance matrix of the surface S_i. By concatenating all the vectors $\vec{p_i}^T$ into one vector $\vec{p} = [\vec{p_1}^T, \vec{p_2}^T, ., ., ., \vec{p_N}^T]^T$ equation (3) can be written as a function of the parameter vector $\vec{p}$ and we get the following objective function:

$$F(\vec{p}) = J = \vec{p}^T \mathcal{H} \vec{p}, \qquad \mathcal{H} = \begin{bmatrix} H_1 & (0) & . & (0) \\ (0) & H_2 & . & (0) \\ (0) & . & . & (0) \\ (0) & . & (0) & H_N \end{bmatrix} \tag{6}$$

Such a function is convex if and only if the matrix $\mathcal{H}$ is positive, which is the case. Besides, under the above form, the objective equation contains separate terms for the data and the parameters. The data matrix $\mathcal{H}$ can be thus computed off-line before the optimization.

The objective function could be taken as the likelihood of the range data given the parameters (with a negative sign since we want to minimize). The likelihood function has the advantage of accounting for the statistical aspect of the measurements. As a first step, we have chosen the least squares function. The integration of the data noise characteristics in the LS function can be done afterwards with no particular difficulty, leading to the same estimation of the likelihood function in the case of the Gaussian distribution.

4.2. CONSTRAINT FORMULATION

The different constraints are implemented under a matrix formulation. The matrix notation leads to a compact form and avoids expressions with many variables in particular for the second order derivatives that may be eventually needed in the optmization algorithm. This allows a fast, automatic and easy implementation of the constraints.

Some intrinsic constraints, for instance circularity of quadric surfaces could be imposed implicitly by choosing a suitable form of the surface equation. However, the implementation of the reduced form in the optimization algorithm may cause some complexity. Indeed, because of the nonlinearity of these forms, it has not been possible to get an objective function with separated terms for the data and the parameters. Thus, the data terms could not be computed off-line. This may increase the computational cost dramatically. Examples of how constraints can be implemented are found in section 6.

4.3. THE OPTIMIZATION ALGORITHM

Optimization techniques fall into two broad branches namely Operation Research techniques and the recent evolutionary techniques.

Evolutionary computation techniques [10, 11] have been having increasing attraction for their potential to solve complex problems. In short they are stochastic optimization methods. They are conveniently presented using the metaphor of natural evolution: they start from a randomly generated set of points or solutions of the search space (population of individuals). Then this set evolves following a process close the natural selection principle. At each stage a new population is generated using simulated genetic operations such as mutation or crossover. The probability of survival of the new solutions depends on how well they fit a given evaluation function. The best are kept with high probability and the worst are discarded. This process is repeated until the set of solutions converges to the one best fitting the evaluation function.

The main advantages of the evolutionary techniques is that they do not have many mathematical requirements about the optimization problem. They are 0-order methods, in the sense that they operate only on the objective function and they can handle linear or nonlinear problems, constrained or unconstrained.

The main drawback of these techniques is that they are highly time consuming. This is due to the fact that to ensure convergence, the number of generated solutions has to be high, and at each iteration all the solutions have to be evaluated. This increases the computation time dramatically.

The second branch of the optimization techniques are the classical operation research techniques. They are more mature than the evolutionary techniques. They involve search techniques, numerical analysis and differential tools. Most of these techniques use an iterative scheme. A reasonable initialisation causes significant speedup in convergence. A detailed review and analysis of these optimization techniques could be found in [8, 9].

We believe that the evolutionary techniques are suitable mainly to the optimization cases where objective functions and constraints are very complex, presenting hard-handled aspects such nonlinearity, non-differentiability, or do have not explicit forms. Indeed the earlier mentioned characteristics of the evolutionary techniques allow them to by-pass these problems.

As our optimization problem does not have these problems, the operational research techniques are more appropriate. This argument is supported by the time-consuming characteristic of the evolutionary techniques, where the average scale of the processing time is on the order of hours. This characteristic makes these methods not appropriate for interactive user environments and impractical for a static verification and checking of the results when experiments have to be repeated many times. The other important reason for opting for search techniques is that we can obtain a reasonable initial estimate of the model parameters. This initial solution is the estimation of the model parameters without considering the constraints. This estimation is not far away from the optimal one since it is obtained from the real object prototype.

Theoretically a solution of the problem stated in (2) is given by finding the set $(\vec{p}, \lambda_1, \lambda_2, .., \lambda_k)$ minimizing the following equation:

$$\begin{aligned} E(\vec{p}) &= F(\vec{p}) + \sum_{k=1}^{M} \lambda_k C_k(\vec{p}) \\ F(\vec{p}) &= \vec{p}^T \mathcal{H} \vec{p} \\ C_k(\vec{p}) &= \vec{p}^T A_k \vec{p} + B_k^T \vec{p} + C_k \end{aligned} \tag{7}$$

Under the Khun-Tucker conditions [8](Chapter 9), namely that the objective function and the constraint functions are continuously differentiable and the gradients of the constraint functions are linearly independent, the

optimal set $(\vec{p}, \lambda_1, \lambda_2, .., .., \lambda_k)$ minimizing (7) is the solution of the system:

$$\frac{\partial F}{\partial \vec{p}} + \sum_{k=1}^{M} \lambda_k \frac{\partial C_k}{\partial \vec{p}} = 0 \tag{8}$$

In some particular cases it is possible to get a closed form solution for (8) such as the generalized eigenvalues methods. This depends on the characteristics of the constraint functions and whether it is possible to combine them efficiently with the objective function. When the constraints are linear (having the form $A\vec{p} + B = 0$) the standard quadratic programming methods could be applied to solve this system.

However the geometric constraints are mainly non-linear. Generally it is not trivial to develop an analytical solution for such problem. In this case an algorithmic numerical approach could be of great help taking into account the increasing capabilities of computing.

Now if we look to the objective function and the constraint functions in (7) we see that they are explicitly defined as a function of the parameters, they are smooth, differentiable and they both have a quadratic structure. From (5) we can notice that each submatrix H_i of $\mathcal{H}$ in (6) is the sum of cross-product terms $\vec{h_i^l}\vec{h_i^l}^T$. Thus H_i as well as $\mathcal{H}$ are positive definite. Consequently the objective function is convex. Such functions could be efficiently minimized. Besides it has the important property that its minimum is global. If the constraint functions are squared, thus enforced to be also convex, the optimization problem (7) would be a convex optimization problem for $\lambda_k > 0$. For such problem an optimal solution exists, moreover this solution corresponds to the solution of the system (8) defined by the Khun-Tucker conditions [17](section 27,28).

The problem would be to determine the set $(\vec{p}, \lambda_1, \lambda_2, .., .., \lambda_k)$ minimizing:

$$E(\vec{p}) = F(\vec{p}) + \sum_{k=1}^{M} \lambda_k (C_k(\vec{p})^2),\ \lambda_k > 0 \tag{9}$$

To provide a numerical solution of this problem we have been investigating an approach in the framework of sequential unconstrained minimization. The basic idea is to attach different penalty functions to the objective function $F(\vec{p})$ in such a way that the optimal solutions of successive unconstrained problems approach the optimal solution of the problem (9). Indeed the term $\sum_{k=1}^{M} \lambda_k (C_k(\vec{p})^2)$ could be seen as a penalty function controlling the constraints satisfaction. The scheme then increments the set of λ_k iteratively, at each step minimize (9) by a standard non-constrained technique, update the solution $\vec{p}$, and repeat the process until the constraints are satisfied. For equal values of λ_k, Fiacco and McCormick [6] have shown that the solutions of (9) converge towards the same solution of the problem (2) when λ_k tends to infinity.

In more detail the proposed algorithm is: We start with a parameter vector $\vec{p}^{[0]}$ that minimizes the least squares objective function and attempt to find a nearby vector $\vec{p}^{[1]}$ that minimizes (9) for small values λ_k. Then we iteratively increase the set of λ_k slightly and solve for a new optimal parameter $\vec{p}^{[n+1]}$ using the previous $\vec{p}^{[n]}$. At each iteration *n*, the algorithm increases each λ_k by a certain amount and a new $\vec{p}^{[n]}$ is found such that the optimization function is minimized by means of the standard Levenberg-Marquardt algorithm (see Appendix). The parameter vector $\vec{p}^{[n]}$ is then updated to the new estimate $\vec{p}^{[n+1]}$ which becomes the initial estimate at the next values of λ_k. The algorithm stops when the constraints are satisfied to the desired degree or when the parameter vector remains stable for a certain number of iterations. A simplified version of the algorithm is illustrated in Figure 7.a in which a single λ is associated to the constraints. At each iteration λ is increased by multiplying it by a factor inversely proportional to the constraint value decrease.

A computational problem associated with this algorithm emerges when λ_k become too large. This problem arises in the Hessian matrix of the optimization function (9) involved in Levenberg-Marquardt algorithm. This matrix becomes ill-conditioned for high values of λ_k. To overcome this problem we have used the technique developed by Broyden *et al* [3] for updating the parameter vector $\vec{p}$ at the level of the Levenberg-Marquardt algorithm.

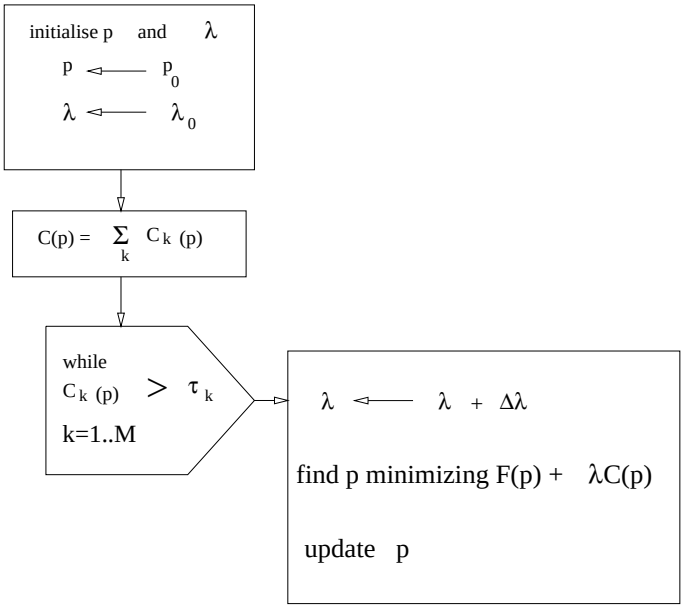


Figure 7. Optim: the constraint optimization algorithm.

The initialization of the parameter vector is crucial to guarantee the convergence of the algorithm to the desired solution. For this reason the initial vector was the one which best fitted the set of data in the absence of constraints. This vector can be obtained by estimating each surface's parameter vector separately and then concatenating the vectors into a single one. Naturally, the option of minimizing the objective function $F(\vec{p})$ alone has to be avoided since it leads to the trivial null vector solution. On the other hand, the initial values λ_k have to be large enough to avoid the above trivial solution and to give the constraints a certain weight. A convenient value for the initial λ_k is :

$$\lambda_k^{[0]} = \frac{F(\vec{p}^{[0]})}{C_k(\vec{p}^{[0]})} \tag{10}$$

where $\vec{p}^{[0]}$ is the initial parameter estimation obtained by concatenating the unconstrained estimates.

5. Implementation

First, the algorithm was developed and implemented under MATLAB, mainly to check the behaviour and the convergence of the algorithm as well as the validity of the results. This version rapidly turned out to be inconvenient since a new implementation is needed to be done for each part. The next step was then to develop a program which can hold any part and automatically convert the information given by the user about the object (surfaces and constraints) into a structure (set of objective function and constraint functions) ready to be integrated into the optimization algorithm. A simple constraint language compiler was developed under C++ for this purpose. The input file is a list of statements in which the user declares the surfaces, their identifications and the files where the associated 3D measurement points are stored. Then the constraints are declared with their associated values and tolerances. Figure 8 shows the structure of the input file and the language statements.

The whole package (the compiler and the optimization algorithm) has been implemented on a 200*Mhz* SUN Ultrasparc workstation. The computation time is in the range of 3-10 minutes for the different test objects. This range is suitable for CAD work.

6. A simple example

Consider a simple polyhedral object, for instance a partial tetrahedron. Suppose that the tetrahedron is composed from three surfaces, S_1, S_2 and S_3 (Fig.3).

```
SURFACES                        /* begin of surfaces declaration */
SURFACE TYPE    Identifier      data file
END_SURFACES                    /* end of surfaces declaration */

CONSTRAINTS                     /*begin of constraints declaration */
PARALLEL_PLANES
Identifier1     Identifier2
END_PARALLEL_PLANES
INTRINSIC_CONSTRAINTS
Identifier      Constraint Type     Tolerance
END_INTRINSIC_CONSTRAINTS
ORIENTATION_PLANE_PLANE
Identifier1     Identifier2     Angle     Tolerance
END_ORIENTATION_PLANE_PLANE

ORIENTATION_PLANE_QUADRIC
Identifier1     Identifier2     Angle     Tolerance
END_ORIENTATION_PLANE_QUADRIC


END_CONSTRAINTS                 /*end of constraints declaration */
```

Figure 8. Structure of the input file for the constraint language compiler: the upper case words are the key words of the language

Following the paradigm of Section 4.1, each surface is represented by the equation:

$$\vec{h}_j^{\,i\,T}\,\vec{p}_i = 0\,; \quad i = 1..3$$

$$\vec{h}_j^{\,i} = [x_j^i, y_j^i, z_j^i, 1]^T; \qquad \vec{p}_i = [n_x^i, n_y^i, n_z^i, d_i]^T$$

The object is then represented by the parameter vector:

$$\vec{p} = [n_x^1, n_y^1, n_z^1, d_1, n_x^2, n_y^2, n_z^2, d_2, n_x^3, n_y^3, n_z^3, d_3]$$

The objective function is expressed by:

$$F(\vec{p}) = J = \vec{p}^T \mathcal{H} \vec{p} = \begin{bmatrix} H_1 & (0)_4 & (0)_4 \\ (0)_4 & H_2 & (0)_4 \\ (0)_4 & (0)_4 & H_3 \end{bmatrix}$$

where

$$H_i = \sum_j (\vec{h_j^i})(\vec{h_j^i})^T$$

The surfaces have three orientation constraints reflecting the three angles 90^0, 90^0 and 120^0 between the three surface normals $\vec{n}_1$, $\vec{n}_2$ and $\vec{n}_3$. These constraints are represented by the following equations

$$\begin{aligned} \vec{n_1}^T \vec{n}_2 &= -0.5 \\ \vec{n_1}^T \vec{n}_3 &= 0 \\ \vec{n_2}^T \vec{n}_3 &= 0 \end{aligned}$$

from which the constraint functions are deduced:

$$\begin{aligned} Angle_1(\vec{p}) &= (\vec{p}^T A_1 \vec{p} + 0.5)^2 = 0 \\ Angle_2(\vec{p}) &= (\vec{p}^T A_2 \vec{p})^2 = 0 \\ Angle_3(\vec{p}) &= (\vec{p}^T A_3 \vec{p})^2 = 0 \end{aligned}$$

where

$$A_1 = \begin{cases} A_1(i,j) = A_1(j,i) = 1/2 & \text{if } i = 1+t,\ j = 5+t, \quad 0 \le t \le 2 \\ A_1(i,j) = A_1(j,i) = 0 & \text{otherwise} \end{cases}$$

$$A_2 = \begin{cases} A_2(i,j) = A_2(j,i) = 1/2 & \text{if } i = 1+t,\ j = 9+t, \quad 0 \le t \le 2 \\ A_2(i,j) = A_2(j,i) = 0 & \text{otherwise} \end{cases}$$

$$A_3 = \begin{cases} A_3(i,j) = A_3(j,i) = 1/2 & \text{if } i = 5+t,\ j = 9+t, \quad 0 \le t \le 2 \\ A_3(i,j) = A_3(j,i) = 0 & \text{otherwise} \end{cases}$$

The surfaces normals are also constrained to be unit. This leads to the following unit constraints:

$$\begin{aligned} Unit_1(\vec{p}) &= (\vec{p}^T U_1 \vec{p} - 1)^2 = 0 \\ Unit_2(\vec{p}) &= (\vec{p}^T U_2 \vec{p} - 1)^2 = 0 \\ Unit_3(\vec{p}) &= (\vec{p}^T U_3 \vec{p} - 1)^2 = 0 \end{aligned}$$

where U_1, U_2 and U_3 are diagonal matrices defined by

$$U_1 = \begin{cases} U_1(i,i) = 1 & \text{for } i = 1..3 \\ U_1(i,i) = 0 & \text{otherwise} \end{cases}$$

$$U_2 = \begin{cases} U_2(i,i) = 1 & \text{for } i = 5..7 \\ U_2(i,i) = 0 & \text{otherwise} \end{cases}$$

```
SURFACES
PLANE     S1     "S1.dat"
PLANE     S2     "S2.dat"
PLANE     S3     "S3.dat"
END_SURFACES

CONSTRAINTS
ORIENTATION_PLANE_PLANE
S1     S2     120     10e-4
S1     S3     90      10e-4
S2     S3     90      10e-4
END_ORIENTATION_PLANE_PLANE

END_CONSTRAINTS
```

Figure 9. Input file of the tetrahedron object.

$$U_3 = \begin{cases} U_3(i,i) = 1 & \text{for } i = 9..11 \\ U_3(i,i) = 0 & \text{otherwise} \end{cases}$$

The expression of the optimization function is then

$$\vec{p}^T \mathcal{H} \vec{p} + \sum_{l=1}^{3} \lambda_{unit}^{l} Unit_l(\vec{p}) + \sum_{l=1}^{3} \lambda_{angle}^{l} Angle_l(\vec{p})$$

The input file related to this object is shown in Figure 9.

7. Experiments

The experiments were carried out on real parts having planar and quadric surfaces (cylinder, cone, sphere). The process of extracting the different surfaces of a given part (Fig.10) starts by scanning the part by a 3D laser triangulation range sensor. With this device a cloud of 3D points representing the shape of the object are obtained. The next step is to segment the points into sets associated to the different surfaces of the object. This is achieved using the *rangeseg* program [12]. To be fully measured, most of the objects have to be scanned at different views. Therefore the measurement data points obtained in each view have to be registered to the same reference frame. This operation is carried out manually by visualising the data points associated to the different views and manipulating the set of points by hand. Since the user relies only on his eye to judge the quality of the registration the data points locations are expected to be additionally corrupted by systematic errors. Actually we have intentionally performed the registration by hand to check the sensitivity of the algorithm with respect to the registration errors.

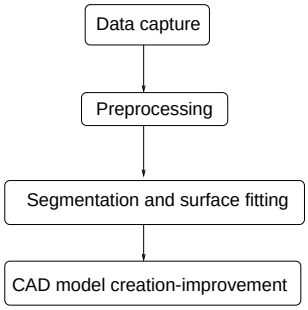


Figure 10. Steps of the object modelling process

This section will present two experiments carried out on two multi-quadric objects. These experiments check the behaviour and the convergence of the algorithm as well as the impact of constraint satisfaction on the quality of object shape reconstruction.

In order to save some space, the expressions of the different constraints and the way how they were set up will not be developed. The readers can could find more details in [21].

7.1. RECONSTRUCTION EXPERIMENT 1

The object (Fig.11) tested in this experiment comprises two lateral planes S_1 and S_2, a back plane S_3, a bottom plane S_4, a cylindrical surface S_5 and a conic surface S_6. The cylinder surface and the back plane surface contain more than twenty thousands points each. The number of points for each of the other surfaces range from four to nine thousand. The cylindrical patch is less than a half cylinder (40% arc), the conic patch occupies a small area of the whole cone (less than 30%)

The surfaces of the object have the following constraints:

1. S_1 makes an angle of 120^o with S_2 (we consider the angle between normals).
2. S_1 and S_2 are perpendicular to S_3.
3. S_1 and S_2 make an angle of 120^o with S_4.
4. S_3 is perpendicular to S_4.
5. The axis of the cylindrical patch S_5 is parallel to S_3's normal.
6. The axis of the cone patch S_6 is parallel to S_4's normal.
7. The cylindrical patch is circular.
8. The cone patch is circular.

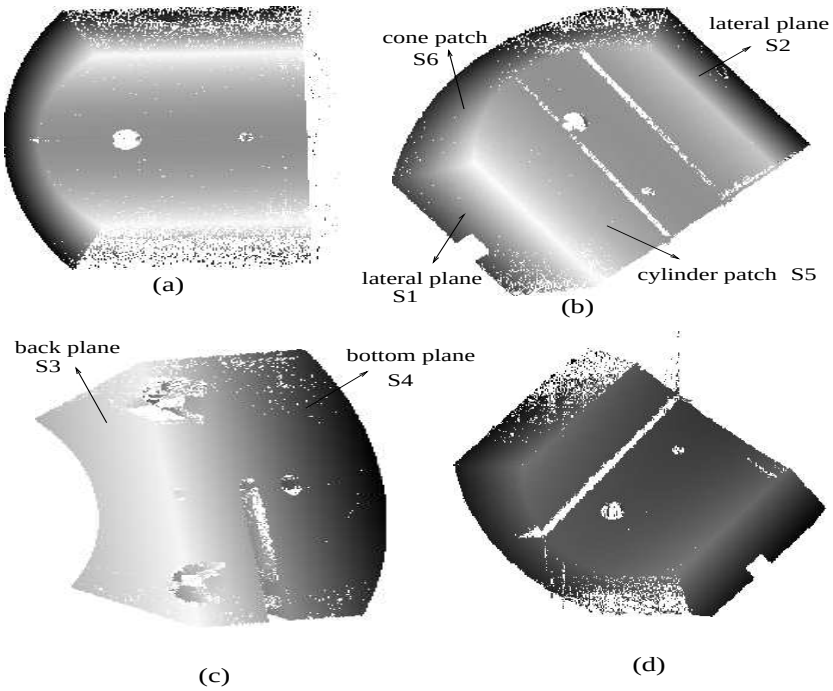


Figure 11. four views of the multi-quadric object

Constraints 5 and 6 are imposed by associating the normals to S_3 and S_4 respectively to the orientation vectors of the cylinder axis and the cone axis and thus could be combined with the angle constraints (see [21] for explicit development).

The complete optimisation function is then given by the expression:

$$E(\vec{p}) = \vec{p}^T H \vec{p} + \lambda_1 C_{unit}(\vec{p}) + \lambda_2 C_{ang}(\vec{p}) + \lambda_3 C_{circ_{cyl}}(\vec{p}) + \lambda_4 C_{circ_{cone}}(\vec{p})$$

Since the surfaces cannot be recovered from a single view, four views (Fig.11) have been registered by hand. 100 estimations were carried out for statistical comparison. At each trial 50% of the surface's points are selected randomly. The results shown in this section are the average of these estimations. The results regarding the algorithm convergence are shown in Figure 12. The behaviour of the different constraints during the optimization have been mapped as a function of the associated λ_i as well as the least squares residual and the sum of the constraint functions. The figures show a nearly linear logarithmic decrease of the constraints. It is also noticed that at the end of the optimization all the constraints are highly satisfied. The least squares error converges to a stable value and the constraint function vanishes at the end of the optimization. Thus, the final part shape satisfies the shape constraints at a slight increase in the least squares fitting error. The figures also show that it is possible to continue the optimization further until a higher tolerance is

reached, however this is limited practically by the numerical accuracy of the machine.

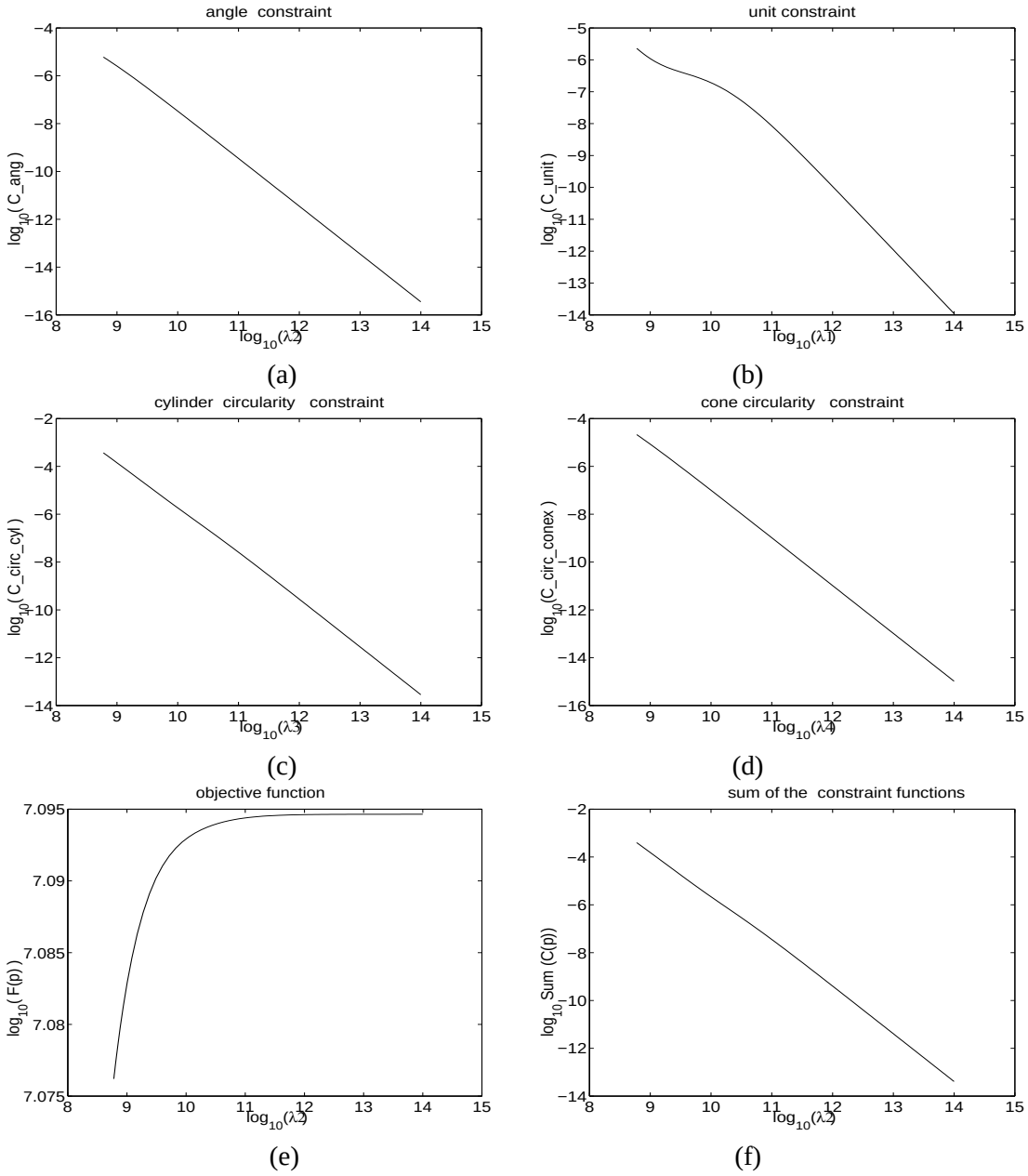


Figure 12. a,b,c,d : Decrease of the different constraint errors as function of the related λ. e,f : Variation of the least squares error and the total constraint error.

The angles between the different fitted planes are presented in Table II. It should be noticed that all the angles converge to the actual values. Table III and Table IV contain the estimated values of some attributes of the cylinder and the cone. The values show that each of the axis constraints are perfectly satisfied, the estimated radius and the cone half angle α improve when the constraints are introduced.

Table II. The surface's relative angle estimation with and without constraints.

angle	(S_1, S_2)	(S_1, S_3)	(S_1, S_4)	(S_2, S_3)	(S_2, S_4)	(S_3, S_4)
without constraints	119.76	92.08	121.01	87.45	119.20	90.39
with constraints	120.00* [1]	90.00*	120.00*	90.00*	120.00*	90.00*
actual values	120	90	120	90	120	90

We notice the good shape improvement, relative to the unconstrained least squares method, given by a reduction of bias of about $22mm$ and 3^0 respectively in the radius and the half angle estimation. The standard deviations of the estimations have been reduced as well.

The radius estimation is within the hoped tolerances, a systematic error of about $0.5mm$ is quite nice. However the cone half angle estimation involves a larger systematic error (about 1.8^o). Two factors may contribute to this. The registration error may be too large since the registration was done by hand and the limited area of the cone patch which covers less then 30 % of the whole cone. It is known that when a quadric patch does not contain enough information concerning the curvature, the estimation is very biased, even when robust techniques are applied, because it is not possible to predict the variation of the surface curvature.

Table III. The cylinder characteristic estimates with and without constraints.

cylinder parameters	angle(axis, S_3's normal)	radius	standard deviation of radius
without constraints	2.34	37.81	0.63
with constraints	0.00*	59.65	0.08
actual values	0	60	0

[1] * means that the estimated value is constrained to be equal to the true value.

Table IV. The cone characteristic estimates with and without constraints.

cone attributes	angle(axis, S_4's normal)	α	standard deviation of α
without constraints	6.08	26.01	0.30
with constraints	0.00^*	31.83	0.13
actual values	0	30	0

7.2. RECONSTRUCTION EXPERIMENT 2

The object (Fig.13) contains six plane surfaces $S_1, S_2, S_3, S_4, S_5, S_6$, a cylindrical surface S_7 and a spherical surface S_8. The surfaces S_1, S_2, S_3, S_4, S_5 form a square prism, the surface S_5 is a square plane surface. The cylindrical patch is a whole cylinder and the spherical patch occupies a half sphere.

About 10, 000 and 3000 points were measured from the cylinder and sphere respectively. 1500 points in average were measured from each of the surface planes except for the plane surface S_6 from which less than 300 points were measured.

The surfaces of the object have the following relationships:

1. S_1, S_3 are parallel.
2. S_2, S_4 are parallel.
3. S_5, S_6 are parallel.
4. S_1, S_3 are orthogonal to S_2, S_4.
5. S_5, S_6 are orthogonal to S_1, S_3 and S_2, S_4.
6. S_1, S_3 and S_2, S_4 are separated by the same distance.
7. The cylinder axis is parallel to S_1, S_2, S_3 and S_4 and orthogonal to S_5, S_6.
8. The cylinder axis is located midway between S_1 and S_3 and midway between S_2 and S_4.
9. The cylindrical patch is circular.
10. The sphere centre lies on the cylinder axis.
11. The radius of the cylinder is equal to the radius of sphere.
12. The length diagonal of surface S_5 is equal to the cylinder diameter.

Constraints $1, 2$ and 3 allow a single normal to be associated with each pair of planes (S_1, S_3), (S_2, S_4) and (S_5, S_6). Constraint 7 is imposed by associating S_5's normal to the axis of the cylinder and thus combined with the angle constraints. The other constraints are explicitly defined.

The optimization function is expressed by:

$$E(\vec{p}) = \vec{p}^T H \vec{p} + \lambda_1 C_{unit}(\vec{p}) + \lambda_2 C_{angl}(\vec{p}) + \lambda_3 C_{dist}(\vec{p}) + \lambda_4 C_{axe\_pos}(\vec{p}) + \lambda_5 C_{circ}(\vec{p}) + \lambda_6 C_{sph_{center}}(\vec{p}) + \lambda_7 C_{equ_{radius}}(\vec{p}) + \lambda_8 C_{median}(\vec{p})$$

The surfaces of the objects were recovered from 4 views shown in Figure

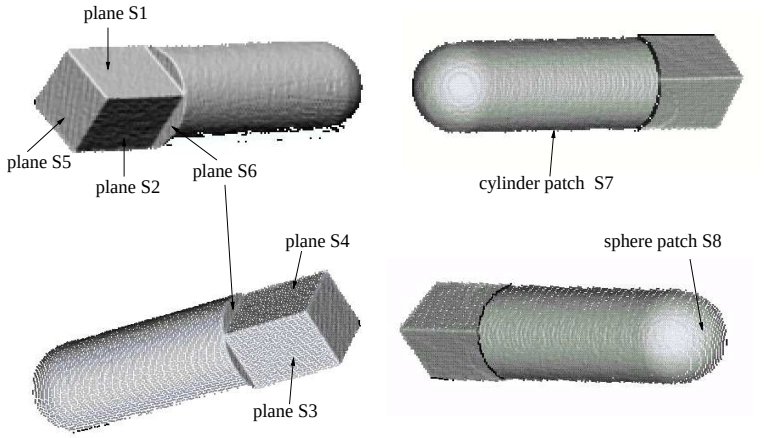


Figure 13. Four views of the multi-quadric object.

13. Similarly to the previous object 100 trials were performed. At each of them 50% of the surfaces's points are selected randomly leading to a different initialisation each trial. In all the trials, the decrease of all the constraint errors and the high level of satisfaction of the constraints at the end of the optimization for a slight increase of the least squares error are essentially similar to that observed in the previous experiments and so similar graphs are not shown here.

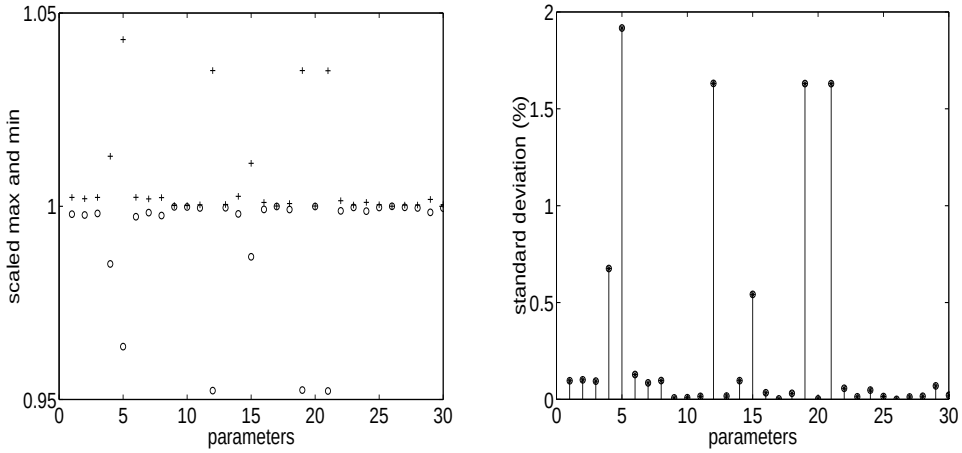
7.3. ALGORITHM EVALUATION EXPERIMENTS

Other experiments trials were carried out in order to give answers to the following questions:

1. How stable is the convergence of the algorithm ?
2. How close is the estimation to the actual optimal value ?
3. What are the effects of leaving some features unconstrained ?
4. What is the effect of constraint invalidity ?
5. What is the effect of constraint inconsistency ?
6. Does the global shape improve with local constraints ?

7.3.1. *Stability of the convergence*

The different estimations resulting from the 100 trials were examined statistically. Figure 14.a shows the maximum and the minimum value (scaled by the absolute value of the mean) for each parameter. The closeness of each pair of extrema is well noticed. This aspect is further confirmed by the standard deviations of the parameters illustrated in Figure 14.b. The distribution of the least squares errors of the different estimations is shown in Figure 15. The related relative variance is 1.94%. Thus we can conclude that the algorithm is stable.



(a) (b)

Figure 14. (a): maximum (+) and minimum (o) value for each parameter scaled by the absolute value of the mean. (b): relative standard deviation of the parameters.

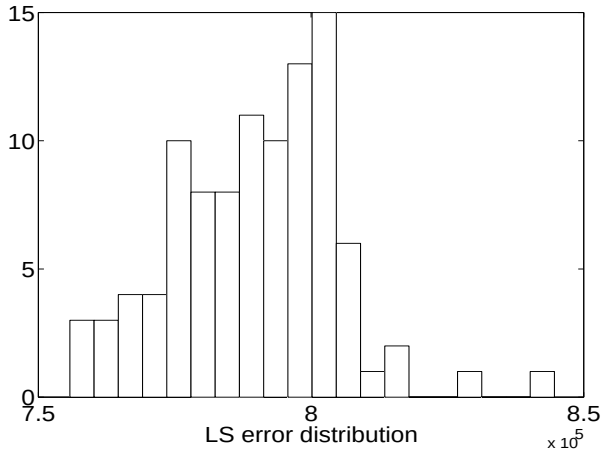


Figure 15. Distribution of the least squares errors.

7.3.2. *Closeness to the actual optimal solution*

By "actual optimal solution" we mean the estimation obtained from a process where the constraints are defined, incorporated and satisfied within the least squares error formulation. The solution provided in this case completely satisfies the constraints. So one may ask how close is the estimate obtained by our approach to this optimal solution. As we have mentioned previously, such an ideal and elegant formulation is difficult or impossible to achieve for many objects due to the complexity and to the non-linearity of the geometric constraints. In fact one purpose and motivation of our approach is to overcome this problem. Nevertheless it is possible for some simple particular cases to combine the constraints with the least squares error.

So, in order to make a comparison with the optimal solution a sub-part of the multi-quadric object shown in Figure 13 was considered. It is composed of the two parallel planes S_1 and S_3. The objective is to estimate the planes' orientation taking into account the parallel constraint. For the first case, the parallel constraint is implicitly considered by associating one normal to both planes. The optimization function is then:

$$\vec{n}^T H \vec{n} + \lambda(1 - \vec{n}^T \vec{n})$$

where H is the appropriate data matrix. The second term of the function is the unit constraint. A closed form solution is provided by the eigenvalue method.

In the second case each plane was assigned a different normal vector. The equality of the two normals has to be satisfied through the optimization process. According to our approach the objective function is:

$$\vec{n_1}^T H_1 \vec{n_1} + \vec{n_3}^T H_3 \vec{n_3} + \lambda_1(1 - \vec{n_1}^T \vec{n_1})^2 + \lambda_2(1 - \vec{n_3}^T \vec{n_3})^2 + \lambda_3(1 - \vec{n_1}^T \vec{n_3})^2$$

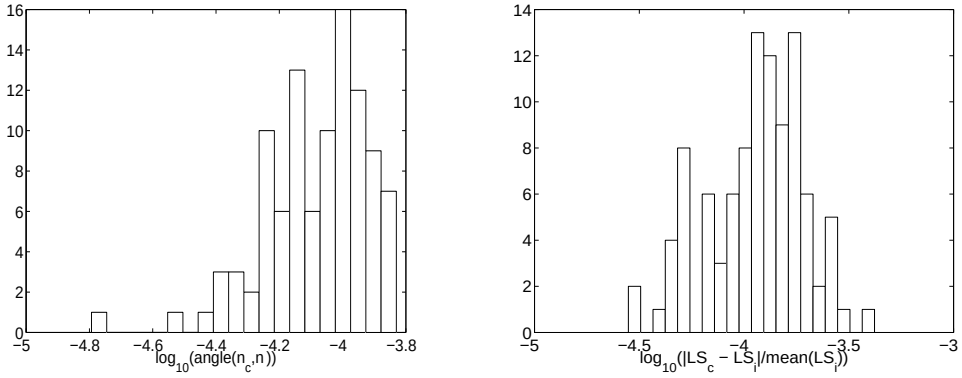
100 tests were applied for each of the two cases. The average of the results are summarized in Table V. The estimations are similar in the two

Table V. Mean estimates of S1 and S3 normal and LS error in the two types of solutions.

	$\vec{n}$	$\vec{n_1}$	$\vec{n_3}$	angle($\vec{n_1}, \vec{n_3}$) (degree)	LS error
Closed form	0.5316 0.6733 0.5139	-	-	-	9.07
Optimization	-	0.5316 0.6733 0.5139	0.5316 0.6733 0.5139	0.00	9.06

cases. This shows that both solutions converge to the same value and almost

equally minimize the least squares error. The LS of the second solution is slightly lower than the optimal solution one. This is because in the optimal case the constraint is perfectly satisfied so the least squares error has to absorb all the error. The same convergence of the two solutions is further confirmed from the distribution of the difference between the two approaches (angle $(\vec{n}, \vec{n_c})$ where $\vec{n_c}$ is the mean of $\vec{n_1}$ and $\vec{n_3}$) and the difference between the related LS residuals from the 100 trials (Fig.16). Thus we conclude that our optimization process leads us to solutions that are very close to the optimal.



(a) (b)

Figure 16. (a): Distribution of the estimation difference. (b): Distribution of the LS residuals difference.

7.3.3. *Leaving some features unconstrained*

Another series of tests has been performed without considering the diagonal constraint (constraint 12). This is in order to check if this will affect the position of the four plane surfaces with respect to the cylinder axis and therefore the estimation of the edge of the square surface S_5. Results are shown in Table VI with the previous results for comparison. It is noticed that the radius estimation is not affected but the incorporation of the additional constraints slightly reduces the diagonal length error.

7.3.4. *Invalidity of the constraints*

Suppose that one or more constraints do not reflect the actual relationships between features and therefore are invalid. What would be the behaviour of the algorithm? Will these "false constraints" be satisfied? What could be the resulting estimated model ?

To answer these questions, some angle constraints were set to an incorrect values. Three tests were carried out, in the first the angle $(\vec{n_1}, \vec{n_2})$ was set to $\pi/3$, in the second the angle $(\vec{n_1}, \vec{n_5})$ was set to $\pi/3$ and in the third test both angles $(\vec{n_1}, \vec{n_5})$ and $(\vec{n_2}, \vec{n_5})$ were set to $\pi/3$ (note that the correct angles are $\pi/2$ for both angles).

Table VI. comparison of the estimation without median constraints with previous results.

	distance(S_1, S_3)	distance(S_2, S_4)	diagonal of S_5	cylinder radius
without constraints	–	–	–	14.64
with all constraints	21.17	21.17	29.94	14.97
without median constraint	21.15	21.15	29.91	14.97
actual values	21.28	21.28	30.02	15.01

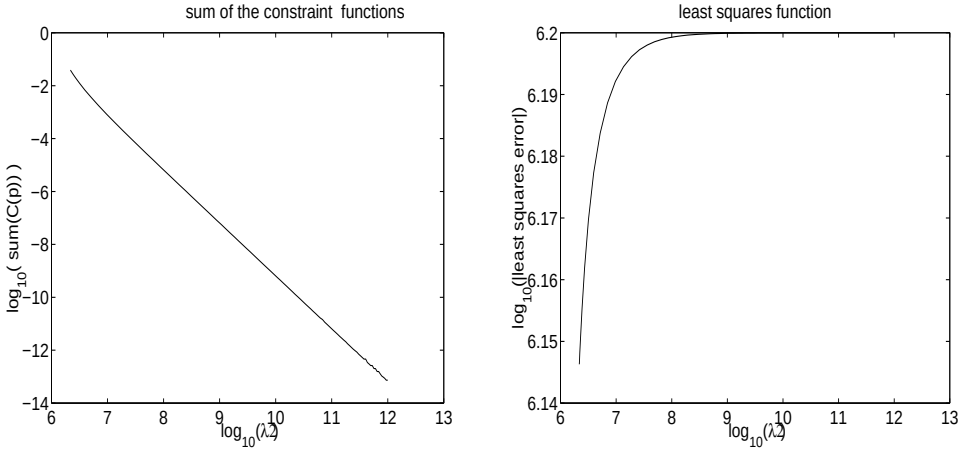
In all these tests the behaviour and the convergence of the algorithm were qualitatively similar to those of the previous experiments. The algorithm converges, the least squares error stabilizes and all the constraints are satisfied at the end of the process although the least squares error is greater than the valid constraints case (Figure 17). Table VII summarizes the estimated model characteristics in each of the three tests.

Table VII. The object characteristic estimates for invalid constraints and true constraints (last row).

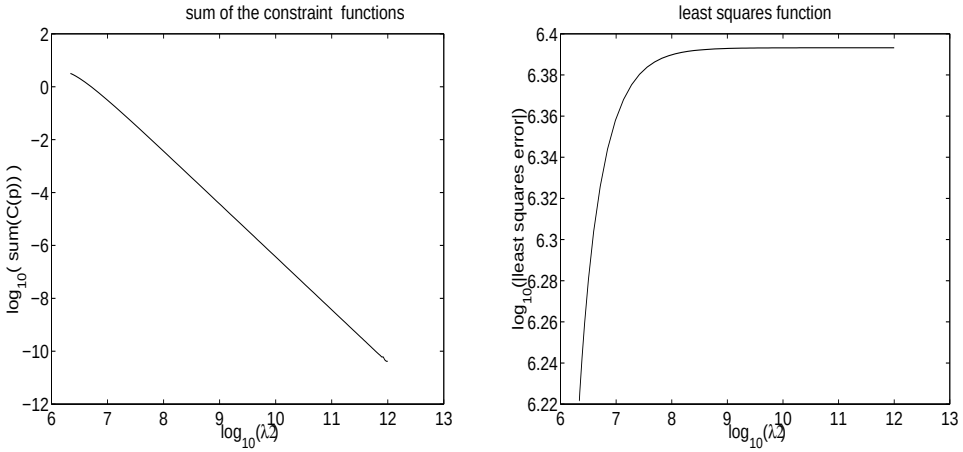
	$\vec{n_1}$	$\vec{n_2}$	$\vec{n_5}$	R_{cyl}	R_{sph}	axe_{cyl}	$Center_{sph}$
$(\vec{n_1}, \vec{n_2}) = \pi/3$	-0.61 -0.47 -0.62	-0.58 0.52 -0.62	0.72 -0.02 -0.69	14.97	14.97	0.72 -0.02 -0.69	86.30 -87.38 17.44
$(\vec{n_1}, \vec{n_5}) = \pi/3$	-0.08 -0.60 -0.78	-0.46 0.72 -0.50	0.72 -0.02 -0.69	14.97	14.97	0.72 -0.02 -0.69	86.31 -87.41 17.44
$(\vec{n_1}, \vec{n_5}) = \pi/3$ $(\vec{n_2}, \vec{n_5}) = \pi/3$	-0.02 -0.68 -0.72	0.05 0.72 -0.68	0.72 -0.02 -0.69	14.97	14.97	0.72 -0.02 -0.69	86.31 -87.42 17.44
true constraints	-0.52 -0.67 -0.51	-0.45 0.73 -0.50	0.72 -0.02 -0.69	14.97	14.97	0.72 -0.02 -0.69	86.30 -87.38 17.44

An examination of Table VII leads to the following observations:

1. In all of the three tests the cylinder and the sphere characteristics are not affected by the invalid constraints.
2. The normal $\vec{n_1}$ which is involved in each of the invalid constraints is affected in three tests.
3. The normal $\vec{n_2}$ is changed in the first and third test where it is involved in the invalid constraints whereas it is unchanged in the second test where it is not involved.



(a)



(b)

Figure 17. (a):Constraint error function and least squares error function function for valid constraints. (b):Constraint error and least squares error function for invalid constraints (3^{rd} test).)

4. The normal $\vec{n_5}$ is kept unchanged in all the tests even in those where it is involved in the invalid constraints.

From these observations we can deduce that invalid constraints affect the object feature's locations by shifting the involved features toward positions where these constraints are satisfied. Consequently, this will increase the least squares error. The locations and the characteristics of the surfaces which are not involved in the invalid constraints are not affected (the sphere and the cylinder). However the normal $\vec{n_5}$ seems not to satisfy this rule since its orientation stays unchanged for all the cases where it is involved in an invalid constraint. This is explained by the fact that contrary to $\vec{n_1}$ and $\vec{n_2}$,

$\vec{n}_5$ is also involved in other constraints, in particular it is constrained to have the same orientation as the cylinder axis. The satisfaction of this constraint keeps it collinear to the cylinder axis and prevents its orientation from being affected. Thus the algorithm satisfies the invalid constraints in which $\vec{n}_5$ is involved by acting on the other normals involved in these constraints.

7.3.5. *Inconsistency of the constraints*

In this test we investigated what the behaviour of the algorithm would be when some constraints are inconsistent and have a contradiction between them. For this purpose we introduced two additional inconsistent angle constraints (imposing the angles $(\vec{n}_1, \vec{n}_2)$ and $(\vec{n}_1, \vec{n}_5)$ to be $\pi/3$) that conflict with the two original consistent constraints (defining each pair of $(\vec{n}_1, \vec{n}_2)$ and $(\vec{n}_1, \vec{n}_5)$ as orthogonal vectors). The trial carried out with these inconsistent constraints revealed that the algorithm converges normally (Figure 18) with both the least squares and the constraint functions stabilizing at the end of the algorithm. From Figure 18.a we notice that the angle constraints are not satisfied. This is obvious because it is not possible to satisfy conflicting constraints simultaneously. The converging values of the constraint function (the sum of all the constraints) in Figure 18.b and the angle constraints error are practically equal at the end of the optimization process. This shows that the other consistent constraints are satisfied. This result is quite useful, it means that the set of constraints affected by the inconsistency can be detected by observing the convergence of each constraint error function rather than its reduction to zero. More analysis is needed to detect the smallest subset of constraints causing the inconsistency however.

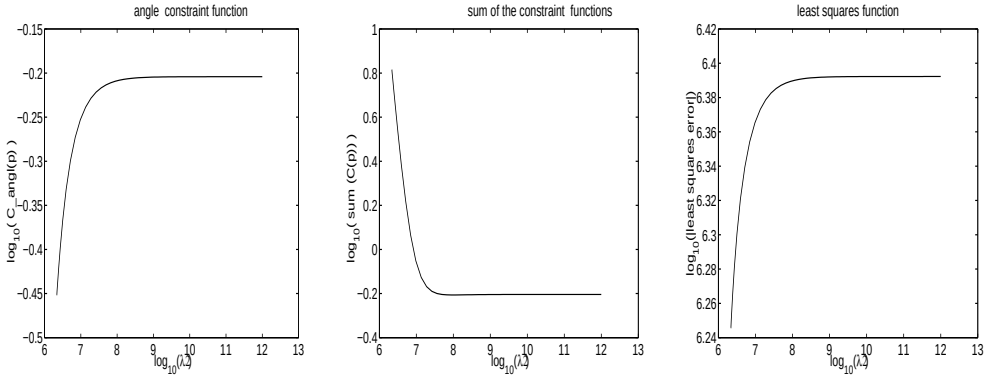


Figure 18. (a): The sum of the angle constraints' errors. (b): the constraint function. (c) the least squares error

7.3.6. *Global shape improvement*

The different tables shown in this section compare the geometric characteristics of the object for an optimization with and without constraints and show the improvement of the object characteristic estimates when constraints are applied. The results presented in the tables are the average of the 100 estimations. The angles between each pair of surfaces $(S_1, S_2), (S_1, S_5)$ and (S_2, S_5) were set as constraints and the constraints were nearly perfectly satisfied. From Table VIII we notice the satisfaction of the square property of the prism, illustrated by the equality of the two distances separating (S_1, S_3) and (S_2, S_4). Their values are close to the actual length of the edge of the square plane S_5 and closeness of the estimated value of the diagonal of S_5 to the actual value when the constraints are considered. The distances between these last surfaces for an optimization without constraints is not mentioned in this table since the estimated surfaces are not parallel.

Table VIII. Improvement of the prism characteristic estimates.

	distance(S_1, S_3)	distance(S_2, S_4)	diagonal of S_5
with constraints	21.17	21.17	29.95
standard deviation/mean	0.03 %	0.03%	0.03%
actual values	21.28	21.28	30.02

The improvement of the quadric surface estimation is confirmed again for this object (Table IX and Table X). The radius estimation error is less than $0.04mm$ for both the cylinder and the sphere. The standard deviations of the cylinder and the sphere radius have been significantly reduced as well.

Table IX. Improvement of the cylinder characteristic estimates.

cylinder parameters	angle(axis, S_5's normal)	radius (mm)
without constraints	1.55	14.64
σ/mean without constraints	-	0.12%
with constraints	0.00^*	14.97
σ/mean with constraints	-	0.03%
actual values	0	15.01

Table X. Improvement of the sphere characteristic estimates.

sphere parameters	distance(centre, cylinder axis)	radius (mm)
without constraints	1.36	16.02
σ/mean without constraints	-	0.11%
with constraints	0.00*	14.97
σ/mean with constraints	-	0.03%
actual values	0	15.01

8. Conclusion

This work presents a method for the reconstruction of shape incorporating geometric constraints. It can hold a large number of varied constraint types and incorporates them integrally without the need for linearization.

The experiments carried out on the different objects confirm the convergence of the algorithm. The parameter optimization search does produce shape fitting that satisfies almost perfectly the constraints. They show in particular that the least squares error grows slightly as the constraints are applied and the weighting values increased, but it stabilizes above certain values of the λ_k while the constraint errors are still decreasing. Thus it is possible to satisfy the constraints up to the desired tolerance without seriously affecting the quality of the data fitting. This allows the user to control the degree of satisfaction of the constraints and to set the tolerances as high as necessary. The processing time for the different objects is typically a few minutes and is is expected to be further reduced with more optimized versions of the implementation.

The above observations suggest that the proposed approach allows flexibility in the incorporation of the constraints, as well as for their satisfaction. Indeed the low computing time of the algorithm allows an interactive user environment. This is not possible with techniques requiring several hours computing time such as techniques based on genetic algorithms.

Regarding the slight increases of the LS error, we have to bear in mind that the increase of the least squares residuals value may not reflect a bad estimation in the case when measurement errors are systematic, e.g. miscalibration and registration error. This last type of error is expected in our data since the registration process is performed by hand. We believe that the slight increase of the least squares error as a consequence of the constraints satisfaction is a result of the object being located more accurately. However we

intend to investigate a more robust form for the objective function involving the data noise statistics.

The different trials applied on the multi-quadric objects confirm the stability of the convergence of the algorithm. The low values of the parameters' variances illustrates the stability of the solution provided by the optimization search process. The tests have shown as well that the proposed approach leads to an estimate which is close to the optimal solution in the case where the constraints could be combined with the least squares error. The experiments also show that applying the constraints to only some features does not seriously affect the estimation of the unconstrained surfaces. The estimation is still improved compared to the case of unconstrained optimization.

The examination of some constraint invalidity cases has shown the constraints are always satisfied whether they are valid or not and the behaviour of the algorithm is typically the same. The satisfaction of an invalid constraint leads to the relocation of the involved and less constrained features (having more degrees of freedom) toward positions where the inconsistency is removed. However, this will result in a false object model. The trial performed with constraint inconsistency revealed the same behaviour regarding the convergence of the algorithm but the inconsistent constraints are not satisfied at the end of the optimization. This suggests that constraint validity and consistency checking have to be done before starting the optimization process, or at least examination of the constraint error results to determine if a set of inconsistent constraints have been supplied.

Regarding the shape estimation accuracy, the comparison of the object dimension estimates with those from unconstrained fitting confirms that the proposed approach improves the quality of the shape reconstruction to a high degree. For the second quadric object the radius of the cylinder and the sphere have an estimation error in the range of $0.04mm$, the edge of the square prism has an estimation error around $0.1mm$. The radius of the cylinder patch estimated from the registered half cylinder has an estimation error around $0.01mm$. For a single view it is less than $0.5mm$. The same range of error is obtained for the radius of the cylinder patch of the first multi-quadric object.

Results for the cone patch are less satisfactory for the first multi-quadric object. This is mainly due to the relatively small area of the conic patch. Actually, we intentionally chose to work with small patches because unconstrained fitting surface techniques fail to give reasonable estimates in this case (see the radius estimation in Table III) even with robust algorithms due to the "poorness" of the information embodied in the patch.

Although the experiments presented in this work were performed on single objects, the proposed approach can hold for multiple objects. Indeed, generally industrial parts are designed to fit to each other, so geometric relationships between the parts may be considered and the resulting constraints can be incorporated as well in the optimization process.

Another area we are starting to investigate is how one might automatically identify inter-surface relationships that can have a constraint applied. In manufacturing objects, simple angular and spatial relationships are given by design. So, it should be straightforward to define simple statistical tests that hypothesize standard feature relationships, subject to the feature's statistical position distribution. With this analysis, a computer program could propose a variety of constraints that a human could either accept or reject, after which shape reconstruction could occur.

The proposed technique restricts its scope to applications where a reasonable initial solution is available. Also the approach can hold only geometric constraints that can be represented by continuous and differentiable functions. More complex objects with higher order surfaces can use by the approach as far as this condition is fulfilled.

Finally, although this work is mainly intended for object modelling it may be extended to any constrained built environment application, for example modelling of different parts of an industrial plant (pipes, reservoirs, etc) needs the consideration of the geometric relationships between these different parts in order that the whole model will be consistent. The same is true as well for modelling different compartments of buildings. Cities are probably too under-constrained.

ACKNOWLEDGEMENT

The work presented in this paper was funded by UK EPSRC grant GR/L25110.

References

1. R. Anderl, R Mendegen. *Modelling with constraints: Theoretical foundation and application.* CAD, Vol. 28, No 3, pp 155-168 1996.
2. R.M.Bolle, D.B.Cooper. *On Optimally Combining Pieces of Information, with Application to Estimating 3-D Complex-Object Position from Range Data.* IEEE Trans. PAMI, Vol.8, No.5, pp.619-638, September 1986.
3. C.G. Broyden, N.F. Attia. *Penalty Functions, Newton's Method and Quadratic Programming.* Journal of optimization theory and applications, Vol.58, No.3, pp.377-385., 1988.
4. J.De Geeter, H.V.Brussel, J.De Schutter, M. Decreton. *A Smoothly Constrained Kalman Filter.* IEEE Trans. PAMI pp.1171-1177, No.10, Vol.19, October 1997.
5. Chang-Xue Feng, A. Kusiak. *Constraints-based design of parts.* CAD, Vol.27, No.5, 1995 pp 343-352.
6. A.V. Fiacco, G.P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* John Wiley and Sons, New York 1968.
7. A.F. Fitzgibbon, D.W. Eggert, R.B. Fisher. *High-level CAD model acquisition from range images.* CAD, Vol.29, No.4, pp.321-330, 1997.
8. R.Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 1987.
9. P.E.Gill, W.Murray, M.H.Wright. *Practical Optimization.* Academic Press, 1981.
10. D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.
11. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, 1996.
12. 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. PAMI, Vol.18, No.7, pp.673-689, July 1996.
13. S.L.S. Jacoby, J.S. Kowalik, J.T.Pizzo. *Iterative Methods for Nonlinear Optimization Problems.* Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1972.
14. W. Murray. *An Algorithm for Constrained Minimization.* Optimization, Ed. R.Fletcher, pp.247-258, Academic Press, London, 1969.
15. J.Porrill. *Optimal Combination and Constraints for Geometrical Sensor Data.* International Journal of Robotics Research, Vol.7, No.6, pp.66-78, 1988.
16. A.A.G. Requicha. *Representation of Tolerances in Solid Modelling: Issues and Alternative Approaches.* Solid Modelling by Computers: from Theory to Applications, J.W.Boyse amd M.S.Pickett, Eds. New York: Plenum, 1984, pp.3-22.
17. R.T. Rockafellar. *Convex Analysis.* Princeton University Press, 1970.
18. A.P. Rockwood, J. Winget. *Three-dimensional object reconstruction from two-dimensional images.* CAD, Vol.29, No.4, pp.279-286, 1997.
19. B.S. Shin, Y.G. Shin. *Fast 3D solid model reconstruction from orthographic views.* CAD, Vol.30, No.1, pp.63-76.
20. T. Varady, R. R. Martin, J. Cox. *Reverse engineering of geometric models, an introduction.* CAD, Vol.29, No.4, pp. 255-268, 1997.

21. N.Werghi, R.B.Fisher, A.Ashbrook, C.Robertson. *Modelling Objects Having Quadric Surfaces Incorporating Geometric Constraint.* Proc. ECCV'98, pp.185-201, Freiburg, Germany, June 1998.

22. Q.W. Yan, C.L.P. Chen, Z. Tang. *Reconstruction of 3D objects from orthographic projections.* CAD, Vol.26, No.9, 1994.

Appendix: Levenberg-Marquadt algorithm

Here are the main steps of the Levenberg-Marquardt algorithm applied to a simple optimization function:

$E(\vec{p}) = F(\vec{p}) + C(\vec{p})$

$\alpha = \alpha_0$ % initialization
$E_{decrease}$ = big value

while $E_{decrease} > \varepsilon$ % a threshold

Do G_E=Grad($E(\vec{p})$) $= \frac{\partial}{\partial \vec{p}}(E(\vec{p}))$

Loop: H_E=Hessian($E(\vec{p})$) $= \frac{\partial^2}{\partial^2 \vec{p}}(E(\vec{p}))$
$H_E = H_E + \alpha(diag(H_E))$
solve $H_E \vec{\delta p} = -G_E$
$\vec{p}_{updated} = \vec{p} + \vec{\delta p}$
$E_{decrease} = E(\vec{p}_{updated}) - E(\vec{p})$

if $E_{decrease} > 0$
increase α
go to Loop
else
$\vec{p} = \vec{p}_{updated}$
decrease α

end if
end while