Available online at www.sciencedirect.com
ScienceDirect
Computer-Aided Design 39 (2007) 987–1002
COMPUTER-AIDED DESIGN
www.elsevier.com/locate/cad

Dynamic B-spline surface reconstruction: Closing the sensing-and-modeling loop in 3D digitization

Yunbao Huang, Xiaoping Qian*

Mechanical, Materials and Aerospace Engineering, Illinois Institute of Technology, Chicago, IL 60616, United States

Received 7 February 2007; accepted 12 June 2007

Abstract

In this paper, we present a new B-spline surface reconstruction approach, called *dynamic surface reconstruction*, aiming to close the sensing-and-modeling loop in 3D digitization. At its core, this approach uses a recursive least squares method, the *Kalman filter*, to dynamically reconstruct the B-spline surface as the surface data are acquired. That is, the acquired data are dynamically incorporated into the surface model and the updated surface model is then used to dynamically guide further data acquisition. It thus enables a closed-loop shape sensing-and-modeling methodology for 3D digitization.

Our technical contribution lies on the exploitation of the recursive nature of the Kalman filter for B-spline surface reconstruction. This enables dynamic parameterization of data points, dynamic determination of next optimal sensing locations, and low-discrepancy based efficient sensing and reconstruction. Experiments demonstrate that such dynamic surface reconstruction leads to *more efficient* data acquisition and *better* surface reconstruction.

Keywords: 3D digitization; Reverse engineering; Dynamic surface reconstruction; B-spline surface

1. Introduction

Three-dimensional (3D) digitization (a.k.a. reverse engineering) is a process to obtain digital models (often CAD models) from physical objects. It is widely used in aerospace, automobile, biomedical, and consumer product industries to facilitate product design, analysis and manufacturing from pre-existing products. The processing steps from physical objects to digital models can be roughly divided into two: (1) data acquisition where various modalities of 3D sensors, either tactile, optical, magnetic, acoustic, or x-ray, are used individually or in combination to obtain a 3D point cloud of the physical objects; (2) post-sensing data processing where a shape model, either a mesh, a surface or a solid model, is reconstructed. These two steps are typically sequential with data acquisition preceding the shape reconstruction and the reconstruction is done offline. Such sequential and separate data acquisition and offline reconstruction essentially form an open loop process in 3D digitization. Such open-loop processing can potentially lead to inefficient sensing since there is no timely feedback from the reconstructed surface to sensing. It may also lead to poor surface reconstruction due to potential data missing and outliers in the acquired point cloud.

This paper presents an approach aiming to close such a gap between the 3D sensing and reconstruction. The rapidly growing 3D sensing techniques and ever-advancing computing power have made it possible now to reconstruct the surface as the data are collected. In the proposed approach, the acquired data are dynamically incorporated into the surface model and the updated surface model is then used to dynamically guide further data acquisition. It thus enables a closed-loop shape sensing-and-modeling methodology for 3D digitization.

The approach presented in this paper, called *dynamic B-spline surface reconstruction*, is based on the Kalman filter. The new approach has the following distinguishing characteristics:

- More efficient sensing through dynamic sensing planning

  When a surface is reconstructed as the data are collected, the reconstructed surface can be used to find the next best sensing location based on, *e.g.* surface curvatures, root-mean-squared (RMS) error, or the surface uncertainty.

[*] Corresponding author. Tel.: +1 312 567 5855.
E-mail address: qian@iit.edu (X. Qian).

0010-4485/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2007.06.008

This way, sensing only takes place at the most desirable locations such as the missing data area or at the high uncertainty area. We present two methods: one dynamically determining the next best sensing location based on its effect on minimizing the surface uncertainty, and the other determining the sensing location based on the low-discrepancy sequences generated by a quasi-Monte-Carlo (QMC) method. This dynamic sensing approach extends our earlier work on uncertainty-based multisensor dynamic sensing-and-modeling [9] into a generalized closed-loop framework for 3D sensing and digitization, which is applicable to both single sensor sensing and multisensor sensing.

- Better quality in reconstructed surface through dynamic data parameterization.

  As described in [13,17], the base surface for data parameterization in B-spline surface fitting is a significant factor affecting the resulting surface quality. As the base surface approximates the true surface better, it leads to better data parameterization, thus a better reconstructed surface. Consequently many existing reconstruction approaches often involve *iterative* parameterization where the same data points are parameterized against an evolving base surface *multiple* times. In the *dynamic* parameterization presented in this paper, the same data points will be only parameterized once. The dynamic B-spline surface reconstruction updates the surface as more data are collected. Thus the subsequently collected data are parameterized on the new surface updated with previously collected data, which utilizes all available prior measurement data, and approximates the true surface better and leads to better surface quality.

  Besides the application in dynamic sensing, the dynamic parameterization is also applicable for surface reconstruction from the static point cloud. In such cases, the sequence of incorporating data points into the recursive surface reconstruction can be judiciously determined, e.g. using QMC, to improve the resulting surface quality.

- Eliminating the need for large storage space to store the point cloud or wide transmission band to transmit the point cloud.

  With the advancement of various 3D sensors, some of these sensors can output point clouds of megabytes or even gigabytes size and they thus need large storage space. The use of recursive surface updating allows the measurement points to be incorporated into the surface model as they are collected. Thus it avoids the need for large storage space. This is especially useful in a networked or remote sensing environment where storage space or data bandwidth might be limited.

The remainder of this paper is organized as follows. Section 2 reviews prior work in surface reconstruction. Section 3 presents the mathematical basis for dynamic surface reconstruction through the Kalman filter. Section 4 discusses the properties of the dynamic surface reconstruction. Section 5 describes the advancements enabled by the dynamic surface reconstruction. Section 6 presents the experimental results. The computational complexity of the dynamic surface reconstruction is analyzed in Section 7. This paper concludes in Section 8.

2. Literature review

Surface reconstruction has been an active research topic due to its broad applications such as reverse engineering [22] and quality inspection [12]. In the mechanical computer aided design community, a tensor B-spline surface is the standard surface representation and its reconstruction has been studied extensively [13,17,25]. To obtain an accurate and smooth surface, hierarchical B-spline surface fitting [6], multilevel B-spline surface fitting [11], and local surface updating [14] techniques have been introduced. The dynamic surface reconstruction has also been reported in the computational geometry community to obtain a fine reconstruction surface [1,2]. However, they only address the post-sensing reconstruction, not addressing how to couple the sensing and reconstruction.

In the computer vision community, the Kalman filter has been used to build a tensor parametric surface [20,23,24] from multiple sensor data. Other forms of the recursive least squares based methods have been used for surface reconstruction [4,5]. However they were just utilized for statically integrating different sensor data, not for closed-loop sensing and modeling. In the computer aided design area, the incremental updating nature of the Kalman filter has been used to interactively deform the free-form surface [19], but not used for reconstructing the surface.

During the B-spline surface fitting, parameterization is a critical issue [21] because a poor choice of the base surface for parameterization may lead to a poor reconstruction. So an iterative process is often used to achieve a better base surface [13,17]. A dynamic base surface has been proposed by iteratively projecting the increased grid points from the base surface to the point cloud, and then reconstructing a new base surface from those projected points in the point cloud until the termination criterion is satisfied [16]. However, in our dynamic parameterization approach, the base surface can be updated with an arbitrary number of data points acquired at any location.

3. Mathematical basis for dynamic surface reconstruction

This section gives the mathematical basis for our dynamic surface reconstruction, including (1) B-spline surface representation, and (2) the Kalman filter for surface updating.

Note, it is assumed in this paper that the object has been properly segmented [22] so that only one B-spline surface is reconstructed for a given point cloud.

3.1. B-spline surface

B-spline surfaces are widely used to model free-form shapes in product design and manufacturing in automotive, aerospace and consumer products industries.

A bi-cubic B-spline surface has the form:

$$S(u, v) = \sum_{i,j} N_i(u) N_j(v) \mathbf{P}_{ij} \tag{1}$$

where N is the B-spline shape function and $\mathbf{P}_{ij}$ is the ij-th control point. The equation can also be expressed in a compact form:

$$S(u, v) = \mathbf{A}(u, v)\mathbf{P} \tag{2}$$

where $\mathbf{A}(u, v)$ and $\mathbf{P}$ are vectors of length n and n is the number of control points. See [18] for details on B-spline surface representation.

3.2. Kalman filter for surface updating

In order to fuse noisy sensor data (here we assume the noise is independent, white and Gaussian) into a B-spline surface, we choose the Kalman filter [10,26] to produce the statistically optimal estimate of the surface.

For any point on the B-spline surface $S(u, v)$, its sensor measurement is z, and its parameter is (u_z, v_z), we can get from Eq. (2)

$$z = \mathbf{A}(u_z, v_z)\mathbf{P} + \varepsilon \tag{3}$$

where ε is the measurement noise.

In the terminology of the Kalman filter, the above B-spline surface equation represents a linear system between the internal surface state $\mathbf{P}$ and external observation z. That is, the collection of control points $\mathbf{P}$ constitutes the internal state of the object shape, the measurement z with its uncertainty Λz forms the external observation of B-spline surface. $\mathbf{A}(u_z, v_z)$ corresponds to the measurement matrix H in [26]. Then we can get the Kalman gain [9] as

$$\begin{aligned}\mathbf{K}_l = {} & \mathbf{\Lambda P}_{l-1}\mathbf{A}^{\mathrm{T}}(u_z, v_z) \\ & \times \left(\mathbf{A}(u_z, v_z)\mathbf{\Lambda P}_{l-1}\mathbf{A}^{\mathrm{T}}(u_z, v_z) + \Lambda z\right)^{-1}\end{aligned} \tag{4}$$

where $\mathbf{K}_l$ is the l-th step Kalman gain, and $\mathbf{\Lambda P}_{l-1}$ is the state uncertainty at the $(l-1)$-th step.

The surface state and its uncertainty updating equation can be obtained as

$$\mathbf{P}_l = \mathbf{P}_{l-1} + \mathbf{K}_l\left(z - \mathbf{A}(u_z, v_z)\mathbf{P}_{l-1}\right) \tag{5}$$

$$\begin{aligned}&\text{(a): } \mathbf{\Lambda P}_l = \left(\mathbf{I} - \mathbf{K}_l\mathbf{A}(u_z, v_z)\right)\mathbf{\Lambda P}_{l-1} \quad \text{or} \\ &\text{(b): } (\mathbf{\Lambda P}_l)^{-1} = (\mathbf{\Lambda P}_{l-1})^{-1} \\ &\qquad + \mathbf{A}^{\mathrm{T}}(u_z, v_z)\,(\Lambda z)^{-1}\,\mathbf{A}(u_z, v_z).\end{aligned} \tag{6}$$

That is, for any new measurement z and its variance Λz at the l-th step, we can get the updated surface estimate $\mathbf{P}_l$ and uncertainty $\mathbf{\Lambda P}_l$ (its dimension is $n \times n$) through Eqs. (5) and (6) based on the prior surface state $\mathbf{P}_{l-1}$ and its variance $\mathbf{\Lambda P}_{l-1}$. Such recursive updating forms the basis of our dynamic B-spline surface reconstruction.

4. Dynamic surface reconstruction

In this section, we describe how we can use the recursive nature of the Kalman filter for dynamic surface reconstruction in two modes, and then also examine the effectiveness of the dynamic surface reconstruction by comparing it with the least squares based reconstruction. Here, the *dynamic surface reconstruction* refers to a surface reconstruction process in which the surface is reconstructed or updated from an *a priori* surface in an incremental manner, in which the points are incorporated into the surface model as they are collected.

For a given set of measurements $\{z_i, i = 1, \ldots m\}$ with the corresponding noise characteristics Λz_i, and the *a priori* surface estimate defined by $\mathbf{P}_0$ and $\mathbf{\Lambda P}_0$, we can iteratively update the surface from those m measurements with Eqs. (5) and (6). This surface reconstruction mode is defined here as the *incremental mode*. The pseudo-code for this incremental surface updating can be written as in Box I.

If all the measurements $\{z_i, i=1, \ldots m\}$ with corresponding noise Λz_i are parameterized with reference to one surface and given the same *a priori* surface estimate $\mathbf{P}_0$ and $\mathbf{\Lambda P}_0$, the resulting surface through Eqs. (5) and (6) can also be computed through the following equations.

$$\begin{aligned}\mathbf{P}_m = {} & \left((\mathbf{\Lambda P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) \right. \\ & \left. \times\ (\Lambda z_i)^{-1}\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \\ & \times \left((\mathbf{\Lambda P}_0)^{-1}\mathbf{P}_0 + \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1} z_i\right)\end{aligned} \tag{7}$$

$$\begin{aligned}\mathbf{\Lambda P}_m = {} & \left((\mathbf{\Lambda P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) \right. \\ & \left. \times\ (\Lambda z_i)^{-1}\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1}\end{aligned} \tag{8}$$

where $\mathbf{P}_0$ and $\mathbf{\Lambda P}_0$ are the initial estimate of surface and its uncertainty estimate, $\mathbf{A}(u_{z_i}, v_{z_i})$ is the B-spline shape function matrix corresponding to the measurement z, and Λz_i is the uncertainty of the measurement z_i.

In Eqs. (7) and (8), we can see that all the measurements are processed at once to produce the surface. This fitting mode is referred to as the *batch mode*. The pseudo-code for surface reconstruction in this batch mode is as follows:

```
Initialize M with $(\mathbf{\Lambda P}_0)^{-1}$, and W with $(\mathbf{\Lambda P}_0)^{-1}\mathbf{P}_0$.
For $i = 1$ to $m$
    $\mathbf{M} = \mathbf{M} + \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})$.
    $\mathbf{W} = \mathbf{W} + \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1} z_i$.
End for
Compute $\mathbf{\Lambda P}_m$ through $(\mathbf{M})^{-1}$.
Compute $\mathbf{P}_m$ through $\mathbf{\Lambda P}_m\mathbf{W}$.
```

These two modes of surface reconstruction have the following properties.

Property 1. *Assume all the measurements are parameterized with reference to one initial surface, for a given set of measurements $\{z_i, i = 1, \ldots m\}$ with corresponding noise Λz_i, the reconstructed surface, its control point $\mathbf{P}_m$ and its*

```
For $i = 1$ to $m$
    Compute the Kalman gain $\mathbf{K}_i = \boldsymbol{\Lambda}\mathbf{P}_{i-1}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\left(\mathbf{A}(u_{z_i}, v_{z_i})\boldsymbol{\Lambda}\mathbf{P}_{i-1}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) + \Lambda z_i\right)^{-1}$.
    Compute $\mathbf{P}_i$ through $\mathbf{P}_i = \mathbf{P}_{i-1} + \mathbf{K}_i\left(z_i - \mathbf{A}(u_{z_i}, v_{z_i})\mathbf{P}_{i-1}\right)$.
    Compute $\boldsymbol{\Lambda}\mathbf{P}_i$ through $\boldsymbol{\Lambda}\mathbf{P}_i = \left(\mathbf{I} - \mathbf{K}_i\mathbf{A}(u_{z_i}, v_{z_i})\right)\boldsymbol{\Lambda}\mathbf{P}_{i-1} = \boldsymbol{\Lambda}\mathbf{P}_{i-1} - \mathbf{K}_i\mathbf{A}(u_{z_i}, v_{z_i})\boldsymbol{\Lambda}\mathbf{P}_{i-1}$.
End for
```

Box I.

uncertainty covariance $\boldsymbol{\Lambda}\mathbf{P}_m$, can be obtained equivalently in a batch mode through Eqs. (7) and (8) or in an incremental mode through Eqs. (5) and (6). (The proof can be seen in the Appendix A).

Property 2. *If the measurements are parameterized with one initial surface, $\mathbf{P}_m$ and $\boldsymbol{\Lambda}\mathbf{P}_m$ computed from the Kalman filter are independent of the measurement sequence of z_i, z_j, $(i \neq j)$.*

Proof. This can be easily seen from the batch fitting mode equations (Eqs. (7) and (8)). □

Property 3. *If the measurements (total m points) are parameterized with reference to one initial surface, with the same m measurements and parameterization, the reconstructed surface from the weighted least squares equals that from the Kalman filter if*

(1) *the initial determinant* $\det(\boldsymbol{\Lambda}\mathbf{P}_0) \to \infty$ *and it is fused with the m points using the Kalman filter (Condition* 1*), or*

(2) *the initial surface as characterized by $\mathbf{P}_0$ and $\boldsymbol{\Lambda}\mathbf{P}_0$ is estimated with the weighted least squares from initial m_0 points and it is then fused with the remaining $(m - m_0)$ points using the Kalman filter (Condition* 2*).*

Proof. For the same measurements and parameterization, we can also reconstruct a B-spline surface of the same number of control points by employing the weighted least squares method (details are in the Appendix B). The reconstructed surface and its uncertainty can be represented by

$$\mathbf{P}_m = \left(\sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \times \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i \tag{9}$$

$$\boldsymbol{\Lambda}\mathbf{P}_m = \left(\sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1}. \tag{10}$$

Under condition (1)

From Eqs. (7) and (8), when the determinant of the initial surface uncertainty $\det(\boldsymbol{\Lambda}\mathbf{P}_0) \to \infty$, then $\det((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}) \to 0$, $(\Lambda\mathbf{P}_0)^{-1}\mathbf{P}_0 \to \mathbf{0}$, and $(\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} \to \mathbf{0}$ (in this case, the *a priori* shape is a surface with very large uncertainty), the reconstructed surface from m measurements through the Kalman filter changes to

$$\boldsymbol{\Lambda}\mathbf{P}_m = \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} = \left(\sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \tag{11}$$

and

$$\begin{aligned}\mathbf{P}_m &= \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \\ &\quad \times \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\,\mathbf{P}_0 + \sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i\right) \\ &= \left(\sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \\ &\quad \times \left(\sum_{i=1}^{m}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i\right).\end{aligned} \tag{12}$$

Comparing Eqs. (11) and (12) with Eqs. (9) and (10), we can see that the two surfaces and their uncertainty are actually equivalent when the determinant of the initial surface uncertainty $\det(\boldsymbol{\Lambda}\mathbf{P}_0) \to \infty$.

Under condition (2)

In Eqs. (7) and (8), the initial surface $\mathbf{P}_0$ and $\boldsymbol{\Lambda}\mathbf{P}_0$ can be estimated from a subset of the total measurements $(m_0, m_0 \leq m)$ by using the weighted least squares. From Eqs. (9) and (10), the $\mathbf{P}_0$ and $\boldsymbol{\Lambda}\mathbf{P}_0$ can be got by

$$\begin{aligned}\mathbf{P}_0 &= \left(\sum_{i=1}^{m_0}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \\ &\quad \times \sum_{i=1}^{m_0}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i \\ \boldsymbol{\Lambda}\mathbf{P}_0 &= \left(\sum_{i=1}^{m_0}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1}.\end{aligned} \tag{13}$$

Then, we can get the final reconstructed surface and its uncertainty based on the initial surface $\mathbf{P}_0$ and $\boldsymbol{\Lambda}\mathbf{P}_0$ and the other $m - m_0$ measurements with Eqs. (7) and (8) as

$$\begin{aligned}&\mathbf{P}_{m-m_0} \\ &= \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m-m_0}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\right. \\ &\qquad \left. \times\ (\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1}\end{aligned}$$

$$
\begin{aligned}
&\times \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\,\mathbf{P}_0 + \sum_{i=1}^{m-m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\, z_i \right) \\
&= \left(\sum_{i=1}^{m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right. \\
&\quad \left. + \sum_{i=1}^{m-m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right)^{-1} \\
&\quad \times \left(\sum_{i=1}^{m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\, z_i \right. \\
&\quad \left. + \sum_{i=1}^{m-m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\, z_i \right) \\
&= \left(\sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right)^{-1} \\
&\quad \times \left(\sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\, z_i \right) \qquad (14)
\end{aligned}
$$

$$
\begin{aligned}
&\boldsymbol{\Lambda}\mathbf{P}_{m-m_0} \\
&= \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m-m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) \right. \\
&\quad \left. \times\ (\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right)^{-1} \\
&= \left(\sum_{i=1}^{m-m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right. \\
&\quad \left. + \sum_{i=1}^{m_0} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right)^{-1} \\
&= \left(\sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i}) \right)^{-1}. \qquad (15)
\end{aligned}
$$

Comparing Eqs. (14) and (15) with Eqs. (9) and (10), we can see that the two surfaces and their uncertainty are equivalent when $\mathbf{P}_0$ and $\boldsymbol{\Lambda}\mathbf{P}_0$ are computed directly with the weighted least squares from m_0 points and are subsequently fused with $(m - m_0)$ points through the Kalman filter. □

This equivalency property between the Kalman filter and the weighted least squares gives us an intuitive sense why the Kalman filter can be used for surface reconstruction, since the least squares method is a common way for reconstructing the B-spline surface.

5. New advancements enabled by dynamic *B*-spline surface reconstruction

This section presents how dynamic surface reconstruction enables *dynamic parameterization* for a sequential data acquisition process, in which subsequently collected data are parameterized on the previously updated surface. This dynamic parameterization makes it possible to dynamically optimize or plan the sensing operations based on available information such as fitting error and its distribution in the reconstructed surface, surface uncertainty, curvature and other geometric properties. Based on this dynamic parameterization, we present two sensing strategies below:

- *Uncertainty minimization based sensing*: based on the previously sensed data, dynamically determining optimal subsequent sensing locations to minimize the surface uncertainty, which can result in more effective sensing and better surface quality.
- *Low-discrepancy based sensing*: acquiring shape data in a surface domain with the low-discrepancy sequences generated by the quasi-Monte-Carlo method, which can also lead to more efficient sensing and better surface quality. With this sensing strategy, the acquired point set can be augmented one point at a time with the goal of keeping all the points being evenly distributed in the sampling domain.

Note, these methods are also applicable for a static point cloud whereby the sequence of points being incorporated into the recursive surface update can be planned similarly.

5.1. Dynamic parameterization

Data point parameterization is a key step in the B-spline surface reconstruction. It involves a process mapping a point in 3D space to one parameter pair (u, v) on the parametric domain of a base surface. The corresponding surface point's parameter pair (u, v) is chosen as the parameter for the 3D point. Thus, for a given point cloud, the base surface is the key to obtain a better parameterization and a better surface.

In Fig. 1, we select three kinds of base surfaces to illustrate the effect of data parameterization on surface reconstruction. The first is a planar surface, the second is a Coons surface defined by the four boundary curves of the point cloud, and the last is the surface reconstructed from the point cloud using the Kalman filter, in which the data is first parameterized by the planar surface (Fig. 1(b)). From the fitting accuracy of resulting surfaces, we can see that the resulting surface has smaller RMS error when the base surface better approximates the underlying shape of the point cloud.

Therefore, to achieve a higher accuracy in surface reconstruction, a better initial base surface for parameterization is desired. Hence currently an iterative fitting process is often applied [13,17]. Let $z_1, z_2 \cdots z_m$ be the discrete sensed points and S_0 be initial base surface for parameterization, this common iterative process can be described in Fig. 2(a). In this iterative parameterization process, the previously reconstructed surface S_l is used as the base surface for parameterizing the entire point set $\{z_i\}$. Upon completion, the entire point set is used to reconstruct the surface S_{l+1}. A termination criterion such as the change of root mean squared error Δe between S_{l+1} and $\{z_i\}$ is used to determine whether such iteration should continue. As such, the iterative process includes several times of parameterization $\varphi\,(S_l, z_i, i = 1, \ldots m)$ and surface reconstruction that involves the entire point cloud. Assuming q is the times of iterative parameterization and surface reconstruction, the total number of point parameterization for the total m measurements is $q \times m$.

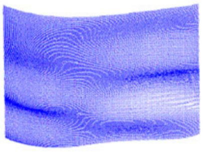
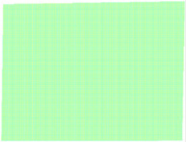
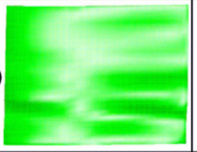
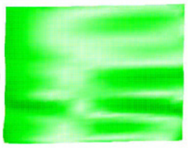
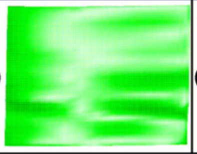
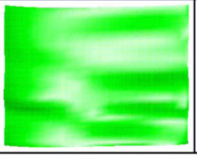
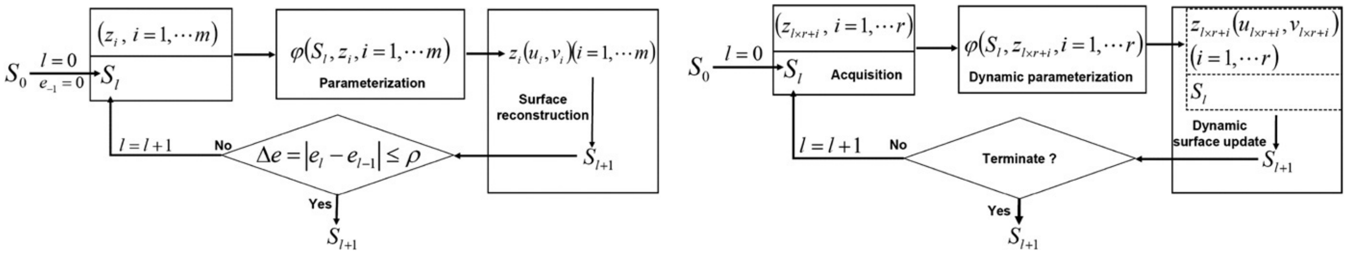
	Base surface for parameterization		Reconstructed surface	Fitting RMS error
 Point cloud with 104562 points	Planar surface	(a) 	(b) 	0.061888
	Coons surface	(c)	(d) 	0.061651
	Surface (b)	(e) 	(f) 	0.059990

Fig. 1. Different base surfaces for parameterization and the resulted fitting accuracy.



(a) Iterative parameterization.

(b) Dynamic parameterization.

Fig. 2. Iterative parameterization vs dynamic parameterization in surface reconstruction.

Instead of $q \times m$ times of iterative point parameterization, we can dynamically parameterize any number of dynamically acquired r points (or $1 \le r \le m$ for any static point cloud) and update the surface with just the parameterized r points (Fig. 2(b)). Such dynamic parameterization and surface update constitute our new dynamic parameterization scheme, which enables us to achieve a better reconstructed surface with fewer parameterization steps.

Assume the current surface estimate after fusing the point z_i is S_i' and the next measurement is z_{i+1}. The parameter of z_{i+1} can be obtained as follows:

$$(u_{i+1}, v_{i+1}) = \varphi\left(S_i', z_{i+1}\right).$$

Then the surface S_i' can be dynamically updated with parameterized z_{i+1} through Eq. (5). A similar parameterization can be applied in the batch mode through Eq. (7). In this manner, the base surface is dynamically updated and the subsequent measurements are then dynamically parameterized based on the updated surface. Therefore, we call it *dynamic parameterization*.

In order to achieve a better parameterization for the subsequent sampling points, we need to plan the data sampling sequence so that the dynamically updated base surface approximates more closely to the true surface. Therefore, there arises a sensing planning issue in dynamic surface reconstruction: what would be a desirable point sampling sequence so that the resulting base surface is beneficial for the subsequent data points' parameterization?

5.2. *Uncertainty minimization based sensing*

We show here how dynamic surface reconstruction can be useful in efficiently determining the next best sensing locations to minimize the surface uncertainty. This is especially useful for adding additional points at the missing data area during the sensing process. So a next best point (NBP) problem can be formulated as follows:

Let z be the next measurement on the free-from surface with parameter (u_z, v_z) and $\mathbf{\Lambda P}_i$ be the uncertainty covariance matrix at the time step i. From Eq. (6)(b) by setting $l = i$, we can get the updated uncertainty covariance matrix at time step $(i + 1)$ as

$$(\mathbf{\Lambda P}_{i+1})^{-1} = (\mathbf{\Lambda P}_i)^{-1} + \mathbf{A}^{\mathrm{T}}(u_z, v_z)\,(\Lambda z)^{-1}\,\mathbf{A}(u_z, v_z). \tag{16}$$

Then $\mathbf{\Lambda P}_{i+1} = \mathbf{\Lambda P}_i(\mathbf{I}+\mathbf{\Lambda P}_i\mathbf{A}^{\mathrm{T}}(u_z, v_z)(\Lambda z)^{-1}\mathbf{A}(u_z, v_z))^{-1}$. From reference [9], the determinant of $\mathbf{\Lambda P}_{i+1}$ can be got by

$$\det(\mathbf{\Lambda P}_{i+1}) = \det(\mathbf{\Lambda P}_i)/(1 + \mathbf{A}(u_z, v_z)\mathbf{\Lambda P}_i \times \mathbf{A}^{\mathrm{T}}(u_z, v_z)\,(\Lambda z)^{-1}). \tag{17}$$

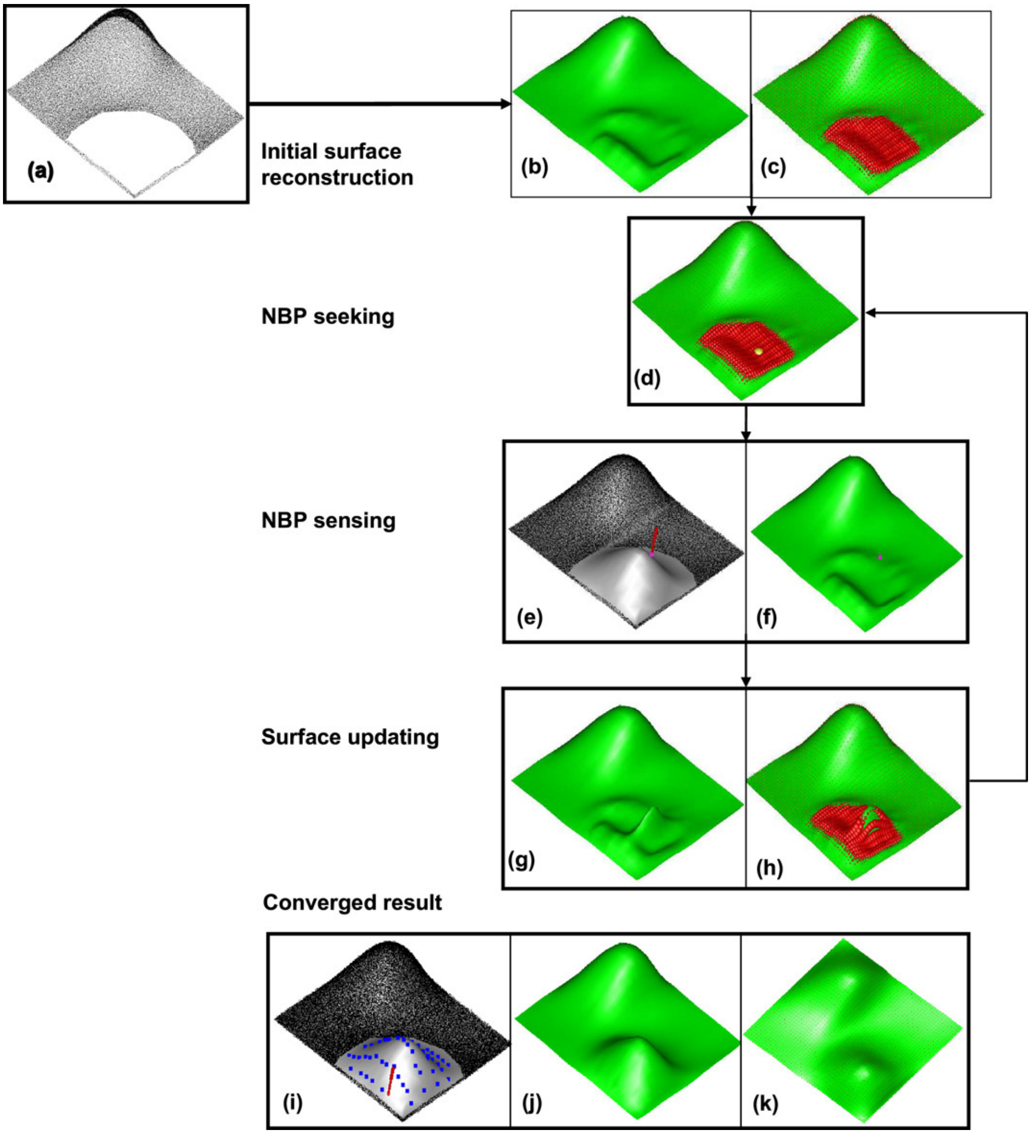


Fig. 3. Dynamic seeking–sensing–modeling procedures for dynamic surface reconstruction.

From Eq. (17), we can see that minimizing the determinant of $\mathbf{\Lambda P}_{i+1}$ is equivalent to maximizing $\mathbf{A}(u_z, v_z)\mathbf{\Lambda P}_i\mathbf{A}^{\mathrm{T}}(u_z, v_z)$. That is,

$$\min(\det(\mathbf{\Lambda P}_{i+1})) \sim \max\left(\mathbf{A}(u_z, v_z)\mathbf{\Lambda P}_i\mathbf{A}^{\mathrm{T}}(u_z, v_z)\right). \qquad (18)$$

So the NBP problem can be cast as *finding the optimal point on the reconstructed B-spline surface with maximal uncertainty to maximally reduce the reconstructed surface uncertainty*.

With the NBP, we can dynamically determine the best position on the physical surface from the dynamically reconstructed surface, and then the updated surface can again be used to determine the next best sensing location. Cycling the seeking, sensing and surface reconstruction process, we can efficiently reconstruct the surface. So the dynamic seeking–sensing–reconstruction basically involves the following three steps:

- *Seeking*: dynamically determine the next best sensing location to minimize the surface uncertainty by solving Eq. (18). One approach based on divide-and-conquer strategy was presented in [9] to dynamically seek such NBPs.
- *Sensing*: dynamically sense the object at the above computed NBP location.
- *Reconstruction*: dynamically update the surface based on the above sensed point through the Kalman filter (Eqs. (5) and (6)).

In Fig. 3, an initial surface (Fig. 3(b)) is first reconstructed from incomplete data (Fig. 3(a)). Based on the reconstructed surface uncertainty (Fig. 3(c)), the next best point can be found and shown in Fig. 3(d). Then we can determine the sensing location with the largest uncertainty on the physical surface and sense the next actual point (Fig. 3(e)). Fig. 3(f) shows the present surface and the actual acquired point. Fig. 3(g) and (h) show the updated surface and its uncertainty. Repeat such seeking, sensing and updating procedures until the dynamically reconstructed surface's uncertainty satisfies specified criteria. In the converged state, the initial point cloud and the dynamically acquired points are shown in Fig. 3(i), and the final surface and its uncertainty are shown in Fig. 3(j) and Fig. 3(k).

So here the dynamic surface reconstruction is the key to provide an updated surface to guide further sensing. It has been proven that such an uncertainty minimizing process converges monotonously [9].

5.3. Low-discrepancy based sensing

In this subsection, we present a low-discrepancy sampling method which can take advantage of our dynamic parameterization to improve the sensing efficiency and reconstruction quality. More specifically, a quasi-Monte Carlo (QMC) method is used to generate the low-discrepancy sensing sequence in the parametric domain.

The discrepancy of sampled points is used to characterize the quality of the even distribution of discrete points $X = \{x_i | 0 \le x_i < 1, 1 \le i \le m\}$ over a given interval (here we assume the interval is [0, 1]). It is defined as

$$D_m = \sup_{J \in J^*} \left| \frac{B(J; X)}{m} - |J| \right| \tag{19}$$

where J^* is the set of intervals $[0, t]$ with $0 < t \le 1$, $B(J; X)$ is the number points in X that fall into the particular interval of J, and $|J|$ is the length of the interval J. From Eq. (19), we can see that

(1) D_m is in between 0 and 1;

(2) more evenly distributed points means a smaller value of D_m. An overly denser or sparser distribution of points in one particular interval will lead to a larger value of D_m.

There are several well-known low-discrepancy sequence construction methods such as Faure [3], Halton [7] and Niederreiter [15]. Since the Halton sequence construction method performs well in lower dimensions, in this paper, we select the Halton construction to successively generate points with a low discrepancy. Consider a prime base b, the number i can be written in the form

$$i = d_j b^j + \cdots d_2 b^2 + d_1 b + d_0, \quad 0 \le d_j < b. \tag{20}$$

Then the i-th Halton sequence point X_i is defined by

$$X_i = \frac{d_0}{b} + \frac{d_1}{b^2} + \cdots + \frac{d_j}{b^{j+1}}. \tag{21}$$

In Eq. (21), the i-th Halton sequence point X_i is in the open interval (0, 1).

The 2D Halton point can be composed by using a product of two 1D Halton points with a different base, e.g. $b = 2$ in the u direction and $b = 3$ in the v direction. Since the 2D Halton point is generated in the open interval $(0, 1) \times (0, 1)$, no sampling takes place on the surface boundary. Since data points near the vicinities of surface boundary are critical to surface reconstruction, we introduce an additional one dimensional quasi-Monte Carlo (QMC) sequence on the boundary of the parametric domain as shown in Fig. 4 (here i starts from 0 in order to include the four corner points and the base $b = 2$).

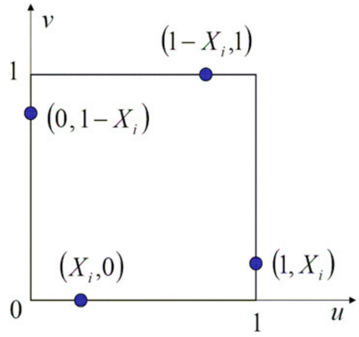


Fig. 4. Sampling scheme for QMC on the boundary of the parametric domain.

Assume we want to obtain r points through QMC. Of these points, $[t \times r/4]$ points are sampled on each of the four sides of the boundary, where t is the percentage of r points to be sampled on the boundary ($0 \le t < 1$). Denote the initial base surface as S_0, we can present such a dynamic sampling, sensing and reconstruction strategy through the quasi-Monte Carlo method as follows.

Dynamic sampling, sensing, and reconstruction strategy

Step 1. Reconstruct the initial surface S from available sensed data points and the *a priori* surface S_0 through the batch fitting mode of the Kalman filter.

Step 2. Identify the areas on the parametric domain of the surface S requiring additional sensing, e.g. missing data areas. (Here we assume the area interval is $[u_1{}^*, u_2{}^*] \times [v_1{}^*, v_2{}^*]$.)

Step 3. Dynamic sampling, sensing and surface updating.

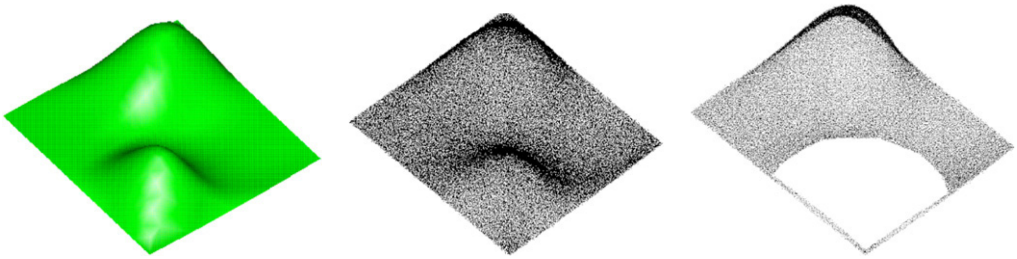
- Generate $[t \times r/4]$ points on each of the four boundary curves bounding the surface parametric domain with the one-dimensional Halton low-discrepancy sequence construction method (base $b = 2$).
- Generate the interior $r - [t \times r/4] \times 4$ low-discrepancy sequence points (u_i, v_i), $0 \le u_i, v_i \le 1$, $i = 1, \ldots, r - t * r$ and transform $\{(u_i, v_i)\}$ into the interval $[u_1^*, u_2^*] \times [v_1^*, v_2^*]$ by $u_i = u_1^* + u_i \times (u_2^* - u_1^*)$ and $v_i = v_1^* + v_i \times (v_2^* - v_1^*)$.
- Acquire shape data $\{z_i\}$ on the physical part surface at the locations closest to $\{S(u_i, v_i)\}$ where (u_i, v_i) is the sequence of r points generated above.
- Update the surface S with the acquired point set $\{z_i\}$ through the Kalman filter and get the updated surface S'. Set $S = S'$.
- Repeat the step 3 until a termination criterion is met. For example, a rule of thumb can be that the number of sensed points should be several times the number of model parameters.

6. Examples

Four examples are shown below to demonstrate the capabilities enabled by the dynamic surface reconstruction.

6.1. Example 1: Simulated surface

In Fig. 5, 9×10^4 data points (Fig. 5(b)) are uniformly sampled from a known bi-cubic B-spline surface (28×12 control points) (Fig. 5(a)). Gaussian noise was added (variance

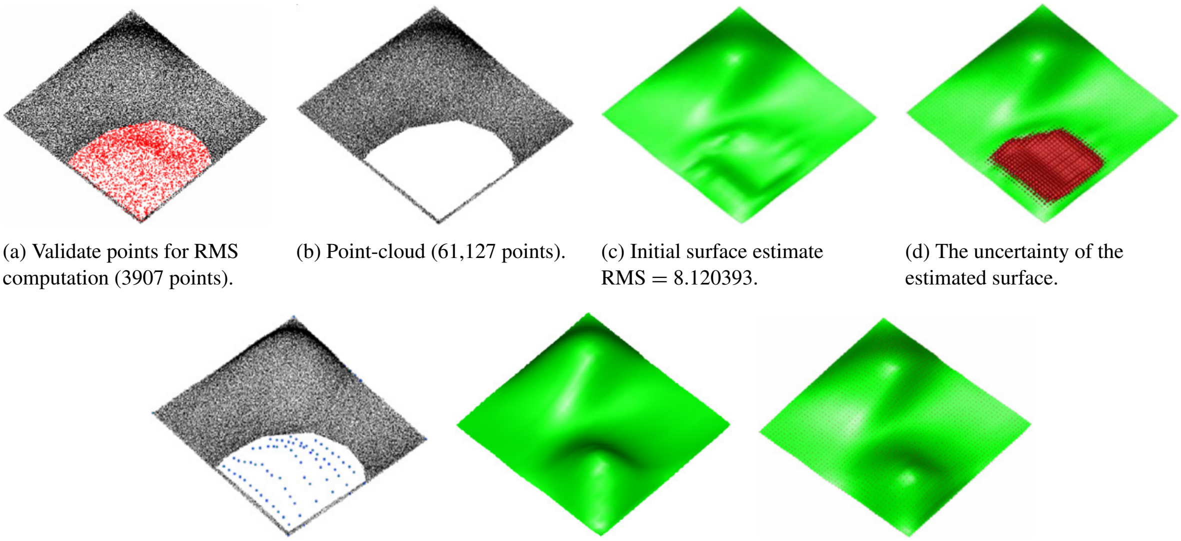


(a) A B-spline surface (28 × 12).

(b) Uniformly sampled 9×10^4 points from (a).

(c) Re-sampled 61127 points from (b).

Fig. 5. Sampled point cloud from a known B-spline surface.



(a) Validate points for RMS computation (3907 points).

(b) Point-cloud (61,127 points).

(c) Initial surface estimate RMS = 8.120393.

(d) The uncertainty of the estimated surface.

(e) Point-cloud + additional points (79 points).

(f) Dynamically reconstructed surface RMS = 0.067915.

(g) The uncertainty of the reconstructed surface.

Fig. 6. The final surface and its uncertainty through dynamic reconstruction. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

$\Lambda z^x = \Lambda z^y = \Lambda z^z = 0.01$). In the noisy data, 61,127 data points (Fig. 5(c)) are then selected to represent the acquired point cloud and to simulate the measurement with missing data on the surface.

Since the model structure of the surface is known, a planar surface with 28 × 12 control points bounding the point cloud is firstly selected as the *a priori* shape and the unit matrix is defined as the covariance matrix of its control points. Then the points are parameterized with reference to this *a priori* surface and the Kalman filter in the batch fitting mode is applied to estimate the initial surface and its uncertainty. A B-spline surface is first reconstructed with 28 × 12 control points. The dynamic seeking, sensing and surface updating is then iterated to minimize the surface uncertainty as described in Section 5.2. The 79 additional optimal points are added to reduce the surface uncertainty. It took total 88.75 s (incremental fitting with the Kalman filter: 0.14 s; seeking time: 88.61 s).

In Fig. 6, the initial surface estimate (Fig. 6(c)) and its uncertainty (the red ellipsoids in Fig. 6(d)) are obtained from the point cloud (Fig. 6(b)), and then dynamic seeking, sensing and surface reconstruction is run and 79 accurate points with variance $\Lambda z^x = \Lambda z^y = \Lambda z^z = 0.0001$ are added to obtain the surface with lower uncertainty (Fig. 6(f)). We can see that the final reconstructed surface has a lower uncertainty (Fig. 6(g)) and a smaller RMS error, which is computed with the randomly sampled 3907 points (Fig. 6(a)) in the missing data area at the actual surface.

To validate the sensing efficiency through our dynamic seeking–sensing–reconstruction approach, we compare it with a static plan-reconstruction method and an *ad hoc* method. The static plan-reconstruction is to pre-plan the optimal points in the parametric domain according to an initial reconstructed surface and its static surface uncertainty distribution, and then to map them to the physical surface for sensing. This differs from our dynamic approach in that the parameterization is done with a fixed static base surface, which is estimated from the initial point cloud (Fig. 6(c)). The *ad hoc* method refers to a way of randomly sensing the points in the missing data areas with a static base surface for parameterization.

Given an estimated surface and its physical counterpart, and a point's (u, v) parameter, we can compute the corresponding 3D point on the estimated surface and its normal, and then mapping it onto the physical surface by intersecting the physical surface with the ray going through the point and along the

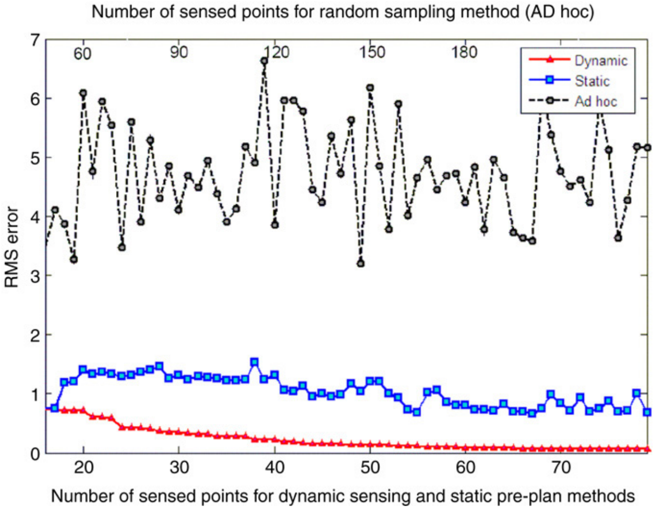


Fig. 7. Reconstructed surface accuracy with sensed points.

normal direction. This approach for acquiring the sensed point is used for all three methods.

In Fig. 7, the parameterization based on point projection to the base surface (i.e. the shortest distance between the point and the base surface) is applied, and the comparison of the reconstructed surface accuracy shows that (1) dynamic sensing and reconstruction method steadily converges to a much lower RMS error than the other two methods, (2) with three times of measured points in the dynamic approach, the *ad hoc* method still cannot achieve the same steady accuracy surface, (3) the static plan-reconstruction method also takes many more additional sensed points to reconstruct a good surface.

From the initial surface in Fig. 6(a) and (b), we can see a local bump missing in the missing data area, which leads to a significant difference between the computed surface points at the initial estimated surface and the mapped 3D points on the physical surface. Further, the static plan-reconstruction and *ad hoc* methods only use the initial estimated surface as the parameterization surface. However, the dynamic sensing and reconstruction approach utilizes the dynamically evolving surface as the base surface for parameterization. As more data points are collected, it approaches the physical surface better and thus achieves better parameterization results. Consequently, more effective sensing is achieved.

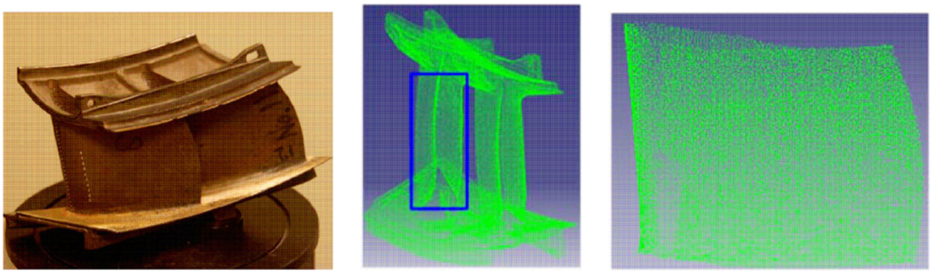
This example demonstrates that dynamic seeking, sensing, and surface updating (1) *can achieve more accurate surface digitization with fewer sensed points than static plan-reconstruction and ad hoc methods*, (2) *can achieve a higher-quality surface in terms of fitting accuracy due to the use of dynamic parameterization.*

6.2. Example 2: Aero nozzle

In Fig. 8, an aero nozzle part was scanned in 12 views with an area sensor (Minolta Vivid 910). In the merged point cloud, data missing still occurs due to the occlusion between the airfoils. Such missing data leads to large surface uncertainty. In [9], we have applied the dynamic seeking, sensing and surface reconstruction to invoke a tactile sensor to reduce the surface uncertainty. Instead of uncertainty-based dynamic seeking–sensing-and-modeling, we here employ the QMC based dynamic sensing method to improve the surface quality at the missing data area. The comparison of uniform sampling (zero discrepancy) with a static base surface for parameterization and the low-discrepancy QMC sampling with dynamic parameterization is given in Fig. 9.

As shown in Fig. 9, 21,763 data points for one surface (variance $\Lambda z^x = \Lambda z^y = \Lambda z^z = 0.01$ mm^2) are obtained through Minolta Vivid 910, and the initial surface (the number of control points is 19×35) is estimated with the multilevel Kalman filter method [8] since the underlying model structure is unknown. In the area where there are missing points, this area corresponds to the parametric domain $[0, 1] \times [0.52, 0.64]$ and the number of control points affected is 19×8. Thus, in this area, we used both the uniform sampling method (Fig. 9(d), (e)) and the QMC sampling method (Fig. 9(g) and (h)) to acquire 152 points with the touch probe (variance $\Lambda z^x = \Lambda z^y = \Lambda z^z = 0.0001$ mm^2) in a coordinate measurement machine. In order to compare the resulting surface quality of the two methods, an additional 2799 points (Fig. 9(a)) are measured with the touch probe in the missing data area. RMS errors are evaluated between the resulting surfaces and these additional measured points.

Comparing Fig. 9(f) and (i), we can see that the pre-planned uniform sampling (zero discrepancy) in the parametric grid with the static base surface (reconstructed from the initial point cloud shown in Fig. 9(c)) has larger RMS error (0.071987) than

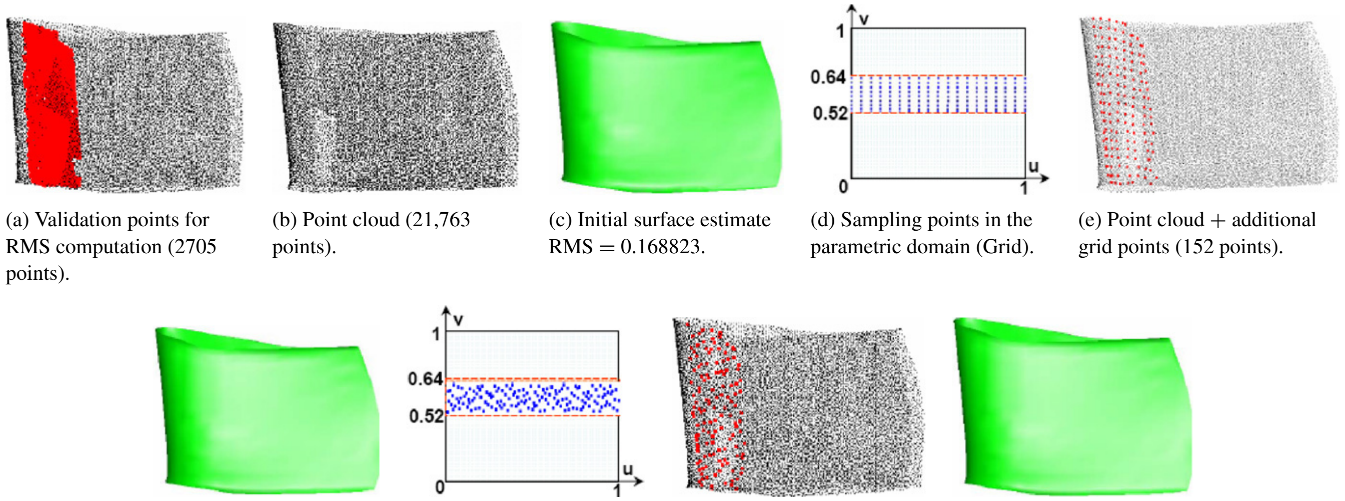


(a) Nozzle part.

(b) Scanned point cloud (185,734 points).

(c) Incomplete data due to occlusion (21,763 points).

Fig. 8. Scanned incomplete point cloud of nozzle part due to occlusion.



(a) Validation points for RMS computation (2705 points).

(b) Point cloud (21,763 points).

(c) Initial surface estimate RMS = 0.168823.

(d) Sampling points in the parametric domain (Grid).

(e) Point cloud + additional grid points (152 points).

(f) Reconstructed surface RMS = 0.071987.

(g) Sampling points in the parametric domain (QMC).

(h) Point cloud + additional QMC points (152 points).

(i) Dynamically reconstructed surface RMS = 0.062432.

Fig. 9. Reconstructed surfaces before and after fusing the sensed points.

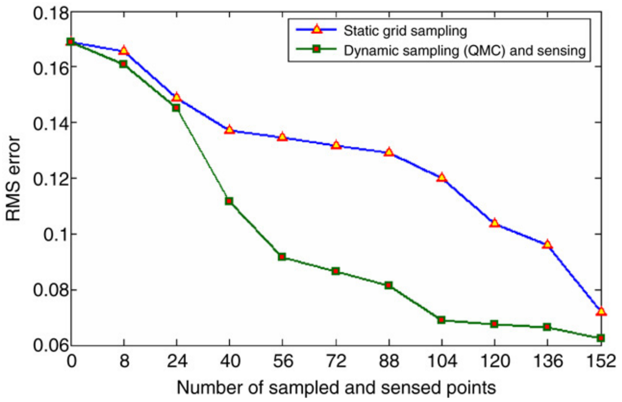


Fig. 10. The RMS errors vs. the number of additional updating points.

that of the low-discrepancy sequences generated by the QMC ($r = 8, t = 0$) method with dynamic parameterization (RMS error 0.062432).

Fig. 10 further compares the RMS error during the sequential process when we add uniform sampled points with static parameterization and QMC generated points with dynamic parameterization into the surface reconstruction. We can see that *dynamic sensing and surface reconstruction through quasi-Monte Carlo converges much faster to a better stable value than the static planned method due to the combination of low-discrepancy sampling and dynamic parameterization.*

6.3. Example 3: Manufactured free-form surface

In Fig. 11, a manufactured surface was scanned with Minolta Vivid910 and 104,562 data points (variance $\Lambda z^x = \Lambda z^y = \Lambda z^z = 0.01$ mm^2) were obtained.

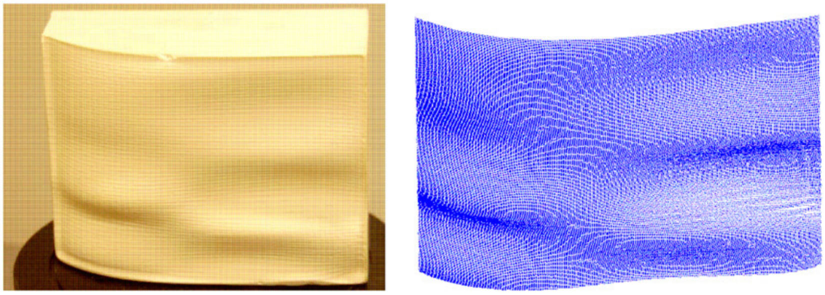
In Fig. 12, a planar base surface (the number of the control points $\mathbf{P}_0 : n = 28 \times 36$ and uncertainty covariance matrix $\mathbf{\Lambda P}_0 = I$ with the dimension 1008 × 1008) is used for parameterizing the point cloud, and the dynamic sampling and surface reconstruction described in Section 5.3 with different number of samples each time ($r = 2000, t = 0.24$; $r = 1000, t = 0.24$) are employed to reconstruct the surface. The RMS error between the point cloud and the reconstructed surface are then calculated. Comparing the resulting surfaces' error with that from the conventional iterative parameterization method shown in Fig. 1, we can see that the dynamic parameterization (both strategies in Fig. 12) can lead to a more accurate surface than the iterative parameterization method (Fig. 1(d), RMS = 0.061651 mm) using the Coons surface as the base surface.

In addition, we also examined the process of dynamic sampling and surface reconstruction.

Fig. 13 gives the comparison of RMS errors between the surfaces and data points during the process of dynamic parameterization and iterative parameterization. In the horizontal axis is the frequency of sampling r data points. For example 15/30 represents 2000 data points have been sampled with QMC for 15 times and 1000 data points have been sampled with QMC for 30 times. Note, during the dynamic parameterization, different data points are sampled from the point set (104,562 points) due to the use of QMC. Also in Fig. 13 is the iterative parameterization shown as a blue dotted line where the entire point set (104,562 points) is parameterized for each iteration.

Fig. 13 shows 20% of data points by the QMC sampling (at 10/20) with dynamic parameterization can lead to smaller surface fitting RMS error than the parameterizing the entire point cloud on the static base surface. Dynamically parameterizing the entire point cloud once with sequences generated by the QMC method (at 52/104) would lead to smaller surface fitting RMS error than iteratively parameterizing the entire point cloud five times on a static base surface.

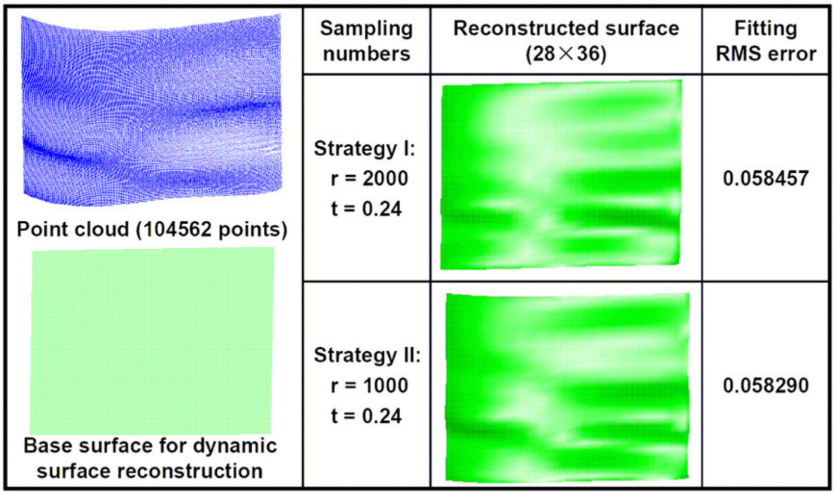
This example demonstrates that high quality surface can be obtained much faster by dynamic parameterization of the point



(a) Measured surface.

(b) Scanned point cloud (104,562 points).

Fig. 11. Manufactured surface and scanned point cloud.



	Sampling numbers	Reconstructed surface (28×36)	Fitting RMS error

Point cloud (104562 points)	Strategy I:
	r = 2000
	t = 0.24
			0.058457

Base surface for dynamic surface reconstruction	Strategy II:
	r = 1000
	t = 0.24
			0.058290

Fig. 12. The resulting surface through dynamic sampling and reconstruction (unit: mm).

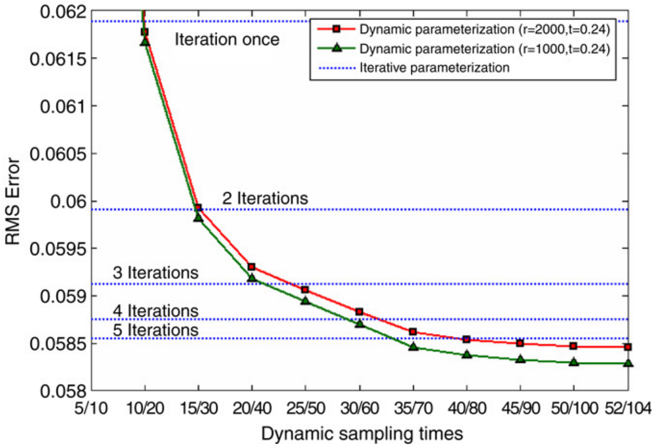


Fig. 13. RMS errors of the dynamic parameterization and iterative parameterization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

cloud with an appropriate dynamic sampling scheme (QMC in this paper) than iterative parameterization.

6.4. Example 4: Complex surface reconstruction

In Fig. 14, a human face was scanned by Minolta Vivid910 and 27,927 data points (variance $\Lambda z^x = \Lambda z^y = \Lambda z^z = 0.01\ \text{mm}^2$) were obtained.

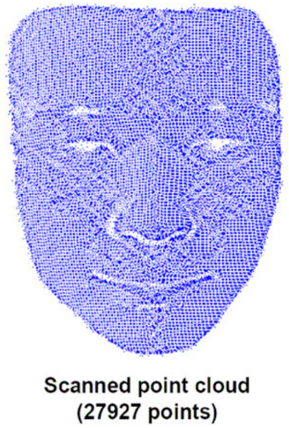
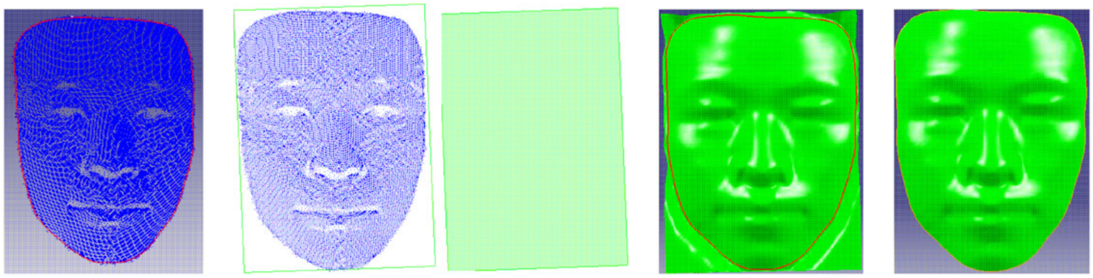


Fig. 14. A scanned human face and its point cloud.

From the scanned point cloud (Fig. 15(a)), we use a plane (Fig. 15(b)) as the base surface for parameterization and the initial surface estimate has the number of control points $\mathbf{P}_0$: 40×30 and uncertainty covariance matrix $\mathbf{\Lambda P}_0 = \mathbf{I}$ ($\mathbf{I}$ is the unit matrix with the dimension 1200×1200). Then the whole measured points are parameterized to reconstruct the surface (Fig. 15(c)) using the dynamic sampling and surface reconstruction strategy as described in Section 5.3 ($r = 2000, t = 0.04$). Finally we project the boundary curve defined from the boundary points to the reconstructed surface



(a) The scanned point cloud (27,927 points).

(b) The planar base surface for dynamic sampling and surface reconstruction.

(c) The reconstructed surface and the boundary curve.

(d) The trimmed surface with the boundary curve.

Fig. 15. The reconstructed surface through dynamic surface reconstruction.

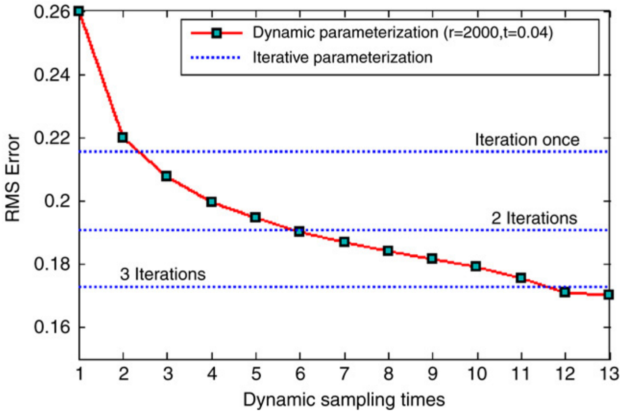


Fig. 16. RMS errors of dynamic parameterization and iterative parameterization. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

and trimmed the reconstructed surface to get the 3D face (Fig. 15(d)).

Fig. 16 gives the comparison of RMS error variations for dynamic parameterization and iterative parameterization. It can be seen that the dynamic parameterization enables dynamic sampling and surface reconstruction to achieve a more accurate surface faster than that of the iterative parameterization method. With about 23% data points (after sampling 2000 data points 3 times) from the dynamic parameterization, the RMS error is smaller than that using the iterative parameterization after fitting all the measurements once. After fusing once all the measurements (sample times = 13 times) from the dynamic parameterization, the RMS error is smaller than that of three times of iterative parameterization.

It can be seen from Figs. 15 and 16 that *for a dense data point cloud from a single sensor, a surface with many complicated shape undulations can still be dynamically reconstructed through the Kalman filter. A more accurate surface is obtained due to the use of dynamic parameterization.*

7. Computational complexity analysis

In the dynamic surface reconstruction, the use of two fitting modes of the Kalman filter facilitates the dynamic surface reconstruction for different sizes of sensing data. Suppose m is the number of sensing data points, n is the number of reconstructed surface's control points. From the increment mode procedure and its pseudo code described in Section 4, we can analyze the computational time as shown in Table 1.

So its complexity for the Kalman filter is $O(n^2)$ when fitting each single data point in the increment mode, and is $O(m \times n^2)$ when fitting all m measurements. On the other hand, we can get the computation complexity for the Kalman filter in the batch mode to fit the whole m measurements (Table 2).

So only $O(m + n^3)$ is required for the Kalman filter in the batch mode to fit the whole m measurements. Therefore, the two modes have the following significance in fitting different sizes of sensing data:

- The batch fitting capability allows the large amount of point data ($m >> n$) to be processed efficiently once (*its computational complexity is* $O(m + n^3)$) as opposed to the large number of iterative use of incremental updating (*its computational complexity is* $O(m \times n^2)$).
- The incremental update allows any additional single point or a few sensed data to be dynamically and efficiently fused with the reconstructed surface model without reference to the large number of sensed data from scratch (*the computational complexity is* $O(n^2)$ for one single point).

8. Conclusion

In this paper, a new approach for 3D digitization is presented. It enables dynamic B-spline surface reconstruction as point data are acquired. It thus closes the sensing and modeling loop for 3D digitization through dynamically determining sensing locations to improve the quality of the reconstructed surface. The new approach is based on the Kalman filter.

Technically, we exploit the recursive nature of the Kalman filter in the context of B-spline surface reconstruction. The resulting surface is equivalent to that from the weighted least squares method under certain conditions. We present two modes of dynamic surface reconstruction using the Kalman filter: batch mode and incremental mode. We demonstrate their benefits through several examples. These dynamic surface reconstruction modes enable the dynamic parameterization of data points, dynamic determination of next optimal sensing locations, and low-discrepancy based efficient sensing and reconstruction. Experiments demonstrate that high quality surfaces

Table 1
Computational complexity in the incremental mode of the Kalman filter

Computational items	Computation complexity
$\boldsymbol{\Lambda}\mathbf{P}_{i-1}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})$	$n \times 16$
$\mathbf{A}(u_{z_i}, v_{z_i})\boldsymbol{\Lambda}\mathbf{P}_{i-1}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})$	$n \times 16 \times 16$
$\mathbf{K}_i = \boldsymbol{\Lambda}\mathbf{P}_{i-1}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})(\mathbf{A}(u_{z_i}, v_{z_i})\boldsymbol{\Lambda}\mathbf{P}_{i-1}\mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) + \Lambda z)^{-1}$	$(n \times 16 \times 16 + 1) + n$
$\mathbf{K}_i\mathbf{A}(u_{z_i}, v_{z_i})\boldsymbol{\Lambda}\mathbf{P}_{i-1}$	$[(n \times 16 \times 16 + 1) + n] + n \times n$
$\boldsymbol{\Lambda}\mathbf{P}_i = \boldsymbol{\Lambda}\mathbf{P}_{i-1} - \mathbf{K}_i\mathbf{A}(u_{z_i}, v_{z_i})\boldsymbol{\Lambda}\mathbf{P}_{i-1}$	$[(n \times 16 \times 16 + 1) + n] + n \times n + n \times n$
$\mathbf{A}(u_{z_i}, v_{z_i})\mathbf{P}_{i-1}$	16
$\mathbf{K}_i(z - \mathbf{A}(u_{z_i}, v_{z_i})\mathbf{P}_{i-1})$	$17 + n$
$\mathbf{P}_i = \mathbf{P}_{i-1} + \mathbf{K}_i(z - \mathbf{A}(u_{z_i}, v_{z_i})\mathbf{P}_{i-1})$	$17 + n + n$
$\mathbf{P}_i, \boldsymbol{\Lambda}\mathbf{P}_i$	$2n^2 + 259n + 18$

Table 2
Computational complexity in the batch mode of the Kalman filter

Computational items	Computation complexity
$\sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})$	$m \times 16 \times 16$
$(\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})$	$m \times 256 + n \times n$
$\boldsymbol{\Lambda}\mathbf{P}_m = \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1}$	$m \times 256 + n \times n + n^3$
$\sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i$	$m \times 16$
$\mathbf{P}_m = \boldsymbol{\Lambda}\mathbf{P}_m\left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\,\mathbf{P}_0 + \sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i\right)$	$m \times 16 + n + n \times n$
$\mathbf{P}_m, \boldsymbol{\Lambda}\mathbf{P}_m$	$m \times 272 + n + 2n^2 + n^3$

can be obtained much faster by dynamic parameterization of the point cloud with an appropriate dynamic sampling scheme (e.g. through uncertainty minimization or low-discrepancy sequences) than iterative parameterization. Further, it does not need to store the point cloud during the data acquisition process since all the points are directly incorporated into the surface model during the sensing process.

Future work will aim to extend this dynamic surface reconstruction to other types of surface representations, as well as considering non-Gaussian noise in the sensed data. Future work will also study the sensing strategies for dynamic parameterization to optimize the quality of the reconstructed surface.

Acknowledgements

This work was supported in part by the U.S. National Science Foundation Award #0529165 and the Air Force Office of Scientific Research Award #FA9550-07-1-0241. The authors also want to thank Hendrik Speleers for suggesting the quasi-Monte-Carlo method.

Appendix A. Kalman filter for surface reconstruction in the batch mode

The Kalman filter for the measurements in the batch mode is shown in this appendix.

Lemma 1. *Let m be the number of measured points $z_1, z_2 \ldots z_m$ with uncertainty $\Lambda z_1, \Lambda z_2 \ldots, \Lambda z_m$, and $\mathbf{P}_0$ with uncertainty $\boldsymbol{\Lambda}\mathbf{P}_0$ be the set of control points of the initial surface estimate, the final surface through the Kalman filter can be written as*

$$\begin{aligned}\mathbf{P}_m = {} & \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) \right. \\ & \left. \times\ (\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1} \\ & \times \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\,\mathbf{P}_0 + \sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i})\,(\Lambda z_i)^{-1}\,z_i\right)\end{aligned} \tag{22}$$

$$\begin{aligned}\boldsymbol{\Lambda}\mathbf{P}_m = {} & \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m} \mathbf{A}^{\mathrm{T}}(u_{z_i}, v_{z_i}) \right. \\ & \left. \times\ (\Lambda z_i)^{-1}\,\mathbf{A}(u_{z_i}, v_{z_i})\right)^{-1}\end{aligned} \tag{23}$$

where $\mathbf{A}(u_{z_i}, v_{z_i})$ is B-spline shape function matrix corresponding to z_i, which is noted as $\mathbf{A}_i$ in the following.

Proof. Let $\mathbf{K}_1$ be the Kalman gain when fitting the point z_1 into the B-spline surface, and $\mathbf{A}_1$ be the B-spline shape function matrix corresponding to z_1.

The internal state estimate $\mathbf{P}_1$ can be got by

$$\mathbf{P}_1 = \mathbf{P}_0 + \mathbf{K}_1(z_1 - \mathbf{A}_1\mathbf{P}_0) \tag{24}$$

and its uncertainty $\boldsymbol{\Lambda}\mathbf{P}_1 = (\mathbf{I} - \mathbf{K}_1\mathbf{A}_1)\,\boldsymbol{\Lambda}\mathbf{P}_0$.

Eq. (24) can also be written as

$$\mathbf{P}_1 = (\mathbf{I} - \mathbf{K}_1\mathbf{A}_1)\,\mathbf{P}_0 + \mathbf{K}_1 z_1. \tag{25}$$

From Eq. (25), we can further infer $\mathbf{P}_2$ as

$$\begin{aligned}\mathbf{P}_2 &= (\mathbf{I} - \mathbf{K}_2\mathbf{A}_2)\,\mathbf{P}_1 + \mathbf{K}_2 z_2 \\ &= (\mathbf{I} - \mathbf{K}_2\mathbf{A}_2)\,((\mathbf{I} - \mathbf{K}_1\mathbf{A}_1)\,\mathbf{P}_0 + \mathbf{K}_1 z_1) + \mathbf{K}_2 z_2\end{aligned}$$

$$
\begin{aligned}
&= (\mathbf{I}-\mathbf{K}_2\mathbf{A}_2)(\mathbf{I}-\mathbf{K}_1\mathbf{A}_1)\mathbf{P}_0 + (\mathbf{I}-\mathbf{K}_2\mathbf{A}_2)\mathbf{K}_1 z_1 + \mathbf{K}_2 z_2 \\
&= \prod_{i=1}^{2}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\mathbf{P}_0 + \prod_{i=2}^{2}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\mathbf{K}_1 z_1 + \mathbf{K}_2 z_2 \qquad (26)
\end{aligned}
$$

and its uncertainty $\boldsymbol{\Lambda}\mathbf{P}_2 = (\mathbf{I}-\mathbf{K}_2\mathbf{A}_2)\boldsymbol{\Lambda}\mathbf{P}_1 = (\mathbf{I}-\mathbf{K}_2\mathbf{A}_2) \times (\mathbf{I}-\mathbf{K}_1\mathbf{A}_1)\boldsymbol{\Lambda}\mathbf{P}_0 = \prod_{i=1}^{2}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\boldsymbol{\Lambda}\mathbf{P}_0$.

After all the measurements are fitted, the final state estimate $\mathbf{P}_m$ can be denoted by

$$
\begin{aligned}
\mathbf{P}_m &= \prod_{i=1}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\mathbf{P}_0 + \prod_{i=2}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i) \\
&\quad \times \mathbf{K}_1 z_1 + \cdots \mathbf{K}_m z_m \qquad (27)
\end{aligned}
$$

and its uncertainty

$$
\boldsymbol{\Lambda}\mathbf{P}_m = \prod_{i=1}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\boldsymbol{\Lambda}\mathbf{P}_0. \qquad (28)
$$

For any $(1 \le r \le m)$, Eq. (28) can be written as

$$
\begin{aligned}
&\prod_{i=r}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\prod_{i=1}^{r-1}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\boldsymbol{\Lambda}\mathbf{P}_0 \\
&= \prod_{i=r}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\boldsymbol{\Lambda}\mathbf{P}_{r-1} = \boldsymbol{\Lambda}\mathbf{P}_m. \qquad (29)
\end{aligned}
$$

Multiplying $(\boldsymbol{\Lambda}\mathbf{P}_{r-1})^{-1}$ to both sides of Eq. (29), we can get

$$
\prod_{i=r}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i) = \boldsymbol{\Lambda}\mathbf{P}_m(\boldsymbol{\Lambda}\mathbf{P}_{r-1})^{-1}. \qquad (30)
$$

Substituting Eq. (30) with $r = 1, 2, \ldots, m$ into Eq. (27), we can get the final state estimate $\mathbf{P}_m$ as

$$
\begin{aligned}
\mathbf{P}_m &= \prod_{i=1}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\mathbf{P}_0 \\
&\quad + \prod_{i=2}^{m}(\mathbf{I}-\mathbf{K}_i\mathbf{A}_i)\mathbf{K}_1 z_1 + \cdots + \mathbf{K}_m z_m \\
&= \boldsymbol{\Lambda}\mathbf{P}_m(\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\mathbf{P}_0 + \boldsymbol{\Lambda}\mathbf{P}_m(\boldsymbol{\Lambda}\mathbf{P}_1)^{-1}\mathbf{K}_1 z_1 \\
&\quad + \cdots + \boldsymbol{\Lambda}\mathbf{P}_m(\boldsymbol{\Lambda}\mathbf{P}_m)^{-1}\mathbf{K}_m z_m. \qquad (31)
\end{aligned}
$$

From the uncertainty updating equation (Eq. (6)(b)) of the Kalman filter, $\boldsymbol{\Lambda}\mathbf{P}_r(1 \le r \le m)$ in Eq. (31) can be got by

$$
(\boldsymbol{\Lambda}\mathbf{P}_r)^{-1} = (\boldsymbol{\Lambda}\mathbf{P}_{r-1})^{-1} + \mathbf{A}_r^{\mathrm{T}}(\Lambda z_r)^{-1}\mathbf{A}_r. \qquad (32)
$$

Multiplying $\mathbf{K}_r z_r$ to both sides of Eq. (32), we can get

$$
\begin{aligned}
(\boldsymbol{\Lambda}\mathbf{P}_r)^{-1}\mathbf{K}_r z_r &= \left((\boldsymbol{\Lambda}\mathbf{P}_{r-1})^{-1} + \mathbf{A}_r^{\mathrm{T}}(\Lambda z_r)^{-1}\mathbf{A}_r\right)\boldsymbol{\Lambda}\mathbf{P}_{r-1}\mathbf{A}_r^{\mathrm{T}} \\
&\quad \times \left(\Lambda z_r + \mathbf{A}_r\boldsymbol{\Lambda}\mathbf{P}_{r-1}\mathbf{A}_r^{\mathrm{T}}\right)^{-1} z_r \\
&= \left(\mathbf{A}_r^{\mathrm{T}} + \mathbf{A}_r^{\mathrm{T}}(\Lambda z_r)^{-1}\mathbf{A}_r\boldsymbol{\Lambda}\mathbf{P}_{r-1}\mathbf{A}_r^{\mathrm{T}}\right) \\
&\quad \times \left(\Lambda z_r + \mathbf{A}_r\boldsymbol{\Lambda}\mathbf{P}_{r-1}\mathbf{A}_r^{\mathrm{T}}\right)^{-1} z_r \\
&= \mathbf{A}_r^{\mathrm{T}}(\Lambda z_r)^{-1}\left(\Lambda z_r + \mathbf{A}_r\boldsymbol{\Lambda}\mathbf{P}_{r-1}\mathbf{A}_r^{\mathrm{T}}\right) \\
&\quad \times \left(\Lambda z_r + \mathbf{A}_r\boldsymbol{\Lambda}\mathbf{P}_{r-1}\mathbf{A}_r^{\mathrm{T}}\right)^{-1} z_r \\
&= \mathbf{A}_r^{\mathrm{T}}(\Lambda z_r)^{-1} z_r. \qquad (33)
\end{aligned}
$$

Substituting Eq. (33) with $r = 1, 2, \ldots, m$ into Eq. (31), we get the final state estimate $\mathbf{P}_m$ as:

$$
\begin{aligned}
\mathbf{P}_m &= \boldsymbol{\Lambda}\mathbf{P}_m(\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\mathbf{P}_0 + \boldsymbol{\Lambda}\mathbf{P}_m\mathbf{A}_1^{\mathrm{T}}(\Lambda z_1)^{-1} z_1 \\
&\quad + \cdots \boldsymbol{\Lambda}\mathbf{P}_m\mathbf{A}_m^{\mathrm{T}}(\Lambda z_m)^{-1} z_m \\
&= \boldsymbol{\Lambda}\mathbf{P}_m\left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\mathbf{P}_0 + \sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1} z_i\right). \qquad (34)
\end{aligned}
$$

From Eq. (32), we can easily calculate the final state uncertainty $\boldsymbol{\Lambda}\mathbf{P}_m$ by

$$
(\boldsymbol{\Lambda}\mathbf{P}_m)^{-1} = (\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1}\mathbf{A}_i \qquad (35)
$$

or

$$
\boldsymbol{\Lambda}\mathbf{P}_m = \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1}\mathbf{A}_i\right)^{-1}. \qquad (36)
$$

Substituting Eq. (36) into Eq. (34), we can get the final state estimate $\mathbf{P}_m$ as

$$
\begin{aligned}
\mathbf{P}_m &= \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1} + \sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1}\mathbf{A}_i\right)^{-1} \\
&\quad \times \left((\boldsymbol{\Lambda}\mathbf{P}_0)^{-1}\mathbf{P}_0 + \sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1} z_i\right). \qquad (37)
\end{aligned}
$$

So the lemma is proved. □

Note, there is an assumption in this lemma that the measurements are first parameterized based on one base surface.

Appendix B. Weighted least squares for surface reconstruction

The weighted least squares method for surface reconstruction is discussed in this appendix.

The weighted least squares method is often used to reconstruct the surface from the measured points in the statistical optimal sense. The optimal object function $f(\mathbf{P})$ can be defined as

$$
P_m = \arg\min_{\mathbf{P}}\left(f(\mathbf{P}) = \sum_{i=1}^{m}\frac{(z_i - \mathbf{A}_i\mathbf{P})^2}{\Lambda z_i}\right). \qquad (38)
$$

Differentiating $f(\mathbf{P})$ against $\mathbf{P}$, we can get

$$
\begin{aligned}
\frac{\partial f}{\partial \mathbf{P}} &= -2\left(\sum_{i=1}^{m}\frac{\mathbf{A}_i^{\mathrm{T}}(z_i - \mathbf{A}_i\mathbf{P})}{\Lambda z_i}\right) \\
&= -2\left(\sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1} z_i - \sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1}\mathbf{A}_i\mathbf{P}\right). \qquad (39)
\end{aligned}
$$

Let $\frac{\partial f}{\partial \mathbf{P}} = 0$, the optimal value $\mathbf{P}_m$ can be calculated by

$$
\mathbf{P}_m = \left(\sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1}\mathbf{A}_i\right)^{-1}\sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1} z_i. \qquad (40)
$$

Eq. (40) can also be written as

$$\begin{aligned}
\mathbf{P}_m &= \left(\begin{bmatrix}\mathbf{A}_1^{\mathrm{T}} & \cdots & \mathbf{A}_m^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}(\Lambda z_1)^{-1} & & \\ & \ddots & \\ & & (\Lambda z_m)^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{A}_1\\ \vdots \\ \mathbf{A}_m\end{bmatrix}\right)^{-1} \\
&\quad\times \begin{bmatrix}\mathbf{A}_1^{\mathrm{T}} & \cdots & \mathbf{A}_m^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}(\Lambda z_1)^{-1} & & \\ & \ddots & \\ & & (\Lambda z_m)^{-1}\end{bmatrix}\begin{bmatrix}z_1\\ \ddots \\ z_m\end{bmatrix} \\
&= (\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{Z} \qquad (41)
\end{aligned}$$

where

$$\mathbf{A} = \begin{bmatrix}\mathbf{A}_1\\ \vdots \\ \mathbf{A}_m\end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix}(\Lambda z_1)^{-1} & & \\ & \vdots & \\ & & (\Lambda z_m)^{-1}\end{bmatrix}$$

$$\text{and} \quad \mathbf{Z} = \begin{bmatrix}z_1\\ \vdots \\ z_m\end{bmatrix}.$$

From the linear relationship between $\mathbf{P}_m$ and $\mathbf{Z}$ in Eq. (41), we can easily get the uncertainty $\mathbf{\Lambda P}_m$ as

$$\begin{aligned}
\mathbf{\Lambda P}_m &= (\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{\Lambda Z}((\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{U})^{\mathrm{T}} \\
&= (\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{U}\begin{bmatrix}\Lambda z_1 & & \\ & \ddots & \\ & & \Lambda z_m\end{bmatrix}\mathbf{U}^{\mathrm{T}}\mathbf{A}((\mathbf{A}^{\mathrm{T}}\times\mathbf{U}\times\mathbf{A})^{-1})^{\mathrm{T}} \\
&= (\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A}(\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1} \\
&= (\mathbf{A}^{\mathrm{T}}\mathbf{U}\mathbf{A})^{-1} \\
&= \left(\begin{bmatrix}\mathbf{A}_1^{\mathrm{T}} & \cdots & \mathbf{A}_m^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}(\Lambda z_1)^{-1} & & \\ & \ddots & \\ & & (\Lambda z_m)^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{A}_1\\ \vdots \\ \mathbf{A}_m\end{bmatrix}\right)^{-1} \\
&= \left(\sum_{i=1}^{m}\mathbf{A}_i^{\mathrm{T}}(\Lambda z_i)^{-1}\mathbf{A}_i\right)^{-1} \qquad (42)
\end{aligned}$$

References

[1] Allegre R, Chaine R, Akkouche S. Convection-driven dynamic surface reconstruction. In: Proceedings of shape modeling international. 2005. p. 33–42.

[2] Boissonnat JD, Cazals F. Coarse-to-fine surface simplification with geometric guarantees. Computer Graphics Forum 2001;20(3):490–9.

[3] Bratley P, Fox BL, Niederreiter H. Implementation and tests of low-discrepancy sequences. ACM Transactions on Mathematical Software 1992;14:88–100.

[4] Calway A. Recursive estimation of 3D motion and surface structure from local affine flow parameters. IEEE Transactions on Pattern Analysis and Machine Intelligence 2005;27(4):562–74.

[5] Chessa M, Sabatini SP, Solari F, Bisio GM. A recursive approach to the design of adjustable linear models for complex motion analysis. In: Proceedings of signal processing, pattern recognition, and applications. 2007.

[6] Forsey DR, Bartels RH. Surface fitting with hierarchical splines. ACM Transactions on Graphics 1995;22(4):134–61.

[7] Halton JH. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Journal of Numeric Mathematics 1960;2:84–90.

[8] Huang Y, Qian X. A stochastic approach to surface reconstruction. In: Proceedings of IDETC/CIE 2006, ASME 2006 international design engineering technical conference & computers and information in engineering conference. 2006.

[9] Huang Y, Qian X. A dynamic sensing-and-modeling approach to 3D point- and area-sensor integration. ASME Transactions Journal of Manufacturing Science and Engineering 2007;129(June):623–35.

[10] Kalman RE. A new approach to linear Filtering and prediction problems. Transactions of the ASME Journal of Basic Engineering 1960;82(series D):35–45.

[11] Lee S, Wolberg G, Shin SY. Scattered data interpolation with multilevel B-splines. IEEE Transactions on Visualization and Computer Graphics 1997;3(3):228–44.

[12] Li YD, Gu PH. Free-form surface inspection techniques state of the art review. Computer-Aided Design 2004;36(13):1395–417.

[13] Ma W, Kruth JP. Parameterization of randomly measured points for least square fitting of B-spline curves and surfaces. Computer-Aided Design 1995;27(9):663–75.

[14] Ma W, He P. B-spline surface local updating with unorganized points. Computer-Aided Design 1998;30(11):853–62.

[15] Niederreiter H. Quasi-Monte Carlo methods and pseudo-random numbers. Bulletin of American Mathematical Society 1978;84(6):957–1041.

[16] Phillip NA. Parameterization of clouds of unorganized points using dynamic base surfaces. Computer-Aided Design 2004;36(7):607–23.

[17] Piegl L, Tiller W. Parameterization for surface fitting in reverse engineering. Computer-Aided Design 2001;33(8):593–603.

[18] Piegl L, Tiller W. The NURBS book. NY: Springer-Verlag; 1997.

[19] Rappoport A, Hel-Or Y, Werman M. Interactive design of smooth objects with probabilistic point constraints. ACM Transactions on Graphics 1994;13(2):156–76.

[20] Szeliski R. Probabilistic modeling of surfaces. Geometric methods in computer vision, SPIE, vol. 1570. 1991. p. 154–65.

[21] Varady T, Martin R. Reverse engineering handbook of CAGD. North Holland; 2002. p. 651–81.

[22] Varady T, Martin RR, Cox J. Reverse engineering of geometric models — an introduction. Computer-Aided Design 1997;29(4):256–68.

[23] Wang YF. A new method for sensor data fusion in machine vision. Geometric methods in computer vision, SPIE, vol. 1570. 1991. p. 31–42.

[24] Wang YF, Wang JF. On 3D model construction by fusing heterogeneous sensor data. CVGIP: Image Understanding 1994;60(2):210–29.

[25] Weiss V, Andor L, Renner G, Varady T. Advanced surface fitting techniques. Computer Aided Geometric Design 2002;19(1):19–42.

[26] Welch G, Bishop G. An Introduction to the Kalman filter. Technical report, TR-95-041. Department of Computer Science, University of North Carolina at Chapel Hill; 2004.