University of Bath

Fitting quadrics with a Bayesian prior

Daniel Beale[1] (✉), **Yong-Liang Yang**[1], **Neill Campbell**[1], **Darren Cosker**[1], **and Peter Hall**[1]

Abstract   Quadrics are a compact mathematical formulation for a range of primitive surfaces. A problem arises when there are not enough data points to compute the model but knowledge of the shape is available. This paper presents a method for fitting a quadric with a Bayesian prior. We use a matrix normal prior in order to favour ellipsoids when fitting to ambiguous data. The results show the algorithm copes well when there are few points in the point cloud, competing with contemporary techniques in the area.

Keywords   geometry; statistics; graphics; computer vision

1   Introduction

Surface fitting is one of the most important research areas in computer graphics and geometric modelling. It studies how to approximate unorganized geometric data using regular surfaces with explicit algebraic forms, which benefits many important graphics applications, including shape approximation, surface reconstruction, local geometric feature analysis, etc. In the fitting process, the choice of the target surface form usually depends on the application, which might require algebraic surfaces of specific orders, or freeform surfaces, such as B-Splines.

Among all types of surfaces, quadrics and hypersurfaces represented by a quadratic polynomial in the embedding space may be the most popular form in surface fitting for several reasons. Firstly, they can represent a variety of common primitive surfaces,

such as planes, spheres, cylinders, etc. Secondly, many real world shapes, ranging from industrial products to architectural designs, can be represented by a union of quadrics. Thirdly, it is the form with the least order from which it is possible to estimate second order differential properties, such as curvatures. Fourthly, the simple quadratic form allows for efficient numerical computations which are usually in closed-form.

In this work, we present a novel probabilistic quadric fitting method. Alternative to previous algorithms, we assume an additive Gaussian noise model on the algebraic distance. A Bayesian prior is placed on the parameters, allowing certain shapes to be favoured when few data are available. We have tested our quadric fitting method on various datasets, both synthetic and from the real world. The experiments and comparisons show that our method is not only efficient, but also robust to contaminated data.

2   Related work

There are a large number of approaches to fitting both implicit and parametric surfaces to unordered point clouds. We focus the review on existing methods in quadric fitting and then go on to show how they can be used in a variety of applications in computer graphics.

2.1   Quadric fitting

The popularity of quadrics has led to many research papers on fitting them to unordered data points. Methods in this area can be broken into two main areas, either minimising the *algebraic* or *geometric* distance of each point to the surface. The former is simpler, since it is the value computed by evaluating the implicit equation itself (defined in Eq. (1)).

---

1   University of Bath, Claverton Down, Bath, BA2 7AY, UK. E-mail: D. Beale, d.beale@bath.ac.uk (✉); Y.-L. Yang, y.yang2@bath.ac.uk; N. Campbell, n.campbell@bath.ac.uk; D. Cosker, d.p.cosker@bath.ac.uk; P. Hall, p.m.hall@bath.ac.uk.

Geometric distances are formulated in terms of Euclidean distances between the points and the surface; methods in this area produce a fit which favours each point equally, although methods tend to be iterative due to the complex relationship between the implicit equation and the solution set.

One of the most popular methods to use a geometric distance is from Taubin [1], who approximates the true distance in terms of the normalised Euclidean distance to the quadric surface. Knowledge of the explicit parametric equations of the surface can be useful for more compact least-squares parameter estimates [2–8]. Fast estimation of the closest point on the surface to a general point in space is described in Ref. [9], which also makes use of RANSAC to add robustness to outliers.

Finding a least-squares estimate based on the algebraic distance yields fast and numerically stable solutions which can be solved using matrix algebra. In this case, literature in the area seeks to constrain the surface to lie on a particular surface shape, such as an ellipsoid, e.g., Refs. [10–13]. The work of Li and Griffiths [12] shows how to fit ellipsoids and Dai et al. [14] present a similar method for fitting paraboloids.

Bayesian probabilistic methods exist for fitting general quadrics: for example, Subrahmonia et al. [15] show how to fit by casting a probability distribution over the geometric Taubin-type errors. Their use of priors is to avoid overfitting or parameters which produce a minimal error but undesirable results.

2.2 Quadric fitting applications

Quadric fitting has been applied to many important graphics problems. In the context of shape approximation, Cohen-Steiner et al. [16] present a variational shape approximation (VSA) algorithm, where a set of planar proxies are iteratively fitted to an input mesh based on Lloyd's clustering on mesh faces, resulting in a simplified mesh representation.

To better approximate curved parts and sharp features, Wu and Kobbelt [17] extend the previous approach by using proxies other than planes, where various types of quadrics, such as spheres or cylinders, are allowed. Another extension of the VSA algorithm was presented in Ref. [18], using general quadric proxies. For surface reconstruction,

quadric fitting has been used to better represent the underlying geometry and improve reconstruction quality. Schnabel et al. [19] present a RANSAC-based approach to fit different types of quadrics to noisy data. Li et al. [20] further improve the reconstruction quality by optimizing global regularities, such as symmetries, of the fitted surface arrangements. In addition to joining the geometric primitives, it is possible to create an implicit surface using a multi-level partition [21]. This kind of approach can help to smooth the resulting surface in regions which have not been accurately modelled.

In shape analysis, local geometric features (e.g., curvatures) which are invariant under certain transformations benefit greatly from quadric fitting. A common approach is to fit quadrics to a local neighbourhood so that differential properties can be estimated with the help of the fitted surface [22].

The use of computer vision in robotics applications needs the use of geometric fitting algorithms for estimating object properties for physical interaction, such as the curvature of an object [23]. Segmenting objects into primitive surfaces is also possible [24], allowing a robot to discriminate between surfaces before planning.

2.3 Our work

Our method assumes a probability distribution over the algebraic errors, and results in a compact matrix computation for the maximum a priori estimate of the parameters, given the data points. The use of a prior allows us to choose specific primitives from the collection of quadric surfaces.

3 Bayesian quadrics

We begin by defining what a quadric is, and then extend it with the use of a noise model. We define a conditional distribution which provides the likelihood of the data given the parameters, from which we infer the probability of the parameters with the use of a suitable prior.

Quadrics are polynomial surfaces of degree two. Each is a variety given by the implicit equation:
$$\boldsymbol{z}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{z} + \boldsymbol{b}^{\mathrm{T}} \boldsymbol{z} + c = 0 \qquad (1)$$
with $\boldsymbol{A}$ is an $M \times M$ matrix, $\boldsymbol{b}$ is a vector, c is a scalar, and $\boldsymbol{z} \in \mathbb{R}^M$.

Before defining more notation we note that $\boldsymbol{A}$ can be assumed to be symmetric, without loss of

generality, as explained in Appendix A. Positive definiteness is determined from the eigenvalues. Letting $\boldsymbol{\psi}$ denote each element of $\boldsymbol{A}$, $\boldsymbol{b}$, and c. Further, let $\boldsymbol{\theta_A}$ denote the values in $\boldsymbol{\psi}$ that correspond to the values in $\boldsymbol{A}$, i.e.,

$$\boldsymbol{\theta_A} = [A_{11}, \ldots, A_{MM}, A_{12}, \ldots, A_{(M-1)M}]^{\mathrm{T}} \quad (2)$$

so

$$\boldsymbol{\psi} = \begin{bmatrix} \boldsymbol{\theta_A} \\ \boldsymbol{b} \\ c \end{bmatrix} \quad (3)$$

The data from the input point cloud are collected into a data matrix $\boldsymbol{X} = [\boldsymbol{x}_0, \ldots, \boldsymbol{x}_\ell, \ldots, \boldsymbol{x}_N]^{\mathrm{T}}$, so that

$$\boldsymbol{x} = (z_1^2, \ldots, z_M^2, 2z_1 z_2, \ldots, z_1, \ldots, z_M, 1)^{\mathrm{T}} \quad (4)$$

where $\boldsymbol{x}$ is an arbitrary column of $\boldsymbol{X}$, and

$$\boldsymbol{x} \cdot \boldsymbol{\psi} = \boldsymbol{z}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{z} + \boldsymbol{b}^{\mathrm{T}} \boldsymbol{z} + c \quad (5)$$

In order to use Bayesian statistics, an additive noise model is assumed on each of the data points, i.e., we assume that

$$\boldsymbol{X} \boldsymbol{\psi} = \boldsymbol{\varepsilon} \quad (6)$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ is a single draw from a unit multivariate Gaussian distribution.

In order to solve the matrix equation $\boldsymbol{X} \boldsymbol{\psi} = \boldsymbol{0}$, without a prior on $\boldsymbol{\psi}$ one can find the eigenvector corresponding to the smallest eigenvalue of $\boldsymbol{X}^{\mathrm{T}} \boldsymbol{X}$. If we consider the following eigenvalue problem:

$$\boldsymbol{X}^{\mathrm{T}} \boldsymbol{X} \boldsymbol{v} = s \boldsymbol{v} \quad (7)$$

it is sufficient to find an eigenvector $\boldsymbol{v}$ associated with eigenvalue $s = 0$. Existence of such a solution is guaranteed if the matrix $\boldsymbol{X}^{\mathrm{T}} \boldsymbol{X}$ is singular. In practice, however, we take the smallest eigenvalue.

The solution to Eq. (7) is a least-squares solution of the equation $||\boldsymbol{X} \boldsymbol{\psi}||^2 = \boldsymbol{0}$ since the value $||\boldsymbol{X} \boldsymbol{\psi}||^2$ is minimal when all of the partial derivatives with respect to $\boldsymbol{\psi}$ are zero. Consider

$$\frac{\partial}{\partial \boldsymbol{\psi}} \left(||\boldsymbol{X} \boldsymbol{\psi}||^2 \right) = \frac{\partial}{\partial \boldsymbol{\psi}} (\boldsymbol{\psi}^{\mathrm{T}} \boldsymbol{X}^{\mathrm{T}} \boldsymbol{X} \boldsymbol{\psi}) \quad (8)$$

$$= 2 \boldsymbol{X}^{\mathrm{T}} \boldsymbol{X} \boldsymbol{\psi} \quad (9)$$

Setting Eq. (9) equal to zero and dividing out the constant, one can see that a minimal solution using eigenvalue decomposition is a least-squares estimate.

The least-squares estimate introduced above places no constraint on the matrix $\boldsymbol{A}$, and therefore can produce a matrix with negative eigenvalues. If one wishes to ensure that the matrix is an ellipsoid, for example, $\boldsymbol{A}$ is required to have a full set of positive eigenvalues. This is what motivates our use of Bayesian statistics. Not only does the noise model explicitly consider Gaussian errors, but it also allows us to enforce prior knowledge on the parameters. We extend the least-squares solution by introducing the model in Eq. (6) and then use Bayes' Law, which allows us to combine the likelihood of the data $p(\boldsymbol{X}|\boldsymbol{\psi})$ with a prior $p(\boldsymbol{\psi})$ in order to produce the posterior:

$$p(\boldsymbol{\psi}|\boldsymbol{X}) \propto p(\boldsymbol{X}|\boldsymbol{\psi})p(\boldsymbol{\psi}) \quad (10)$$

The distribution $p(\boldsymbol{X})$ is constant with respect to the parameters $\boldsymbol{\psi}$ and so we replace it with the proportionality operator ($\propto$). This is useful because traditionally, one would compute $p(\boldsymbol{X}) = \int_{\boldsymbol{\psi}} p(\boldsymbol{X}|\boldsymbol{\psi})p(\boldsymbol{\psi})$ which can be intractable. Certain properties are maintained across proportionality, for example, the maximum of $p(\boldsymbol{\psi}|\boldsymbol{X})$ is the same as the maximum of $\kappa p(\boldsymbol{\psi}|\boldsymbol{X})$ for any κ. Since we only seek a maximum a priori estimate of the parameters, we only use equivalence of distributions up to proportionality.

With a careful choice of prior (discussed in Section 4), one can find the maximum a priori estimate of the parameters, given the data, by finding zeros of the log-posterior derivatives. In the context of fitting a general quadric and following from Eq. (6), we assume additive Gaussian noise on Eq. (1) in which case the likelihood is taken to be

$$p(\boldsymbol{X}|\boldsymbol{\psi}) = \mathcal{N}(\boldsymbol{X} \boldsymbol{\psi}|\boldsymbol{0}, \boldsymbol{I}) \quad (11)$$

In other words, the errors in computing Eq. (1) are jointly Gaussian, with a spherical unit covariance $\boldsymbol{I}$. The assumption of a spherical covariance is equivalent to saying that the variables are independent, given the parameters $\boldsymbol{\psi}$. This is true, since the error from the original surface of one point tells us nothing about the error of a neighbouring point. The covariances are assumed to be identical as a matter of simplicity. The identity is chosen rather than some scalar multiple, since the parameter becomes superfluous when the prior is introduced. The hyper-parameter σ (introduced later in the text) provides all of the information necessary to model the surface.

It would be possible at this point to use a prior for the vector $\boldsymbol{\psi}$. Rather than doing this we introduce a bijection between $\boldsymbol{A}$ and $\boldsymbol{\psi}$ through the vector $\boldsymbol{\theta}$ defined in Section 3. This allows the hyper-

parameters to be set in terms of the matrix rather than the somewhat unintuitive values of $\boldsymbol{\psi}$. In fact, using a matrix normal on $\boldsymbol{A}$ with scalar parameters is equivalent to assuming a multivariate Gaussian on the first elements of $\boldsymbol{\psi}$.

We only place a prior distribution over the matrix $\boldsymbol{A}$ since it contains information about the shape of the surface. This is discussed in detail in Section 7, with the choice of hyper-parameters discussed more in Section 4.

The matrix normal distribution is defined as

$$\mathcal{MN}(\boldsymbol{A}\,|\,\boldsymbol{W},\,\boldsymbol{U},\,\boldsymbol{V}) = \kappa \mathrm{e}^{[-\frac{1}{2}\mathrm{tr}(\boldsymbol{V}^{-1}(\boldsymbol{A}-\boldsymbol{W})^{\mathrm{T}}\boldsymbol{U}^{-1}(\boldsymbol{A}-\boldsymbol{W}))]} \tag{12}$$

where $\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{M\times M}$ and

$$\kappa = ((2\pi)^{np/2}|\boldsymbol{V}|^{n/2}|\boldsymbol{U}|^{n/2})^{-1} \tag{13}$$

The posterior is then found by multiplying Eq. (11) by Eq. (12) to give

$$p(\boldsymbol{\psi}|\boldsymbol{X}) \propto \mathcal{N}(\boldsymbol{X}\boldsymbol{\psi}|\boldsymbol{0},\boldsymbol{I})\mathcal{MN}(\boldsymbol{A}\,|\,\boldsymbol{W},\,\boldsymbol{U},\,\boldsymbol{V}) \tag{14}$$

$$\propto \mathrm{e}^{-\frac{1}{2}||\boldsymbol{X}\boldsymbol{\psi}||^2}\mathrm{e}^{-\frac{1}{2}\mathrm{tr}(\boldsymbol{V}^{-1}(\boldsymbol{A}-\boldsymbol{W})^{\mathrm{T}}\boldsymbol{U}^{-1}(\boldsymbol{A}-\boldsymbol{W}))} \tag{15}$$

If we then choose $\boldsymbol{U} = \boldsymbol{V} = \sigma\boldsymbol{I}$, and ensure that $\boldsymbol{W}$ is diagonal and symmetric, the posterior can be simplified to

$$p(\boldsymbol{\psi}|\boldsymbol{X}) \propto \mathrm{e}^{-\frac{1}{2}||\boldsymbol{X}\boldsymbol{\psi}||^2}\mathrm{e}^{-\frac{1}{2\sigma^2}\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}-2\boldsymbol{W}\boldsymbol{A}+\boldsymbol{W}^2)} \tag{16}$$

$$\propto \mathrm{e}^{-\frac{1}{2}||\boldsymbol{X}\boldsymbol{\psi}||^2}\mathrm{e}^{-\frac{1}{2\sigma^2}(\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})-\mathrm{tr}(2\boldsymbol{A}\boldsymbol{W})+\mathrm{tr}(\boldsymbol{W}^2))} \tag{17}$$

Taking the natural log and then the first partial derivative yields:

$$\frac{\partial\mathrm{ln}p(\boldsymbol{\psi}|\boldsymbol{X})}{\partial\boldsymbol{\psi}} = -\frac{1}{2}\frac{\partial}{\partial\boldsymbol{\psi}}||\boldsymbol{X}\boldsymbol{\psi}||^2$$
$$-\frac{1}{2\sigma^2}\frac{\partial}{\partial\boldsymbol{\psi}}\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}) - \frac{1}{\sigma^2}\frac{\partial}{\partial\boldsymbol{\psi}}\mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{A}) \tag{18}$$

Before computing the vector derivatives, we first note the matrix derivatives for a symmetric $\boldsymbol{A}$ and $\boldsymbol{W}$ can be written as (see Ref. [25] for more details):

$$\frac{\mathrm{d}(\mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{A}))}{\mathrm{d}\boldsymbol{A}} = \boldsymbol{W} + \boldsymbol{W}^{\mathrm{T}} - \boldsymbol{W}\circ\boldsymbol{I} \tag{19}$$

$$\frac{\mathrm{d}(\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}))}{\mathrm{d}\boldsymbol{A}} = 2(\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}} - \boldsymbol{A}\circ\boldsymbol{I}) \tag{20}$$

The operator $\circ$ is the Hadamard product for matrices, and produces a matrix of equal size by element wise multiplication: if $\boldsymbol{S} = \boldsymbol{Q}\circ\boldsymbol{R}$ then $S_{ij} = Q_{ij}R_{ij}$ for each i and j. More intuitively, Eqs. (19) and (20) represent a doubling of all the off-diagonal elements, leaving the diagonals unchanged.

Since $\boldsymbol{A}$ is symmetric we change notation for the derivatives and set

$$\hat{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}} - \boldsymbol{A}\circ\boldsymbol{I} \tag{21}$$

$$\hat{\boldsymbol{W}} = \boldsymbol{W} + \boldsymbol{W}^{\mathrm{T}} - \boldsymbol{W}\circ\boldsymbol{I} \tag{22}$$

The final derivative can then be calculated as

$$\frac{\partial\mathrm{ln}p(\boldsymbol{\psi}|\boldsymbol{X})}{\partial\boldsymbol{\psi}} = -\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{\psi} - \frac{1}{\sigma^2}\begin{bmatrix}\boldsymbol{\theta}_{\hat{A}}\\\boldsymbol{0}\end{bmatrix} + \frac{1}{\sigma^2}\begin{bmatrix}\boldsymbol{\theta}_{\hat{W}}\\\boldsymbol{0}\end{bmatrix} \tag{23}$$

In the final step we separate $\boldsymbol{\psi}$ from $\boldsymbol{X}$ and σ by distributing multiplication over addition. Here $\boldsymbol{\psi}$ depends on $\boldsymbol{A}$ and $\boldsymbol{\theta}_{\hat{A}}$ on $\hat{\boldsymbol{A}}$. Let the matrix $\boldsymbol{J}$ be a diagonal matrix which takes a value of 1 on the diagonal when the corresponding value of $\boldsymbol{\theta}_{\hat{A}}$ is a diagonal element, and 2 otherwise. Setting the partials to zero it then follows that

$$\left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} + \frac{1}{\sigma^2}\begin{bmatrix}\boldsymbol{J} & \boldsymbol{0}\\\boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\right)\boldsymbol{\psi} - \frac{1}{\sigma^2}\begin{bmatrix}\boldsymbol{\theta}_{\hat{W}}\\\boldsymbol{0}\end{bmatrix} = 0 \tag{24}$$

Rearranging yields

$$\boldsymbol{\psi} = \left(\sigma^2\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X} + \begin{bmatrix}\boldsymbol{J} & \boldsymbol{0}\\\boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\right)^{-1}\begin{bmatrix}\boldsymbol{\theta}_{\hat{W}}\\\boldsymbol{0}\end{bmatrix} \tag{25}$$

4 Choice of prior

We use a matrix normal prior rather than a Wishart or inverse Wishart prior for the matrix $\boldsymbol{A}$. The Wishart distributions are commonly used, well principled, and suitable for use with empirical data. However, computing a maximum a priori estimate in a similar fashion to that used in the previous section leads to a solution that must be determined by search, as opposed to the closed-form of Eq. (25). The requirement that Wishart and inverse Wishart matrices are symmetric positive definite is another limitation, particularly if cylinders or hyperboloids are necessary. Section 7 discusses how to construct a mean matrix $\boldsymbol{W}$ in order to favour different shapes of surface.

The prior is only cast over the matrix $\boldsymbol{A}$ since the parameters $\boldsymbol{b}$ and c contain information about the center and also the width of the resulting surface. These values are also coupled, in the sense that a change in center affects both $\boldsymbol{b}$ and c. Section 6 discusses how to extract the center and width from the parameters, but their relationship is pathological. Constraining $\boldsymbol{A}$ alone provides just enough information on the shape of the surface

to ensure that it does not become a hyperboloid, without affecting the quality of the fit.

The matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ determine the variance of the matrix $\boldsymbol{A}$ from the mean $\boldsymbol{W}$. Smaller values more strongly favour the mean as opposed to the data. For our experiments we simply choose the parameter $\sigma \in \mathbb{R}^+$, which gives control over how close the matrix $\boldsymbol{A}$ is to the identity. The value σ is free, in the sense that it can be chosen arbitrarily or depending on a larger dataset. The results in Section 8 describe how it is chosen for our experiments; however, it could also be computed using a training set containing prior information about the variance of the observed quadric from the mean.

5   3D data

The results in Section 3 apply to data of arbitrary dimension. This section provides explicit formulae for 3 dimensions. We note that the same principles apply for 2 dimensional data, allowing us to model ellipses and hyperbola.

The parameters $\boldsymbol{A}$ and $\boldsymbol{b}$ can be written as follows:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad (26)$$

In this case $\boldsymbol{\psi}$ is defined as

$$\boldsymbol{\psi} = \begin{bmatrix} \boldsymbol{\theta_A} \\ \boldsymbol{b} \\ c \end{bmatrix} \quad (27)$$

where

$$\boldsymbol{\theta_A} = [a_{11}, a_{22}, a_{33}, a_{12}, a_{13}, a_{23}]^{\mathrm{T}} \quad (28)$$

The matrix $\boldsymbol{X}$ is the quadratic data matrix:

$$\boldsymbol{X} = [\boldsymbol{x}_0, \dots, \boldsymbol{x}_N] \quad (29)$$

where

$$\boldsymbol{x}_i = (z_{1i}^2, z_{1i}^2, z_{3i}^2, 2z_{1i}z_{2i}, \dots, z_{1i}, z_{2i}, z_{3i}, 1)^{\mathrm{T}} \quad (30)$$

The matrix $\boldsymbol{J}$ is then taken to be

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & 2\boldsymbol{I} \end{bmatrix} \quad (31)$$

and

$$\boldsymbol{\theta}_{\hat{\boldsymbol{W}}} = \begin{bmatrix} \boldsymbol{1} \\ \boldsymbol{0} \end{bmatrix} \quad (32)$$

6   Parametrisation

The quadric can be parameterised by finding the transformation between the canonical shape, such as the unit sphere, and the resulting surface.

Consider the following:

$$(\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Lambda} (\boldsymbol{x} - \boldsymbol{\mu}) = \tau \quad (33)$$

Multiplying out and rearranging leads to

$$\boldsymbol{x}^{\mathrm{T}} \boldsymbol{\Lambda} \boldsymbol{x} - \boldsymbol{\mu}^{\mathrm{T}} (\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^{\mathrm{T}}) \boldsymbol{x} + \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\Lambda} \boldsymbol{\mu} - \tau = 0 \quad (34)$$

If we then take

$$\boldsymbol{A} = \boldsymbol{\Lambda} \quad (35)$$

$$\boldsymbol{b} = -(\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^{\mathrm{T}}) \boldsymbol{\mu} \quad (36)$$

$$c = \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{\Lambda} \boldsymbol{\mu} - \tau \quad (37)$$

it can be seen that Eq. (1) takes on a canonical form of $\boldsymbol{y}^{\mathrm{T}} \boldsymbol{W} \boldsymbol{y} = 1$ after an affine transformation of the data, where $\boldsymbol{W}$ is a diagonal matrix with entries from the set $\{1, 0, -1\}$.

For an ellipsoid, the matrix $\boldsymbol{W}$ is the identity. Faces and vertices on a unit sphere are generated using a spherical coordinate system, and we transform only the vertices. Letting $\boldsymbol{v}$ be an arbitrary vertex on the unit sphere, and $\boldsymbol{USU}^{\mathrm{T}} = \boldsymbol{A}$ be the eigendecomposition of $\boldsymbol{A}$ ($\boldsymbol{A}$ is Hermitian, so $\boldsymbol{U}^{\mathrm{T}} = \boldsymbol{U}^{-1}$), it follows that

$$\hat{\boldsymbol{v}} = \boldsymbol{U} \left(\frac{\boldsymbol{S}}{\tau} \right)^{-\frac{1}{2}} \boldsymbol{v} + \boldsymbol{\mu} \quad (38)$$

where $\hat{\boldsymbol{v}}$ is the required ellipsoid vertex. The same principle can be used for any canonical shape, including cylinders, ensuring that the principal axis is aligned with the zero entry of $\boldsymbol{W}$.

Alternative methods exist for computing the solutions of Eq. (1): isosurface algorithms, for example. A parametric approach is beneficial for its speed; however, it then also requires algebraic methods for merging with other surfaces, which can become complicated.

7   Eigendecomposition

The shape of the quadric can be determined by analysing the eigenvalues of $\boldsymbol{A}$.

Since $\boldsymbol{A}$ is real and symmetric it is a special case of a Hermitian matrix, and so it follows that all of its eigenvalues are real. In the case of a symmetric positive definite matrix all of the eigenvalues are positive, and so the quadric forms an ellipsoid. If any of the eigenvalues are zero then the axis of the corresponding eigenvector is indeterminate; in the case of 2 positive eigenvalues it is a cylinder, for example. A single negative eigenvalue leads to a hyperboloid.

The final classifications can be obtained by studying the canonical form of the quadric

introduced in Section 6, $\boldsymbol{y}^{\mathrm{T}}\boldsymbol{W}\boldsymbol{y} = 1$, where $\boldsymbol{y}$ is an affine transformation of the arbitrary data values $\boldsymbol{z}$, and $\boldsymbol{W}$ is a diagonal matrix. Multiplying this equation out for different combinations of sign on the diagonal of $\boldsymbol{W}$ reduces to a scalar equation which can be compared to studied quadratic forms, as presented in Andrews and Séquin [3].

It is important to note that Eq. (1) is not only invariant to scale, but also negation, in the sense that multiplying the whole equation by a scalar $\alpha \in \mathbb{R}$ does not change the solution set. Consider the following expansion:

$$\boldsymbol{z}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{z} + c = 0 \tag{39}$$

$$\Longleftrightarrow \alpha(\boldsymbol{z}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{z} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{z} + c) = 0 \tag{40}$$

$$\Longleftrightarrow \boldsymbol{z}^{\mathrm{T}}(\alpha\boldsymbol{A})\boldsymbol{z} + \alpha\boldsymbol{b}^{\mathrm{T}}\boldsymbol{z} + \alpha c = 0 \tag{41}$$

Further, let $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^{\mathrm{T}}$ be the eigendecomposition of $\boldsymbol{A}$. Then $\alpha\boldsymbol{A} = \boldsymbol{U}^{\mathrm{T}}(\alpha\boldsymbol{S})\boldsymbol{U}$. This means that a scaling and negation of the eigenvalues of $\boldsymbol{A}$ only affects the other parameters, but not the solutions, e.g., a full set of negative eigenvalues yields the same solution as a full set of positive eigenvalues. Since the signs of the eigenvalues of $\boldsymbol{A}$ are the same as the signs of the diagonal entries of $\boldsymbol{W}$ we can relate them to the shape of the surface.

A range of different surface shapes is attainable in this way. For example, a hyperboloid can be made from two negative and one positive eigenvalues, which is the same as two positive and one negative eigenvalues. A table showing the classification of the surface for different eigenvalues is shown in Table 1, and represents only a sample of the possible combinations. We note that the parameters $\boldsymbol{b}$ and c also have an affect on the shape. For example, if $\boldsymbol{b} = \boldsymbol{0}$ and we have one positive eigenvalue and two zero, the resulting shape is no longer a paraboloid but a pair of planes. Examples of these shapes can be seen in Fig. 1.

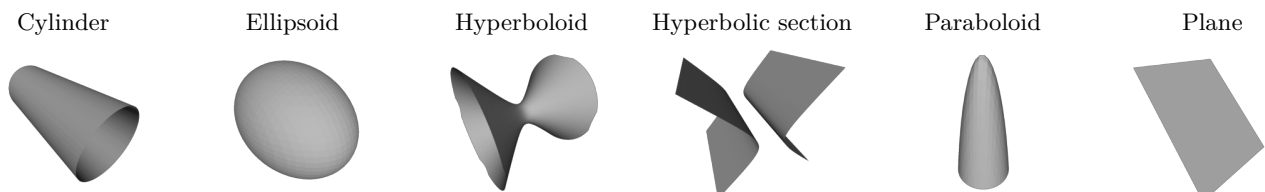# 8   Results

The results consider two scenarios. Firstly, it is

Table 1   Relationship between eigenvalues and shape of the corresponding quadric

Shape	Positive	Zero	Negative
Ellipsoid	3	0	0
Cylinder	2	1	0
Paraboloid	2	1	0
Plane	0	0	0
Hyperboloid	2	0	1
Hyperbolic section	1	1	1

shown that the method is able to perform well on data drawn from simulated quadrics, ellipsoids with known parameters. The algorithm is then shown to perform well on empirical data from a collection of 3D point clouds. The results are compare to quadric fitting without a prior and also against the work of Li and Griffiths [12], who solve a generalised eigenvalue problem similar to that of Ref. [11].

The work presented by Li and Griffiths is a candidate example of a modern ellipsoid fitting algorithm, from the larger set of algorithms available (e.g., Refs. [2–8]). Our method is a Bayesian model, which allows us to provide a parameter which expresses a measure of how close the shape should be to a canonical shape, such as a sphere. Rather than comparing to all other algorithms we simply show that a hard constraint to fit an ellipse does not always lead to the best fit to the data, particularly when there is a large percentage of missing or excess noise.

In the final section of the results, we give an example of a fit to a simulated hyperboloid, primarily to show that our algorithm also works for different surface types. It also demonstrates the value in choosing different diagonals on the mean matrix $\boldsymbol{W}$.

8.1   Algorithmic complexity

One of the most attractive attributes of a direct method for fitting, such as ours and Refs. [12] or [11], is its efficiency. A well known alternative is to compute the squared Euclidean (or *geometric*) distance between each point and the surface, and then minimise the sum of squares [9]. This is

| Cylinder | Ellipsoid | Hyperboloid | Hyperbolic section | Paraboloid | Plane |

Fig. 1   A collection of canonical quadric shapes.

difficult since computing the Euclidean distance is a non-linear problem (see Refs. [26, 27] for more details), and the complexity is dependent on a polynomial root finding algorithm. Minimising the sum of squared errors in this way requires a non-linear least-squares algorithm such as Levenberg–Marquardt. Computing the error alone has an order of $\mathcal{O}(NM^3)$, which is a lower bound on the final fitting algorithm as it involves iteratively computing the error and parameter gradients at each step. It is possible, however, to approximate the distance function, as shown in Refs. [1] and [6], and fit using generalised eigenvalues or using a linear least-squares estimate; these methods have a much lower complexity, but do not allow us to use a Bayesian prior when fitting.

The complexity of the algorithm presented in the paper arises in the computation of the scatter matrix $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ in Eq. (25), which is $\mathcal{O}(NM^2)$; the subsequent matrix operations have much lower complexity ($\mathcal{O}(M^3)$ at worst). This makes ours an efficient approach, particularly since the scatter can be computed independently of the value σ. Faster algorithms exist for computing covariances, for example Ref. [28], at the expense of memory, although it may be generally quicker to use parallel computation, such as a GPU, to improve performance.

8.2 Simulated data

We first show results on a simulated dataset. Data is drawn from an ellipsoid of known dimensions, some of the data is removed, and spherical Gaussian noise is added. We show that with the correct choice of hyper-parameters, the Bayesian method fits best.

Figure 2 provides an example of fitting an ellipsoid to a simulated dataset, with missing data. In this experiment we only use 30% of the training data from the original ellipsoid and add Gaussian noise with a standard deviation of 0.01. It can be seen from the results that a value of $\sigma = 10$ produces a result most plausibly in agreement with the original, whereas the result of fitting with no prior produces a hyperboloid. It is clear from this example that lower values of σ produce a result closer to a sphere.

In order to determine the quality of the fit, we compare the vertices on the original quadric with the fitted quadric using the Hausdorff distance. Firstly, the distance between a point $\boldsymbol{x}_i \in X$ and a point cloud $Y = \{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_N\}$ is taken to be

$$\varrho(\boldsymbol{x}_i, Y) = \min_{j \in \{1, \ldots, N\}} (|\boldsymbol{x}_i - \boldsymbol{y}_j|) \qquad (42)$$

The Hausdorff distance is then:

$$H(X, Y) = \max(\max_{i \in \{1, \ldots, M\}} (\varrho(\boldsymbol{x}_i, Y)), \\ \max_{j \in \{1, \ldots, N\}} (\varrho(\boldsymbol{y}_j, X))) \qquad (43)$$
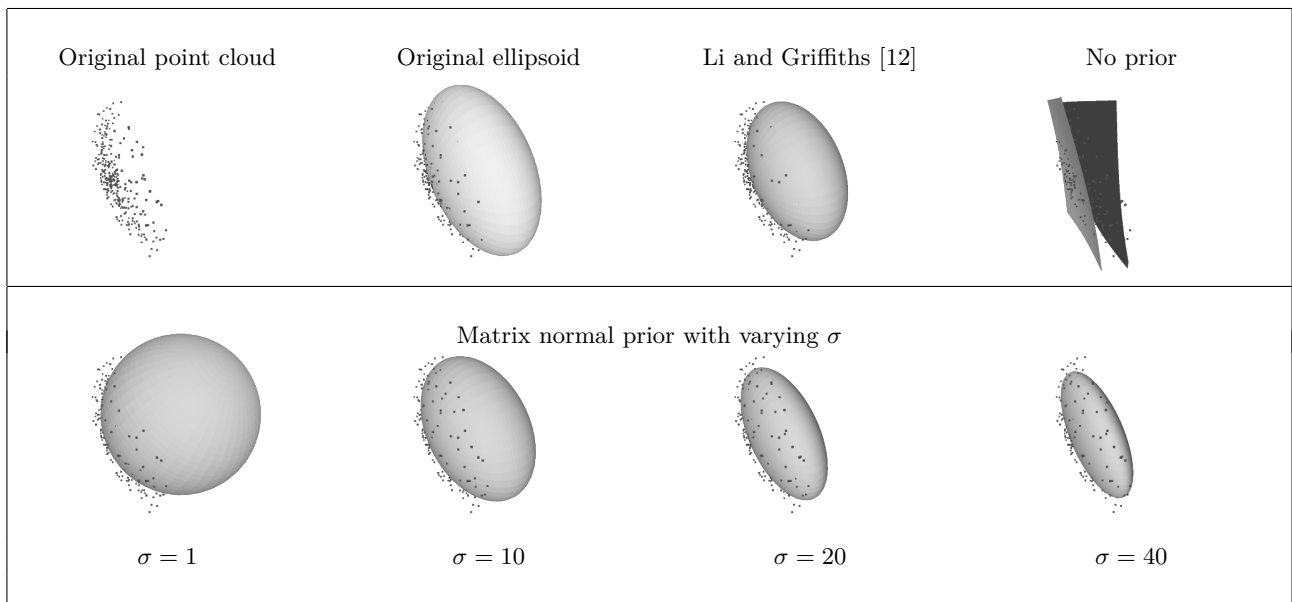
Fig. 2 A collection of ellipses fitted to data. Bottom: results of fitting with a prior. Top: results of fitting with no prior. The first two images are the point cloud and the ellipsoid that they were generated from.

The first experiment considers how each of the algorithms perform when there is missing data. We generate an arbitrary ellipsoid and remove $N\%$ of the data, and add spherical Gaussian noise with 0.01 standard deviations. The parameters of the ellipsoid are drawn as follows:

$$\boldsymbol{\Lambda} \sim \mathcal{W}(\omega^2 \boldsymbol{I}, 20) \tag{44}$$

$$\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{0}, \omega^2 \boldsymbol{I}) \tag{45}$$

$$\tau \sim \mathcal{N}(1, \omega^2) \tag{46}$$

with $\omega^2 = 0.01$. The parameters $\boldsymbol{A}$, $\boldsymbol{b}$, and c are computed as in Section 6. The percentage of data removed is varied between 10% and 100%, and we choose the hyper-parameter $0.1 < \sigma < 30$ which produces the best Hausdorff distance. Using a multivariate matrix normal prior is compared with (1) using no prior at all and (2) the ellipsoid specific least-squares method of Li and Griffiths [12].

The results of the first experiment can be seen in Fig. 3. Our method consistently outperforms both methods, although the distance drops below the standard deviation in Gaussian noise in all methods after 60% of the data is used. Fitting without a prior also gives reasonable results if enough of the data is used.

In the second simulation experiment we demonstrate how the method is robust to noise by varying the amount of additive Gaussian noise added to each sampled ellipsoid. We denote the noise level by $\hat{\tau}$. In order to demonstrate the quality of fit over a range of values we sample 10 quadrics and compute error statistics. In this experiment we use 100% of the points, i.e., there is no missing data.
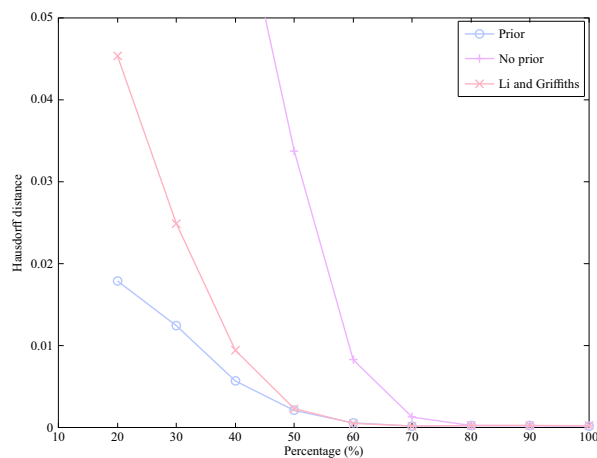
The hyper-parameter is fixed at $\sigma = 10$.

The results from the second experiment can be seen in Fig. 4. Our method produces a Hausdorff distance which is comparable to Ref. [12] for lower values of $\hat{\tau}$, while as $\hat{\tau}$ increases the use of a prior proves to be beneficial. Although fitting without a prior gives reasonable results when $\hat{\tau}$ is small, it becomes very inaccurate when $\hat{\tau}$ is greater than 0.06. This tends to be due to negative eigenvalues in the matrix $\boldsymbol{A}$ which does not produce an ellipsoid.

In order to show that the algorithm works for other types of quadric surface, we simulated a hyperboloid and removed a percentage of the points. Three quadrics were fitted to the data using different priors for each of: an ellipsoid, no prior, and a hyperboloid. For the ellipsoid, $\boldsymbol{W} = \boldsymbol{I}$ and $\sigma = 1\mathrm{e}^{-6}$; for the hyperboloid $\boldsymbol{W} = \mathrm{Diag}(-1, 1, 1)$ and $\sigma = 10$. Note that the variance for the ellipsoid needs to be low enough or it will favor a different shape. The hyperboloid hyper-parameter is aligned with the x-axis, as for the simulated data, based on our knowledge of the shape. Fitting without a prior chooses the hyperboloid's medial axis to be aligned with the z-axis. The results of this experiment can be seen in Fig. 5. The original quadric is in the top left of the figure. It is clear that using a hyperboloid prior is the best option for this data.

8.3 Empirical data

We provide some examples which demonstrate the use of our algorithm on real data from point clouds extracted from a visual structure from a motion pipeline. In this scenario, regions of the point cloud
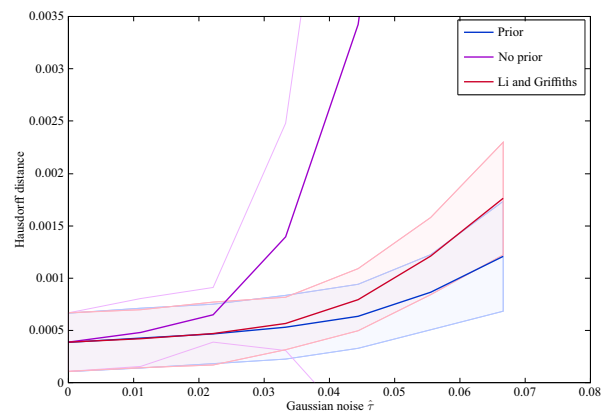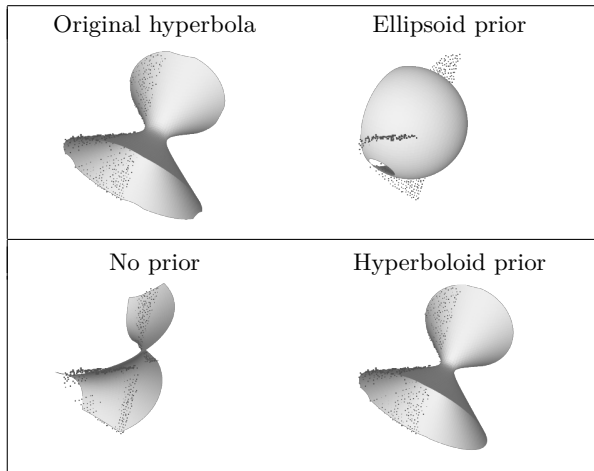

Fig. 3 The Hausdorff distance for a varying amount of missing data.

Fig. 4 The Hausdorff distance for a varying amount of Gaussian noise. The shaded regions represent one standard deviation from the mean.

Fig. 5 Fitting a hyperbola using a variety of different priors. The original points are shown as black dots.

can be left without any reconstructed points. An example is shown in Fig. 6. A smooth surface must be fitted to the roof of the phone box, presenting a challenge for most surface fitting algorithms due to the noisy and shapeless extracted points. Previous algorithms either do not fit at all, such as the one in Li and Griffiths [12], or do not allow control over the curvature of the data, which is the case when using no prior. The surface gradient can be chosen to match the roof curvature by varying σ and produces a final result as shown.

Some examples of the algorithm on point cloud data are shown in Fig. 7. The point clouds are computed from a collection of photographs comprising a post box, rugby ball, and roof. The surfaces are visually plausible even when there are large regions of missing data. This is particularly visible on the roof data, for which only the front is visible in each of the photographs.

9   Conclusions

We conclude by observing that state of the art methods for quadric fitting give reasonable results on noisy point clouds. Our algorithm provides a means to enforce a prior, allowing the algorithm to better fit the quadric that the points were drawn from, particularly when there is missing data or a large amount of noise. This has a practical use for real datasets, since it allows a user to specify the curvature of the surface where there are few points available.
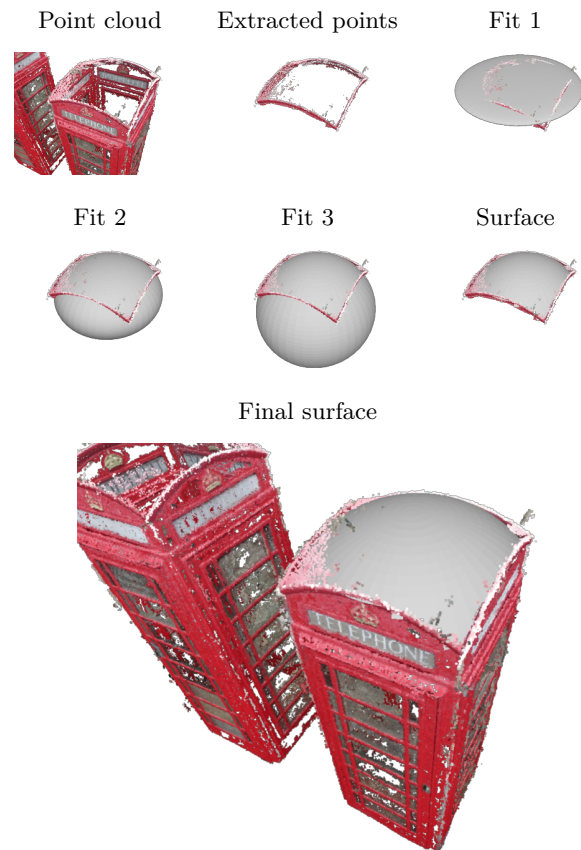
Fig. 6   Phone box dataset. The top left shows the original point cloud and the extracted points that a quadric must be fitted to, followed by 3 examples with varying sigma values and the final surface at the bottom right.

Appendix A   Symmetry of the matrix $\boldsymbol{A}$

The matrix $\boldsymbol{A}$ is assumed symmetric without loss of generality. This is possible since the function:

$$f : \mathbb{R}^{M \times M} \to \mathbb{R}^M \qquad (47)$$

$$\boldsymbol{A} \mapsto \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} \qquad (48)$$

is not injective for any $\boldsymbol{x} \in \mathbb{R}^M$. For example, if we take

$$\boldsymbol{B} = \frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}}) \qquad (49)$$

which is a symmetric matrix, then

$$\boldsymbol{x}^{\mathrm{T}} \boldsymbol{B} \boldsymbol{x} = \boldsymbol{x}^{\mathrm{T}}(\frac{1}{2}(\boldsymbol{A} + \boldsymbol{A}^{\mathrm{T}}))\boldsymbol{x} \qquad (50)$$

$$= \frac{1}{2}(\boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} + \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{x}) \qquad (51)$$

Since $\boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{x}$ is a scalar, it is also symmetric, and so

$$\boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{x} = (\boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{x})^{\mathrm{T}} \qquad (52)$$

$$= \boldsymbol{x}^{\mathrm{T}}(\boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}})^{\mathrm{T}} \qquad (53)$$

$$= \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} \qquad (54)$$

Point cloud   Surface

Post box

Rugby ball

Roof
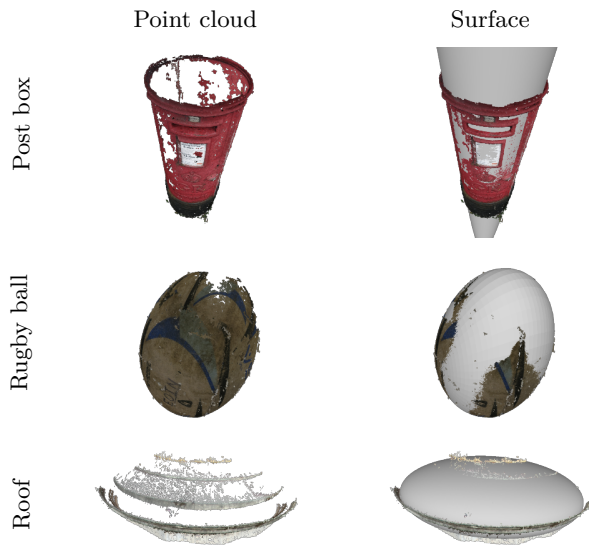
Fig. 7 Some examples of the quadric fitting algorithm on point cloud data.

It then follows that

$$\boldsymbol{x}^{\mathrm{T}} \boldsymbol{B} \boldsymbol{x} = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} \qquad (55)$$

The choice of symmetry then follows, because if $\boldsymbol{A}$, $\boldsymbol{b}$, and c satisfy $\boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} + \boldsymbol{b}^{\mathrm{T}} \boldsymbol{x} + c = 0$, then so does $\boldsymbol{B}$, and $\boldsymbol{B}$ is symmetric.

References

[1] Taubin, G. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 13, No. 11, 1115–1138, 1991.

[2] Ahn, S. J.; Rauh, W.; Cho, H. S.; Warnecke, H.-J. Orthogonal distance fitting of implicit curves and surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 24, No. 5, 620–638, 2002.

[3] Andrews, J.; Séquin, C. H. Type-constrained direct fitting of quadric surfaces. *Computer-Aided Design and Applications* Vol. 11, No. 1, 107–119, 2014.

[4] Atieg, A.; Watson, G. A. A class of methods for fitting a curve or surface to data by minimizing the sum of squares of orthogonal distances. *Journal of Computational and Applied Mathematics* Vol. 158, No. 2, 277–296, 2003.

[5] Gander, W.; Golub, G. H.; Strebel, R. Least-squares fitting of circles and ellipses. *BIT Numerical Mathematics* Vol. 34, No. 4, 558–578, 1994.

[6] Lukács, G.; Martin, R.; Marshall, D. Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation. In: *Lecture Notes in Computer Science, Vol. 1406*. Burkhardt, H.; Neumann, B. Eds. Springer Berlin Heidelberg, 671–686, 1998.

[7] Petitjean, S. A survey of methods for recovering quadrics in triangle meshes. *ACM Computing Surveys* (*CSUR*) Vol. 34, No. 2, 211–262, 2002.

[8] Ruiz, O.; Arroyave, S.; Acosta, D. Fitting of analytic surfaces to noisy point clouds. *American Journal of Computational Mathematics* Vol. 3, No. 1A, 18–26, 2013.

[9] Rouhani, M.; Sappa, A. D. Implicit polynomial representation through a fast fitting error estimation. *IEEE Transactions on Image Processing* Vol. 21, No. 4, 2089–2098, 2012.

[10] Pratt, V. Direct least-squares fitting of algebraic surfaces. *ACM SIGGRAPH Computer Graphics* Vol. 21, No. 4, 145–152, 1987.

[11] Fitzgibbon, A.; Pilu, M.; Fisher, R. B. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 21, No. 5, 476–480, 1999.

[12] Li, Q.; Griffiths, J. G. Least squares ellipsoid specific fitting. In: Proceedings of Geometric Modeling and Processing, 335–340, 2004.

[13] Halir, R.; Flusser, J. Numerically stable direct least squares fitting of ellipses. In: Proceedings of the 6th International Conference in Central Europe on Computer Graphics and Visualization, Vol. 98, 125–132, 1998.

[14] Dai, M.; Newman, T. S.; Cao, C. Least-squares-based fitting of paraboloids. *Pattern Recognition* Vol. 40, No. 2, 504–515, 2007.

[15] Subrahmonia, J.; Cooper, D. B.; Keren, D. Practical reliable Bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 18, No. 5, 505–519, 1996.

[16] Cohen-Steiner, D.; Alliez, P.; Desbrun, M. Variational shape approximation. *ACM Transactions on Graphics* Vol. 23, No. 3, 905–914, 2004.

[17] Wu, J.; Kobbelt, L. Structure recovery via hybrid variational surface approximation. *Computer Graphics Forum* Vol. 24, No. 3, 277–284, 2005.

[18] Yan, D.-M.; Liu, Y.; Wang, W. Quadric surface extraction by variational shape approximation. In: *Lecture Notes in Computer Science, Vol. 4077*. Kim, M.-S.; Shimada, K. Eds. Springer Berlin Heidelberg, 73–86, 2006.

[19] Schnabel, R.; Wahl, R.; Klein, R. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum* Vol. 26, No. 2, 214–226, 2007.

[20] Li, Y.; Wu, X.; Chrysathou, Y.; Sharf, A.; Cohen-Or, D.; Mitra, N. J. GlobFit: Consistently fitting primitives by discovering global relations. *ACM Transactions on Graphics* Vol. 30, No. 4, Article No. 52, 2011.

[21] Ohtake, Y.; Belyaev, A.; Alexa, M.; Turk, G.; Seidel, H.-P. Multi-level partition of unity implicits. In: Proceedings of ACM SIGGRAPH 2005 Courses, Article No. 173, 2005.

[22] Hamann, B. Curvature approximation for triangulated surfaces. In: *Computing Supplementum, Vol. 8*. Farin, G.; Noltemeier, H.; Hagen, H.; Kndel, W. Eds. Springer Vienna, 139–153, 1993.

[23] Vona, M.; Kanoulas, D. Curved surface contact patches with quantified uncertainty. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 1439–1446, 2011.

[24] Powell, M. W.; Bowyer, K. W.; Jiang, X.; Bunke, H. Comparing curved-surface range image segmenters. In: Proceedings of the 6th International Conference on Computer Vision, 286–291, 1998.

[25] Petersen, K. B.; Pedersen, M. S. *The Matrix Cookbook*. Technical University of Denmark. Version: November 15, 2012. Available at https://www.math.uwaterloo.ca/∽hwolkowi/matrixcookbook.pdf.

[26] Lott, G. K. Direct orthogonal distance to quadratic surfaces in 3D. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 36, No. 9, 1888–1892, 2014.

[27] Eberly, D. Distance from a point to an ellipse, an ellipsoid, or a hyperellipsoid. Geometric Tools, LLC, 2011. Available at http://www.geometrictools.com/Documentation/DistancePointEllipseEllipsoid.pdf.

[28] Kwatra, V.; Han, M. Fast covariance computation and dimensionality reduction for sub-window features in images. In: *Lecture Notes in Computer Science, Vol. 6312*. Daniilidis, K.; Maragos, P.; Paragios, N. Eds. Springer Berlin Heidelberg, 156–169, 2010.

Daniel Beale is a post-doctoral research officer at the University of Bath. He obtained his Ph.D. degree in computer science and master degree of mathematics (M.Math.) from the University of Bath. He has worked in industrial positions as a software and systems engineer. His interests are in the application of mathematics, statistics, and probability theory to problems in computing, particularly in the area of computer vision.

Yong-Liang Yang is a lecturer (assistant professor) in the Department of Computer Science at the University of Bath. He received his Ph.D. degree and master degree in computer science from Tsinghua University. His research interests include geometric modelling, computational design, and computer graphics in general.

Neill Campbell is a lecturer in computer vision, graphics, and machine learning in the Department of Computer Science at the University of Bath. Previously he was a research associate at University College London (where he is an honorary lecturer) working with Jan Kautz and Simon Prince. He

completed his M.Eng. and Ph.D. degrees in the Department of Engineering at the University of Cambridge under the supervision of Roberto Cipolla with George Vogiatzis and Carlos Hernàndez at Toshiba Research. His main area of interest is learning models of shape and appearance in 2D and 3D.

Darren Cosker is a Royal Society Industrial Research Fellow at Double Negative Visual Effects, London, and a reader (associate professor) at the University of Bath. He is the director of the Centre for the Analysis of Motion, Entertainment Research and Applications (CAMERA), starting from September 2015. Previously, he held a Royal Academy of Engineering Research Fellowship, also at the University of Bath. His interests are in the convergence of computer vision, graphics, and psychology, with applications in movies and video games.

Peter Hall is a professor of visual computing at the University of Bath. His research interests focus around the use of computer vision for computer graphics applications. He is known for automatically processing real photographs and video into art, especially abstract styles such as Cubism, Futurism, etc. More recently he has published papers on classification and detection of objects regardless of how they are depicted such as photos, drawings, paintings, etc. He is also interested in using computer vision methods to acquire 3D dynamic models of complex natural phenomena such as trees, water, and fire for use in computer games, TV broadcast, and film special effects.

Other papers from this open access journal are available free of charge from http://www.springer.com/journal/41095. To submit a manuscript, please go to https://www.editorialmanager.com/cvmj.