**Robotics: Science and Systems 2007**
**Atlanta, GA, USA, June 27-30, 2007**

# BS-SLAM: Shaping the World

Luis Pedraza*, Gamini Dissanayake†, Jaime Valls Miro†, Diego Rodriguez-Losada* and Fernando Matia*
*Universidad Politecnica de Madrid (UPM)
C/ Jose Gutierrez Abascal, 2. 28006, Madrid, Spain
Email: luis.pedraza@ieee.org
†Mechatronics and Intelligent Systems Group. University of Technology Sydney (UTS)
NSW2007, Australia

***Abstract*— This paper presents BS-SLAM, a simultaneous localization and mapping algorithm for use in unstructured environments that is effective regardless of whether features correspond to simple geometric primitives such as lines or not. The coordinates of the control points defining a set of B-splines are used to form a complete and compact description of the environment, thus making it feasible to use an extended Kalman filter based SLAM algorithm. The proposed method is the first known EKF-SLAM implementation capable of describing both straight and curve features in a parametric way. Appropriate observation equation that allows the exploitation of virtually all observations from a range sensor such as the ubiquitous laser range finder is developed. Efficient strategies for computing the relevant Jacobians, perform data association, initialization and expanding the map are presented. The effectiveness of the algorithms is demonstrated using experimental data.**

## I. Introduction

Developing an appropriate parameterization to represent the map is the key challenge for simultaneous localization and mapping in unstructured environments. While a substantial body of literature exists in methods for representing unstructured environments during robot mapping, most of these are unsuitable for use with the Kalman filter based simultaneous localization and mapping. For example, the use of popular occupancy grid approach [13], based on dividing the environment into small cells and classify these as occupied or unoccupied, and its variants such as those using quad-trees would result in an impractically large state vector.

In much of the early SLAM work the map is described by a set of points [6], [9], [11]. While this simplifies the formulation and the complexity of the SLAM estimator, there are two main disadvantages in relying solely on point features. The obvious problem arises if the environment does not have sufficient structure to be able to robustly extract point features, for example in an underground mine [12]. The more significant issue is the fact that much of the information acquired from a typical sensor, such as a laser range finder, does not correspond to point features in the environment. Therefore, the raw information from the sensor needs to be analysed and observations corresponding to stable point features extracted. During this process, typically more than 95% of the observations are discarded and the information contained in these observations wasted. One strategy to exploit this additional information is to model the environment using alternative geometric primitives such as line segments [3], [15] or polylines [18]. While this has been successful in many indoor environments, presence of curved elements can create significant problems. In particular, attempting to interpret information from a sensor using an incorrect model is one of the major causes of failure of many estimation algorithms. Some efforts have been made in the past when circle features are available [20], but that's still a big simplification. Thus a more generic representation of the environment can potentially improve the robustness of the SLAM implementations.

Alternative to using a specific environmental model is to use information from sensor observations from different robot poses to obtain an accurate relationship between these poses. While this strategy has been successfully used to generate accurate visualizations of complex structures [7] and detailed maps of indoor environments [8], it can not exploit the inherent information gain that occur in traditional SLAM due to the improvement of map quality.

In this paper, it is proposed to use B-splines to represent the boundary between occupied and unoccupied regions of a complex environment. B-splines provide naturally compact descriptions consisting of both lines and curves. Vertices of the control polygons that describe a set of B-splines are used to represent a complex environment by a state vector. Computationally efficient strategies for (a) initializing and extending the state vector when new parts of the environment are discovered, (b) formulating a suitable observation equation that allows the exploitation of virtually all the information gathered from a laser range finder, and (c) evaluation of appropriate Jacobians for easy implementation of Kalman filter equations are presented in detail. The algorithms proposed are evaluated for effectiveness using data gathered from real environments.

The paper is organized as follows. Section II introduces some general concepts and properties of B-spline curves. Section III shows how these powerful tools fit into the EKF-SLAM framework. Finally, some experimental results and conclusions are presented in sections IV and V.

## II. Splines Fundamentals

In this section, a brief introduction to the fundamental concepts of the B-splines theory is presented. The term *spline* is used to refer to a wide class of functions that are used in applications requiring interpolation or smoothing of data in a flexible and computationally efficient way. A spline of

degree $k$ (order $k-1$) is a piecewise polynomial curve; i.e. a curve divided into several pieces, where each of these pieces is described by a polynomial of degree $k$, and the different pieces accomplish with certain continuity properties at the joint points or knots. Their most common representation is based on the utilization of B-splines (where B stands for *basic*).

### A. B-Splines Definition

Letting $\mathbf{s}(t)$ be the position vector along the curve as a function of the parameter $t$, a spline curve of order $k$, with control points $\mathbf{x}_i\ (i=0\dots n)$ and knot vector $\Xi = \{\xi_0,\dots,\xi_{n+k}\}$ can be expressed as:

$$\mathbf{s}(t) = \sum_{i=0}^{n} \mathbf{x}_i \beta_{i,k}(t) \tag{1}$$

where $\beta_{i,k}(t)$ are the normalized B-spline basis functions of order $k$ which are defined by the Cox-de Boor recursion formulas [16], [4]:

$$\beta_{i,1}(t) = \begin{cases} 1 & \text{if } \xi_i \le t \le \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and

$$\beta_{i,k}(t) = \frac{(t-\xi_i)}{\xi_{i+k-1}-\xi_i}\beta_{i,k-1}(t) + \frac{(\xi_{i+k}-t)}{\xi_{i+k}-\xi_{i+1}}\beta_{i+1,k-1}(t) \tag{3}$$

The knot vector is any nondecreasing sequence of real numbers ($\xi_i \le \xi_{i+1}$ for $i=0,\dots,n+k-1$) and can be defined in two different ways: *clamped*, when the multiplicity of the extreme knot values is equal to the order $k$ of the B-spline, and *unclamped* [16, 14]. When clamped knot vectors are used, first and last control polygon points define the beginning and end of the spline curve.

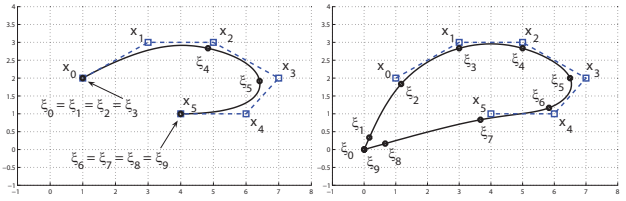


Fig. 1. Examples of splines in B-spline form. **a)** Cubic spline with clamped knot vector ($\Xi_c : \xi_0 = \dots = \xi_3 \le \dots \le \xi_6 = \dots = \xi_9$). **b)** Cubic spline with unclamped knot vector ($\Xi_c : \xi_0 \le \dots \le \xi_9$).

### B. Properties of Spline Curves

In this section, some properties of spline curves, defined as linear combination of B-splines, are summarized.

1) The curve generally follows the shape of the control polygon.
2) Any affine transformation is applied to the curve by applying it to the control polygon vertices.
3) Each basis function $\beta_{i,k}(t)$ is a piecewise polynomial of order $k$ with breaks $\xi_i,\dots,\xi_{i+k}$, vanishes outside the interval $[\xi_i, \xi_{i+k})$ and is positive on the interior of that interval:

$$\beta_{i,k}(t) > 0 \quad \Leftrightarrow \quad \xi_i \le t < \xi_{i+k} \tag{4}$$

4) As a consequence of property 3, the value of $\mathbf{s}(t)$ at a site $\xi_j \le t \le \xi_{j+1}$ for some $j \in \{k-1,\dots,n\}$ depends only on $k$ of the coefficients:

$$\mathbf{s}(t) = \sum_{i=j-k+1}^{j} \mathbf{x}_i \beta_{i,k}(t) \tag{5}$$

5) The sum of all the B-spline basis functions for any value of the parameter $t$ is 1:

$$\sum_{i=0}^{n} \beta_{i,k}(t) = 1 \tag{6}$$

6) The derivative of a B-spline of order $k$ is a spline of one order $k-1$, whose coefficients can be obtained differencing the original ones [4].

$$\frac{d\mathbf{s}(t)}{dt} = \mathbf{s}'(t) = (k-1)\sum_{i=0}^{n} \frac{\mathbf{x}_i - \mathbf{x}_{i-1}}{\xi_{i+k-1}-\xi_i}\beta_{i,k-1}(t) \tag{7}$$

For further information and justification of the previous properties, please see [4], [14], [1] and [16].

### C. Curve Fitting

Here we consider the problem of obtaining a spline curve that fits a set of data points $\mathbf{d}_j$, $j=0\dots m$. If a data point lies on the B-spline curve, then it must satisfy equation 1:

$$\mathbf{d}_j = \beta_{0,k}(t_j)\,\mathbf{x}_0 + \dots + \beta_{n,k}(t_j)\,\mathbf{x}_n, \quad j=0\dots m$$

system of equations which can be more compactly written as

$$\mathbf{d} = \mathbf{B}\mathbf{x} \begin{cases} \mathbf{d} &= \begin{bmatrix} \mathbf{d}_0 & \mathbf{d}_1 & \dots & \mathbf{d}_m \end{bmatrix}^T \\ \mathbf{x} &= \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \dots & \mathbf{x}_n \end{bmatrix}^T \\ \mathbf{B} &= \begin{bmatrix} \beta_{0,k}(t_0) & \dots & \beta_{n,k}(t_0) \\ \vdots & \ddots & \vdots \\ \beta_{0,k}(t_m) & \dots & \beta_{n,k}(t_m) \end{bmatrix} \end{cases} \tag{8}$$

Matrix $\mathbf{B}$, usually referred to as the *collocation matrix*, has for each of its rows at most $k$ non-null values. The parameter value $t_j$ defines the position of each data point $\mathbf{d}_j$ along the B-spline curve, and can be approximated by the chord length between data points:

$$\left.\begin{aligned} t_0 &= 0 \\ t_j &= \textstyle\sum_{s=1}^{j} |\mathbf{d}_s - \mathbf{d}_{s-1}|, \quad j \ge 1 \end{aligned}\right\} \tag{9}$$

being the total length of the curve

$$\ell = \sum_{s=1}^{m} |\mathbf{d}_s - \mathbf{d}_{s-1}| \tag{10}$$

which is taken as the maximum value of the knot vector.

The most general case occurs when $2 \le k \le n+1 < m+1$; the problem is over specified and can only be solved in a mean sense. A least squares solution can be computed making use of the pseudoinverse matrix of $\mathbf{B}$:

$$\mathbf{x} = \left[\mathbf{B}^T\mathbf{B}\right]^{-1}\mathbf{B}^T\mathbf{d} = \mathbf{\Phi}\mathbf{d} \tag{11}$$

Once the order of the B-spline bases $k$ is predefined, the number of control polygon vertices $n+1$, and the parameter

values along the curve are known (as calculated from equation 9), the basis functions $\beta_{i,k}(t_j)$ and hence the matrix $\mathbf{B}$ can be obtained.

In the work here presented, clamped knot vectors are generated taking the total length of the curve $\ell$ (equation 10), and defining a knot density which depends on the complexity of the environment. Recall that knots are the joints of the polynomial pieces a spline is composed of, so complex environments need a high knot density, while segments can be described properly using longer polynomial pieces (lower knot density). The interested reader is referred to [5] and [2], where more information about curve fitting methods can be found.

## III. SLAM WITH B-SPLINES

### A. Data Management

A commonly used sensor in mobile robotics is the laser range-finder. Whenever a robots makes an observation of its environment with such a device, a set of $m$ data points $\mathbf{d}_i \in \Re^2$ is obtained (we're considering here the 2D scenario). In this section, methods for extracting parametric splines representing the physical detected objects are presented. It is also shown the way of performing data association, establishing comparisons between the detected splines and the map splines.

*1) Obtaining Splines:* When a new data set is obtained from an observation of the environment, splines are extracted in a three-stages process (see Fig. 2):

- **Primary segmentation**: Data stream is split into pieces separated by measurements out of range, if any. A set of $N_1$ data vectors is obtained, called primary segmentation objects, and denoted $F_{1,1}, F_{1,2}, \ldots, F_{1,N_1}$ (see Fig. 2.c).
- **Secondary segmentation**: An analysis of the relative positions of consecutive data points is performed. The aim is to detect points close enough as to belong to the same feature, and also detect corners, but allowing points not to lie on the same segment (Fig. 2.f). A set of $N_2$ secondary segmentation features is obtained (Fig. 2.d).
- **Fitting**: Each of the secondary segmentation objects is fitted by a cubic spline, as described in section II-C.

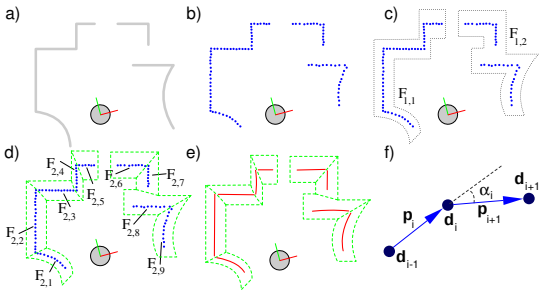


Fig. 2. Splines detection steps: **a)** Robot and environment. **b)** A new data set is acquired. **c)** Primary segmentation. **d)** Secondary segmentation. **e)** Obtained splines, fitting each of the data sets in d). **f)** Data points relative position. $|\alpha_i| \leq \alpha_{max}$ and $max(|\mathbf{p}_i|, |\mathbf{p}_{i+1}|) \leq \eta \cdot min(|\mathbf{p}_i|, |\mathbf{p}_{i+1}|)$ are checked (typically $\alpha_{max} \in [0, \pi/4]$ and $\eta \in [1.5, 2]$).

*2) Splines Association:* In this section, the data association process is described. At each sampling time, $k$, a new set of splines is obtained, being necessary to achieve a good feature association between the curves that are already contained in the map $(\mathbf{s}_{m,1}, \ldots, \mathbf{s}_{m,N})$, and the new ones that have just been detected $(\mathbf{s}_{o,1}, \ldots, \mathbf{s}_{o,N_2})$.

The process is explained with an example. Take the situation displayed in figure 3. Figure 3.a shows the initial configuration: a map containing one spline $\mathbf{s}_{m,1}(t)$, and two new features which have just been detected, $\mathbf{s}_{o,1}(u)$ and $\mathbf{s}_{o,2}(v)$ (notice the lack of correspondence between the parameters of different curves). A rough measure of the distances between the splines can be obtained by measuring the distances between the control points of each spline (recall here property 1). In order to improve this search, the map can be simplified choosing only the splines that are close to the robot's position.

At each sampling time, the position of the observed splines is calculated taking into account the best available estimation for the robot's position and orientation. Then, the distances from the control points of each of the detected splines to the control points of the splines contained in the map are calculated, and each of the observed splines is matched to the closest one on the map, following this criterion:

$$min\left(dist\left(\mathbf{x}_{m,i}, \mathbf{x}_{o,j}\right)\right) \leq d_{min}, \begin{cases} i = 1 \ldots n_m \\ j = 1 \ldots n_o \end{cases} \qquad (12)$$

where $\mathbf{x}_{m,i}$ are the control points of the map spline, $\mathbf{x}_{o,j}$ are the control points of the observed spline, $n_m$ and $n_o$ are, respectively, the number of control points of the spline in the map and the detected spline, and $dist\left(\mathbf{x}_{m,i}, \mathbf{x}_{o,j}\right)$ represents the euclidean distance between two control points.

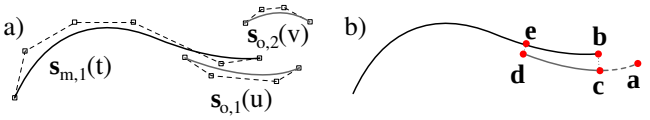


Fig. 3. Data association process. Comparison of control points positions (a) and parameters correspondence (b).

If no spline in the map is close enough to a detected spline (as occurs with $\mathbf{s}_{o,2}$ in figure 3.a) then, this new object is added to the map once its position has been updated. If a spline is associated with a map feature, then it is necessary to obtain a matching between their points, as depicted in figure 3.b. This matching provides information about the correspondence between the parameterizations of both curves, and is very useful when a spline extension is required. The process is as follows:

- One of the extreme points of the observed spline is considered (point $\mathbf{a}$)
- The closest point [19] on the map spline to the point $\mathbf{a}$ is calculated (point $\mathbf{b}$)
- If $\mathbf{b}$ is one of the extreme points of the map's spline, then, the closest point to $\mathbf{b}$ on the observed spline is calculated (point $\mathbf{c}$). Else, point $\mathbf{a}$ is matched with point $\mathbf{b}$.

- The process is repeated taking as starting point the other extreme of the observed spline (point $\mathbf{d}$ in the picture), which is associated with the point $\mathbf{e}$ on the map spline.

At the end of this process, not only correspondent pairs of points $(\mathbf{c}, \mathbf{b})$ and $(\mathbf{d}, \mathbf{e})$ are obtained, but also a correspondence between the map spline parameter $t$ and the observed spline parameter $u$: $(u_{ini}, t_{ini})$ and $(u_{fin}, t_{fin})$, given that

$$\begin{array}{ll} \mathbf{c} = \mathbf{s}_{o,1}(u_{ini}) & \mathbf{b} = \mathbf{s}_{m,1}(t_{ini}) \\ \mathbf{d} = \mathbf{s}_{o,1}(u_{fin}) & \mathbf{e} = \mathbf{s}_{m,1}(t_{fin}) \end{array} \tag{13}$$

The simple data association process described in this section, though quite simple and based upon euclidean distance metric, has performed very robustly in our experiments. However, benefits of parametric representation are not yet fully exploited, and further research is being undertaken in this sense.

### B. The State Model

The state of the system is composed by the robot (the only mobile element) and all the map features, modeled in the work here presented by cubic splines. When these splines are expressed as linear combination of B-splines, the state of each of them can be represented by the positions of their control polygon vertices, given a fixed and known knot vector which generates a basis of B-splines for each of the map features.

Referring all the positions and orientations to a global reference system $\{\mathbf{u}_W, \mathbf{v}_W\}$, and assuming the robot as the first feature in the map ($F_0$) the following expressions describe the state of the system at a certain time $k$:

$$\mathbf{x}_{F_0} = \mathbf{x}_r = [x_r, y_r, \phi_r]^T \tag{14}$$

$$\mathbf{x}_{F_i} = \mathbf{x}_{s_i} = [x_{i,0}, \ldots, x_{i,n_i}, y_{i,0}, \ldots, y_{i,n_i}]^T \quad i = 1, \ldots, N \tag{15}$$

and finally

$$\mathbf{x} = \left[\mathbf{x}_r^T, \mathbf{x}_{s_1}^T, \ldots, \mathbf{x}_{s_N}^T\right]^T \tag{16}$$

In the previous equations, $N$ is the number of map static elements (map features) and $n_i$ is the number of control points for each of them. Note that the number of control points for each of the splines contained in the map can be variable, as features are progressively extended as new areas of the environment are explored.

$$\mathbf{x}(k) \sim N(\hat{\mathbf{x}}(k|k), \mathbf{P}(k|k)) \tag{17}$$

where

$$\hat{\mathbf{x}}(k|k) = \left[\begin{array}{cccc} \hat{\mathbf{x}}_r(k|k) & \hat{\mathbf{x}}_{s_1}(k|k) & \ldots & \hat{\mathbf{x}}_{s_N}(k|k) \end{array}\right] \tag{18}$$

and

$$\mathbf{P}(k|k) = \begin{bmatrix} \mathbf{P}_{rr}(k|k) & \mathbf{P}_{rs_1}(k|k) & \ldots & \mathbf{P}_{rs_N}(k|k) \\ \mathbf{P}_{s_1r}(k|k) & \mathbf{P}_{s_1s_2}(k|k) & \ldots & \mathbf{P}_{s_1s_N}(k|k) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}_{s_Ns_1}(k|k) & \mathbf{P}_{s_Ns_2}(k|k) & \ldots & \mathbf{P}_{s_Ns_N}(k|k) \end{bmatrix} \tag{19}$$

### C. The Observation Model

The implementation of a EKF-SLAM algorithm requires of an observation model; i.e. some expression which allows to predict the measurements that are likely to be obtained by the robot sensors given the robot pose and the current knowledge of the environment. This measurement model can be understood, in our particular case, as the calculation of the intersection of the straight line defined by a laser beam (for each position across the laser angular range) with the splines contained in the map (Fig. 4). Unfortunately, calculating the intersection of a straight line with a parametric curve, in the form $\mathbf{s}(t) = [s_x(t), s_y(t)]^T$ is not suitable for an explicit mathematical formulation. There is a whole field of research regarding this complex problem known as *ray tracing* [17].

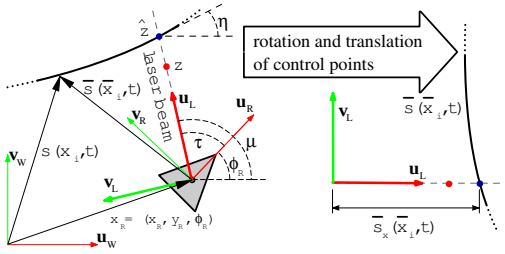


Fig. 4. Observation model

In this document, the predicted measurement is calculated in an iterative way, making use of property 2 in section II-B and the Newton-Raphson method for calculating the zeros of a function. The first step is to define an orthonormal reference system $\{\mathbf{u}_L, \mathbf{v}_L\}$, centered in the robot reference system $\{\mathbf{u}_R, \mathbf{v}_R\}$ and with the abscissas axis $\mathbf{u}_L$ defined by the laser beam direction and orientation (see Fig. 4). Let $\bar{\mathbf{s}}(\bar{\mathbf{x}}_i(\mathbf{x}_i, \mathbf{x}_r), t)$ be the position vector along a spline curve expressed in such a reference system (we're making here explicit the functional dependency between a spline and its control points). The relationship between control points $\mathbf{x}_i = [x_i, y_i]^T$ and $\bar{\mathbf{x}}_i = [\bar{x}_i, \bar{y}_i]^T$, $i = 0 \ldots n$, is given by:

$$\begin{bmatrix} \bar{x}_i \\ \bar{y}_i \end{bmatrix} = \begin{bmatrix} cos\mu & sin\mu \\ -sin\mu & cos\mu \end{bmatrix} \begin{bmatrix} x_i - x_r \\ y_i - y_r \end{bmatrix} \tag{20}$$

being $\mu$ the angle of the considered laser beam in the global reference system; i.e. given the laser orientation in the robot reference system, $\tau$:

$$\mu = \phi_r + \tau \tag{21}$$

In this context, the measurement prediction $\hat{z} = h(\mathbf{x}_i, \mathbf{x}_r)$ is given by the value of $\bar{s}_x(\bar{x}_i(\mathbf{x}_i, \mathbf{x}_r), t^*)$, where $t^*$ is the value of the parameter $t$ that makes $\bar{s}_y(\bar{y}_i(\mathbf{x}_i, \mathbf{x}_r), t^*) = 0$. As only a reduced set of $k$ control points affect the shape of the curve for each value of the parameter $t$, a small number of this points need to be rotated and translated to the new reference system. The initial guess for each map spline associated with an observation can be obtained directly from the data association stage, and for each of the laser beams positions the solution obtained in the previous prediction can be used as initial guess.

During the experiments, a maximum of two iterations were needed for this calculation, with a precision of 0.0001 m.

Despite the lack of an explicit observation model, it is still possible to calculate the derivatives of this expected measurements with respect to the map state in an approximate way. Once calculated the value $t^*$ which makes $\bar{s}_y(t^*) = 0$, the expected measurement in the nearness of this parameter location, assuming small perturbations in the state vector, can be approximated by:

$$h(\mathbf{x}_i, \mathbf{x}_r) = \bar{s}_x\left(\bar{x}_i(\mathbf{x}_i, \mathbf{x}_r), t^* - \frac{\bar{s}_y(\bar{y}_i(\mathbf{x}_i, \mathbf{x}_r), t^*)}{\bar{s}'_y(\bar{y}_i(\mathbf{x}_i, \mathbf{x}_r), t^*)}\right) \quad (22)$$

and derivating with respect to the control points positions:

$$\begin{aligned}\frac{\partial h}{\partial \mathbf{x}_i} &= \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial \mathbf{x}_i} + \bar{s}'_x(t^*)\frac{\frac{\partial \bar{s}'_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i}\bar{s}_y(t^*) - \frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i}\bar{s}'_y(t^*)}{\left[\bar{s}'_y(t^*)\right]^2} \\ &= \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial \mathbf{x}_i} - \frac{1}{\tan(\eta - \mu)}\frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i} \quad (23)\end{aligned}$$

The partial derivatives in equation 23 can be easily obtained looking at equations 1 and 20. For example from equation 1 we can obtain:

$$\frac{\partial \bar{s}_x}{\partial \bar{x}_i} = \frac{\partial \bar{s}_y}{\partial \bar{y}_i} = \beta_{i,k}(t) \quad (24)$$

So, finally, we can write:

$$\frac{\partial h}{\partial x_i} = \beta_{i,k}(t^*)\left[cos\mu + \frac{sin\mu}{\tan(\eta - \mu)}\right] \quad (25)$$

$$\frac{\partial h}{\partial y_i} = \beta_{i,k}(t^*)\left[sin\mu - \frac{cos\mu}{\tan(\eta - \mu)}\right] \quad (26)$$

Similarly, making use of property 5, and equations 1 and 20, we can obtain:

$$\frac{\partial h}{\partial x_r} = -cos\mu - \frac{sin\mu}{\tan(\eta - \mu)} \quad (27)$$

$$\frac{\partial h}{\partial y_r} = -sin\mu + \frac{cos\mu}{\tan(\eta - \mu)} \quad (28)$$

$$\frac{\partial h}{\partial \phi_r} = \frac{\hat{z}}{\tan(\eta - \mu)} \quad (29)$$

These equations will allow the efficient calculation of the relevant Jacobians in the following sections.

### D. Applying the EKF

In this section, results obtained in previous sections are put together and combined in the working frame of the Extended Kalman Filter with the aim of incrementally building a map of an environment modeled with cubic splines.

*1) Kalman Filter Prediction:* Between the times $k$ and $k+1$ the robot makes a relative movement, given by the vector

$$\mathbf{u}(k+1) \sim N(\hat{\mathbf{u}}(k+1), \mathbf{Q}(k+1)) \quad (30)$$

Under the hypothesis that the only moving object in the map is the robot, the a priori estimation of the state at time $k+1$ is given by:

$$\hat{\mathbf{x}}_r(k+1|k) = \mathbf{f}_r(\hat{\mathbf{x}}_r(k|k), \hat{\mathbf{u}}(k+1)) \quad (31)$$

$$\hat{\mathbf{x}}_{s_i}(k+1|k) = \hat{\mathbf{x}}_{s_i}(k|k) \quad (32)$$

and its covariance:

$$\begin{aligned}\mathbf{P}(k+1|k) = \mathbf{F}_x(k+1)\mathbf{P}(k|k)\mathbf{F}_x^T(k+1) + \\ +\mathbf{F}_u(k+1)\mathbf{Q}(k+1)\mathbf{F}_u^T(k+1) \quad (33)\end{aligned}$$

The Jacobian matrices are

$$\mathbf{F}_x(k+1) = \begin{bmatrix} \frac{\partial \mathbf{f}_r}{\partial \mathbf{x}_r}\Big|_{\hat{\mathbf{x}}_r(k|k), \hat{\mathbf{u}}(k+1)} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{I}_{n_N} \end{bmatrix} \quad (34)$$

$$\mathbf{F}_u(k+1) = \begin{bmatrix} \frac{\partial \mathbf{f}_r}{\partial \mathbf{u}}\Big|_{\hat{\mathbf{x}}_r(k|k), \hat{\mathbf{u}}(k+1)} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \quad (35)$$

where $\mathbf{f}_r$ depends on the mobile platform being considered.

*2) Kalman Filter Update:* Once obtained the expected measurements for each of the laser beams positions of an observation associated with a map spline, the innovation covariance matrix is given by [6]:

$$\mathbf{S}(k+1) = \mathbf{H}_{\mathbf{x}}(k+1)\mathbf{P}(k+1|k)\mathbf{H}_{\mathbf{x}}^T(k+1) + \mathbf{R}(k+1) \quad (36)$$

where the Jacobian is:

$$\mathbf{H}_{\mathbf{x}}(k+1) = \begin{bmatrix} \frac{\partial \mathbf{h}}{\partial \mathbf{x}_r} & \mathbf{0} \dots \mathbf{0} & \frac{\partial \mathbf{h}}{\partial \mathbf{x}_{s_i}} & \mathbf{0} \dots \mathbf{0} \end{bmatrix} \quad (37)$$

In the previous equation, the term $\frac{\partial \mathbf{h}}{\partial \mathbf{x}_r}$ is calculated making use of equations 27, 28 and 29, and term $\frac{\partial \mathbf{h}}{\partial \mathbf{x}_{s_i}}$ is calculated from 25 and 26. The gain matrix is calculated as follows

$$\mathbf{W}(k+1) = \mathbf{P}(k+1|k)\mathbf{H}_{\mathbf{x}}^T(k+1)\mathbf{S}^{-1}(k+1) \quad (38)$$

Finally, the state estimation and its covariance are updated according to:

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\mathbf{h}(k+1) \quad (39)$$

$$\mathbf{P}(k+1|k+1) = [\mathbf{I} - \mathbf{W}(k+1)\mathbf{H}_x(k+1)]\mathbf{P}(k+1|k) \quad (40)$$

### E. Extending the Map

The map is incrementally built in two ways: adding new objects, and extending objects already contained in the map.

*1) Adding New Objects to the Map:* Whenever a new observation is not associated with any of the objects already contained in the map, it is considered as a new object, and the spline which defines its shape is added to the map. Given a map containing $N$ static features, and a set of measurements $z_i, i = p \dots p+q$ corresponding to a new detected feature, the augmented state vector is:

$$\mathbf{x}^a = \mathbf{g}(\mathbf{x}, \mathbf{z}) \Leftrightarrow \begin{cases} \mathbf{x}_r^a = \mathbf{x}_r \\ \mathbf{x}_{s_i}^a = \mathbf{x}_{s_i} \quad ,i = 1, \dots, N \\ \mathbf{x}_{s_{N+1}}^a = \mathbf{g}_{s_{N+1}}(\mathbf{x}_r, \mathbf{z}) \end{cases} \quad (41)$$

where $\mathbf{g}_{s_{N+1}}(\mathbf{x}_r, \mathbf{z})$ is defined by the fitting of the $q$ new data points as described in section II-C:

$$\begin{bmatrix} x_{N+1,0} \\ \vdots \\ x_{N+1,n_N} \end{bmatrix} = \mathbf{\Phi} \begin{bmatrix} x_r + z_p \cos(\phi_r + \tau_p) \\ \vdots \\ x_r + z_{p+q} \cos(\phi_r + \tau_{p+q}) \end{bmatrix} \quad (42)$$

$$\begin{bmatrix} y_{N+1,0} \\ \vdots \\ y_{N+1,n_N} \end{bmatrix} = \mathbf{\Phi} \begin{bmatrix} y_r + z_p \sin(\phi_r + \tau_p) \\ \vdots \\ y_r + z_{p+q} \sin(\phi_r + \tau_{p+q}) \end{bmatrix} \quad (43)$$

The new covariance matrix for the augmented state vector is:

$$\mathbf{P}^a = \mathbf{G}_\mathbf{x} \mathbf{P} \mathbf{G}_\mathbf{x}^T + \mathbf{G}_\mathbf{z} \mathbf{R} \mathbf{G}_\mathbf{z}^T \quad (44)$$

and the Jacobians $\mathbf{G}_\mathbf{x} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ and $\mathbf{G}_\mathbf{z} = \frac{\partial \mathbf{g}}{\partial \mathbf{z}}$:

$$\mathbf{G}_\mathbf{x} = \begin{bmatrix} \mathbf{I}_r & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{I}_{n_N} \\ \hline \frac{\partial \mathbf{g}_{s_{N+1}}}{\partial \mathbf{x}_r} & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix}, \mathbf{G}_\mathbf{z} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \hline \frac{\partial \mathbf{g}_{S_{N+1}}}{\partial \mathbf{z}} \end{bmatrix} \quad (45)$$

with

$$\frac{\partial \mathbf{g}_{s_{N+1}}}{\partial \mathbf{x}_r} = \begin{bmatrix} \mathbf{\Phi} \begin{bmatrix} 1 & 0 & -z_p \sin \mu_p \\ \vdots & \vdots & \vdots \\ 1 & 0 & -z_{p+q} \sin \mu_{p+q} \end{bmatrix} \\ \hline \mathbf{\Phi} \begin{bmatrix} 0 & 1 & z_p \cos \mu_p \\ \vdots & \vdots & \vdots \\ 0 & 1 & z_{p+q} \sin \mu_{p+q} \end{bmatrix} \end{bmatrix} \quad (46)$$

$$\frac{\partial \mathbf{g}_{s_{N+1}}}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{\Phi} \begin{bmatrix} \cos \mu_p & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \cos \mu_{p+q} \end{bmatrix} \\ \hline \mathbf{\Phi} \begin{bmatrix} \sin \mu_p & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sin \mu_{p+q} \end{bmatrix} \end{bmatrix} \quad (47)$$

*2) Extending Map Objects:* Frequently, observations are only partially associated with a map feature (as in figure 3). This means that a new unexplored part of a map object is being detected and, consequently, this spline must be extended. Take for instance the situation displayed in figure 5, where the $j$-th map spline has been partially associated with an observation, and the information contained in a new set of data points

$$\mathbf{d_i} = \begin{bmatrix} d_i^x \\ d_i^y \end{bmatrix} = \begin{bmatrix} x_r + z_i \cos \mu_i \\ y_r + z_i \sin \mu_i \end{bmatrix}, \quad i = q, \dots, q+m \quad (48)$$

must be integrated into the map feature. The extended state vector will be:

$$\mathbf{x}^e = \mathbf{g_e}(\mathbf{x}, \mathbf{z}) \Leftrightarrow \begin{cases} \mathbf{x}_r^e &= \mathbf{x}_r \\ \mathbf{x}_{s_i}^e &= \mathbf{x}_{s_i}, \quad i \neq j \\ \mathbf{x}_{s_j}^e &= \mathbf{g}_{s_j}^e(\mathbf{x}_r, \mathbf{x}_j, \mathbf{z}) \end{cases} \quad (49)$$

With the objective of calculating $\mathbf{g}_{s_j}^e(\mathbf{x}_r, \mathbf{x}_j, \mathbf{z})$, a similar scheme to the one used during the data approximation is followed.

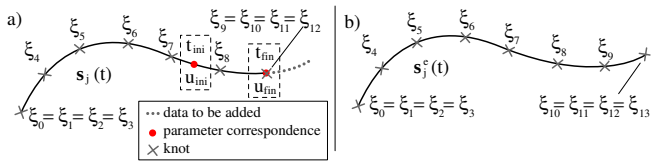


Fig. 5. Map spline extension with new data after Kalman filter update.

In [14] an unclamping algorithm is proposed, which is successfully applied in [10] with the goal of extending a B-spline curve to a single given point. Our problem is to extend a map spline, given a new set of measured data points. The following equations show how, combining the unclamping algorithm with the approximation scheme previously proposed, it is possible to extend the map features as new measurements are obtained. Given a spline curve, defined by a set of control points $\mathbf{x}_i$ and clamped knot vector in the form:

$$\Xi : \underbrace{\xi_0 = \dots = \xi_{k-1}}_{k} \leq \xi_k \leq \dots \leq \xi_n \leq \underbrace{\xi_{n+1} = \dots = \xi_{k+n}}_{k}$$

the unclamping algorithm proposed in [14] calculates the new control points corresponding to the unclamped knot vectors:

$$\Xi_r: \underbrace{\xi_0 = \dots = \xi_{k-1}}_{k} \leq \xi_k \leq \dots \leq \xi_n \leq \bar{\xi}_{n+1} \leq \dots \leq \bar{\xi}_{k+n}$$

$$\Xi_l: \bar{\xi}_0 \leq \dots \leq \bar{\xi}_{k-1} \leq \xi_k \leq \dots \leq \xi_n \leq \underbrace{\xi_{n+1} = \dots = \xi_{k+n}}_{k}$$

The right-unclamping algorithm (on the side corresponding to higher values for the parameter $t$) of a spline of order $k$, converting the knot vector $\Xi$ into $\Xi_r$, and obtaining the new control points $\mathbf{x}_i^r$ is as follows:

$$\mathbf{x}_i^r = \begin{cases} \mathbf{x}_i, & i = 0, 1, \dots, n-k+1 \\ \mathbf{x}_i^{k-2}, & i = n-k+2, n-k+3, \dots, n \end{cases} \quad (50)$$

where

$$\mathbf{x}_i^j = \begin{cases} \mathbf{x}_i^{j-1}, & i = n-k+2, n-k+3, \dots, n-j \\ \frac{\mathbf{x}_i^{j-1} - (1-\alpha_i^j)\mathbf{x}_{i-1}^j}{\alpha_i^j}, & i = n-j+1, \dots, n \end{cases} \quad (51)$$

being

$$\alpha_i^j = \frac{\bar{\xi}_{n+1} - \bar{\xi}_i}{\bar{\xi}_{i+j+1} - \bar{\xi}_i} \quad (52)$$

and the initial values are set to

$$\mathbf{x}_i^0 = \mathbf{x}_i, \quad i = n-k+2, n-k+3, \dots, n \quad (53)$$

If all the splines are cubic B-splines ($k = 4$), the previous expressions can be reduced to:

$$\left.\begin{aligned} \mathbf{x}_i^r &= \mathbf{x}_i, \quad i = 0, \dots, n-2 \\ \mathbf{x}_{n-1}^r &= -\Gamma_{n-1}^2 \mathbf{x}_{n-2} + \tfrac{1}{\gamma_{n-1}^2} \mathbf{x}_{n-1} \\ \mathbf{x}_n^r &= \Gamma_n^2 \Gamma_{n-1}^2 \mathbf{x}_{n-2} - \left(\tfrac{\Gamma_n^2}{\gamma_{n-1}^2} + \tfrac{\Gamma_n^1}{\gamma_n^2}\right) \mathbf{x}_{n-1} + \tfrac{1}{\gamma_n^1 \gamma_n^2} \mathbf{x}_n \end{aligned}\right\} \quad (54)$$

being

$$\gamma_i^j = \tfrac{\bar{\xi}_{n+1} - \bar{\xi}_i}{\bar{\xi}_{i+j+1} - \bar{\xi}_i} \quad \text{and} \quad \Gamma_i^j = \tfrac{1-\gamma_i^j}{\gamma_i^j} \quad (55)$$

Similar results can be obtained when converting a clamped knot vector $\Xi$ into a left-unclamped one $\Xi_l$:

$$\left.\begin{array}{l} \mathbf{x}_0^l = \frac{1}{\omega_0^1\omega_0^2}\mathbf{x}_0 - \left(\frac{\Omega_0^2}{\omega_1^2} + \frac{\Omega_0^1}{\omega_0^2}\right)\mathbf{x}_1 + \Omega_0^2\Omega_0^1\mathbf{x}_2 \\ \mathbf{x}_1^l = \frac{1}{\omega_1^2}\mathbf{x}_1 - \Omega_1^2\mathbf{x}_2 \\ \mathbf{x}_i^l = \mathbf{x}_i, \quad i = 2, \ldots, n \end{array}\right\} \tag{56}$$

with

$$\omega_i^j = \frac{\bar{\xi}_{k-1} - \bar{\xi}_{i+k}}{\bar{\xi}_{i+k-j-1} - \bar{\xi}_{i+k}} \text{ and } \Omega_i^j = \frac{1-\omega_i^j}{\omega_i^j} \tag{57}$$

These results can be combined with the methodology proposed in section II-C for obtaining new splines as new data is acquired, being aware of the following considerations:

- A parametrization for the new measurements to be integrated in the spline, congruent with the map spline parametrization, can be obtained easily from the data association stage.
- The knot vector needs to be unclamped and might need to be extended with $p$ extra knots in order to make room for the new span being added. New knots number and spacing are chosen according to the specified knot density, and as many new control points as new knots need to be added.

This way, the system of equations 8 is written for the new data points, extended with previous equations 54 and/or 56, and its least-squares solution provides a matrix-form linear relationship between the old control points $\mathbf{x}_j$ and the sampled data $\mathbf{d}_i$, and the new control points $\mathbf{x}_j^e$:

$$\begin{bmatrix} x_{j,0}^e \\ \vdots \\ x_{j,n_j+p}^e \end{bmatrix} = \mathbf{\Phi^e} \begin{bmatrix} d_q^x \\ \vdots \\ d_{q+m}^x \\ \hline x_{j,0} \\ \vdots \\ x_{j,n_j} \end{bmatrix}, \begin{bmatrix} y_{j,0}^e \\ \vdots \\ y_{j,n_j+p}^e \end{bmatrix} = \mathbf{\Phi^e} \begin{bmatrix} d_q^y \\ \vdots \\ d_{q+m}^y \\ \hline y_{j,0} \\ \vdots \\ y_{j,n_j} \end{bmatrix} \tag{58}$$

Note that once chosen an unclamped knot vector $\Xi^e$ for the extended spline, $\Phi^e$ can be considered a constant matrix. The new covariance matrix after extending the $j$-th spline is:

$$\mathbf{P}^e = \mathbf{G}_\mathbf{x}^e \mathbf{P} \mathbf{G}_\mathbf{x}^{eT} + \mathbf{G}_\mathbf{z}^e \mathbf{R} \mathbf{G}_\mathbf{z}^{eT} \tag{59}$$

where the involved Jacobians $\mathbf{G}_\mathbf{x}^e = \frac{\partial \mathbf{g}^e}{\partial \mathbf{x}}$ and $\mathbf{G}_\mathbf{z}^e = \frac{\partial \mathbf{g}^e}{\partial \mathbf{z}}$ have the following appearance:

$$\mathbf{G}_\mathbf{x}^e = \begin{bmatrix} \mathbf{I}_r & \mathbf{0} & \ldots & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_1} & \ldots & \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\mathbf{g}_{s_j}^e}{\partial \mathbf{x}_r} & \vdots & \ddots & \frac{\partial \mathbf{g}_{s_j}^e}{\partial \mathbf{x}_{s_j}} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \ldots & \mathbf{I}_{n_N} \end{bmatrix}, \mathbf{G}_\mathbf{z}^e = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \frac{\partial \mathbf{g}_{s_j}^e}{\partial \mathbf{z}} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{60}$$

being

$$\frac{\partial \mathbf{g}_{s_j}^e}{\partial \mathbf{x}_r} = \begin{bmatrix} \mathbf{\Phi}^e \begin{bmatrix} 1 & 0 & -z_p \sin\mu_p \\ \vdots & \vdots & \vdots \\ 1 & 0 & -z_{p+q} \sin\mu_{p+q} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\ \mathbf{\Phi}^e \begin{bmatrix} 0 & 1 & z_p \cos\mu_p \\ \vdots & \vdots & \vdots \\ 0 & 1 & z_{p+q} \sin\mu_{p+q} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{bmatrix}, \frac{\partial \mathbf{g}_{s_j}^e}{\partial \mathbf{x}_{s_j}} = \begin{bmatrix} \mathbf{\Phi}^e \begin{bmatrix} \mathbf{0} \\ \hline \mathbf{I} \end{bmatrix} \\ \mathbf{\Phi}^e \begin{bmatrix} \mathbf{0} \\ \hline \mathbf{I} \end{bmatrix} \end{bmatrix}$$

and

$$\frac{\partial \mathbf{g}_{s_j}^e}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{\Phi}^e \begin{bmatrix} \cos\mu_p & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \cos\mu_{p+q} \\ \hline \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix} \\ \mathbf{\Phi}^e \begin{bmatrix} \sin\mu_p & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \sin\mu_{p+q} \\ \hline \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix} \end{bmatrix}$$

## IV. Experimental Results

Several experiments have been performed with real data in order to validate the results here presented. Figure 6 shows a map representing an environment with predominant flat features (segments), with 81 splines defined by 332 control points. Figure 7 depicts the map of a bigger and more

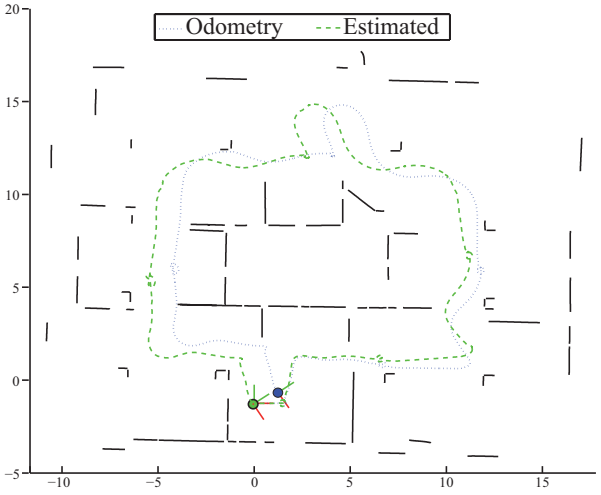


Fig. 6. Map of a career fair held at the School of Industrial Engineering (UPM) in Madrid.

complex environment with a mixture of both straight and curved features. In this case 461 control points defining a total of 96 splines are necessary. In both experiments, a B21r robot with a SICK laser was used for the data acquisition with a sampling frequency of 5 Hz, and density of 2 knots/m was used for knot vectors generation. No geometric constraints have been used in the map building process.

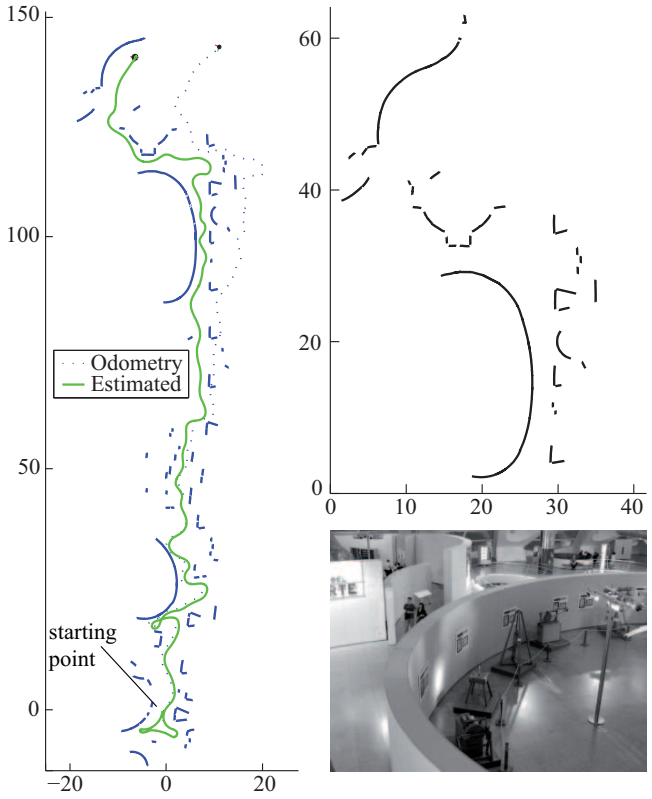


Fig. 7. Map of the Museum of Science "Principe Felipe" (Valencia, Spain) (left). Detail view and photography (right).

## V. Conclusion

A new methodology for simultaneous localization and mapping in arbitrary complex environments has been proposed. For the first time, it has been shown how the powerful and computationally efficient mathematical tools that splines are, fit into the EKF-SLAM framework allowing the representation of complex structures in a parametric way. Despite the apparent difficulty that this symbiosis between EKF-SLAM and splines theory could involve, simple and easily programmable matrix-form expressions have been obtained. Specifically, the new ray tracing method here proposed, constitutes a mathematical artifice that not only provides an effective observation model, but also makes easier the obtaining of the Jacobians for the Kalman filter update. It seems clear that, when simple descriptions of the environment are insufficient or unfeasible, any other SLAM algorithm could benefit from the versatility of this new parametric representation.

No information is lost, as control points incrementally encapsulate the uncertainties of laser measurements, thanks to the application of the new matrix-form algorithms here proposed for splines enlargement. Moreover, these concepts and techniques are suitable for a 3D extension. Finally, this new representation constitutes a big step forward, compared to current SLAM techniques based on geometric maps, allowing the mathematical interpretation and reasoning over the maps being built regardless the shape of the contained features.

## Acknowledgement

The first author thanks to the "Consejería de Educación de la Comunidad de Madrid" for the PhD scholarship being granted to him, and to the "ACR Centre of Excellence for Autonomous Systems" for having him as visiting postgraduate research student during some months in 2006.

## References

[1] J. H. Ahlberg, E. N. Nilson, and J. L. Walsh. *The Theory of Splines and their Applications.* Academic Press, New York, USA, 1967.

[2] A. Atieg and G. A. Watson. A class of methods for fitting a curve or surface to data by minimizing the sum of squares of orthogonal distances. *J. Comput. Appl. Math.*, 158(2):277–296, 2003.

[3] J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós. The SPmap: A probabilistic framework for simultaneous localization and map building. *IEEE Trans. Robot. Automat.*, 15(5):948–952, October 1999.

[4] C. de Boor. *A Practical Guide to Splines*. Springer, 1978.

[5] C. de Boor and J. Rice. Least squares cubic spline approximation I – fixed knots. Technical Report CSD TR 20, Dept. of Computer Sciences, Purdue University, 1968.

[6] M. W. M. G. Dissanayake, P. M. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. Robot. Automat.*, 17(3):229–241, 2001.

[7] R. Eustice, H. Singh, and J. Leonard. Exactly sparse delayed-state filters. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2428–2435, Barcelona, Spain, April 2005.

[8] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2443–2448, 2005.

[9] J. Guivant, E. Nebot, and H. Durrant-Whyte. Simultaneous localization and map building using natural features in outdoor environments. *Intelligent Autonomous Systems 6*, 1:581–586, July 2000.

[10] S.-M. Hu, C.-L. Tai, and S.-H. Zhang. An extension algorithm for B-Splines by curve unclamping. *Computer-Aided Design*, 34(5):415–419, April 2002.

[11] J. J. Leonard, H. F. Durrant-Whyte, and I. J. Cox. Dynamic map building for an autonomous mobile robot. *Int. J. Rob. Res.*, 11(4):286–298, 1992.

[12] R. Madhavan, G. Dissanayake, and H. F. Durrant-Whyte. Autonomous underground navigation of an LHD using a combined ICP–EKF approach. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3703–3708, 1998.

[13] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 116–121, March 1985.

[14] L. Piegl and W. Tiller. *The NURBS Book (2nd ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[15] D. Rodriguez-Losada, F. Matia, and R. Galan. Building geometric feature based maps for indoor service robots. *Robotics and Autonomous Systems*, 54(7):546–558, July 2006.

[16] D. F. Rogers. *An introduction to NURBS: With historical perspective*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[17] A. J. Sweeney and R. H. Barrels. Ray-tracing free-form B-spline surfaces. *IEEE Computer Graphics & Applications*, 6(2):41–49, February 1986.

[18] M. Veeck and W. Burgard. Learning polyline maps from range scan data acquired with mobile robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1065–1070, 2004.

[19] H. Wang, J. Kearney, and K. Atkinson. Robust and efficient computation of the closest point on a spline curve. In *Proc. of the 5th International Conference on Curves and Surfaces*, pages 397–406, San Malo, France, 2002.

[20] S. Zhang, L. Xie, M. Adams, and F. Tang. Geometrical feature extraction using 2d range scanner. In *Proc. of the 4th International Conference on Control and Automation (ICCA'03)*, pages 901–905, 2003.