

Local Parametric Surface Approximation With Automatic Order Selection From Position Data

Michael R. Walker II, *Member, IEEE*

Abstract—Acquiring an anatomical map from position data is important for medical applications where catheters interact with soft tissues. To improve autonomous navigation in these settings, we require information beyond nonparametric maps typically available. We present an algorithm for local surface approximation from position data with automatic surface order selection. The traditional surface fitting objective function is derived from a Bayesian perspective. Posterior probabilities from the occupancy map are incorporated as weights on points selected for surface fitting. Our novel iterative algorithm incorporates surface order selection using the Bayesian information criterion. Simulations demonstrate the ability to automatically select surface order consistent with the latent surface in the presence of noise. Results on human procedure data are also presented.

Index Terms—Catheterization surgery, Medical robotics, Robotics and automation, Surface fitting

I. INTRODUCTION

Anatomic maps used by human navigators exhibit significant interpolation when contrasted against occupancy maps aggregating smoothed position data (see Figure 1). In cardiac catheter ablation surgery, the occupancy map’s utility extends beyond traditional autonomous navigation tasks (e.g. path planning, obstacle avoidance) as catheter-tissue interactions affect catheter response [1]. Here we present a robust algorithm for local parametric surface approximation providing new information from noisy, incomplete data in real time (1Hz updates).

This paper has three main contributions. First we describe a novel algorithm for point selection from an occupancy map for non-planar surfaces. Second, we augment the traditional objective function for surface fitting to include posterior probabilities available from occupancy maps. Third, and most significantly, we incorporate automatic surface order selection in the iterative minimization algorithm, which is critical for distinguishing curvature of the latent surface from noise (errors) in the data.

Here we approximate anatomic surfaces using position data, from the therapeutic catheter, alone. The process is outlined in Figure 2. Position data are typically available with sub-mm precision [2]. These data are subject to strong heartbeat and respiratory motion [3] which we suppress using an unpublished algorithm [4]. Since localization data is available, our problem does not include simultaneous localization and mapping (SLAM) [5], and constructing an

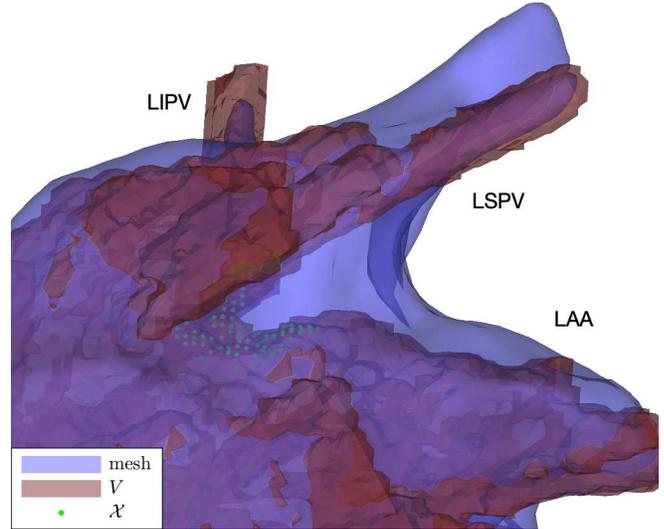


Fig. 1. Anatomic mapping data in left atrium. The surface presented to the physician for navigation is labeled *mesh*. A level set of the occupancy map is labeled V . Additionally, we indicate a local selection of boundary voxels \mathcal{X} . The left inferior pulmonary vein (LIPV), left superior pulmonary vein (LSPV), and left atrial appendage (LAA) are labeled. The voxels indicated by \mathcal{X} are on the ridge between the pulmonary veins and the LAA. For size reference, voxels are 1mm^3 .

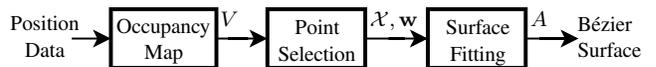


Fig. 2. Process for local surface approximation.

occupancy map [6] is straight forward. Posterior probabilities can be assigned to voxels based on additional sensor data (e.g. tip force or magnetic torque). From the occupancy map, we consider a dense binary matrix, $V \in \{0, 1\}^{n \times m \times p}$, thresholding the known interior volume. From this, we identify an unorganized collection of points, $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^3\}$, which we interpret as a noisy, non-uniform sampling of the latent surface. From these points, we determine a set of control points, $A \in \mathbb{R}^{(n_u+1) \times (n_v+1) \times 3}$, defining a Bézier surface. Parameters $n_u, n_v \in \mathbb{N}_{>0}$ set the surface order and are determined automatically.

There is considerable published work on the fitting of Bézier curves and related generalizations to point clouds. At a high level, we distinguish gradient-free search methods [7], [8] from gradient-based search methods (e.g. Gauss-Newton) [9]–[12] (among others). In most cases, the surface order is assumed fixed. We find the work of Iglesias et al.

This work was supported by Stereotaxis, Inc.

M. R. Walker II is with Stereotaxis, Inc., 4320 Forest Park Ave., St. Louis, Missouri, 63108, USA mwalkerii@wustl.edu

Source code and example data are available: github.com/mrw2ee/BezBic

an interesting exception for their inclusion of the Bayesian information criterion (BIC) for surface order selection [8]. Our approach to surface fitting can be summarized as alternating updates of the location parameters and control points using the distance minimization method (PDM) (see [11] for broader context). Our approach is unique in that we utilize the BIC to conditionally increase the surface order at each update of the control points. Once the surface order is fixed, other methods have demonstrated faster convergence rates [13], [14] and could be used to refine surface approximations more efficiently if necessary.

The rest of this paper is organized as follows. In Section II we present analytic and statistical models guiding algorithm design. In Section III we describe algorithms for point selection, model fitting, and model selection. In Section IV we quantify performance using analytic simulations and demonstrate results on human procedure data. Final remarks are given in Section V.

II. MODEL

This work is focused on fitting a parametric surface to available data. To this end we describe two models: an analytic model for the surface, and a stochastic model for the data. We will subsequently describe surface fitting as maximum-likelihood estimation, and the same stochastic models will be used to determine surface order automatically.

A. Parametric surface

We choose Bézier surfaces for our analytic model based on their broad adoption and efficient use of parameters. Fundamental to the definition of a Bézier surface are the Bernstein polynomials $b : \mathbb{R}^1 \rightarrow \mathbb{R}^1$

$$b(u; i, n) = \binom{n}{i} u^i (1-u)^{n-i}, \quad \forall i \in [0, \dots, n]. \quad (1)$$

We collect the values of all basis functions as the vector-valued function $\mathbf{b} : \mathbb{R} \rightarrow \mathbb{R}^{n+1}$

$$\mathbf{b}(u; n) = [b(u; 0, n) \quad b(u; 1, n) \quad \dots \quad b(u; n, n)]^T. \quad (2)$$

The parameter n determines the length of \mathbf{b} , $n+1$, and we will omit it when clear from context.

A Bézier surface is simply a weighted combination of Bernstein polynomials. For each coordinate, we define $s : \mathbb{R}^2 \rightarrow \mathbb{R}$

$$s(u, v; A_{**k}) = \mathbf{b}(u; n_u)^T A_{**k} \mathbf{b}(v; n_v). \quad (3)$$

Here we use n_u and n_v to represent the polynomial order in directions u and v , respectively. The function is parameterized by a portion of the matrix $A \in \mathbb{R}^{(n_u+1) \times (n_v+1) \times 3}$. We define the Bézier surface, concatenating three copies of (3), as a mapping $\mathbf{s} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$

$$\mathbf{s}(u, v; A) = \begin{bmatrix} s(u, v; A_{**1}) \\ s(u, v; A_{**2}) \\ s(u, v; A_{**3}) \end{bmatrix}. \quad (4)$$

In subsequent expressions it will be convenient to reshape A as a two-dimensional matrix $\bar{A} \in \mathbb{R}^{(n_u+1)(n_v+1) \times 3}$. We

make this distinction as n_u, n_v are clear from A , but ambiguous from \bar{A} alone. We restate (3) and (4)

$$s(u, v; A_{**k}) = \text{vec}(A_{**k})^T (\mathbf{b}(v; n_v) \otimes \mathbf{b}(u; n_u)) \quad (5)$$

$$\mathbf{s}(u, v; A) = \bar{A}^T (\mathbf{b}(v; n_v) \otimes \mathbf{b}(u; n_u)) \quad (6)$$

using \otimes to represent the Kronecker product.

B. Stochastic Data

Let $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^3 : i = 1, \dots, n_x\}$ represent an indexed set of points. We model the data

$$\mathbf{x}_i = \mathbf{s}(u_i, v_i; A) + \mathbf{y}_i, \quad i \in \{1, \dots, n_x\}. \quad (7)$$

Here $\mathbf{y}_i \in \mathbb{R}^3$ represents a stochastic error term. This offset is due to a number of factors including quantization of the volume, biological motion, and model error. Motivated by the central limit theorem, we consider the errors to be Gaussian distributed. For simplicity, we model these terms as independent and distributed as $\mathbf{y}_i \sim \mathcal{N}(0, I_3 \sigma^2 / w_i^2)$. The weights, $w_i \in \mathbb{R}_{>0}$, represent the posterior probabilities of the occupancy map encoding our confidence in each $x_i \in \mathcal{X}$. Reducing w_i , associated with x_i , increases the modeled variance of corresponding y_i .

Let $\theta = \{\mathbf{u}, \mathbf{v}, A, \sigma^2\}$ collectively refer to the model parameters. We define column vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{n_x}$ comprising the coordinates (u_i, v_i) associated with the indexed elements of \mathcal{X} . The log-likelihood of the data, parameterized by θ , reads [15]

$$\begin{aligned} \ell(\mathcal{X}; \theta) = & -n_x \frac{3}{2} \ln(2\pi) \\ & + \sum_{i=1}^{n_x} \left[-\frac{3}{2} \ln\left(\frac{\sigma^2}{w_i^2}\right) - \frac{1}{2} \frac{w_i^2}{\sigma^2} \|\mathbf{x}_i - \mathbf{s}(u_i, v_i; A)\|_{\ell_2}^2 \right]. \end{aligned} \quad (8)$$

The maximum likelihood estimate of θ are the model parameters maximizing (8). This expression will also prove useful in selecting n_u and n_v which determine the size of A .

III. ALGORITHMS

For our problem, \mathcal{X} are not immediately available. They must be determined based on a record of the known volume. In the following, we first present an algorithm for selecting \mathcal{X} , and w_i , from a binary occupancy map. Then, we fit the surface parameters A , \mathbf{u} , and \mathbf{v} for fixed n_u, n_v . Finally, we consider selection of n_u, n_v .

A. Point Selection

Let $V \in \{0, 1\}^{n \times m \times p}$ indicate voxels visited by the catheter obtained by thresholding the occupancy map. Our objective in this section is to identify a subset of these voxels which locally approximate a single surface. We identify interior boundary voxels using convolution and thresholding

$$\tilde{V} = (W * V) \geq \epsilon \quad (9)$$

where W is a discrete convolutional kernel matrix and ϵ is a scalar threshold. There is some flexibility in selecting W . We use the three-dimensional Laplacian kernel

$$[W_\epsilon]_{i,j,k} = \begin{cases} 26 & i = j = k = 2 \\ -1 & \text{otherwise} \end{cases} \quad (10)$$

such that the threshold specifies the minimum number of exterior voxels in a $3 \times 3 \times 3$ neighborhood. In application, we found $\epsilon = 9$ provided reasonable balance of sensitivity and specificity for our problem.

The voxels indicated by \tilde{V} typically compose multiple surfaces. We attribute this to mislabeled voxels in matrix V . For example, some interior voxels may not be visited by the catheter. Next we seek to identify a local subset of voxels, indicated by \tilde{V} , associated with a single surface.

We consider two points (i, j, k) , and (i', j', k') associated with the same surface when they are both indicated by \tilde{V} and are within the same neighborhood. We summarize these requirements

$$\left[\tilde{V}\right]_{i,j,k} = \left[\tilde{V}\right]_{i',j',k'} = [J_l * \delta_{i,j,k}]_{i',j',k'} = 1. \quad (11)$$

In the final equality, we use $\delta_{i,j,k}$ to represent the indicator matrix where the element (i, j, k) is one. We use J_l to indicate the $l \times l \times l$ matrix of all ones. Starting with an initial point (i, j, k) such that $\left[\tilde{V}\right]_{i,j,k} = 1$, we expand the collection of points iteratively convolving with J . At each iteration, the iteration number approximates the distance along the surface from newly added points to the initial point. The point selection process is summarized in Algorithm 1.

Algorithm 1 POINTSELECTION. Identify points locally approximating a single surface about a query point. Input binary matrix V indicates interior voxels, and δ indicates the query point. We assume the size of kernels W, J do not exceed the size of V . The indicated points are associated with nonzero entries in R . The values of R approximate an inverse distance, along the surface, to the query point. We use $*$ to indicate 3D convolution and \circ to indicate element-wise multiplication.

Input: $V, \delta \in \{0, 1\}^{n \times m \times p}$; $W, J \in \mathbb{Z}^{n' \times m' \times p'}$

Output: $R \in \mathbb{N}_{\geq 0}^{n \times m \times p}$

- 1: $\tilde{V} \leftarrow (W * V) \geq 0$
 - 2: $R \leftarrow (J * \delta) \circ \tilde{V}$
 - 3: **for** $t = 1, 2, \dots$ **do**
 - 4: $R \leftarrow R + (J * R) \circ \tilde{V}$
 - 5: **end for**
-

In general, we assume Algorithm 1 is applied locally such that 3D convolutions are performed quickly. When V represents a truncated set of the known volume, voxels along the perimeter of \tilde{V} may be mislabeled. A maximum number of iterations should be enforced such that $J * R$ never indicates perimeter voxels.

In remaining algorithms, we will not reference V or R . Rather, we use R to determine an indexed set of points $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^3 : i = 1, \dots, n_x\}$. Each element, $\mathbf{x}_i \in \mathcal{X}$, is associated with a nonzero entry of R . Additionally, we account for an indexed set of weights $w_i > 0$. These could be used to emphasize points closer to the query point, or to represent stochastic priors from an occupancy map [6].

B. Model Fitting

Only \mathbf{s} in (8) is affected by the parameters A , \mathbf{u} , and \mathbf{v} . The maximum likelihood estimates of these parameters are found minimizing

$$f(A, \mathbf{u}, \mathbf{v}) = \frac{1}{2} \sum_{i=1}^{n_x} w_i^2 \|\mathbf{x}_i - \mathbf{s}(u_i, v_i; A)\|_{\ell_2}^2. \quad (12)$$

This expression is challenging to minimize since (3) involves a high-order product of its arguments. However, the arguments provide a natural decomposition for block coordinated descent [16]. We define

$$\mathcal{P}1 : \quad \mathbf{u}^{(t+1)}, \mathbf{v}^{(t+1)} = \arg \min_{\mathbf{u}, \mathbf{v}} f(A^{(t)}, \mathbf{u}, \mathbf{v}) \quad (13)$$

$$\mathcal{P}2 : \quad A^{(t+1)} = \arg \min_A f(A, \mathbf{u}^{(t+1)}, \mathbf{v}^{(t+1)}) \quad (14)$$

and iteratively update estimates of the control points and location parameters.

By fixing A , (12) becomes separable. In this way, $\mathcal{P}1$ decomposes as n_x two-dimensional optimization problems. For each $x_i \in \mathcal{X}$, we solve

$$\hat{u}_i, \hat{v}_i = \arg \min_{u, v \in \mathbb{R}} g(u, v; \mathbf{x}_i, A) \quad (15)$$

where $g : \mathbb{R}^2 \rightarrow \mathbb{R}^1$

$$g(u, v; \mathbf{x}, A) := \frac{1}{2} \|\mathbf{x} - \mathbf{s}(u, v; A)\|_{\ell_2}^2. \quad (16)$$

The gradient and Hessian of (16), with respect to u and v , are available analytically (see Appendix A for details). They are not constant with respect to u and v , and the Hessian is not guaranteed to be positive definite. However, a local minimum can be identified quickly using Newton's method with Armijo backtracking [16].

In contrast, $\mathcal{P}2$ has a closed-form solution. For convenience we horizontally concatenate \mathbf{x}_n as $X \in \mathbb{R}^{3 \times n_x}$, and define $B \in \mathbb{R}^{(n_u+1)(n_v+1) \times n_x}$

$$B = [\mathbf{b}([\mathbf{v}]_1) \otimes \mathbf{b}([\mathbf{u}]_1) \quad \dots \quad \mathbf{b}([\mathbf{v}]_{n_x}) \otimes \mathbf{b}([\mathbf{u}]_{n_x})]. \quad (17)$$

Using w_i , we compose the diagonal matrix $\Lambda \in \mathbb{R}_{>0}^{n_x \times n_x}$. We then restate (12) as a matrix equation

$$f(A, \mathbf{u}, \mathbf{v}) = \frac{1}{2} \|\Lambda B^T \bar{A} - \Lambda X^T\|_F^2, \quad (18)$$

replacing the vector norm with the matrix norm. In this form, the solution to $\mathcal{P}2$ is obvious.

Often $\mathcal{P}2$ is poorly scaled, and without regularization the optimal A will include control points far from the elements of \mathcal{X} . Regularization has been addressed previously (e.g. Tikhonov [17], and the fairing term [11], [14]). Using Tikhonov regularization the solution remains available in closed form. Without loss of generality, we assume \mathcal{X} are centered about the origin. This assumption may require translation of the initial point cloud $\tilde{\mathcal{X}}$ before fitting and translation of A after fitting summarized as

$$X = \tilde{X} - \mathbf{x}_0 \mathbf{1}^T, \quad \tilde{A} = \bar{A} + \mathbf{1} \mathbf{x}_0^T.$$

Here we use $\mathbf{1}$ to represent a vector of all ones. We define the regularized function

$$f_\lambda(A, \mathbf{u}, \mathbf{v}) = \frac{1}{2} \|\Lambda B^T \bar{A} - \Lambda X^T\|_F^2 + \frac{\lambda}{2} \|\bar{A}\|_F^2. \quad (19)$$

The A minimizing (19) is found solving the linear system of equations

$$(B\Lambda^2 B^T + \lambda I) \bar{A} = B\Lambda^2 X^T. \quad (20)$$

So far, we have assumed n_u and n_v constant. They determine the size of A and B but have no affect on the size of \mathbf{u} or \mathbf{v} . In this way, each iteration of $\mathcal{P}2$ provides an opportunity to change n_u, n_v . Next, we consider selection of these parameters.

C. Model Selection

We can force (8) arbitrarily small by selecting large n_u and n_v . However, increasing these parameters leads to higher order surfaces which are not anatomically accurate. We cast the problem of choosing n_u and n_v as model selection. For this we employ the Bayesian information criteria (BIC).

Let q index the candidate models θ_q . In our case, each θ_q comprises the optimal parameters associated with a unique surface order (n_u, n_v) . We seek the model which most-likely generated the available data. As n_x increases, the BIC asymptotically approximates the joint log likelihood of the data and model [18]

$$\ln p(\mathcal{X}, q) \approx \ell(\mathcal{X}; \theta_q) - \frac{d_q}{2} \ln n_x. \quad (21)$$

In this approximation θ_q represents the maximum likelihood estimates for model q . The scalar d_q indicates the total number of model parameters associated with the model q

$$d_q = 2n_x + 3(n_u + 1)(n_v + 1) + 1. \quad (22)$$

Expression (21) provides a regularized objective function for model order selection. Gains in $\ell(\mathcal{X}; \theta_q)$ are offset by additional model parameters.

In (21) we make use of (8) which requires σ^2 . Given maximum-likelihood estimates $A, \mathbf{u}, \mathbf{v}$ (as detailed in Section III-B), the maximum-likelihood estimate of σ^2 is then

$$\begin{aligned} \hat{\sigma}^2 &= \frac{1}{3n_x} \sum_{i=1}^{n_x} w_i^2 \|\mathbf{x}_i - \mathbf{s}_i\|_{\ell_2}^2 \\ &= \frac{2}{3n_x} f(A, \mathbf{u}, \mathbf{v}). \end{aligned} \quad (23)$$

Plugging (23) into (8), we retain only terms of σ_q and d_q and define the statistic

$$t_q = -3n_x \ln \sigma_q^2 - d_q \ln n_x. \quad (24)$$

Here q indexes candidate models θ_q , which includes σ_q^2 , and determines d_q according to (22). The model selection maximizing (21) is equivalent to selecting q maximizing the statistic t_q . Algorithm 2 describes the process of estimating A for multiple pairs (n_u, n_v) and selecting the result yielding the largest t .

The surface fitting process is given in Algorithm (3). We assume \mathbf{u}, \mathbf{v} have been initialized, for example, projecting

Algorithm 2 MDLSELECT. Estimate A and statistic t while increasing model order. Return the model with the largest statistic. Here COMPUTEA refers to the solution for (20), and COMPUTESIGMA2 refers to (23). While the parameters \mathbf{u}, \mathbf{v} are returned in θ , they are not changed.

Input: $X \in \mathbb{R}^{3 \times n_x}; \mathbf{w} \in \mathbb{R}_{>0}^{n_x}; \mathbf{u}, \mathbf{v} \in \mathbb{R}^{n_x}; n_u, n_v \in \mathbb{N}_{>0}; \lambda \in \mathbb{R}_{\geq 0}$

Output: θ

- 1: $q \leftarrow 0$
- 2: **for** $n'_v = n_u$ to $n_u + 1$ **do**
- 3: **for** $n'_u = n_v$ to $n_v + 1$ **do**
- 4: $q \leftarrow q + 1$
- 5: $A \leftarrow \text{COMPUTE}A(X, \mathbf{w}, \mathbf{u}, \mathbf{v}, n'_u, n'_v, \lambda) \triangleright (20)$
- 6: $\sigma^2 \leftarrow \text{COMPUTE}SIGMA2(X, \mathbf{w}, A, \mathbf{u}, \mathbf{v}) \triangleright (23)$
- 7: $t_q \leftarrow \text{COMPUTET}(\theta) \triangleright (24)$
- 8: **end for**
- 9: **end for**
- 10: $i \leftarrow \arg \max_{i \in \{1, \dots, q\}} t_i \quad \triangleright \text{Maximize BIC}$
- 11: $\theta \leftarrow \theta_i$

X onto a 2D subspace using SVD. With each update of A , in MDLSELECT, we potentially increase surface order according to the BIC. Changes to \mathbf{u}, \mathbf{v} , and σ^2 are all useful for stopping criteria (not addressed here).

Algorithm 3 FITSURFACE. Fit a surface to a collection of points iteratively updating \mathbf{u}, \mathbf{v} , and A . UPDATEPOINT refers to (15), which requires an iterative solver.

Input: $X \in \mathbb{R}^{3 \times n_x}; \mathbf{u}, \mathbf{v} \in \mathbb{R}^{n_x}; \mathbf{w} \in \mathbb{R}_{>0}^{n_x}; \lambda > 0$

Output: θ

- 1: $n_u, n_v \leftarrow 1$
- 2: $\theta \leftarrow \text{MDLSELECT}(X, \mathbf{w}, \mathbf{u}, \mathbf{v}, n_u, n_v, \lambda) \triangleright \text{Algorithm 2}$
- 3: **for** $t = 1, 2, \dots$ **do**
- 4: $A, n_u, n_v \leftarrow A(\theta), n_u(\theta), n_v(\theta) \quad \triangleright \text{Expand last } \theta$
- 5: **for** $i = 1$ to n_x **do** $\triangleright \text{Update } \mathbf{u}, \mathbf{v}$
- 6: $u_i, v_i \leftarrow \text{UPDATEPOINT}(\mathbf{x}_i, A, u_i, v_i) \quad \triangleright (15)$
- 7: **end for**
- 8: $\theta \leftarrow \text{MDLSELECT}(X, \mathbf{w}, \mathbf{u}, \mathbf{v}, n_u, n_v, \lambda) \triangleright \text{Update } A$
- 9: **end for**

The statistical interpretation of (21), as an approximation for the joint log-likelihood, is somewhat disingenuous for our problem. The asymptotic approximation of the BIC is not accurate for our problem sizes ($n_x \sim 100$). Additionally, the criterion requires maximum likelihood estimates of the parameters which are not resolved during the iterative algorithm. The implication here is that q , maximizing t_q , may fluctuate while the algorithm converges. Additionally, switching q online (as in Algorithm (3)) may yield different results in contrast with the brute-force approach of solving for each model order independently and then applying model selection. However, the brute-force approach is computationally expensive. Here we use the right hand side of (21) as a regularized objective function guiding allocation of

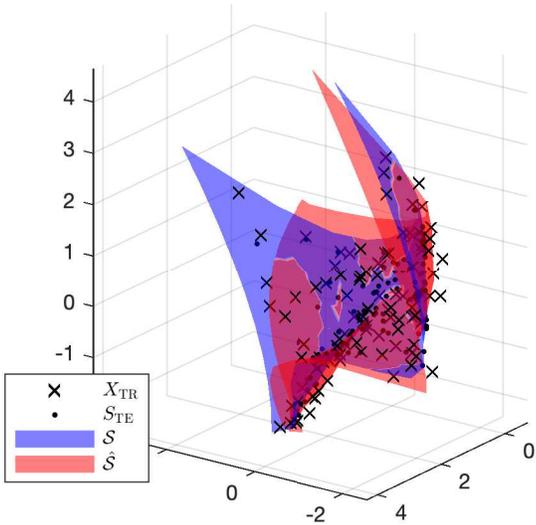


Fig. 3. Visualization of surface approximation. The training and testing data are indicated by X_{TR} and S_{TE} , respectively. The reference surface and approximation are indicated by S and \hat{S} , respectively. Here $n_{\text{TR}} = 100$, and $\sigma_y^2 = 0.02$.

computational resources.

IV. SIMULATIONS AND RESULTS

To demonstrate the benefits of our approach, we quantify performance with simulations and present qualitative results on human procedure data. Simulations are primarily designed to demonstrate automatic surface order selection: avoiding both overfitting and underfitting the latent surface.

We simulate a notional reference surface as an infinite collection of points $\mathcal{S} \subset \mathbb{R}^3$. We then define two unique subsets, $\mathcal{S}_{\text{TR}}, \mathcal{S}_{\text{TE}} \subset \mathcal{S}$, $\mathcal{S}_{\text{TR}} \cap \mathcal{S}_{\text{TE}} = \emptyset$, used for training and testing purposes, respectively. For convenience, we will refer to the cardinality of these sets using $n_{\text{TR}} = |\mathcal{S}_{\text{TR}}|$, $n_{\text{TE}} = |\mathcal{S}_{\text{TE}}|$. Concatenating the elements of \mathcal{S}_{TR} horizontally, we define the matrix $S_{\text{TR}} \in \mathbb{R}^{3 \times n_{\text{TR}}}$. We do not assume S_{TR} is available directly. Instead, we assume the observed data are subject to additive noise

$$X_{\text{TR}} = S_{\text{TR}} + Y \quad (25)$$

consistent with (7). From the available X_{TR} we fit a surface, estimating $\hat{\theta}$. These parameters, or more specifically A , define a surface $\hat{S} \subset \mathbb{R}^3$ as an infinite collection of points indexed by coordinates $(u, v) \in \mathbb{R}$. These quantities are shown for an example surface in Figure 3.

To quantify surface fit, we consider the distance between elements $\mathbf{x}_i \in \mathcal{S}_{\text{TE}}$ and their nearest neighbor $s(\hat{u}_i, \hat{v}_i; A) \in \hat{S}$. Finding neighbor coordinates (\hat{u}_i, \hat{v}_i) requires solving (15). We then define the matrix $\hat{S}_{\text{TE}} \in \mathbb{R}^{3 \times n_{\text{TE}}}$ concatenating $s(\hat{u}_i, \hat{v}_i; A)$ horizontally. We quantify performance with two metrics

$$\hat{\sigma}_{\text{TR}}^2 = \frac{1}{3n_{\text{TR}}} \left\| \hat{S}_{\text{TR}} - X_{\text{TR}} \right\|_F^2 \quad (26)$$

$$\hat{\sigma}_{\text{TE}}^2 = \frac{1}{3n_{\text{TE}}} \left\| \hat{S}_{\text{TE}} - S_{\text{TE}} \right\|_F^2. \quad (27)$$

Here \hat{S}_{TR} are determined by $\hat{\theta}$. When $w_i = 1$, (26) is equivalent to (23).

Two attributes of the training data have a significant impact on algorithm performance: n_{TR} and the variance of Y , σ_y^2 . We demonstrate these effects using a scaled-version of the Rosenbrock function (see Figure 3). We sample the reference function and apply a consistent, randomly generated rotation to all sampled points. We then partition the rotated sample points as $S_{\text{TR}}, S_{\text{TE}}$. Randomly generating i.i.d. elements $[Y]_i \sim \mathcal{N}(0, \sigma_y^2)$, we generate X_{TR} using (25). From X_{TR} we estimate $\hat{\theta}$ using Algorithm 3, which also determines \hat{S}_{TR} and $\hat{\sigma}_{\text{TR}}^2$. From \hat{A} and S_{TE} we estimate \hat{S}_{TE} by solving (15) for each $\mathbf{x}_i \in S_{\text{TE}}$. Using \hat{S}_{TE} , we determine $\hat{\sigma}_{\text{TE}}^2$ according to (27). This process was repeated 100 times for multiple pairs $(n_{\text{TR}}, \sigma_y^2)$. The averages are shown in Table I.

TABLE I

EFFECTS OF PROBLEM SIZE AND NOISE ON RESULTS

n_{TR}	σ_y^2	iter.	size	$\hat{\sigma}_{\text{TR}}^2$	$\hat{\sigma}_{\text{TE}}^2$	ms
50	2.5E-01	9.91	5.94	6.8E-02	4.9E-02	391.3
50	1.0E-02	10	14.21	2.5E-03	4.6E-03	404.0
50	2.5E-03	9.31	17.93	8.1E-04	2.4E-03	353.5
50	1.0E-04	5.84	30.28	2.5E-04	1.6E-03	221.4
100	2.5E-01	10	6.91	6.8E-02	3.3E-02	449.2
100	1.0E-02	9.99	14.89	3.1E-03	1.7E-03	448.1
100	2.5E-03	9.61	18.86	9.6E-04	8.2E-04	434.8
100	1.0E-04	6.5	32.85	1.9E-04	3.4E-04	282.8
1000	2.5E-01	10	12.79	7.0E-02	1.1E-02	1343.2
1000	1.0E-02	9.92	23.09	3.3E-03	3.0E-04	1331.9
1000	2.5E-03	7.63	34.79	8.3E-04	9.9E-05	1044.3
1000	1.0E-04	6.86	60.27	3.7E-05	2.7E-05	937.7

In Table I, the *iter.* column represents the average number of AM iterations (t in Algorithm 3). We limited 10 as the maximum number of iterations which we found sufficient for establishing the appropriate surface order. The *size* column in Table I indicates the number of control points in A : $(n_u + 1)(n_v + 1)$. We emphasize this is not d to avoid dependence on n_{TR} . There are two trends to observe. First, size increases with n_{TR} . Second, size increases as σ_y^2 decreases. Both of these trends are due to the BIC. Similarly, we find that larger model sizes are associated with lower $\hat{\sigma}_{\text{TR}}^2$ and $\hat{\sigma}_{\text{TE}}^2$ since the BIC hedges against over-fitting. The final column depicts the average run time for surface fitting (excluding point selection). Computations were performed in MATLAB on a Late 2016 MacBook Pro (2.9 GHz Quad-Core i7). The separable problem was parallelized among 4 workers with no GPU support.

To further demonstrate the benefits of automatic order selection, we contrast fitting errors of fixed-order models. In addition to the Rosenbrock test surface, we now consider a plane as a second latent test surface. Intuitively we expect low-order models to perform poorly on the Rosenbrock test surface (under fitting), while high order models perform poorly on the planar surface (over fitting). This is confirmed in Figure 4. Additionally we find our approach performs well in both cases, automatically selecting a reasonable order for the latent surface from noisy measurements.

To demonstrate the complete algorithm performance, we

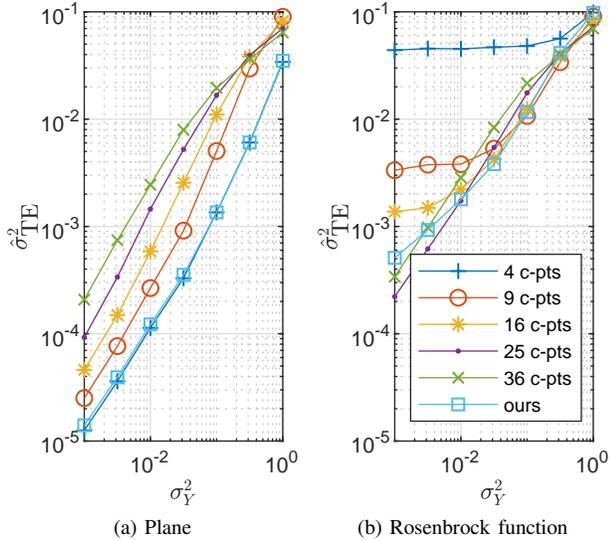


Fig. 4. Contrasting testing error as a function of noise. For Figure 4a and Figure 4b the latent surfaces represent a plane and Rosenbrock function, respectively. The legend indicates the number of control points assumed in fixed-order models.

revisit the human procedure data depicted in Figure 1. The occupancy map was constructed from position data after applying an unpublished algorithm [4] to reduce biological motion [3]. From the occupancy map we extract a dense binary matrix V indicating interior volume with cubic voxels with 1mm edge lengths. We identify \mathcal{X} applying Algorithm 1 to a cubic volume with edge lengths 15mm. This collection was indicated as the point cloud in Figure 1. Using Algorithm 3 we fit a surface \hat{S} to \mathcal{X} . Results are shown in Figure 5. In this case $n_x = 132$, and we used $w_i = 1$. For the fitted surface, $n_u = 1$, $n_v = 3$, and $\hat{\sigma}^2 \approx 0.05$. This case presents a region of the anatomy with high curvature, yet the resulting error is well below quantization of the data. To contrast our results with the surface presented to the physician navigators, we limit *display* to only mesh vertices representing a nearest point to one of the elements of \mathcal{X} . The median distance from \mathcal{X} to the nonparametric display surface (nearest vertex) is 3.38 mm in contrast to 0.24mm (orthogonal distance) to our parametric surface.

V. CONCLUSIONS

We have demonstrated an iterative algorithm for surface approximation that converges quickly and is robust against over fitting. We feel this approach is well suited for the unique challenges associated with autonomous navigation of catheters for cardiac ablation surgery.

Our algorithm for surface fitting only address the local search problem. In other words we assume the initial parameter selections, $\mathbf{u}^{(0)}, \mathbf{v}^{(0)}$, are near the global minimum. We select $\mathbf{u}^{(0)}, \mathbf{v}^{(0)}$ by projection \mathcal{X} onto a 2-dimensional subspace (SVD). The usefulness of this strategy will depend on the curvature of the latent surface. For example, this is not reasonable when the point cloud approximates a cylinder. Limiting the extent of the surface approximation is

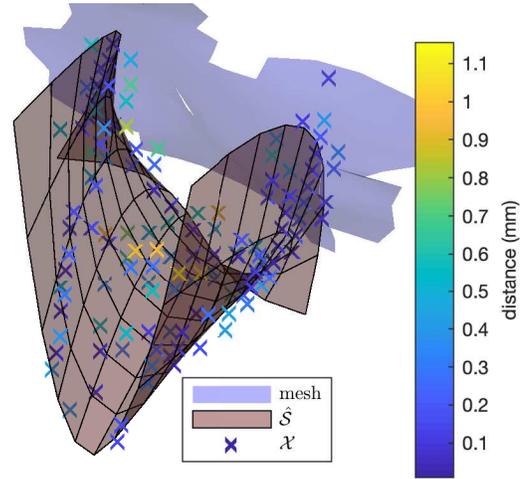


Fig. 5. Local surface approximated from human procedure data. The nonparametric surface presented to human navigators is labeled *mesh*. Our parametric surface approximation and the intermediate point cloud are labeled \hat{S} and \mathcal{X} , respectively. The marker colors for \mathcal{X} indicates the ℓ_2 distance (normal) to the approximated surface in mm.

one mitigation strategy. Larger surfaces may require global search or stitching together multiple surfaces approximations and presents an opportunity for further research.

APPENDIX

A. Gradient and Hessian of the Separable Problem

The Jacobian of (6) can be expressed

$$\mathbf{J}_s = [\mathbf{b}(v) \otimes \mathbf{b}'(u) \quad \mathbf{b}'(v) \otimes \mathbf{b}(u)]^T \bar{A}. \quad (28)$$

Here we use $\mathbf{b}'(u)$ to represent the derivative of (2) with respect to u . Subsequently we will use $\mathbf{b}''(u)$ to represent the second derivative. Their analytic derivation is straight forward from (1). The gradient of (16) is then

$$\nabla g(u, v; \mathbf{x}, A) = -\mathbf{J}_s (\mathbf{x} - \mathbf{s}(u, v; A)) \quad (29)$$

For the convenience, we define the auxiliary scalar values

$$c_u = (\mathbf{b}(v) \otimes \mathbf{b}''(u))^T \bar{A} (\mathbf{x} - \mathbf{s}(u, v; A)) \quad (30)$$

$$c_v = (\mathbf{b}''(v) \otimes \mathbf{b}(u))^T \bar{A} (\mathbf{x} - \mathbf{s}(u, v; A)) \quad (31)$$

$$c_{uv} = (\mathbf{b}'(v) \otimes \mathbf{b}'(u))^T \bar{A} (\mathbf{x} - \mathbf{s}(u, v; A)). \quad (32)$$

The Hessian of (16) is then

$$\nabla^2 g(u, v; \mathbf{x}, A) = \mathbf{J}_s \mathbf{J}_s^T - \begin{bmatrix} c_u & c_{uv} \\ c_{uv} & c_{vv} \end{bmatrix}. \quad (33)$$

ACKNOWLEDGMENT

We are grateful to Dr. Burkhard Hügl for providing procedure data facilitating this research. Nathan Kastelein, Paul F. Rebillot III and Ilker Tunay contributed ideas and helpful comments on the manuscript.

REFERENCES

- [1] I. Tunay, "Spatial continuum models of rods undergoing large deformation and inflation," *IEEE Transactions on Robotics*, vol. 29, pp. 297–307, April 2013.
- [2] F. Bourier, R. Fahrig, P. Wang, P. Santangeli, K. Kurzydum, N. Strobel, T. Moore, C. Hinkel, and A. Al-Ahmad, "Accuracy assessment of catheter guidance technology in electrophysiology procedures," *Journal of Cardiovascular Electrophysiology*, vol. 25, no. 1, pp. 74–83, 2014.
- [3] J. McClelland, D. Hawkes, T. Schaeffter, and A. King, "Respiratory motion models: A review," *Medical Image Analysis*, vol. 17, no. 1, pp. 19 – 42, 2013.
- [4] M. R. Walker II, "Virtual diastole," tech. rep., Stereotaxis, October 2016. Internal technical note.
- [5] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, pp. 1309–1332, Dec 2016.
- [6] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: an efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, pp. 189–206, Apr 2013.
- [7] A. Gálvez and A. Iglesias, "Firefly algorithm for polynomial bézier surface parameterization," *Journal of Applied Mathematics*, vol. 2013, 2013.
- [8] A. Iglesias, A. Gálvez, and C. Loucera, "Two simulated annealing optimization schemas for rational bézier curve fitting in the presence of noise," *Mathematical Problems in Engineering*, vol. 2016, 2016.
- [9] T. A. Pastva, "Bezier curve fitting," Master's thesis, Naval Postgraduate School, 1998.
- [10] H. Pottmann, S. Leopoldseder, and M. Hofer, "Approximation with active b-spline curves and surfaces," in *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.*, pp. 8–25, Oct 2002.
- [11] W. Wang, H. Pottmann, and Y. Liu, "Fitting b-spline curves to point clouds by curvature-based squared distance minimization," *ACM Trans. Graph.*, vol. 25, pp. 214–238, Apr. 2006.
- [12] Y. Liu and W. Wang, "A revisit to least squares orthogonal distance fitting of parametric curves and surfaces," in *Advances in Geometric Modeling and Processing* (F. Chen and B. Jüttler, eds.), (Berlin, Heidelberg), pp. 384–397, Springer Berlin Heidelberg, 2008.
- [13] W. Zheng, P. Bo, Y. Liu, and W. Wang, "Fast b-spline curve fitting by L-BFGS," *CoRR*, vol. abs/1201.0070, 2012.
- [14] P. Bo, R. Ling, and W. Wang, "A revisit to fitting parametric surfaces to point clouds," *Computers & Graphics*, vol. 36, no. 5, pp. 534 – 540, 2012. Shape Modeling International (SMI) Conference 2012.
- [15] T. Moon and W. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall, 2000.
- [16] D. Bertsekas, *Nonlinear Programming*. Athena scientific optimization and computation series, Athena Scientific, 2016.
- [17] Ling Jing and Li Sun, "Fitting b-spline curves by least squares support vector machines," in *2005 International Conference on Neural Networks and Brain*, vol. 2, pp. 905–909, Oct 2005.
- [18] A. D. Lanterman, "Schwarz, wallace, and rissanen: Intertwining themes in theories of model selection," *International Statistical Review / Revue Internationale de Statistique*, vol. 69, no. 2, pp. 185–212, 2001.