Recursive Fusion of Noisy Depth and Position Measurements for Surface Reconstruction

Gerhard Kurz and **Uwe D. Hanebeck**
Intelligent Sensor-Actuator-Systems Laboratory (ISAS)
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
gerhard.kurz@kit.edu, uwe.hanebeck@ieee.org

***Abstract*—We propose an algorithm to combine both depth and position measurements when estimating a continuous surface. Position measurements originate from a fixed point on the surface, whereas depth measurements are determined by the intersection of the surface with a line originating from the depth sensor. Through fusion of both types of measurements, it is possible to benefit from the advantages of different sensors. The surface is obtained through interpolation of control points with splines, which allows a compact representation of the surface. In order to simplify the problem of intersecting the surface with lines originating from the depth sensor, we propose the use of polar or spherical coordinates in surface parameterization. The presented algorithm can be applied in both 2D and 3D settings and is independent of the particular choice of sensors. Our method can recursively include new information as it is obtained by using nonlinear filtering and it considers uncertainties associated with the measurements.**

***Index Terms*—depth camera, stereo camera, surface estimation, spline, point cloud, tracking**

I. Introduction

Many applications require the reconstruction of surfaces based on noisy measurements. For example, various medical applications require the reconstruction of the surface of organs [1], [2], [3]. It is also a relevant topic in other areas such as robotics [4] and computer graphics [1], [5].

For reconstructing a surface, we consider two different types of measurements. First, there are position measurements originating from a certain point located at a fixed position on the surface. Position measurements are typically obtained from landmarks on the surface, for example structured regions that allow 3D reconstruction with a stereo camera system. Second, there are depth measurements that do not originate from a fixed point on the surface and only depend on the distance of a certain point in space to the surface along a given line. Depth measurements can be obtained from depth sensors such as time-of-flight (TOF) cameras or sensors based on structured light such as the Microsoft Kinect.

While position measurements are typically sparse but highly accurate, depth measurements tend to be more plentiful, but less accurate and more susceptible to noise. On the one hand, stereo camera systems may have a high resolution, but perform poorly in non-structured areas. On the other hand, TOF sensors can handle uniform surface areas, but have comparatively limited resolution and accuracy. Thus, it is desirable to combine both types of measurements in order to achieve a more accurate and robust reconstruction of the surface compared to one type of measurements alone. Fusion of information from different types of sensors allows to alleviate the disadvantages of any given sensor type.

Many practical applications are not limited to a static scenario, because the sensors and the surface move relative to each other. Furthermore, the surface may deform and change shape over time. Consequently, our goal is to track surface position and shape over time and to include new information recursively as it is obtained. A priori knowledge may be included to predict the evolution of the surface into the future.

We now outline our main contribution. In this paper, we introduce a novel method for surface reconstruction that is suitable for both 2D and 3D applications. The proposed method combines depth and position measurements to recursively estimate the state of the surface while considering uncertainties. It is independent of the choice of sensors and can be employed in a wide area of applications. Our method is based on a spline representation of the surface whose parameters are recursively estimated using nonlinear filtering techniques. There are separate measurement equations for depth and position measurements in order to deal with their individual characteristics.

The paper is structured as follows. In Sec. II, we give an overview of previous work in the area of surface reconstruction. The required prerequisites are introduced in Sec. III. Our method is derived for the 2D case in Sec. IV and adapted to the 3D case in Sec. V. We propose some further enhancements in Sec. VI and evaluate the proposed algorithms in simulations in Sec. VII. Finally, we form a conclusion in Sec. VIII.

II. Related Work

Traditional methods for surface reconstruction rely on position measurements exclusively. For example, Hoppe et al. presented a method to reconstruct a surface based on unorganized points [5]. As a result of the increasingly widespread use of depth cameras, algorithms based on depth measurements exclusively have been introduced, for example Kinect Fusion [6].

However, approaches that try to combine both types of measurements are still fairly new. In 2008, Guan et al. proposed the use of a graph-cut algorithm to obtain iso-probability surfaces in a probabilistic space occupancy grid [7]. In 2011, Zhu et al. presented an approach based on Markov Random Fields to combine measurements from a stereo camera system and a time-of-flight (TOF) camera while considering the respective uncertainties of both sensor types [8].

There are different ways for representing the reconstructed surface. Depth sensors usually provide depth maps or point clouds as raw data, but a more sophisticated representation is desired. One common approach is to use spatial discretization and represent the surface as voxels [6], [7]. However, voxel-based approaches typically require a lot of memory and computational power if a high resolution is to be achieved. Another common approach is to use triangular meshes [1]. While triangular meshes are usually more computationally efficient than voxels, they require a large number of triangles to provide a satisfactory reconstruction of rounded shapes. An alternative is to describe the surface as a spline, which can be stored in a very compact form as it is uniquely defined by a small number of control points [9], [10]. Splines are very suitable for smooth surfaces without rough edges. Unlike voxels or triangular meshes, splines can be evaluated at an arbitrary resolution and still appear smooth. Stochastic formulations of splines have previously been used by [11]. Gaussian processes can also be used to model uncertain surfaces [12].

III. Prerequisites

Before describing our method for surface reconstruction, we introduce some prerequisites.

A. Interpolation

Let $\underline{p}_1, \ldots, \underline{p}_m \in \mathbb{R}^n$ and $f_1, \ldots, f_m \in \mathbb{R}$. The goal of interpolation is to find a function $f : \mathbb{R}^n \to \mathbb{R}$ with $f(\underline{p}_i) = f_i$ for $1 \leq i \leq m$ where f is smooth in some sense. There are various different types of interpolation functions. An overview can be found in [10]. Our proposed method does not depend on a particular choice of the interpolation method. For later use, we define a function

$$\begin{aligned} &\text{interpolate} : ((\mathbb{R}^n)^m \times \mathbb{R}^m) \to (\mathbb{R}^n \to \mathbb{R}), \\ &(\underline{p}_1, \ldots, \underline{p}_m; f_1, \ldots, f_m) \mapsto f \ , \end{aligned}$$

which maps points $\underline{p}_1, \ldots, \underline{p}_m$ and values $f_1, \ldots, f_m$ to their interpolating function $f \in (\mathbb{R}^n \to \mathbb{R})$.

For the purpose of our experiments, we decided to use Radial Basis Functions (RBF) [2], [13] for interpolation, because they are easy to calculate and are applicable for any dimension n. The interpolating function f is given by

$$f(\underline{p}) = \sum_{j=1}^{m} c_j \cdot \phi(||\underline{p} - \underline{p}_j||) \ ,$$

where $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is the basis function and $c_1, \ldots, c_m \in \mathbb{R}$ are weighting coefficients. A popular choice of basis function

Input: points $p_1, \ldots, p_m \in \mathbb{R}^n$; values $f_1, \ldots, f_m \in \mathbb{R}$

Output: interpolation function $f : \mathbb{R}^n \to \mathbb{R}$

$$\mathbf{A} \leftarrow \begin{pmatrix} \phi(||p_1 - p_1||) & \ldots & \phi(||p_m - p_1||) \\ \vdots & \ddots & \vdots \\ \phi(||p_1 - p_m||) & \ldots & \phi(||p_m - p_m||) \end{pmatrix} ;$$

$$\begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix} \leftarrow \mathbf{A}^{-1} \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix} ;$$

$$f \leftarrow (p \mapsto \textstyle\sum_{j=1}^{m} c_j \cdot \phi(||p - p_j||)) \ ;$$

Fig. 1: Algorithm for interpolation based on RBFs.

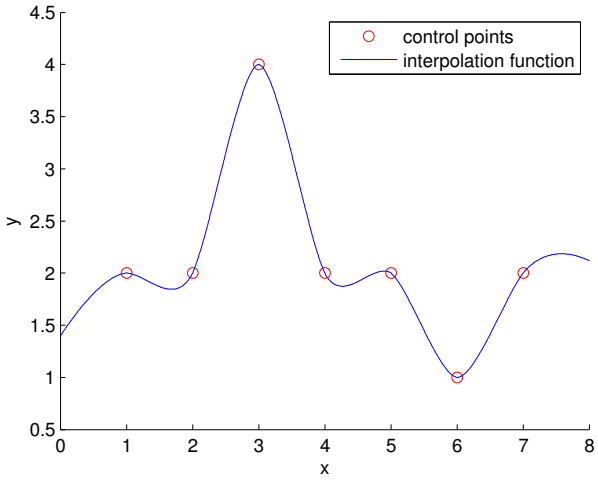


Fig. 2: Example of the interpolation achieved by the presented algorithm for $\mathbb{R} \to \mathbb{R}$ interpolation, i.e., $n = 1$. The control points are $p_1, \ldots, p_7 = 1, 2, 3, 4, 5, 6, 7$ with values $f_1, \ldots, f_7 = 2, 2, 4, 2, 2, 1, 2$ and the basis function $\phi(x)$ is a thin plate spline.

is the thin plate spline (TPS):

$$\phi(x) = \begin{cases} x^2 \log x \ , & x > 0 \\ 0 \ , & x = 0 \end{cases} .$$

The weighting coefficients $c_1, \ldots, c_m$ can be obtained by solving the system of m linear equations

$$f_i = \sum_{j=1}^{m} c_j \cdot \phi(||\underline{p}_i - \underline{p}_j||) \ , \quad 1 \leq i \leq m \ .$$

The algorithm is given in Fig. 1. We show an example of the interpolation produced by this algorithm in Fig. 2.

B. Polar and Spherical Coordinates

While many common approaches rely on Cartesian coordinates exclusively, we use polar coordinates (in 2D) and spherical coordinates (in 3D) to simplify certain computations

similar to [14]. The transformation between Cartesian and polar coordinates is given by

$$x = r\cos(\phi) \ ,$$
$$y = r\sin(\phi) \ ,$$

and

$$r = ||(x,y)^T|| = \sqrt{x^2+y^2} \ ,$$
$$\phi = \text{atan2}(y,x) \ .$$

For spherical coordinates, there are several common definitions. We use the convention

$$x = r\cos(\theta)\cos(\phi) \ ,$$
$$y = r\cos(\theta)\sin(\phi) \ ,$$
$$z = r\sin(\theta) \ ,$$

and

$$r = ||(x,y,z)^T|| = \sqrt{x^2+y^2+z^2} \ ,$$
$$\phi = \text{atan2}(y,x) \ ,$$
$$\theta = \arcsin(z/r) \ .$$

C. State and System Representation

For describing the estimate of the reconstructed surface at time step k, we use a state vector $\underline{x}_k^e \in \mathbb{R}^q$. The uncertainty of the estimate at time step k is given by the covariance matrix $\mathbf{C}_k^e$. A system model

$$\underline{x}_{k+1}^p = a_k(\underline{x}_k^e) + \underline{\xi}_k$$

with system function $a_k : \mathbb{R}^q \to \mathbb{R}^q$ and additive Gaussian noise $\underline{\xi}_k \sim \mathcal{N}(\underline{0}, \mathbf{C}_k^\xi)$ can be used to describe the evolution of the state $\underline{x}_k$. If the system model is linear, a Kalman filter [15] can be used to perform the prediction. Otherwise a nonlinear filter such as the unscented Kalman filter (UKF, [16]) may be applied. If the system does not follow any known dynamics, a random walk model may be used. In static cases, where the surface does not change over time, prediction may be omitted. As our approach for estimating the reconstructed surface is independent of the particular details of the system model, we will focus on the measurement model from now on.

IV. Surface Reconstruction in 2D

Let us first consider the 2D case. Although the 2D case might not seem relevant at first, there are actually a number of applications for 2D surface reconstruction. For example, LIDAR (light detection and ranging) sensors are commonly used in robotics and allow the reconstruction of obstacles as surfaces in 2D [17].

A. Position Measurements

We consider a set of l landmarks on the surface. For tracking these landmarks, we define the state vector at time step k as

$$\underline{x}_k = \left(x_k^{1,1}, x_k^{1,2} \ldots, x_k^{l,1}, x_k^{l,2}\right)^T \in \mathbb{R}^{2l} \ ,$$

where $x_k^{a,b}$ represents the position of landmark $a \in \{1,\ldots,l\}$ in dimension $b \in \{1,2\}$ at time step k. In this case, the measurement model is trivially given by

$$\hat{\underline{y}}_k = \mathbf{I}_{2l\times 2l} \cdot \underline{x}_k + \underline{v}_k \ ,$$

where $\mathbf{I}_{2l\times 2l} \in \mathbb{R}^{2l\times 2l}$ is the identity matrix, $\hat{\underline{y}}_k$ is the measurement at time step k, and $\underline{v}_k$ is additive Gaussian noise with $\underline{v}_k \sim \mathcal{N}(\underline{0}, \mathbf{C}_k^v)$. As the measurement equation is linear, a Kalman filter [15] can be used to perform the measurement update. The surface s_k at time step k can be found by performing an interpolation through the currently estimated positions of the landmarks with any suitable interpolation method.

B. Depth Measurements

In addition to the position measurements, we now want to include depth measurements. For the moment, we assume a single depth camera. Without loss of generality, we define it to be located in the origin of the coordinate system and facing towards $(1,0)^T$. We further assume that the depth camera can obtain r depth measurements $\hat{z}_k^1, \ldots, \hat{z}_k^r$ at r different angles $\alpha_1, \ldots, \alpha_r$. These angles are typically evenly spread across the depth camera's field of view. Consequently, the measurement equation has to calculate the intersections of the line at angle angles $\alpha_1, \ldots, \alpha_r$ with the surface. Depending on the surface representation, calculating this intersection can be very difficult. One of the key ideas of our approach is to use polar coordinates, which nicely circumvents this problem. If we parameterize the surface as a function $s_k : \mathbb{R} \to \mathbb{R}$ which maps angles α to distances $s_k(\alpha)$, the intersection for the lines at angles $\alpha_1, \ldots, \alpha_r$ are trivially calculated as $s_k(\alpha_1), \ldots, s_k(\alpha_r)$.

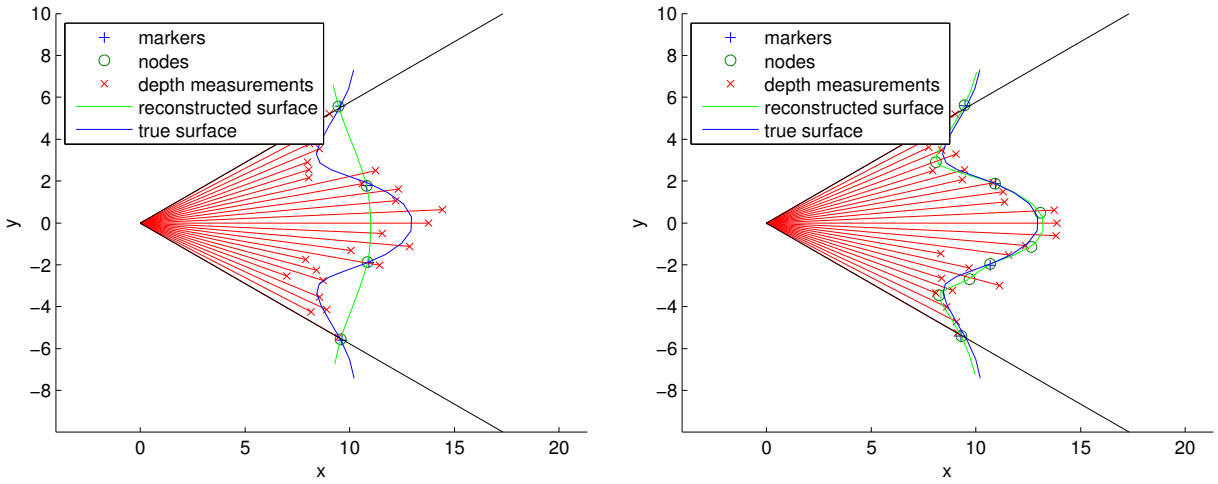
This yields the measurement equation

$$\hat{\underline{z}}_k = \begin{pmatrix} \hat{z}_k^1 \\ \vdots \\ \hat{z}_k^r \end{pmatrix} = \begin{pmatrix} s_k(\alpha_1) \\ \vdots \\ s_k(\alpha_r) \end{pmatrix} + \underline{w}_k \ ,$$
$$s_k(\alpha) = \text{interpolate}(p_k; f_k)(\alpha) \ ,$$
$$p_k = \left(\text{atan2}\left(x_k^{1,2}, x_k^{1,1}\right), \ldots, \text{atan2}\left(x_k^{l,2}, x_k^{l,1}\right)\right) \ ,$$
$$f_k = \left(\left|\left|\begin{pmatrix} x_k^{1,1} \\ x_k^{1,2} \end{pmatrix}\right|\right|, \ldots, \left|\left|\begin{pmatrix} x_k^{l,1} \\ x_k^{l,2} \end{pmatrix}\right|\right|\right)$$

with measurements $\hat{\underline{z}}^k$ and Gaussian noise $\underline{w}_k \sim \mathcal{N}(\underline{0}, \mathbf{C}_k^w)$. The surface function s_k is derived from $\underline{x}_k$ by interpolation, which is in general nonlinear in $\underline{x}_k$. Thus, it is necessary to use a nonlinear filter to perform the depth measurement update. For example, the UKF [16], the S^2KF [18] or the Gaussian filter introduced in [19] may be used.

C. State Augmentation

While it is possible to use depth measurements as just described, the achievable accuracy is still strongly limited by the number of position measurements. The reason for this



(a) Before state augmentation.

(b) After state augmentation.

Fig. 3: Reconstructed and true surface in 2D.

issue is the fact that the number of degrees of freedom of the reconstructed surface are determined by the number of position measurements. An example of this issue is depicted in Fig. 3.

As a result, we augment the state by additional control points that do not correspond to landmarks. One may be tempted to augment the state by the coordinates of points in $\mathbb{R}^2$, which lie on the surface. However, as these points are not located on a fixed position on the surface, their location cannot be uniquely determined from the measurements. Any position in space that leads to the same interpolated surface s_k is just as reasonable to estimate as any other. Consequently, the problem is underdetermined and the state is not observable.

The key idea is to introduce additional nodes not as arbitrary points in $\mathbb{R}^2$ but in polar coordinates as depths at certain fixed angles $\varphi_1, \dots, \varphi_d$. This yields an augmented state

$$\underline{x}_k = (\underbrace{x_k^{1,1}, x_k^{1,2}, \dots, x_k^{l,1}, x_k^{l,2}}_{\text{landmarks}}, \underbrace{x_k^{1,*}, \dots, x_k^{d,*}}_{\text{additional control points}})^T \in \mathbb{R}^{2l+d} ,$$

where $x_k^{1,*} \dots x_k^{d,*}$ are the depths at angles $\varphi_1, \dots, \varphi_d$. These angles are not part of the state, as they are not estimated but chosen as fixed values. This poses the question how to choose these angles. Simple approaches include random angles inside the camera's field of view or angles that lie on an equally spaced grid. More sophisticated ways to choose appropriate angles are discussed in Sec. VI-B.

The measurement equation for positions changes to

$$\hat{\underline{y}}_k = (\mathbf{I}_{2l \times 2l} \mathbf{0}_{2l \times d}) \underline{x}_k + \underline{v}_k ,$$

with identity matrix $\mathbf{I}_{2l \times 2l} \in \mathbb{R}^{2l \times 2l}$ and zero matrix $\mathbf{0}_{2l \times d} \in \mathbb{R}^{2l \times d}$, which just ignores the additional control points. The measurement equation for depth changes to

$$\begin{aligned}
\hat{\underline{z}}_k &= \begin{pmatrix} \hat{z}_k^1 \\ \vdots \\ \hat{z}_k^r \end{pmatrix} = \begin{pmatrix} s_k(\alpha_1) \\ \vdots \\ s_k(\alpha_r) \end{pmatrix} + \underline{w}_k , \\
s_k(\alpha) &= \text{interpolate}(p_k; f_k)(\alpha) , \\
p_k &= \Big(\operatorname{atan2}\left(x_k^{1,2}, x_k^{1,1}\right), \dots, \operatorname{atan2}\left(x_k^{l,2}, x_k^{l,1}\right), \\
&\quad \varphi_1, \dots, \varphi_d \Big) , \\
f_k &= \left(\left\| \begin{pmatrix} x_k^{1,1} \\ x_k^{1,2} \end{pmatrix} \right\|, \dots, \left\| \begin{pmatrix} x_k^{l,1} \\ x_k^{l,2} \end{pmatrix} \right\|, x_k^{1,*}, \dots, x_k^{d,*} \right) ,
\end{aligned}$$

which now includes the additional control points in the interpolation process.

V. Surface Reconstruction in 3D

For many applications that are relevant in practice, 3D surface reconstruction is required. Fortunately, the presented methods can easily be applied to a 3D setting as well.

To accommodate for the third dimension, a few changes are required. Positions in $\mathbb{R}^2$ are replaced with positions in $\mathbb{R}^3$ and polar coordinates are replaced with spherical coordinates. We also change the surface representation to a function $s_k : \mathbb{R}^2 \to \mathbb{R}$ that maps pairs of angles (α, β) to distances $s_k(\alpha, \beta)$. Once again we assume a single depth camera. Without loss of generality, it is located in the origin and facing towards $(1, 0, 0)^T$.

These modifications yield the state representation

$$\underline{x}_k = (\underbrace{x_k^{1,1}, x_k^{1,2}, x_k^{1,3}, \dots, x_k^{l,1}, x_k^{l,2}, x_k^{l,3}}_{\text{landmarks}}, \underbrace{x_k^{1,*}, \dots, x_k^{d,*}}_{\text{additional control points}})^T \in \mathbb{R}^{3l+d} ,$$

where $x_k^{a,b}$ represents the position of landmark a in dimension $b \in \{1,2,3\}$ at time step k. The angles of the additional control points $(\varphi_1, \theta_1), \ldots, (\varphi_d, \theta_d)$ are once again fixed and not part of the state. The measurement equation for positions is now

$$\underline{\hat{y}}_k = (\mathbf{I}_{3l\times 3l}\mathbf{0}_{3l\times d})\underline{x}_k + \underline{v}_k \ ,$$

and the measurement model for depth is

$$\begin{aligned}
\underline{\hat{z}}_k &= \begin{pmatrix} \hat{z}_k^1 \\ \vdots \\ \hat{z}_k^r \end{pmatrix} = \begin{pmatrix} s_k(\alpha_1, \beta_1) \\ \vdots \\ s_k(\alpha_r, \beta_r) \end{pmatrix} + \underline{w}_k \ , \\
s_k(\alpha, \beta) &= \text{interpolate}(p_k; f_k)(\alpha, \beta) \ , \\
p_k &= \left(\begin{pmatrix} \operatorname{atan2}(x_k^{1,2}, x_k^{1,1}) \\ \arcsin\left(\frac{x_k^{1,3}}{||(x_k^{1,1}, x_k^{1,2}, x_k^{1,3})^T||}\right) \end{pmatrix}, \ldots, \right. \\
&\quad \begin{pmatrix} \operatorname{atan2}(x_k^{l,2}, x_k^{l,1}) \\ \arcsin\left(\frac{x_k^{l,3}}{||(x_k^{l,1}, x_k^{l,2}, x_k^{l,3})^T||}\right) \end{pmatrix}, \\
&\quad \left. \begin{pmatrix} \varphi_1 \\ \theta_1 \end{pmatrix}, \ldots, \begin{pmatrix} \varphi_d \\ \theta_d \end{pmatrix} \right) \ , \\
f_k &= \left(\left|\left| \begin{pmatrix} x_k^{1,1} \\ x_k^{1,2} \\ x_k^{1,3} \end{pmatrix} \right|\right|, \ldots, \left|\left| \begin{pmatrix} x_k^{l,1} \\ x_k^{l,2} \\ x_k^{l,3} \end{pmatrix} \right|\right|, x_k^{1,*}, \ldots, x_k^{d,*} \right) \ ,
\end{aligned}$$

where $(\alpha_1, \beta_1), \ldots, (\alpha_r, \beta_r)$ are the angles at which depth measurements are obtained.

VI. Enhancements of the Proposed Method

In this section, we present several enhancements to the proposed method.

A. More Than One Depth Camera

We previously assumed that our surface was observed by just a single depth camera. This assumption can be dropped by transforming the depth measurements of additional cameras into the coordinate system of the first camera. The relation between the coordinate systems and the associated uncertainties can be obtained with standard camera calibration procedures [20].

B. Adaptive Addition of Nodes

When augmenting the state by adding control points, one has to choose the angles where the additional nodes should be located. In the 2D case, we need to determine angles $\varphi_1, \ldots, \varphi_d$ and in the 3D case pairs of angles $(\varphi_1, \theta_1), \ldots, (\varphi_d, \theta_d)$. Simple approaches may involve picking these angles at random within the view of the depth camera or choosing angles that lie on a grid.

However, these choices are usually not optimal. Additional control points should be added adaptively in areas where the error is large or where the expected gain of accuracy is high. One approach for the 2D case is to calculate the RMSE (root mean square error) between estimate and measurement

$$E_k^\tau(\alpha_i) = \sqrt{\frac{1}{\tau}\sum_{j=0}^{\tau-1}(s_{k-j}(\alpha_i) - \hat{z}_{k-j}^i)^2}$$

at time step k within a sliding window of length τ for each angle $\alpha_1, \ldots, \alpha_r$. A large RMSE suggests a control point at this position may be desirable and thus one should choose

$$\varphi = \arg\max_{1\le i\le r}(E_k^\tau(\alpha_i))$$

as the angle of the new control point. This approach can easily be generalized to the 3D case.

In our experiments, we found that adding additional nodes successively tends to give better results than adding several nodes at once.

C. Handling Missing Measurements

In a practical setting, both position and depth measurements may be missing, for example when a tracked landmark is occluded or when the depth sensor is unable to provide a valid depth measurement at a certain angle. It is possible to handle these cases with slight modifications to the proposed method. The measurement models for both position and depth can simply omit the entries of $\underline{\hat{y}}_k$ and $\underline{\hat{z}}_k$ that could not be measured at time step k. Consequently, even a surface that is never visible as a whole at any given time step can be reconstructed over time.

VII. Evaluation

In order to evaluate the proposed algorithm, we have performed several simulations. All simulations use the UKF [16] for nonlinear filtering. We use the following constants:

- initial state x_0^e: random between 0 and 1
- initial covariance: $\mathbf{C}_0^e = \operatorname{diag}(10)$
- initial variance for additional nodes: 10
- noise covariance for position: $\mathbf{C}_k^v = \operatorname{diag}(0.01)$
- noise covariance for depth: $\mathbf{C}_k^w = \operatorname{diag}(1)$

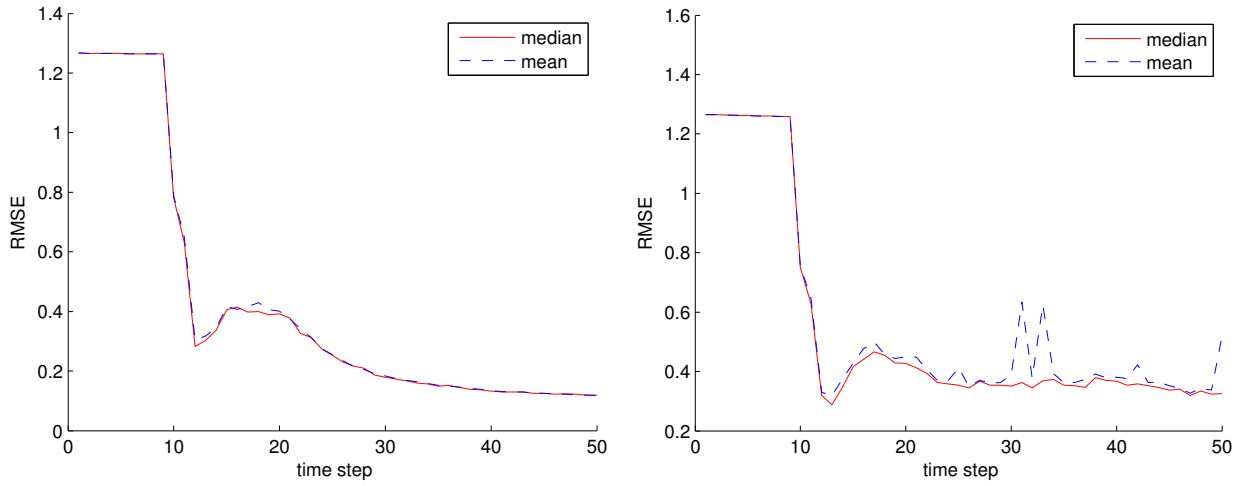
For interpolation, we apply the RBF algorithm depicted in Fig. 1 and use a scaled version of the thin plate spline as the RBF:

$$\phi(x) = \begin{cases} (x/1000)^2 \log(x/1000), & x > 0 \\ 0, & x = 0 \end{cases} \ .$$

A. Simulations in 2D

As a performance measure, we want to determine how similar the reconstructed surface is to the true surface. For this purpose, we choose evaluation angles $\gamma_1, \ldots, \gamma_e$ and define the RMSE E_k of the estimated surface at time step k as

$$E_k = \sqrt{\frac{1}{e}\sum_{i=1}^{e}(s_k(\gamma_i) - s_k^{\text{true}}(\gamma_i))^2} \ ,$$



(a) Static surface.

(b) Dynamic surface.

Fig. 4: Median and mean RMSE for each time step in the 2D case. Additional control points are inserted from time step $k = 10$ until $k = 20$.

where $s_k^{\text{true}}(\cdot)$ is the true surface in polar coordinates. This can be interpreted as the error in depth, measured from the camera towards the surface.

We consider a depth camera with a viewing angle of $60°$ and a resolution of $r = 25$ measurements at equidistant measurement angles. There are 26 evaluation angles, which are equidistant in a $72°$ angle around the camera center, so we evaluate the extrapolation capability of the algorithm as well.

1) Static Case: The true surface that we try to estimate is given by

$$s_k^{\text{true}}(\gamma) = 11 + 2\cos(9 \cdot \gamma)$$

and does not change with time. Thus, we omit the prediction step. This surface is the same as depicted in Fig. 3. We start out with $l = 4$ landmarks and no additional nodes. From time step $k = 10$ to time step $k = 20$ we add one node at each time step, so we have $d = 11$ additional nodes afterwards. The angles $\varphi_1, \ldots, \varphi_d$ are chosen deterministically and are evenly distributed across the camera's view.

The simulation was carried out repeatedly and the median and mean RMSE of the results of 100 Monte Carlo runs are shown in Fig. 4a. As can clearly be seen, the error is very high until time step $k = 10$, because the surface description does not have sufficient degrees of freedom. After all additional nodes are added at $k = 20$, the estimate quickly converges to a point were it has a consistently low error.

2) Dynamic Case: We consider the same situation as in the static case except for the fact that the surface is now time-variant. The moving surface is given by

$$s_k^{\text{true}}(\gamma) = 11 + 2\cos(9 \cdot \gamma) + \sin(0.1 \cdot k)$$

and the system model is assumed to be unknown. Consequently, we use a random walk model for prediction. The system noise is modeled by the covariance matrix $\mathbf{C}_k^{\xi} = \text{diag}(0.1, \ldots, 0.1)$.

Once again, we performed 100 Monte Carlo runs and calculated the mean and median RMSE. The results are depicted in Fig. 4b. Overall the results look similar to the static case, but the overall RMSE is higher as is to be expected. From the deviation between mean and median it can be seen, that there are a few outliers, so estimation is not quite as robust as in the static case.

B. Simulations in 3D

Similar to the 2D case, we choose pairs of evaluation angles $(\gamma_1, \delta_1), \ldots, (\gamma_e, \delta_e)$ and define the RMSE E_k at time step k as

$$E_k = \sqrt{\frac{1}{e}\sum_{i=1}^{e}(s_k(\gamma_i, \delta_i) - s_k^{\text{true}}(\gamma_i, \delta_i))^2} \ ,$$

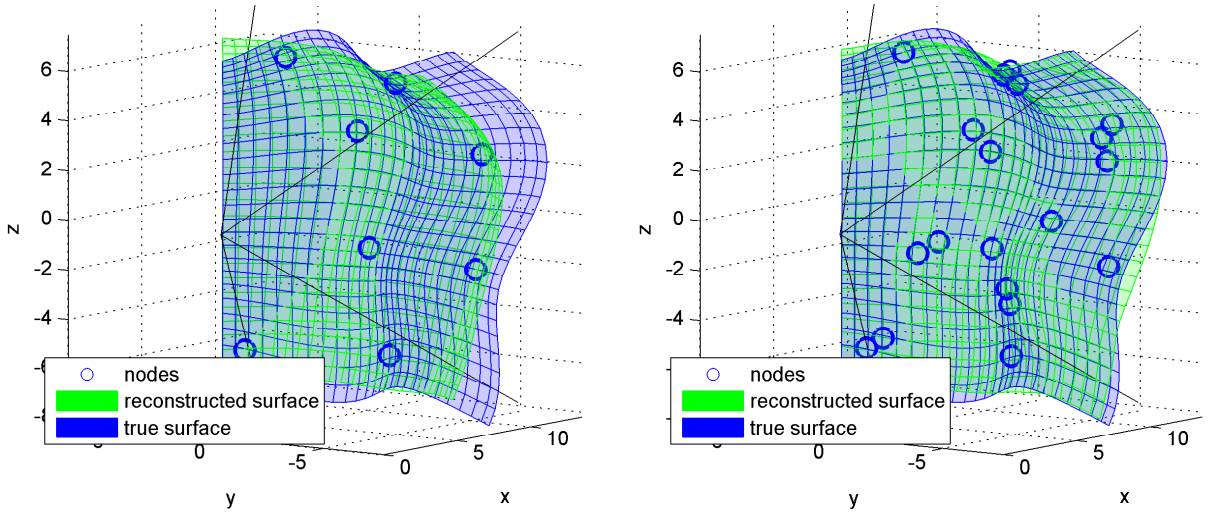
where $s_k^{\text{true}}(\cdot, \cdot)$ is the true surface in spherical coordinates.

We assume a depth camera with a horizontal and vertical viewing angle of $60°$ a resolution of 25×25, so $r = 25^2 = 625$. The measurement angles are located on an equidistant 25×25 grid. For evaluation, we use vertical and horizontal angle of $72°$ and 26×26 equidistant evaluation angles.

1) Static Case: Similar to the static 2D case, we consider a time-invariant surface. In spherical coordinates, it is given by

$$s_k^{\text{true}}(\gamma, \delta) = 12 + \sin(7 \cdot \gamma) + \sin(7 \cdot \delta) \ .$$

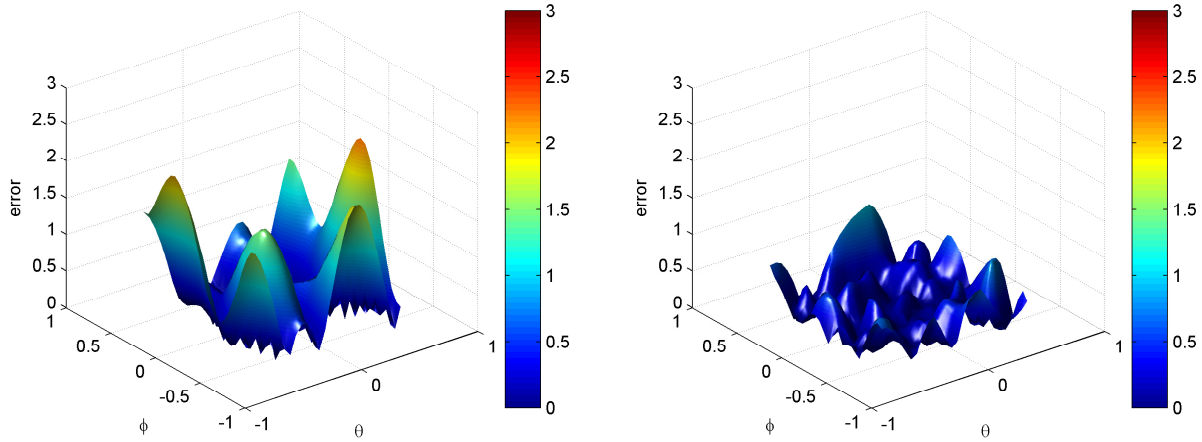
Because the 3D surface has more degrees of freedom, we start with $l = 8$ landmarks. The initial situation is depicted in Fig. 5. Once again, we introduce $d = 11$ additional control points from time step $k = 10$ to $k = 20$. The pairs of angles $(\varphi_1, \theta_1), \ldots, (\varphi_d, \theta_d)$ are evenly distributed across the field of



(a) Time step $k = 9$, without additional control points.

(b) Time step $k = 50$, with additional control points.

Fig. 5: Reconstructed and true surface in 3D.



(a) Time step $k = 9$, without additional control points.

(b) Time step $k = 50$, with additional control points.

Fig. 6: Absolute error in 3D.

view of the camera in a deterministic way. The error can be seen in Fig. 6.

Our results from 100 Monte Carlo runs are shown in Fig. 7a. A comparison with Fig. 4a shows little difference to the 2D case, although the range of the values is different because a different surface is reconstructed.

2) Dynamic Case: For the dynamic 3D case we consider the time-variant surface

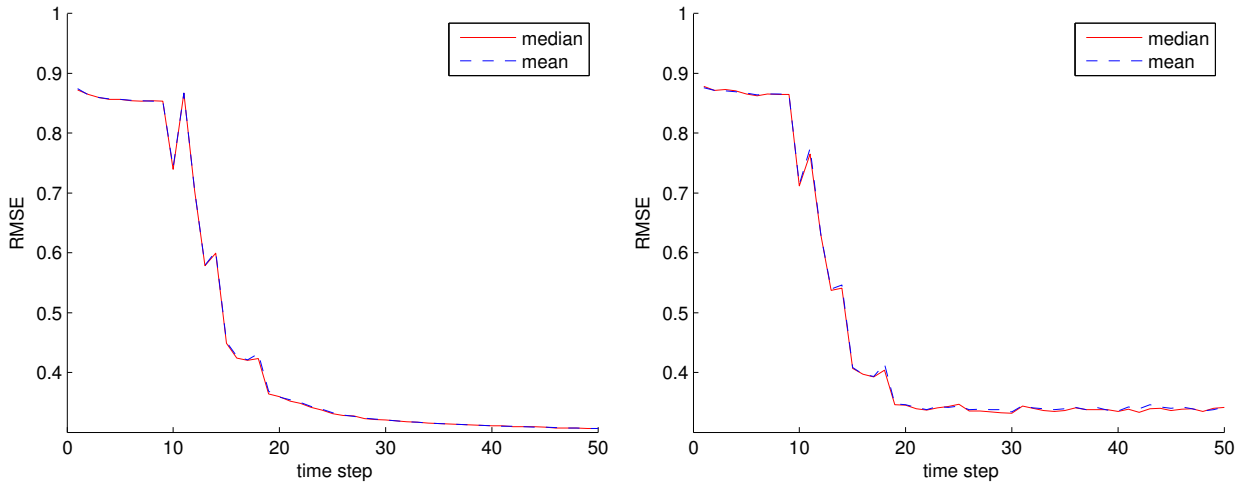
$$s_k^{\text{true}}(\gamma, \delta) = 12 + \sin(7 \cdot \gamma) + \sin(7 \cdot \delta) + \sin(0.1 \cdot k) \ .$$

The system model is a random walk model with system noise $\mathbf{C}_k^{\xi} = \text{diag}(0.1, \ldots, 0.1)$. Fig. 7b shows the results from 100 Monte Carlo runs. This simulation demonstrates that our methods works almost as well in a dynamic as in a static setting.

VIII. Conclusion

We have presented an algorithm for recursively combining depth and position measurements for surface reconstruction under consideration of uncertainties. Surface representation as a spline allows for a compact state representation. The measurement equation for position is trivial and the measurement equation for depth can be calculated easily if polar or spherical coordinates are used.

Through evaluation in simulations we have shown the viability of our approach in both 2D and 3D settings for static as well as dynamic surfaces. Our experiments clearly demonstrate the benefits of adding additional control points in order to better incorporate depth measurements.



(a) Static case.

(b) Dynamic case.

Fig. 7: Median and mean RMSE for each time step in the 3D case. Additional control points are inserted from time step $k = 10$ until $k = 20$.

Future research may include more sophisticated ways to insert additional control points. A practical application of the presented algorithm in a medical setting is planned.

ACKNOWLEDGMENT

This work was partially supported by the German Research Foundation (DFG) within the Research Training Groups RTG 1126 "*Intelligent Surgery - Development of new computer-based methods for the future working environment in visceral surgery*".

REFERENCES

[1] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 163–169, Aug. 1987.

[2] J. C. Carr, W. R. Fright, and R. K. Beatson, "Surface interpolation with radial basis functions for medical imaging," *IEEE Transactions on Medical Imaging*, vol. 16, pp. 96–107, 1997.

[3] D. Stoyanov, G. P. Mylonas, F. Deligianni, A. Darzi, and G. Z. Yang, "Soft-Tissue Motion Tracking and Structure Estimation for Robotic Assisted MIS Procedures," in *Proceedings of the Medical Image Computing and Computer Assisted Interventions (MICCAI 2005)*, vol. 2, 2005, pp. 139–146.

[4] Z. C. Marton, R. B. Rusu, and M. Beetz, "On fast surface reconstruction methods for large and noisy point clouds," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA 2009)*, may 2009, pp. 3218 –3223.

[5] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *SIGGRAPH Comput. Graph.*, vol. 26, no. 2, pp. 71–78, Jul. 1992.

[6] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *In Proc. UIST*, 2011, pp. 559–568.

[7] L. Guan, J.-S. Franco, and M. Pollefeys, "3D Object Reconstruction with Heterogeneous Sensor Data," in *International Symposium on 3D Data Processing, Visualization and Transmission*, Atlanta, États-Unis, 2008.

[8] J. Zhu, L. Wang, R. Yang, J. Davis, and Z. Pan, "Reliability fusion of time-of-flight depth and stereo geometry for high quality depth maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 7, pp. 1400 –1414, july 2011.

[9] S. Lee, G. Wolberg, and S. Y. Shin, "Scattered data interpolation with multilevel B-splines," *Visualization and Computer Graphics, IEEE Transactions*, vol. 3 (3), pp. 228–244, 1997.

[10] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: A survey," *Journal of Electronic Imaging*, vol. 11(2), no. 2, pp. 157–176, apr 2002.

[11] D. Brunn and U. D. Hanebeck, "A Model-Based Framework for Optimal Measurements in Machine Tool Calibration," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA 2005)*, Barcelona, Spain, Apr. 2005.

[12] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, ser. Adaptive Computation And Machine Learning. Mit Press, 2006.

[13] N. Arad and D. Reisfeld, "Image Warping Using Few Anchor Points and Radial Funcions," *Computer Graphics Forum*, vol. 14, pp. 35–46, 1995.

[14] M. Baum and U. D. Hanebeck, "Shape Tracking of Extended Objects and Group Targets with Star-Convex RHMs," in *Proceedings of the 14th International Conference on Information Fusion (Fusion 2011)*, Chicago, Illinois, USA, Jul. 2011.

[15] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[16] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, mar 2004.

[17] C. Ye and J. Borenstein, "Characterization of a 2d laser scanner for mobile robot obstacle negotiation," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA 2002)*, vol. 3, 2002, pp. 2512 –2518.

[18] J. Steinbring and U. D. Hanebeck, "The Smart Sampling Kalman Filter (to appear)," in *Proceedings of the 16th International Conference on Information Fusion (Fusion 2013)*, Istanbul, Turkey, Jul. 2013.

[19] M. F. Huber and U. D. Hanebeck, "Gaussian Filter based on Deterministic Sampling for High Quality Nonlinear Estimation," in *Proceedings of the 17th IFAC World Congress (IFAC 2008)*, vol. 17, no. 2, Seoul, Republic of Korea, Jul. 2008.

[20] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.