2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

A B-spline Mapping Framework for Long-Term Autonomous Operations

Rômulo T. Rodrigues[1], A. Pedro Aguiar[1], and António Pascoal[2]

***Abstract*— This paper presents a 2D B-spline mapping framework for representing unstructured environments in a compact manner. While occupancy-grid and landmark-based maps have been successfully employed by the robotics community in indoor scenarios, outdoor long-term autonomous operations require a more compact representation of the environment. This work tackles this problem by interpolating the data of a high frequency sensor using B-spline curves. Compared to lines and circles, splines are more powerful in the sense that they allow for the description of more complex shapes in the scene. In this work, spline curves are continuously tracked and aligned across multiple sensor readings using lightweight methods, making the proposed framework suitable for robot navigation in outdoor missions. In particular, a Simultaneous Localization and Mapping (SLAM) algorithm specifically tailored for B-spline maps is presented here. The efficacy of the proposed framework is demonstrated by Software-in-the-Loop (SiL) simulations in different scenarios.**

I. Introduction

Within the context of autonomous mobile robots, there are different tasks for which a model of the environment is required or beneficial: localization, motion planning, exploration, robot coordination, and human-robot interaction. Usually, this implies that to successfully accomplish a goal, a robot must build and maintain a map of the environment using the information gathered by its on-board sensors.

The IEEE Standard for robot data representation [1] classifies 2D maps in two major categories: topological and metric. While topological maps describe a connectivity relationship, which are often represented by edges and nodes of a graph, metric maps have a physical or geometric meaning, making it possible to compute a metric distance between two elements in the map. This paper focuses on metric maps, which can be further categorized into landmark, occupancy, and geometric representations [2], which are illustrated in Fig 1.

Landmark-based approaches employ artificial landmarks uniquely identified or salient features extracted from the environment, e.g., corners. Theoretical tools that deal with the probabilistic uncertainty associated with landmarks have been developed over the years, making landmark-based SLAM a mature technique by now [3]. See for instance the seminal book by Thrun et al. [4] that covers landmark-based SLAM algorithms using Extended Kalman Filters and Rao-Blackwellized particle filters. Also, since landmark maps are a compact representation of the environment, the model scales well for large environments and model-related data exchange among robots is relatively efficient. However, proper landmark-based localization requires a densely covered environment. The same applies for the execution of other tasks, including motion planning and exploration, which cannot rely on a sparse landmark map.

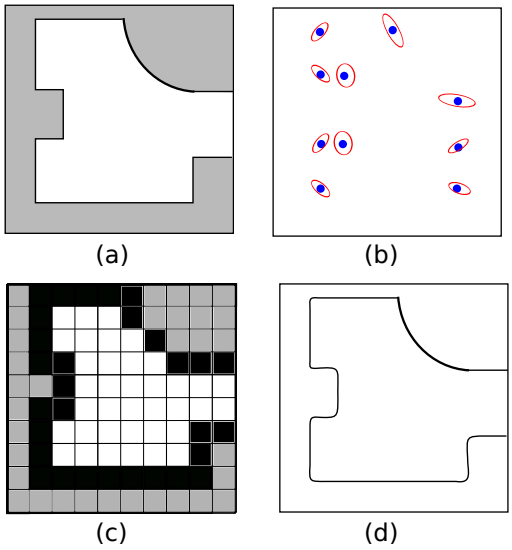

Fig. 1. Different representations of the environment presented in (a): landmark-based map (b), occupancy grid map (c), and geometric map (d).

Early work on occupancy grid maps was done by Moravec and Elfes [5] for robot motion planning and navigation in cluttered environments. The occupancy grid representation consists in decomposing a map into small cells, which are assigned as occupied or free. Multiple readings reinforce the state of the cells as empty or occupied. The occupancy grid maps provide an accurate description of the environment, being well suited for exploration [6] and SLAM [7], [8]. The major drawbacks of this method are: 1) it does not scale well to large environments and 2) it does not model explicitly the spatial constraints required for motion planning.

Geometric maps describe the environment using basic geometric primitives, e.g., lines or polygons. It is a compact representation of the environment that scales well for large scenarios and avoids discretization problems, offering floating point resolution [9]. Crowley [10] proposed one of the first successful geometric representation based on line segments. In addition to lines, Vandorpe et al. [11] param-

This work was supported by projects IMPROVE - POCI-01-0145-FEDER-031823 - funded by FEDER funds through COMPETE2020 - Programa Operacional Competitividade e Internacionalização (POCI) and by national funds (PIDDAC) through FCT/MCTES; PDMA - NORTE-08-5369-FSE-000061, through Programa Operacional Regional do Norte (NORTE 2020); and the Portuguese FCT Project UID/EEA/5009/2013.

[1] R. T. Rodrigues and A. P. Aguiar are with the Faculty of Engineering, University of Porto, Portugal. E-Mail: rtr@fc.up.pt, pedro.aguiar@fe.up.pt

[2] A. Pascoal is with the Laboratory of Robotics and Engineering Systems (LARSyS), ISR/IST, University of Lisbon, Portugal. E-Mail: antonio@isr.tecnico.ulisboa.pt

eterized a map using circles. Wolter et al. [12] proposed a representation that uses polygonal curves and addressed a matching methodology to find previously mapped curves. As pointed in [2], merging geometric primitives is the main challenge with geometric approaches.

The work by Pedraza et al. [13] is a hybrid of geometric and landmark based approaches. First, a B-spline interpolates the sensor data. Then, the control points of the spline curve are individually tracked using an EKF. Later, Liu et al. [14] extended the SLAM framework to take the covariance error of each control point into account.

Selecting the best map representation depends on the specific mission to be accomplished. In particular, for long-term autonomy operations over large areas, e.g., ocean exploration, storing a map becomes cumbersome due to the limited amount of onboard memory. This motivates new map representations which are memory efficient. This paper proposes a geometric 2D B-spline based map that represents sparse environments in a compact fashion. The method is well suit for robotic systems equipped with a high frequency sensor such as laser range finders (aerial and ground robots) or a multibeam sonar (underwater vehicles). The proposed B-spline framework may support different navigation tasks. In particular, a curve alignment based SLAM is discussed. The supplemental video that accompanies this work illustrates the proposed method in different scenarios. Our work differs greatly from [13]. Namely, because it is a pure geometric map approach and the map update relies on curve alignment, rather than control point tracking. Thus, it is less likely to suffer from wrong data association.

This paper is organized as follows. Sec. II introduces the most relevant B-spline concepts that support the present work. Sec. III presents the proposed mapping framework. Sec. IV brings forward a SLAM application that uses the proposed B-spline map. Results are discussed in Sec. V. Finally, Sec. VI addresses final remarks and open challenges.

II. B-SPLINES CURVES

A B-spline is a compact, yet powerful tool that allows for the description of complex geometric shapes. A 2D B-spline curve of degree d, n-dimensional knot vector $\mathbf{t} = (t_i)_{i=1}^{n}$, and control points $C = (\mathbf{c}_i)_{i=1}^{n-d-1}$, where $\mathbf{c}_i = (c_{i,x}, c_{i,y})^T$, is given by

$$\mathbf{s}(\tau) = \sum_i \mathbf{c}_i B_{i,d}(\tau) = C\mathbf{B}_d(\tau)^T,$$

where $\tau \in \mathbb{R}$ is the spline parametric variable, $\mathbf{B}_d(\tau) = (B_{i,d})_{i=1}^{n-d-1}$ and $B_{i,d}(\tau)$ is the B-spline function defined as:

$$\begin{aligned} B_{j,r}(\tau) &= \frac{\tau - t_j}{t_{j+r} - t_j} B_{j,r-1}(\tau) + \frac{t_{j+r+1} - \tau}{t_{j+r+1} - t_j} B_{j+1,r-1}(\tau) \\ B_{j,0}(\tau) &= \begin{cases} 1, & t_j \le \tau < t_{j+1} \\ 0, & \text{otherwise} \end{cases} \end{aligned} . \quad (1)$$

Henceforth, unless otherwise stated, the subscript that indicates the degree of the spline function is omitted. Also, the notations $B_i(\tau|\mathbf{t}), B_i(\tau)$, and B_i are employed interchangeable. The first shows explicitly the dependency on the knot vector $\mathbf{t}$. Similar comments apply to $\mathbf{s}(\tau|\mathbf{t}), \mathbf{s}(\tau), \mathbf{s}$.

There are different knot vector structures: clamped or non-clamped and uniform or non-uniform. This work employs non-clamped uniform knot vectors. Each element of the non-clamped uniform knot vector $\mathbf{t}(n, I, \Delta)$ is given by

$$t_i = (i-1)\Delta + I\Delta, \quad \text{for } i = 1, \ldots, n \ ,$$

where $\Delta \in \mathbb{R}^+$ is the uniform step between consecutive knots and $I \in \mathbb{Z}$ is an offset. In what follows, the basic properties of B-splines required in this paper are introduced. For more details and formal proofs, the reader is referred to [15], [16].

Property 1: (Local knots) The B-spline function $B_{\mu,d}$ depends exclusively on the knots $(t_i)_{i=\mu}^{\mu+d+1}$. The proof follows by inspection of (1). This property implies that $\mathbf{s}(\tau)$ has $n-d-1$ control points.

Property 2: (Local support) The local knot property implies that if $\tau \in [t_\mu, t_{\mu+1})$, then

$$\mathbf{s}(\tau) = \sum_{i=\mu-d}^{\mu} \mathbf{c}_i B_i(\tau) = C_\mu \mathbf{B}_\mu(\tau),$$

where $C_\mu = (\mathbf{c}_{\mu-d}, \ldots, \mathbf{c}_\mu)$ and $\mathbf{B}_\mu = (B_{\mu-d}, \ldots, B_\mu)^T$. Therefore, a spline curve is only valid for $\tau \in [t_{d+1}, t_{n-d-1})$.

Property 3: (Convex combination) From the recursive relation (1), for any valid τ (see local support property), the non-null B-spline functions are positive and add up to 1, i.e., if $\tau \in [t_\mu, t_{\mu+1})$, then $\sum_{i=\mu-d}^{\mu} B_i(\tau) = 1$.

Property 4: (Weighted sum) Consider $\mathbf{s}_a(\tau|\mathbf{t}_a)$ and $\mathbf{s}_b(\tau|\mathbf{t}_b)$, two B-spline curves of degree d, where $\mathbf{t}_a(n_a, I_a, \Delta)$ and $\mathbf{t}_b(n_b, I_b, \Delta)$. The control points of the weighted sum curve, denoted as $\mathbf{s}_{ws}(\tau|\mathbf{t}_{ws})$, are the weighted sum of the control points of $\mathbf{s}_b$ and $\mathbf{s}_a$ associated with the B-spline $B(\tau|\mathbf{t}_{ws})$.

Proof: The knot vector $\mathbf{t}_{ws}(n_{ws}, I_{ws}, \Delta) = \cap(\mathbf{t}_a, \mathbf{t}_b)$ is defined by $I_{ws} = d + \max(I_a, I_b)$ and $n_{ws} = \min(n_a + I_a, n_b + I_b) - I_{ws} - d$. Given the weights $\alpha_a, \alpha_b \in \mathbb{R}$, it follows that

$$\begin{aligned} \mathbf{s}_{ws}(\tau) &= \alpha_a \sum_{i=I_{ws}-I_a+1}^{I_{ws}-I_a+n_{ws}} \mathbf{c}_{a,i} B_i(\tau|\mathbf{t}_a) + \alpha_b \sum_{i=I_{ws}-I_b+1}^{I_{ws}-I_b+n_{ws}} \mathbf{c}_{b,i} B_i(\tau|\mathbf{t}_b) \\ &= \sum_i (\alpha_a \mathbf{c}_{a,i+(I_{ws}-I_a)} + \alpha_b \mathbf{c}_{b,i+(I_{ws}-I_b)}) B_i(\tau|\mathbf{t}_{ws}) \\ &= \sum_i \mathbf{c}_{ws,i} B_i(\tau) \end{aligned}$$

■

III. B-SPLINE MAP

This section presents the proposed B-spline mapping framework. Let $\{M\}$ and $\{B\}$ be an inertial coordinate frame attached to the origin of a map and a body-fixed frame attached to a robot, respectively. Without loss of generality, it is assumed a sensor (e.g., a LIDAR unit) that provides input data is placed at the origin of $\{B\}$. Vectors described in $\{B\}$ are typed with a lagging superscript, e.g., ${}^B\mathbf{x}$. The rigid body transformation from $\{B\}$ to $\{M\}$ is parameterized by $\xi = (x, y, \theta)^T$.

The proposed framework represents the environment using multiple B-spline curves described in $\{M\}$. For the sake of

simplicity, in the remainder of the section it is assumed that ξ is known a priori. Later, Sec. IV introduces a SLAM method to estimate ξ as the mission unfolds. The mapping algorithm, described in Algorithm 1, breaks down into four tasks:

1) Point cloud segmentation: divides the point cloud obtained with a specific sensor into clusters and removes outliers;
2) Data association: tracks a cluster between consecutive readings;
3) Spline interpolation: Interpolates the discrete data using a B-spline curve.
4) Map update: Updates the prior map, which includes extending and removing curve segments.

Algorithm 1 Map building algorithm

Input: Point cloud $({}^B\mathbf{p}_i)_{i=1}^s$
Output: Mapped curves $\mathbf{s}_k^m$
Initialization: ξ, $\bar{\tau}_i^k$, and $\mathbf{s}_k^m$
Given parameters: d, Δ, γ_{rel}, γ_{abs}, γ_{min}, γ_{track} γ_{weight}
1: Transform point cloud to map frame (2)
2: Divide point cloud into clusters (3)
for each cluster **do**
  3.1: Compute the centroid (4)
  3.2: Track cluster between readings (5)
  Solve the fitting problem (6)
  if track succeeded **then**
    4.1.1: Solve the curve alignment problem (9),(11)
    4.1.2: Merge interpolated curve and mapped curve (13) and Table I.
  else
    4.2.1 Initialize curve in the map (12)
  end if
end for

A. Point cloud segmentation

Assume that a high frequency sensor system (e.g. LIDAR-based) provides at the discrete time instant k a 2D point cloud $({}^B\mathbf{p}_i)_{i=1}^s$. The transformation

$$T(\xi,\mathbf{p}_i)=\begin{pmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{pmatrix}\begin{pmatrix}p_{i,x}\\ p_{i,y}\end{pmatrix}+\begin{pmatrix}x\\ y\end{pmatrix} \tag{2}$$

transforms the point cloud to the map frame, i.e., $\mathbf{p}_i = T(\xi,{}^B\mathbf{p}_i)$. The point cloud is divided into clusters or blobs. Points that belong to the same blob represents the same continuous geometric feature in the scene, e.g., a corridor. The i-th data point is classified in the same cluster as that of its preceding neighbour if

$$\begin{aligned}&\|\mathbf{p}_i-\mathbf{p}_{i-1}\|\le\gamma_{rel}\|\mathbf{p}_{i-1}-\mathbf{p}_{i-2}\|,\\ &\|\mathbf{p}_i-\mathbf{p}_{i-1}\|\le\gamma_{abs},\end{aligned} \tag{3}$$

where $\gamma_{rel},\gamma_{abs}\in\mathbb{R}^+$. While γ_{rel} imposes the maximum relative distance between three points, γ_{abs} enforces that two consecutive points are not far apart. In addition, the parameter $\gamma_{min}\in\mathbb{N}$ is introduced, to represent the minimum number of elements a cluster must have in order not be discarded.

B. Data association

The assumption that the sensor sampling rate is sufficiently high allows us to assume that between consecutive readings the snapshot of the environment is almost similar. Thus, blobs can be tracked across multiple readings. In the proposed framework, tracking is devised based on the distance between the centroid of the clusters at instant k and $k-1$. Thus, for each cluster we compute the centroid

$$\bar{\mathbf{p}}_i^k=\frac{1}{m}\sum_j^m\mathbf{p}_{i,j}\quad, \tag{4}$$

where m is the number of points of the i-th cluster. The tracking is said to succeed if

$$\min_j\|\bar{\mathbf{p}}_i^k-\bar{\mathbf{p}}_j^{k-1}\|_2\le\gamma_{track}. \tag{5}$$

where $\gamma_{track}\in\mathbb{R}^+$ is the acceptable distance between two clusters in consecutive sensor readings. If tracking succeeds, it means that the mapped curve $\mathbf{s}_i^m$ (the leading superscript m stands for map) that represents the same geometric feature as the tracked blob has been detected.

C. Spline fitting

The data points in each cluster are interpolated by a B-spline curve. Also known as spline approximation, spline interpolation can be categorized into local or global methods. Local approximation methods such as piecewise linear interpolant and cubic Hermite interpolant preserve well the shape of the data. In contrast, global approximation methods relax the constraint that the curve must pass at each point being interpolated, yielding smoother curves. The reason why we use global interpolation is three-fold: *1)* the number of control points and knot vectors per sensor reading is smaller, *2)* it naturally filters out the sensor noise, and *3)* the map update method requires that the same geometric feature yields similar knot vectors across consecutive readings, which can not be directly obtained using local interpolants due the discrete nature of the input data.

The interpolation problem consists in computing the control points of the curve $\mathbf{s}_i^k(\tau_j|\mathbf{t}_i^k)$, which best fit the m data points in the i-th cluster. The problem can be formulated as

$$\min_{C_i^k}\sum_{j=1}^m\|C_i^k\mathbf{B}(\tau_j)^T-\mathbf{p}_j\|^2, \tag{6}$$

where the spline parametric variable is computed using the cord length parameterization, that is,

$$\begin{aligned}&\tau_j=\tau_{j-1}+\|\mathbf{p}_{i,j}-\mathbf{p}_{i,j-1}\|_2, \qquad j=2,\dots,m\\ &\tau_1=\bar{\tau}_i^k,\end{aligned} \tag{7}$$

and $\bar{\tau}_i^k$ is an offset applied to the parametric variable, which will be discussed further in this Section. The non-clamped uniform knot vector $\mathbf{t}_i^k(n_i^k,I_i^k,\Delta)$ is

$$\mathbf{t}_i^k=(-d\Delta+\lfloor\frac{\tau_1}{\Delta}\rfloor\Delta,\dots,\lceil\frac{\tau_m}{\Delta}\rceil\Delta+d\Delta),$$

where $\lfloor x\rfloor$ ($\lceil x\rceil$) is known as floor (ceiling) function and returns the greatest (least) integer less (greater) or equal

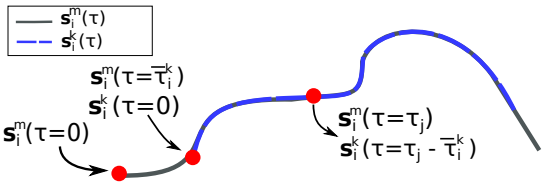

Fig. 2. Two overlapping curves that describe the same geometric feature.

to x. Alternatively, $\mathbf{t}_i^k$ is specified by $I_i^k = \lfloor \frac{\tau_1}{\Delta} \rfloor - d$ and $n_i^k = \lceil \frac{\tau_m}{\Delta} \rceil - \lfloor \frac{\tau_1}{\Delta} \rfloor + 2d + 1$.

The solution for (6) that minimizes the error in a least square sense is $C_i^k = (P^T \bar{B}^\dagger)^T$, where

$$P = (\mathbf{p}_{i,1}, \ldots, \mathbf{p}_{i,m}),$$
$$\bar{B} = \begin{pmatrix} B_1(\tau_1) & \ldots & B_{n_i^k-d-1}(\tau_1) \\ \vdots & \ddots & \vdots \\ B_1(\tau_m) & \ldots & B_{n_i^k-d-1}(\tau_m) \end{pmatrix}, \tag{8}$$

and $\bar{B}^\dagger = (\bar{B}^T \bar{B})^{-1} \bar{B}^T$ is the left pseudo-inverse of $\bar{B}$.

If the cluster being considered does not correspond to any cluster in the previous sensor reading, i.e., tracking has failed, then $\bar{\tau}_i^k = 0$ and the interpolation tasks ends here. However, if track has succeed, the additional constraint $\mathbf{s}_i^k(\tau) = \mathbf{s}_i^m(\tau), \forall \tau \in \cap(\mathbf{t}_i^m, \mathbf{t}_i^k)$ must also hold. This constraint ensures that both curves yield the same value when evaluated in the region both overlap. As illustrated in Fig. 2, this is accomplished by introducing the offset $\bar{\tau}_i^k$ that minimizes the curve alignment error defined as

$$\frac{1}{2} \sum_{j=1}^{r} \| \mathbf{s}_i^m(\tau_j | \mathbf{t}_i^m) - \mathbf{s}_i^k(\tau_j | \mathbf{t}_i^k) \|^2, \tag{9}$$

for $\tau \in \cap(\mathbf{t}_i^m, \mathbf{t}_i^k)$. Notice that r can take any value, as the continuous curve rather than the data points is employed. Simulations show that a small r suffices. Assume that an initial guess for $\bar{\tau}_i^k$, which is available from the previous iteration, feeds (7). The problem can be posed as finding $\Delta\bar{\tau}_i^k$ that minimizes the error

$$\frac{1}{2} \sum_{j=1}^{r} \| \mathbf{s}_i^m(\tau_j) - \mathbf{s}_i^k(\tau_j + \Delta\bar{\tau}_i^k) \|^2.$$

The first order Taylor expansion of $\mathbf{s}_i^k(\cdot)$ yields

$$\frac{1}{2} \sum_{j=1}^{r} \| \mathbf{s}_i^m(\tau_j) - \mathbf{s}_i^k(\tau_j) - \frac{\mathbf{s}_i^k(\tau_j)}{d\tau} \Delta\bar{\tau}_i^k \|^2 \tag{10}$$

The solution for $\Delta\bar{\tau}_i^k$ is computed using the Gauss-Newton method by taking the derivative of (10) with respect to $\Delta\bar{\tau}_i^k$ and setting it to zero, yielding

$$\Delta\bar{\tau}_i^k = -(H_i^k)^{-1} \sum_{j=1}^{r} [\frac{\mathbf{s}_i^k(\tau_j)}{d\tau}]^T [\mathbf{s}_i^m(\tau_j) - \mathbf{s}_i^k(\tau_j)], \tag{11}$$

where $H_i^k = \sum_{j=1}^{r} [\frac{\mathbf{s}_i^k(\tau_j)}{d\tau}]^T [\frac{\mathbf{s}_i^k(\tau_j)}{d\tau}]$. Finally, the interpolation problem (6) is solved once again to re-compute $\mathbf{s}_i^k$, $\mathbf{t}_i^k$, and $\mathbf{C}_i^k$, this time letting $\tau_1 = \bar{\tau}_i^k + \Delta\bar{\tau}_i^k$.

TABLE I

CONTROL POINT UPDATING SCHEME FOR MERGING CURVES

$\mathbf{c}_{new,j}^m$	$\alpha_{new,j}^m$	Condition	Interval
$\mathbf{c}_{i,j}^m$	$\alpha_{i,j}^m$	if $I_i^m \le I_i^k$	$1 \le j \le u_1$
$\mathbf{c}_{i,j}^k$	1	otherwise	
$\mathbf{c}_{i,j-u_1}^{ws}$	$\min(\gamma_{weight}, \alpha_{i,j-u_1+I_{ws}-I_m}^m + 1)$	-	$u_1+1 \le j \le u_2$
$\mathbf{c}_{i,j+u_4-I_i^m}^m$	$\alpha_{i,j}^m$	if $n_i^m + I_i^m \ge n_i^k + I_i^k$	$u_2+1 \le j \le u_3$
$\mathbf{c}_{i,j+u_4-I_i^k}^k$	1	otherwise	

D. Map update

The final step consists in storing or updating the detected curve, that is, the control points and the knot vector. Notice that it is not necessary to keep the knot vector $\mathbf{t}(n, I, \Delta)$ in memory, but only I and n. For a given map, Δ is assumed to be a known constant.

If tracking has failed, the new curve $\mathbf{s}_{new}^m = \mathbf{s}_i^k$ is initialized in the map by storing

$$\mathbf{C}_{new}^m = \mathbf{C}_i^k \quad I_{new}^m = I_i^k \quad n_{new}^m = n_i^k$$
$$\alpha_{i,j}^m = 1 \text{ , for } j = 1, \ldots, n_i^k, \tag{12}$$

where $\alpha_{i,j}^m$ is the weight associated to the j-th control point of $\mathbf{s}_i^m$. Similar to a cell in the occupancy grid approach, as a control point is re-observed, its weight increases until reaching the threshold γ_{weight}.

If tracking has succeeded, then the updated curve is a combination of the mapped curve $\mathbf{s}_i^m$ and its corresponding most recently interpolated curve $\mathbf{s}_i^k$. For the region in which the curves overlap, we compute $\mathbf{s}_i^{ws}(\tau | \mathbf{t}_i^{ws})$, the weighted sum curve, as defined in Sec. II. The control points of $\mathbf{s}_{ws}$ are given by

$$\mathbf{c}_{i,j}^{ws} = \frac{(\alpha_{i,j+I_i^{ws}-I_i^m}^m - 1)\mathbf{c}_{i,j+I_i^{ws}-I_i^m}^m + \mathbf{c}_{i,j+I_i^{ws}-I_i^k}^k}{\alpha_{i,j+I_i^{ws}-I_i^m}^m}$$

for $j = 1, \ldots, n_i^{ws} - d - 1$. The updated curve $\mathbf{s}_{new}^m$ is defined by the knot vector:

$$I_{new}^m = \min(I_i^m, I_i^k) \qquad n_{new}^m = n_i^m + n_i^k - n_i^{ws} - 2d, \tag{13}$$

and its control points are as described in Table I, where

$$\begin{aligned} u_1 &= I_i^{ws} - \min(I_i^m, I_i^k) \\ u_2 &= u_1 + n_i^{ws} - d - 1 \\ u_3 &= u_2 + \max(n_i^m + I_i^m, n_i^k + I_i^k) - \min(n_i^m + I_i^m, n_i^k + I_i^k) + d \\ &= \max(n_i^m + I_i^m, n_i^k + I_i^k) - \min(I_i^m, I_i^k) - d - 1 \\ u_4 &= -u_2 + I_i^{ws} + n_i^{ws} - d - 1 \\ &= \min(I_i^m, I_i^k) \end{aligned}$$

E. Discussion

The parameters d, Δ, γ_{rel}, γ_{abs}, γ_{min},γ_{track}, and γ_{weight} play a major a role on the performance and robustness of the proposed framework. The spline degree d must be at least 2, to ensure that the spline curve has continuous first order derivative, a constraint imposed by (11). The uniform knot step Δ determines the map resolution. A large value smooths

out the sensor noise and sharp geometric shapes, while small values preserves better the shape of the point cloud, but also leads to more oscillations. Furthermore, due to the nature of interpolation, Δ cannot be arbitrarily small. By choosing γ_{abs} smaller than Δ, one ensures that $\bar{B}$ in (8) has a pseudo-inverse and the interpolation problem (6) can be solved. As the value of Δ increases, γ_{rel} becomes crucial to correctly separate the data into clusters. A minimum number of elements per cluster is guaranteed by the threshold γ_{min}. The parameter γ_{track} relies on the assumption that the robot does not move between two consecutive sensor readings. A small value of γ_{track} increases the false negative tracking rejection, while a large value increases the false positive tracking success. The last parameter, γ_{weight}, determines the dynamic of the control points in the presence of noise and changes in the environment.

After several simulations, the following parameter setup is suggested: $d = 3$, $\gamma_{rel} = 3$, $\gamma_{abs} = .7\Delta$, $\gamma_{min} = 4(d+1)$, $\gamma_{track} = \Delta$, $\gamma_{weight} = 50$, and Δ is free.

IV. Simultaneous Localization and Mapping

The SLAM problem requires a robot to estimate the rigid body transformation ξ and to build a map of the environment simultaneously. The solution proposed in this section resorts to curve alignment using the B-spline representation discussed previously.

The algorithm pipeline is similar to the mapping framework presented in Sec. III. However, $\bar{\tau}_i^k$ and ξ must now be estimated. Thus, for every tracked curve, rather than solving (9), the following optimization problem is solved:

$$\min_{\xi,\bar{\tau}_i^k} \frac{1}{2}\sum_{j=1}^{r} \|\mathbf{s}_i^m(\tau_j) - T(\xi, {}^B\mathbf{s}_i^k(\tau_j))\|_2.$$

Notice that the point cloud is not transformed to $\{M\}$. Given an initial estimate $(\xi, \bar{\tau}_i^k)$, the problem consist in finding $(\Delta\xi, \Delta\bar{\tau}_i^k)$ that minimizes the quadratic error

$$\frac{1}{2}\sum_{j=1}^{r} \|\mathbf{s}_i^m(\tau_j) - T(\xi + \Delta\xi_i, {}^B\mathbf{s}_i^k(\tau_j + \Delta\bar{\tau}_i^k))]\|^2$$

Linearizing the above equation using first order Taylor expansion about $(\Delta\xi^T, \Delta\bar{\tau}_i^k) = (\mathbf{0}, 0)$ yields

$$\frac{1}{2}\sum_{j=1}^{r} \|\mathbf{s}_i^m(\tau_j) - T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) + \\ - \nabla_\xi T(\xi, {}^B\mathbf{s}_i^k(\tau_j))\Delta\xi_i - \nabla_\tau T(\xi, {}^B\mathbf{s}_i^k(\tau_j))\Delta\bar{\tau}_i^k\|^2 ,$$

where

$$\nabla_\xi T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) = \begin{pmatrix} 1 & 0 & -{}^B\mathbf{s}_x^k \sin\theta - {}^B\mathbf{s}_y^k \cos\theta \\ 0 & 1 & {}^B\mathbf{s}_x^k \cos\theta - {}^B\mathbf{s}_y^k \sin\theta \end{pmatrix}$$

$$\nabla_\tau T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) = R(\theta)\frac{\mathbf{s}_i^k(\tau_j)}{d\tau},$$

where $R(\theta)$ is the rotation matrix defined by the rigid body transformation ξ.

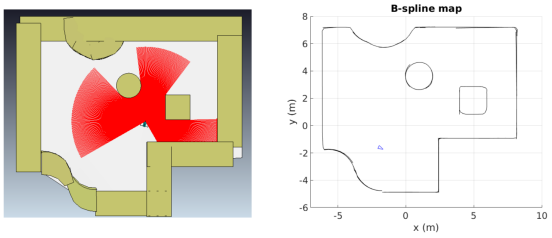

Fig. 3. Evaluation of the proposed mapping framework in an environment with different geometric shapes. (left) shows the V-REP simulation scenario and (right) the B-spline map presented in this work

The solution can be found using the Gauss-Newton method by matching the partial derivative with respect $\Delta\xi_i$ and $\Delta\bar{\tau}_i^k$ to zero, yielding

$$\begin{pmatrix} \Delta\xi_i \\ \Delta\bar{\tau}_i^k \end{pmatrix} = (H_i^k)^{-1} \sum_{j=1}^{m} \left[\begin{array}{c|c} \nabla_\xi T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) & \nabla_\tau T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) \end{array} \right]^T \\ [\mathbf{s}_i^m(\tau_j) - T(\xi, {}^B\mathbf{s}_i^k(\tau_j))],$$

where

$$H_i^k = \sum_{j=1}^{m} \left[\begin{array}{c|c} \nabla_\xi T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) & \nabla_\tau T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) \end{array} \right]^T \\ \left[\begin{array}{c|c} \nabla_\xi T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) & \nabla_\tau T(\xi, {}^B\mathbf{s}_i^k(\tau_j)) \end{array} \right]$$

For better convergence, different map resolutions are employed. This is a well-known technique in laser-based SLAM [7] and visual-SLAM [17]. In B-spline maps, the resolution is determined by the knot spacing Δ. First, the estimation is computed for the coarsest map, that is, the one corresponding to the largest knot spacing. Then, the solution obtained previously is used in the next map resolution.

V. Results

The proposed solution was evaluated using the V-REP[1] simulator of Coppelia Robotics. The software simulates a mobile robot equipped with a Hokuyo UST-10LX[2] laser rangefinder. Figure 3 shows the simulator environment and the proposed B-spline map using $\Delta = 0.2$ m. Storing a B-spline map requires 9 bytes per control point, 8 bytes per knot vector, and 16 bytes per mapped curve. An occupancy grid map usually requires 1 byte per cell. In particular, for the scenario shown in Fig. 3, the B-spline map is almost 10 times more compact than a similar 0.05 m grid map.

The performance of the spline-based SLAM algorithm is assessed in two different scenarios, as illustrated in Fig. 4. In the first scenario, the vehicle follows a wall, while translating in the horizontal plane. In the second scenario, the vehicle rotates around itself, while mapping the environment. For both scenarios, the map represented by the B-spline curves is similar to the ground truth. In each scenario, three map resolutions are employed: $\Delta_1 = 0.4$ m, $\Delta_2 = 0.2$ m, $\Delta_3 = 0.1$ m. Also, as discussed in Sec. IV, a multi-resolution approach that uses $\Delta_1, \Delta_2, \Delta_3$ is evaluated. The results for the

[1] www.coppeliarobotics.com
[2] www.hokuyo-aut.jp

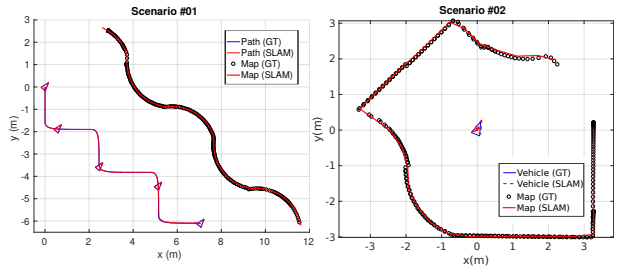

Fig. 4. Missions performed to evaluate the performance of the proposed B-spline SLAM. (left) vehicle moves in the horizontal plane and (right) vehicle rotates 360^o while standing still. GT stands for ground truth.

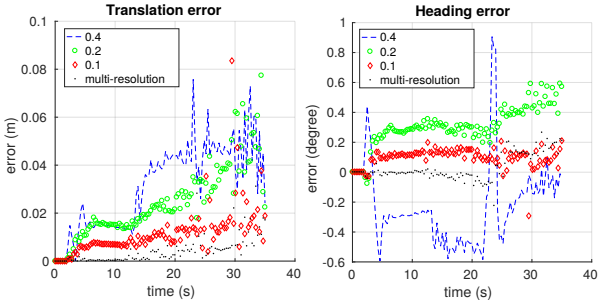

Fig. 5. Estimation error for different knot steps (.4 ,.2, .1) m for the first scenario shown in Fig. 4.

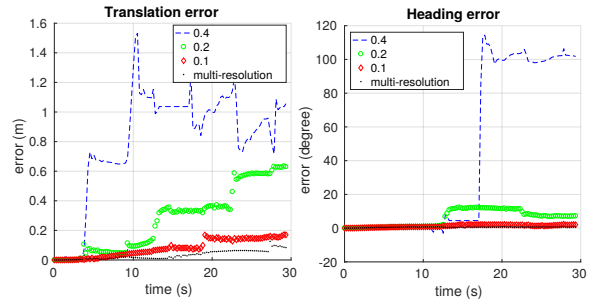

Fig. 6. Estimation error using different knot steps (.4, .2, .1) m for the second scenario shown in Fig. 4.

first scenario are shown in Fig. 5. Higher resolution maps lead to lower translation and orientation estimation errors. However, using the solution of lower resolution maps as a hot start improves the final solution. In the multi-resolution trial, the final translation error was 0.2% of the total length of the trajectory. Similar conclusions are drawn from the second scenario, shown in Fig. 6. Although the lowest resolution map ($\Delta_1 = 0.4$) failed to estimate the orientation, the multi-resolution approach converges to a good estimation. The orientation error after a $360^{o\cdot}$ spin is less then $1^{o\cdot}$.

VI. Conclusions

This work presented a B-spline mapping framework suitable for robot navigation. The framework relies on spline interpolation and the Gauss-Newton gradient method to align B-spline curves tracked across multiple sensor readings. The map is updated with short processing time using the basic properties of B-spline curves. Also, a simultaneous localization and mapping algorithm that exploits the advantages of the proposed framework was presented. The SLAM simulation results confirm that the method can be useful in navigation tasks. The results also show that the method is capable of representing the environment in a reliable fashion, while still keeping the representation compact. Thus, it has the potential to be applied in long-term autonomous operations, for which map scalability is a major concern. The authors are currently investigating loop-closure and matching schemes built upon B-spline properties. This would allow keeping the map clean and more robust against tracking failures.

References

[1] "IEEE standard for robot map data representation for navigation," *1873-2015 IEEE Standard for Robot Map Data Representation for Navigation*, pp. 1–54, Oct 2015.

[2] J. O. Wallgrn, *Hierarchical Voronoi Graphs: Spatial Representation and Reasoning for Mobile Robots*, 1st ed. Springer Publishing Company, Incorporated, 2009.

[3] 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, no. 6, pp. 1309–1332, Dec 2016.

[4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[5] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, Mar 1985, pp. 116–121.

[6] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2002, pp. 540–545 vol.1.

[7] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, Nov 2011, pp. 155–160.

[8] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1271–1278.

[9] D. Sack and W. Burgard, "A comparison of methods for line extraction from range data," in *In Proc. of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV*, 2004.

[10] J. L. Crowley, "World modeling and position estimation for a mobile robot using ultrasonic ranging," in *Proceedings, 1989 International Conference on Robotics and Automation*, May 1989, pp. 674–680 vol.2.

[11] J. Vandorpe, H. V. Brussel, and H. Xu, "Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2d range finder," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, Apr 1996, pp. 901–908 vol.1.

[12] D. Wolter, L. J. Latecki, R. Lakämper, and X. Sun, "Shape-based robot mapping," in *KI 2004: Advances in Artificial Intelligence*, S. Biundo, T. Frühwirth, and G. Palm, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 439–452.

[13] L. Pedraza, D. Rodriguez-Losada, F. Matia, G. Dissanayake, and J. V. Miro, "Extending the limits of feature-based slam with b-splines," *IEEE Transactions on Robotics*, vol. 25, no. 2, pp. 353–366, April 2009.

[14] M. Liu, S. Huang, G. Dissanayake, and S. Kodagoda, "Towards a consistent slam algorithm using b-splines to represent environments," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 2065–2070.

[15] T. Lyche and K. Mrken, "Spline methods," May 1998, draft version.

[16] C. de Boor, *A Practical Guide to Splines*. New York, NY: Springer-Verlag, 1978.

[17] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.