Contour Reconstruction using Recursive Smoothing Splines - Algorithms and Experimental Validation

M. Karasalo, G. Piccolo, D. Kragic and X. Hu

Abstract

In this paper, a recursive smoothing spline approach for contour reconstruction is studied and evaluated. Periodic smoothing splines are used by a robot to approximate the contour of encountered obstacles in the environment. The splines are generated through minimizing a cost function subject to constraints imposed by a linear control system and accuracy is improved iteratively using a recursive spline algorithm. The filtering effect of the smoothing splines allows for usage of noisy sensor data and the method is robust with respect to odometry drift. The algorithm is extensively evaluated in simulations for various contours and in experiments using a SICK laser scanner mounted on a PowerBot from ActivMedia Robotics.

Keywords: mapping, optimal control, recursive smoothing splines, implementation

D.1 Introduction

Today robots are expected to operate in at least partially unknown, open-ended environments, detecting and moving around obstacles, building maps of the environment and localizing their position. The area of autonomous mapping has during the past few years matured and there has been a significant amount of work reported using both visual and laser sensors, [8], [26]. However, most methods are still concerned with detecting and maintaining a dense set of discrete features of the environment that are suitable for navigation and localization without retrieving any specific knowledge about the detailed shape of the encountered objects.

In this paper, we consider the problem of estimating representations of objects using a type of continuous closed curves, the recursive periodic smoothing splines. The splines are retrieved from noisy measurements of the true contour. Intended applications include mapping, identification and path planning. It is well known that interpolating splines from noisy measurement data will give a poor result, as the resulting curve will go through every measurement point. Data smoothing has been a classical problem in system and control history [1], [13], [18]. The theory of regular smoothing splines is thoroughly treated in [29] and [30]. It has been further shown in [35] that control theoretic smoothing splines, where the curve is found through minimizing a cost function, act as filters and are better suited for noisy measurements. It is also noted in [27] that smoothing

splines are in some sense band limited so that small changes in one data point will mainly affect the spline in a neighborhood of that point.

In the work presented here, periodic smoothing splines are derived from noisy data by solving an optimal control problem for a linear system. We discuss two approaches to curve estimation using smoothing splines, one closed form and one recursive method. The idea of the latter is to use several data sets taken from the same continuous feature and compute splines iteratively to refine the estimate stepwise.

This paper is organized as follows. In Section D.2 we acknowledge related work in the field and point out new contributions of this paper. The contour estimation problem is formulated in Section D.3, and in Section D.4 we derive a discretized version of the optimization problem, suitable for implementation. Finally simulations and experimental results are presented for a few test objects in Section D.5 and conclusions are drawn in Section D.6.

D.2 Related Work and Contributions

The application of smoothing splines for path planning and contour synthesis has been investigated in for instance [9], [10] and [14]. [28] and [31] investigate methods of determining the appropriate level of smoothing by analyzing properties of the input data. LQ problems with various types of boundary conditions are studied in [7] and [23] but to the extent of our knowledge, little work has been done on the type of periodic boundary conditions studied in this paper.

Adaptive splines, or snakes [25], and bicubic B-splines [21] have been used in contour matching and surface representation schemes in the vision field. However, these algorithms fit splines to segments of the investigated curve or surface and subsequently patch the splines together. The control theoretic periodic splines presented in this paper approximate the entire contour of an object while maintaining the desirable property that local changes in input data leads only to local changes in the output spline.

While we have previously modeled and evaluated the algorithms in simulation for reconstruction of analytic curves, [16], in this paper we use a real robot platform equipped with SICK laser sensors and investigate arbitrary closed curves. [24] contains some of the results of this paper in an abbreviated form. The main contributions of this paper are the derivation of an implementable discrete version of the smoothing spline problem both for the closed form and the recursive approach, a contour reconstruction algorithm including both forms, and a series of experiments that validate convergence of the recursive approach and the applicability of the methods in practice. For an extensive theoretical background on the type of smoothing splines studied in this paper we refer to [15], [17], [19], and [20] and the references therein.

D.3 Problem formulation and motivation

We consider a mobile agent placed in a planar environment and equipped with range sensors and encoders. The task is to construct 2D-representations of encountered objects in the environment, by means of approximating their contour from noisy sensor data. In this framework, polar coordinates are used for contour reconstruction, thus the only requirement on the objects is that the contours $y(t)$ are well defined in polar coordinates, that is $y(t_1) \neq y(t_2) \Rightarrow t_1 \neq t_2, \forall t \in [0, 2\pi]$. Such contours are called star-shaped or star-convex [32] and are formally defined by

Definition D.3.1 *A set $X \in \mathbb{R}^2$ is star-shaped if $\exists t_c \in X$ such that $\forall t \in X$ the line $t - t_c \in X$. A curve $y(t)$ is star-shaped if its interior X is star-shaped.*

Our problem can be formulated as follows.

We are given a data set $D = \{(t_i, z_i) : i = 1, \dots, N\}$, where $t_i \in [0, T]$ is the polar coordinate angle, $T = 2\pi$ and z_i is the radius in polar coordinates. The data in D are noise contaminated measurements $z_i = y(t_i) + \xi_i$ from a closed continuous curve, where ξ_i is a symmetric, zero-mean iid noise with bounded variance. How to find the function $y(t)$ that best represents the underlying curve, with respect to smoothness and closeness to measurement data?

The solution is found by solving the following polar second derivative L_2 smoothing problem:

Problem D.3.1

$$\begin{aligned} \underset{u}{\text{minimize}} \quad J(u, x) = & x_0^T P_0^{-1} x_0 + \int_0^T u(t)^T Q^{-1} u(t) dt + \\ & \sum_{i=1}^N (t_i - t_{i-1}) (z_i - Cx(t_i))^T R_0^{-1} (z_i - Cx(t_i)) \end{aligned} \quad (\text{D.1})$$

$$\begin{aligned} \text{subject to} \quad \dot{x} &= Ax + Bu \\ x_0 &= x(0) = x(T). \end{aligned} \quad (\text{D.2})$$

$A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^n$ and $C^T \in \mathbb{R}^n$, where the pair (A, B) is controllable and (A, C) is observable. P_0 , Q and R are positive definite matrices of suitable dimension. The constraints (D.2) consist of an n -dimensional ODE with relative degree n and periodic boundary condition. The resulting smoothing spline is given by $y(t) = Cx(t)$. We refer to the system

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \quad (\text{D.3})$$

as the spline generator of (D.1). Thus the sought curve y is the output of an n -tuple integrator with state variable x controlled by an input $u = x_n = x_1^{(n)}$, the n :th derivative of x_1 . The dimension of the spline generator determines on which derivative to impose the smoothing penalty. As the magnitude of the second derivative of $x_1(t)$ is proportional to the curvature of $x_1(t)$, $n = 2$ is a natural choice. Therefore the matrices used in our setup are

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad P_0^{-1} = \begin{bmatrix} \delta_1^2 & 0 \\ 0 & \delta_2^2 \end{bmatrix}, \\ C &= \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad Q = 1, \quad R_0 = 1/\varepsilon^2, \quad \varepsilon > 0. \end{aligned} \quad (\text{D.4})$$

Let us have a closer look at the final two terms of the cost function (D.1). The integral imposes a penalty on large magnitude of the input $u(t)$, corresponding to the curvature of the curve $y(t) = Cx(t) = x_1(t)$. The sum punishes large deviations of the curve $y(t)$ from the data (t_i, z_i) . In other words, the solution of Problem D.3.1 is in some sense the optimal compromise between smoothness of the output curve and faithfulness to the data set. The magnitude of $\varepsilon > 0$ determines how much credibility is given to measurement data. A large value brings the spline close to the data points, while a small value yields a smoother spline and thus more filtering.

As the data is noise contaminated, the spline from one data set D is a raw result, especially if the data is sparse. Therefore we introduce a recursive smoothing spline method, where the optimal solution $(x^{k-1}(t), u^{k-1}(t))$ from the previous iteration is used in iteration k together with the new data z_i^k .

Problem D.3.2

$$\begin{aligned} \underset{u^k}{\text{minimize}} \quad J_k(u^k, x^k) = & (x_0^k - x_0^{k-1})^T P_0^{-1} (x_0^k - x_0^{k-1}) + \\ & \int_0^T (u^k(t) - u^{k-1}(t))^T Q^{-1} (u^k(t) - u^{k-1}(t)) dt + \\ & \sum_{i=1}^N (t_i - t_{i-1}) (z_i^k - Cx^k(t_i))^T R_k^{-1} (z_i^k - Cx^k(t_i)) \quad (\text{D.5}) \end{aligned}$$

$$\begin{aligned} \text{subject to} \quad \dot{x}^k &= Ax^k + Bu^k \\ x_0^k &= x^k(0) = x^k(T). \end{aligned} \quad (\text{D.6})$$

Here, $R_k = 1/\varepsilon_k^2$ which can be chosen as a constant parameter or as a function of k with $\varepsilon_{k+1} < \varepsilon_k$ for increased convergence performance. Using a recursive method has the advantage of getting a better spline approximation with each iteration, improving the map stepwise so that the previous map can be used for path planning in the next iteration. Such a path planning algorithm that essentially uses the servoed contour and displaces it orthogonally toward the servoing agent is described in [34]. Localization may also be improved by fusing for instance odometry data with knowledge of approximate position of various features of the contour. Also, if for some reason new data is obtained only for one part of the curve, we can modify that part of the spline separately by performing the next recursion using new data for that part of the curve and old data for the rest. Problem D.3.1 is a continuous time problem with discrete data and periodic boundary condition. Such problems, without the periodicity constraint, have been widely studied in the literature. See for instance [1], [13], and [18]. A thorough treatment of the theoretical properties of periodic smoothing spline problems can be found in [15]. As is discussed in [15], including the term $\frac{1}{2}x_0^T P_0^{-1} x_0$ simplifies the solution of the problem. In this paper, we show that with this term, existence of a solution to Problem D.3.1 is guaranteed even with an empty input data set. For practical purposes and the implementation discussed in the next section this term is however unnecessary and may be omitted. A derivation of the Riccati equation corresponding to Problem D.3.1 is provided in [16]. It is also shown that as the size N of the input data set increases, the solution of Problem D.3.1 approaches the solution of the corresponding problem for continuous data.

Due to the recursive nature of Problem D.3.2, the Riccati equations become rather complex and convergence has, to the extent of our knowledge, yet to be shown analytically. Simulation results in [16] do however suggest convergence of the error as $1/\sqrt{k}$. In this paper the algorithm is evaluated on real experimental data from test runs with a PowerBot from ActivMedia Robotics, using a SICK laser scanner.

D.4 Implementation

In this section we derive a discretized version of the smoothing spline problem. We show that using a proper choice of approximation formulas both Problem D.3.1 and Problem D.3.2 are reduced to unconstrained quadratic programming problems (QP), suitable for implementation. We further show that by choosing the input cleverly to Problem D.3.2 the QP 's become identical. With this particular choice of discretization the periodic boundary condition is embedded in the QP , facilitating the analysis of convexity and solvability for the problem. In the last part of this section we state and prove that the QP is convex and therefore always has a unique solution. For a polar coordinate

spline $r(\theta)$, the interpretation of the state variables and data points is as follows:

$$\begin{aligned} x_1(t) &= y(t) = r(\theta), \\ x_2(t) &= \dot{x}_1(t) = r'(\theta), \\ \dot{x}_2(t) &= r''(\theta) = u(\theta), \\ z_i &= \text{radius measurement}, \\ t_i &= \text{angle measurement}, \\ T &= 2\pi, \\ w_i &= t_i - t_{i-1}. \end{aligned}$$

In Figure 1 the notation is shown together with two splines drawn for the same data set but with different weights ε . Expressing Problem D.3.1 using the polar notation and the matrices defined in

Figure 1: Notation for the smoothing spline problem in polar coordinates.

Equation (D.4) yields

$$\underset{r}{\text{minimize}} \quad J(r) = \delta_1^2 r(0)^2 + \delta_2^2 r'(0)^2 + \int_0^{2\pi} r''(\theta)^2 d\theta + \varepsilon^2 \sum_{i=1}^N w_i (r(t_i) - z_i)^2 \quad (\text{D.7})$$

$$\begin{aligned} \text{subject to} \quad r(0) &= r(2\pi) \\ r'(0) &= r'(2\pi). \end{aligned} \quad (\text{D.8})$$

Note that Equations (D.7) and (D.8) together constitute a minimization over only one variable, r , with the dynamic constraints of Equation (D.3) built in.

D.4.1 Discretization

Let the vector $\hat{r} = \{\hat{r}_m\}$ be the discretization of the wanted curve $r(\theta)$, and let $\hat{\theta} = \{\hat{\theta}_m\}$ be the corresponding discretization of θ . Here $m = 1, \dots, M$ and we define the sampling rate h so that

$(M + 1)h = 2\pi$. We emphasize that $(\hat{\theta}_m, \hat{r}_m)$ are equidistant samplings from the spline (r, θ) while (t_i, z_i) , $i = 1, \dots, N$ are noisy measurement data from the true curve. Note that when the continuous function $r(\theta)$ is expressed in discretized form as a vector pair $(\hat{\theta}, \hat{r})$, the periodicity constraint translates to $(\hat{\theta}_1, \hat{r}_1) = (\hat{\theta}_{M+1}, \hat{r}_{M+1})$, where $M+1$ indicates the point after the last point of the vector. All terms of $J(r)$ can be approximately expressed as functions of $\hat{r}$, using numerical differentiation formulas:

$$\begin{aligned}\hat{r}'_m &= (1/2h)(\hat{r}_{m-1} - \hat{r}_{m+1}) \\ \hat{r}''_m &= (1/h^2)(\hat{r}_{m-1} - 2\hat{r}_m + \hat{r}_{m+1}).\end{aligned}\tag{D.9}$$

Construct the matrix $\Phi \in \mathbb{R}^{M \times M}$:

$$\begin{aligned}\Phi_{m,m-1} &= \Phi_{m,m+1} = 1 \\ \Phi_{1,M} &= \Phi_{M,1} = 1 \\ \Phi_{m,m} &= -2 \\ \Phi_{j,l} &= 0 \quad \text{otherwise.}\end{aligned}\tag{D.10}$$

Then $\hat{r}'' = \frac{1}{h^2}\Phi\hat{r}$. Note that the periodicity of $r(\theta)$ is implicitly expressed in Φ through Equation (D.10). The discretization of the integral is

$$\int_0^{2\pi} r''(\theta)^2 d\theta \approx \frac{1}{h^3} \hat{r}^T \Phi^T \Phi \hat{r}.\tag{D.11}$$

For the term $\delta_2^2 r'(0)^2$ we construct a vector $q \in \mathbb{R}^M$:

$$q_m = \begin{cases} 1 & \text{if } m = 2 \quad \text{or } m = M \\ 0 & \text{otherwise} \end{cases}\tag{D.12}$$

$$\Rightarrow \delta_2^2 r'(0)^2 \approx (\delta_2^2/2h) \hat{r}^T q q^T \hat{r}.\tag{D.13}$$

For $\delta_1^2 r(0)^2$ we introduce the vector $p \in \mathbb{R}^M$:

$$p_m = \begin{cases} 1 & \text{if } m = 1 \\ 0 & \text{otherwise} \end{cases}\tag{D.14}$$

$$\Rightarrow \delta_1^2 r(0)^2 \approx \delta_1^2 \hat{r}^T p p^T \hat{r}.\tag{D.15}$$

The last term of (D.7), the sum, involves measurement data (t_i, z_i) and the spline $r(\theta)$ evaluated at the measured angles t_i :

$$\varepsilon^2 \sum_{i=1}^N w_i (r(t_i) - z_i)^2 = \varepsilon^2 \sum_{i=1}^N w_i r(t_i)^2 + \varepsilon^2 \sum_{i=1}^N w_i z_i^2 - \varepsilon^2 \sum_{i=1}^N 2w_i r(t_i) z_i.\tag{D.16}$$

Since we are minimizing over r we need not consider the constant second term of (D.16). For the first term we construct the matrix $W \in \mathbb{R}^{M \times M}$:

$$\begin{aligned}W_{m,m} &= \begin{cases} w_i & \text{if } \hat{\theta}_m = t_i \quad \text{for some } i \\ 0 & \text{otherwise} \end{cases} \\ W_{j,l} &= 0 \quad \text{for } j \neq l.\end{aligned}\tag{D.17}$$

Similarly, for the last term we construct the vector $s \in \mathbb{R}^M$:

$$s_m = \begin{cases} w_i z_i & \text{if } \hat{\theta}_m = t_i \quad \text{for some } i \\ 0 & \text{otherwise.} \end{cases}\tag{D.18}$$

We get

$$\varepsilon^2 \sum_{i=1}^N w_i (r(t_i) - z_i)^2 \approx \varepsilon^2 \hat{r}^T W \hat{r} + \varepsilon^2 \sum_{i=1}^N w_i z_i^2 - 2\varepsilon^2 s^T \hat{r}. \quad (\text{D.19})$$

All in all, the discretized problem is

$$\underset{\hat{r}}{\text{minimize}} \quad \hat{J}(\hat{r}) = \frac{1}{2} \hat{r}^T H \hat{r} + c^T \hat{r} \quad (\text{D.20})$$

$$\begin{aligned} \text{with } H &= 2 \left(\frac{1}{h^3} \Phi^T \Phi + \left(\frac{\delta_2}{2h} \right)^2 q q^T + \delta_1^2 p p^T + \varepsilon^2 W \right) \\ c &= -\varepsilon^2 s. \end{aligned} \quad (\text{D.21})$$

$H \in \mathbb{R}^{M \times M}$ is a pentadiagonal matrix with additional non-zero elements $H_{1,M-1}$, $H_{1,M}$, $H_{2,M}$, $H_{M-1,1}$, $H_{M,1}$, $H_{M,2}$.

D.4.2 Recursive Version

The polar coordinate formulation of Problem D.3.2 is simplified by defining

$$\begin{aligned} \tilde{z}_i^k &= z_i^k - r^{k-1}(t_i^k) \\ \tilde{r}^k(\theta) &= r^k(\theta) - r^{k-1}(\theta), \end{aligned} \quad (\text{D.22})$$

where (t_i^k, z_i^k) are the data points from iteration k , $r^{k-1}(t_i^k)$ is the resulting spline of iteration $k-1$ evaluated at the angle measurements of the new data set, and $r^k(\theta)$ is the spline output of iteration k . Then Problem D.3.2 becomes

$$\underset{\tilde{r}^k}{\text{minimize}} \quad J(\tilde{r}^k) = \delta_1^2 \tilde{r}^k(0)^2 + \delta_2^2 \tilde{r}^{k'}(0)^2 + \int_0^{2\pi} \tilde{r}^{k''}(\theta)^2 d\theta + \varepsilon_k^2 \sum_{i=1}^N w_i (\tilde{r}^k(t_i^k) - \tilde{z}_i^k)^2 \quad (\text{D.23})$$

$$\begin{aligned} \text{subject to } \tilde{r}^k(0) &= \tilde{r}^k(2\pi) \\ \tilde{r}^{k'}(0) &= \tilde{r}^{k'}(2\pi), \end{aligned} \quad (\text{D.24})$$

and the spline found in iteration k is

$$r^k(\theta) = \tilde{r}^k + r^{k-1}(\theta). \quad (\text{D.25})$$

Note that Equations (D.23) and (D.24) are identical to Equations (D.7) and (D.8) except for the new variable names $\tilde{r}^k$ and $\tilde{z}^k$. Hence the discretization and solution methods for Problem D.3.1 and Problem D.3.2 are identical except that the input to the latter is $\tilde{z}^k$ at iteration k . To summarize, the recursive algorithm, after an initialization step, is carried out in three steps:

Algorithm D.4.1

- 0) *Initialization.* For $k = 0$, input the first data set (t_i^0, z_i^0) into Problem D.3.1 and compute the first spline $r^0(\theta)$.
- 1) Put $k = k + 1$, construct $(t_i, \tilde{z}_i^k)$ from the data set (t_i^k, z_i^k) and the spline $r^{k-1}(\theta)$.
- 2) Input $(t_i, \tilde{z}_i^k)$ into Problem D.3.2 and compute the spline $\tilde{r}^k(\theta)$.
- 3) Construct the spline $r^k(\theta) = \tilde{r}^k(\theta) + r^{k-1}(\theta)$. Go to 1).

D.4.3 Convexity of the QP

We conclude this section by proving that the QP (D.20) always has a solution and that the solution is unique.

Proposition D.4.1 *Assume that at least one of the following conditions holds:*

- $W_{m,m} \neq 0$ for at least one value of $m \in [1, M]$ (this is equivalent to having a non-empty data set, $N \geq 1$).
- $\delta_1 \neq 0$
- $\delta_2 \neq 0$.

Then Equation (D.20) has a unique solution.

Proof Equation (D.20) is a quadratic unconstrained minimization problem. It is well known that if H is positive semi definite and there exists an $\hat{r}^*$ that solves

$$H\hat{r} + c = 0, \quad (\text{D.26})$$

then $\hat{r}^*$ is optimal. Further, if H is positive definite the QP is strictly convex and has a unique solution $\hat{r}^*$. The matrix $\frac{1}{h^3}\Phi^T\Phi$ is by construction a symmetric positive semidefinite matrix, but not positive definite since $\text{rank}(\Phi) = M - 1$. The single zero eigenvalue of Φ has the eigenvector $v_0 = [1, 1, \dots, 1]^T$. The components of the vectors q and p , as defined by (D.12) and (D.14), respectively, are nonnegative. W is diagonal with nonnegative elements and has rank N . Now, any $x \in \mathbb{R}^M$ can be decomposed as

$$x = v + \alpha v_0, \quad (\text{D.27})$$

where α is scalar and $v \perp v_0$, and

$$x^T \Phi^T \Phi x = v^T \Phi^T \Phi v \geq 0, \quad (\text{D.28})$$

with equality only for $v = 0$. Therefore,

$$x^T H x = \frac{1}{h^3} v^T \Phi^T \Phi v + \left(\frac{\delta_2}{2h} \right)^2 (q^T x)^2 + \delta_1^2 (p^T x)^2 + x^T W x. \quad (\text{D.29})$$

If x is nonzero, then at least one of v and α is nonzero. If $v \neq 0$, then $\frac{1}{h^3} v^T \Phi^T \Phi v > 0$, and consequently $x^T H x > 0$. If $v = 0$, $\alpha \neq 0$ we get

$$x^T H x = \alpha^2 \left(\left(\frac{\delta_2}{2h} \right)^2 (q^T v_0)^2 + \delta_1^2 (p^T v_0)^2 + v_0^T W v_0 \right) > 0. \quad (\text{D.30})$$

It follows that H is nonsingular and thereby positive definite. Hence the minimization problem (D.20) has a unique solution. ■

D.5 Experimental Evaluation

In this section we present results for simulated and real data using a mobile agent with range sensors.

D.5.1 Experimental setup

We use the PowerBot from ActivMedia Robotics. It has unicycle dynamics and uses odometry data for localization. The robot is provided with two SICK laser scanners mounted one in front and one on the left side of the chassis as shown in Figure 2. The scanners give range measurements on the interval $[-\pi/2, \pi/2]$ rad from the center of the scanner and with a resolution of 0.5° , or 361 measurements each time step.

Figure 2: The PowerBot platform equipped with two SICK sensors.

D.5.2 Converting measurements to data points

In this section we describe the steps taken to convert measurement data to suitable input to Algorithm D.4.1. Using odometry for localization and SICK range sensors for distance measurements, the gathered data at each time step consists of the robot position (x, y, ϕ) and the distances $S(v)$ to the measured object from the angles v of the SICK sensors. The input to Algorithm D.4.1 should consist of polar coordinate data (t, z) representing the contour of the object.

The robot position (x, y) is defined as the center point on the major wheel axis and its heading ϕ is the deviation from the x -axis which is defined as the heading of the robot at its starting position $(0, 0)$. The sensors are located at distances L_1 and L_2 from (x, y) as shown in Figure 3. For the sensor located in the front, a point (x_w, y_w) on the target object is obtained as follows:

$$\begin{aligned} x_w &= x + L_1 \cos \phi + S_j \cos(\phi + v_j) \\ y_w &= y + L_1 \sin \phi + S_j \sin(\phi + v_j), \end{aligned} \tag{D.31}$$

Figure 3: Notation of angles and distances for the PowerBot.

and using the side sensor we get

$$\begin{aligned} x_w &= x - L_2 \sin \phi - S_i \sin(\phi + v_i) \\ y_w &= y + L_2 \cos \phi + S_i \cos(\phi + v_i). \end{aligned} \quad (\text{D.32})$$

Here $v_i, v_j \in [-\pi/2, \pi/2]$ is the angle of the sensor ray and S_i, S_j is the range measurement associated with it, as seen in Figure 3.

Conversion to suitable input (t_i, z_i) , $i = 1, \dots, N$ is performed according to the following scheme:

- 1) The origin of the polar coordinate system is chosen to coincide with the mass center of the cartesian data (x_w, y_w) . As the SICK sensors produce dense data sets and under the assumption of a star-shaped contour, the mass center is a natural estimate of t_c in Definition D.3.1.
- 2) Define $z_w = \sqrt{x_w^2 + y_w^2}$, $t_w = \tan^{-1}(y_w/x_w)$.
- 3) The odometry data is contaminated with a drifting error. Thus the resulting set (t_w, z_w) gives a skew representation of the true shape. By making a least squares fit between two consecutive data sets from the same object, we find and compensate for the odometry drift at each time step. For details of the procedure we refer to [17].
- 4) The resulting set is dense and possibly contaminated with outliers. A rough prefiltering removes outliers and produces a sparse data set (t, z) which allows for faster computations: The set (t_w, z_w) is divided into N windows. For each consecutive window, the mean and standard deviations of t_w and z_w are computed, points outside one standard deviation are removed, and the mean is computed again. The second mean values (t_i, z_i) , $i = 1, \dots, N$ constitute the input data to Algorithm D.4.1.

D.5.3 Computation Time

From the implementation details of Section D.4, it is clear that the bottleneck of Algorithm D.4.1 is the inversion of the matrix $H \in \mathbb{R}^{M \times M}$, where M is the chosen level of discretization. As $M \rightarrow \infty$

the discretized problem approaches the analytical problem and the accuracy of the solution improves. The price is an increase in computational time. The size N of the input data set determines the number of nonzero elements in the vector c (Equation (D.21)) and thereby also affects the computation time. The magnitude of N is set in the prefiltering step, and while computation time benefits from a small value, a larger one may yield a more rapid convergence, allowing for fewer iterations. In SLAM applications, large association matrices are often inverted online, see for instance [33]. For online applications of Algorithm D.4.1 a similar approach should be used for inversion of the matrix H . As the aim of the simulations and experiments in this work was to evaluate accuracy and convergence properties of Algorithm D.4.1, time-optimization of the code remains for future work.

D.5.4 Results

The relative error between the spline generated by Algorithm D.4.1 at iteration k and the true curve is defined as

$$e_k = \|\hat{r}^k - \hat{r}_{true}\| / \|\hat{r}_{true}\| \quad (\text{D.33})$$

where $\hat{r}^k$ is the discretized smoothing spline output, and $\hat{r}_{true}$ is the corresponding true curve in polar coordinates.

For each test object we plot the error convergence and resulting spline approximation after a number of iterations. The stars plotted around the contours represent the data set (t_i, z_i) used for the final iteration, after the conversion and prefiltering discussed in Section D.5.2. The headings of the figures indicate the values used of ε_0 and M , noise level, number of iterations and the quotient e_{end}/e_1 between initial and final error.

D.5.4.1 Simulated Data

The contours investigated in simulation correspond to lakes and ponds located in the Arbuckle-Simpson Aquifer study area in Oklahoma. Maps of the area are available on line in the form of MATLAB shape files, published by the Oklahoma Water Resources Board in 2004

(<http://www.owrb.state.ok.us/>). The simulation data used to reconstruct the contours of the lakes and ponds was constructed by making samplings from the true contours and adding noise. The samplings may be considered sparse in the sense that the ratio between number of data points and the chosen number of discretization points for the curve (called M in Section D.4) is roughly 1/20. We use $M = 1600$ and $N \approx 80$, depending on the size of the contour. The added noise was normally distributed with mean zero. The magnitude of the standard deviation of the added noise is given as percentage of the mean contour radius. We iterate over 100 data sets for each contour with a decreasing value of ε_k , chosen as $\varepsilon_k = \varepsilon_0 / \sqrt{k}$. This is a natural choice as it corresponds to the expected convergence rate of Algorithm D.4.1. However other functions or even a constant value of ε will work but might require more tuning to find the best value. In total 25 lake contours were investigated. The complete simulation results can be found at <http://www.math.kth.se/~karasalo/spline>. Due to space limitations only a few illustrating examples are included in this paper.

Three Example Contours We illustrate the performance of Algorithm D.4.1 with three example contours (Figures 4 - 6). The magnitude of the added noise corresponds to 5% of the mean contour radius, and $\varepsilon_0 = 28$. For each of the test cases the initial and final error was recorded and can be seen in the captions of Figures 4 - 6. For each of the investigated contours we plot the error at each iteration, together with the true contour (dashed) and the spline approximation at iteration 100.

Figure 4: **Three Example Contours:** Lake 2, $e_1 = 0.12$, $e_{100} = 0.08$, $e_{100}/e_1 = 0.65$. Figure 5: **Three Example Contours:** Lake 23, $e_1 = 0.10$, $e_{100} = 0.04$, $e_{100}/e_1 = 0.43$. Figure 6: **Three Example Contours:** Lake 24, $e_1 = 0.13$, $e_{100} = 0.06$, $e_{100}/e_1 = 0.52$.

Sensitivity to Noise To give a flavor of the algorithm's sensitivity to noise, we perform simulations for an increasing value of the added noise. Figures 7 - 10 show results for one of the lakes using a noise level of 0%, 2.5%, 7.5% and 10%. For each of the investigated noise levels we plot the error at each iteration, together with the true contour (dashed) and the spline approximation at iteration 100. As expected performance is better for low to moderate noise levels, but convergence is still apparent even at a noise level of 10%. Results for all 25 test cases are summarized in Table D.1 for noise levels between 0% and 20%. The table shows initial and final errors and over all error decrease. The last column shows the mean number of iterations required to obtain an error decrease of at least 25%. Significant improvement is generally obtained after a few (≤ 10) iterations. It should be noted that for different expected noise levels one should tune the design parameter ε_0 for best performance. The aim in this section was however to demonstrate robustness of Algorithm D.4.1 even for unexpected increase of noise, all simulations have therefore been run using $\varepsilon_0 = 28$. As seen in Table D.1 an increased level of noise does not necessarily mean a significant deterioration in convergence performance in terms of relative error decrease. However the error convergence displays a noisier behavior as measurement noise grows large and locally we may well have $e_{k+1} > e_k$ even though the general trend is a decreasing error.

Figure 7: **Sensitivity to Noise:** Lake 24, noise = 0%, $e_1 = 0.13$, $e_{100} = 0.06$, $e_{100}/e_1 = 0.52$.

Figure 8: **Sensitivity to Noise:** Lake 24, noise = 2.5%, $e_1 = 0.12$, $e_{100} = 0.06$, $e_{100}/e_1 = 0.51$.

Figure 9: **Sensitivity to Noise:** Lake 24, noise = 7.5%, $e_1 = 0.13$, $e_{100} = 0.07$, $e_{100}/e_1 = 0.50$.

Figure 10: **Sensitivity to Noise:** Lake 24, noise = 10%, $e_1 = 0.13$, $e_{100} = 0.07$, $e_{100}/e_1 = 0.54$.

noise	e_1	e_{100}	e_{100}/e_1	$k : e_k/e_1 \leq 0.75$
0	0.0949	0.0541	0.5575	7
2.5	0.0955	0.0543	0.5580	8
5	0.0968	0.0559	0.5699	7
7.5	0.0978	0.0579	0.5864	8
10	0.1038	0.0640	0.6214	8
12.5	0.1025	0.0666	0.6505	8
15	0.1107	0.0649	0.5816	6
17.5	0.1167	0.0687	0.5943	8
20	0.1227	0.0770	0.6441	10

Table D.1: Convergence results for simulated data. Mean initial and final errors for the 25 test contours, mean over all error decrease and mean number of iterations k needed to obtain a 25% error decrease.

Impact of choice of ε_0 Studying for instance Equation (D.20) it is apparent that a very small value of ε_k may yield an ill conditioned problem. If a decreasing ε is used, the starting value should be chosen so as to not only reflect the fidelity of the sensor data but also match the number of intended iterations. Simulations were run with a noise level of 5% for all 25 test cases with starting values of ε_0 ranging from 1 to 100. One example is shown in Figures 12 - 14 for $\varepsilon_0 = 10, 50$ and 100. Results for all 25 cases are summarized in Figure 11. The left plot shows convergence rate as a function of ε_0 and the right one shows initial and final errors. The largest mean error decrease is 47% and occurs for $\varepsilon_0 = 66$. Generally a smaller value of ε_k yields an error convergence trend closer to the expected rate $1/\sqrt{k}$. This is reasonable since decreasing the weight of the stochastic part of the cost function makes the process more deterministic. The price for this reliable convergence trend may be larger total errors and a slower over all convergence rate since only a small amount of new information is added to the problem at each iteration. Again we stress that ε_0 is a design parameter that should be chosen with respect to information on sensor quality and the intended application.

Figure 11: **Impact of choice of ε_0** : Mean error convergence results for the 25 lake contours for $\varepsilon_0 = 1, \dots, 100$.

D.5.4.2 Experimental Data

Algorithm D.4.1 was further evaluated using real data. In this section we provide convergence results for a square contour and a "shamrock" contour constructed from three circular objects. The test contours were chosen because they present challenges such as sharp edges and narrow corners. 15 data sets were used for each test object. Convergence results are provided in Figures 15 and 16. Figures 17 and 18 show initial and final error and convergence rates for ε_0 ranging from 1 to 100. For $\varepsilon_0 \in [5, 15]$, the error decrease is about 40% in 15 iterations. For contours with sharp edges and narrow corners, the overall error depends highly on whether you get good measurements close to those difficult areas or not. This might explain why the over all initial and final errors are larger for the more complex "shamrock" contour as is seen from Figures 17 and 18.

Studying Figures 17 and 18 we notice that for a large ε_0 there is hardly any error decrease between iteration 1 and 15 - the initial result is basically as accurate as the final one. The SICK scanner produces rather accurate data, and the solution benefits from giving the data high credibility. If this information on the quality of the data sets is available beforehand, one may choose a large ε_0 and perform only one or a few iterations. However, if we have no prior knowledge of the expected noise level, we should use a smaller value of ε_0 and increase the number of iterations. As mentioned in Section D.5.4.1, with a small ε_0 simulations indicate slower but reliable convergence trend for noisy as well as accurate data sets.

Figure 12: **Impact of choice of ε_0 :** Lake 23, convergence results for $\varepsilon_0 = 10$. Figure 13: **Impact of choice of ε_0 :** Lake 23, convergence results for $\varepsilon_0 = 50$. Figure 14: **Impact of choice of ε_0 :** Lake 23, convergence results for $\varepsilon_0 = 100$.

Figure 15: **Experimental Data:** Error convergence of Algorithm D.4.1 for the square object.

Figure 16: **Experimental Data:** Error convergence of Algorithm D.4.1 for the "shamrock" object.

D.6 Conclusions

In this paper we investigated a recursive smoothing spline approach for contour reconstruction. We derived periodic smoothing splines recursively from noisy data by solving an optimal control problem for a linear system. The methods discussed in this paper have previously been mathematically analyzed, see for instance [16]. The main contributions of this paper are the derivation of a discrete version of the smoothing spline problem both for the closed form and recursive approach, and experimental evaluations of the algorithm. We prove that the resulting QP is strictly convex and therefore always has a unique solution. Simulations and experiments show the convergence of the recursive approach and the applicability of the scheme in practice.

Future research comprises a convergence analysis for the recursive method and possibly convergence criteria. We see many possible extensions and new directions for this work, for instance combining sensors in a grid to construct surface estimations or integrating the smoothing spline

Figure 17: **Experimental Data:** Error convergence of Algorithm D.4.1 for the square object for $\epsilon_0=1, \dots, 100$.

Figure 18: **Experimental Data:** Error convergence of Algorithm D.4.1 for the "shamrock" object for $\epsilon_0 = 1, \dots, 100$.

approach in existing SLAM algorithms to manage path planning using sparse data sets. Ideally, to make Algorithm D.4.1 truly automatic, it should include a way of determining the magnitude of ϵ based on the input data without knowledge of the contour. For regular smoothing splines such methods are presented in for instance [28] and [31]. Ongoing work includes adapting such a method to Problem D.3.1 and Problem D.3.2.

D.7 References

- [1] A. E. Bryson and Y. Ho, Applied Optimal Control, Optimization estimation and control, *Halsted Press*, 1975.
- [2] J. A. Castellanos and J. D. Tardós, Mobile Robot Localization and Map Building: A Multi-sensor Fusion Approach, *Kluwer Academic Publishers*, 1999.
- [3] A. J. Davison, Real-Time Simultaneous Localisation and Mapping with a Single Camera, *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2003.
- [4] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Corba, A Solution to the SLAM Building Problem, *IEEE TRA*, vol. 17 no. 3, pp. 229–241, 2001.
- [5] M. Egerstedt, C.F. Martin, and S. Sun, Optimal control, statistics and path planning, *Math. Comput. Modelling*, vol. 33, pp. 237–253, 2001.
- [6] M. Egerstedt and C.F. Martin, Statistical estimates for generalized splines, *ESAIM:Control, Optimisation and Calculus of Variations*, vol. 9, pp. 553–562, 2003.
- [7] A. Ferrante, G. Marro, and L. Ntogramatzidis, A parametrization of the solutions of the finite-horizon LQ problem with general cost and boundary condition, *Automatica*, vol. 41, no. 8, pp. 1359–1366, 2005.
- [8] J. Folkesson, P. Jensfelt, and H.I. Christensen, Graphical SLAM using Vision and the Measurement Subspace, *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005.
- [9] R. Frezza and G. Picci, On Line Path Following by Recursive Spline Updating, *Proc. of the 34th Conference of Decision and Control (CDC)*, pp. 367–393, 1995.
- [10] R. Frezza, S. Soatto, and G. Picci., Visual Path Planning by Recursive Spline Updating, *Proc. of the 36th Conference of Decision and Control (CDC)*, pp. 367–393, 1997.
- [11] L. Goncavles, E. di Bernardo, D. Benson, M. Svedman, J. Ostrovski, N. Karlsson, and P. Pirjanian, A visual fron-end for simultaneous localization and mapping, *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 44–49, 2005.
- [12] S. Isotani, A. de Albuquerque, M. Muratore, and N. Brasil Filho, A recursive spline-based algorithm for sensor calibration design, *Industrial Electronics, Control and Instrumentation*, vol. 3, pp. 1952–1954, 1994.
- [13] A. Jazwinski, Stochastic Processes and Filtering Theory, *Academic Press*, 1970.
- [14] H. Kano, H. Fujioka, M. Egerstedt, and C.F. Martin, Optimal smoothing spline curves and contour synthesis, *Proc. of the 16th Congress of the International Federation of Automatic Control (IFAC)*, 2005.
- [15] H. Kano, M. Egerstedt, H. Fujioka, S. Takahashi, and C.F. Martin, Periodic Smoothing Splines, *Automatica*, vol. 44, pp. 185–192, 2008.
- [16] M. Karasalo, X. Hu, and C.F. Martin, Localization and Mapping using Recursive Smoothing Splines, *Proc. of the European Control Conference (ECC)*, 2007.
- [17] M. Karasalo, X. Hu, and C.F. Martin, Contour Reconstruction and Matching using Recursive Smoothing Splines, in *Modeling, Estimation and Control*, Springer, pp. 193–206, 2007.

- [18] C.T Leondes, *Advances in Control Systems*, vol 16, *Academic Press*, 1965.
- [19] C.F. Martin and J. Smith, Approximation, interpolation and sampling, *Differential geometry, the interface between pure and applied mathematics, Contemporary Mathematics*, vol. 68, pp. 227–252, 1987.
- [20] C.F. Martin, S. Sun, and M. Egerstedt, Control theoretic smoothing splines, *IEEE Transactions on Automatic Control*, vol. 45, pp. 2271–2279, 2000.
- [21] S.M. Naik and R. Jain, Spline-based surface fitting on range images for CAD applications, *Proc. of Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 249–253, 1988.
- [22] P. Newman and K. Ho, SLAM-Loop Closing with Visually Salient Features, *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 644–651, 2005.
- [23] H. Perez and S. Devasia, Optimal output-transitions for linear systems, *Automatica*, vol. 39, no. 2, pp. 181–192, 2003.
- [24] G. Piccolo, M. Karasalo, D. Kragic, and X. Hu, Contour reconstruction using recursive smoothing splines - experimental validation, *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2071–2076, 2007.
- [25] D. Rueckert and P. Burger, Contour fitting using an adaptive spline model, *Proc. of the 1995 British conference on Machine vision*, pp. 207–216, 1995.
- [26] S. Thrun, D. Fox, and W. Burgard, A probabilistic approach to concurrent mapping and localization for mobile robots, *Autonomous Robots*, vol. 5, pp. 253–271, 1998.
- [27] Y. Zhou, W. Dayawansa, and C.F. Martin, Control theoretic smoothing splines are approximate linear filters, *Communications in Information and Systems*, vol. 4, pp. 253–272, 2004.
- [28] P. Craven and G. Wahba, Smoothing Noisy Data With Spline Functions, *Numerische Mathematik*, vol. 31, pp. 377–403, 1979.
- [29] G. Wahba, Spline Models for Observational Data, *CBMS-NSF Series, SIAM*, 1990.
- [30] R.L. Eubank, Nonparametric Regression and Spline Smoothing, 2nd ed., *CRC Press*, 1999.
- [31] C.H. Reinsch, Smoothing by spline functions, *Numerische Mathematik*, vol. 10, pp. 177–183, 1967.
- [32] M. Breen and T. Zamfirescu, A characterization theorem for certain unions of two starshaped sets in $\mathbb{R}^2$, *Journal Geometriae Dedicata, Springer Netherlands*, vol. 22, no. 1, pp. 95–103, 1987.
- [33] D. Hähnel, S. Thrun, B. Wegbreit, and W. Burgard, Towards Lazy Data Association in SLAM, *Proc. of the International Symposium on Robotics Research (ISRR)*, 2003.
- [34] M. Karasalo, X. Hu, L. Johansson, and K. Johansson, Multi-Robot Terrain Servoing with Proximity Sensors, *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 367–393, 2005.
- [35] M. Egerstedt and C.F. Martin, Statistical estimates for generalized splines, *ESAIM:Control, Optimisation and Calculus of Variations*, vol. 9, pp. 553–562, 2003.