2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Online Trajectory Planning and Filtering for Robotic Applications via B-spline Smoothing Filters

Luigi Biagiotti, Claudio Melchiorri

Abstract— **In this paper, a novel technique for online generating trajectories in the 3-D space is presented. The trajectory planner is based on cubic B-splines. However, while the definition of B-splines requires the solution of a global problem that involves the entire set of via-points to be interpolated/approximated, and therefore it is not suitable for online implementation, the proposed generator is able to approximate spline functions with the prescribed precision on the basis of local computations, which only need the knowledge of a very limited number of via-points. FIR filters are the foundation of this result. As a matter of fact the planner is composed by a first FIR filter for the computation of the control points from the sequence of desired via-points, followed by a chain of moving average filters. Therefore, the generator combines the characteristics of B-spline trajectories (smoothness and minimum curvature) and those of FIR filters (simple structure and computational efficiency). Moreover, besides standard cubic curves, the so-called smoothing B-splines have been considered for online trajectory generation. This allows to find a tradeoff between the possibility of exactly crossing the given via-points and the smoothness of the resulting trajectory. A simple teleoperation task with a Puma 560 industrial manipulator has been arranged for experimentally validating the proposed method.**

I. Introduction

Spline functions are one of the main mathematical tools for designing trajectories for robotic systems, in particular in the workspace where complex motions are generally defined by the user by means of a (possibly very large) set of via-points to be interpolated or approximated. The parameters that define these functions (coefficients of the polynomial form or control points of the B-form) are computed offline by solving a global problem that requires all the given points [1]. Additionally, these trajectories may be optimized with the purpose of minimizing the total travelling time of robot subject to constraints of velocity, acceleration and jerk [2], or to globally minimize some quantities, such as acceleration [3] or jerk [4]. More recently, in [5] an optimization of spline trajectories in the frequency domain has been proposed in order to suppress vibrations that may arise in those robotic applications where compliance of links and joints are not negligible. The proposed solution exploits the fact that spline functions expressed in the so-called B-form can be efficiently generated by means of a chain of linear filters properly fed with the sequence of the control points that determine the shape of the curves in the space. The trajectory generator devised in [6] is composed by p moving average filters of order N and an algorithm that transforms the desired points $\boldsymbol{q}_j$ in the set of control points $\boldsymbol{p}_j$ used for defining the input sequence $\boldsymbol{p}(k) = \sum_{j=0}^{n} \boldsymbol{p}_j B^0(k - jN)$, where

$$B^0(k) = B^0(kT_s) = \begin{cases} 1, & \text{if } k = 0, 1, \ldots, N-1 \\ 0, & \text{otherwise.} \end{cases}$$

In Fig. 1 the sequence $\boldsymbol{p}(k)$ and the discrete-time spline $\mathbf{s}(k)$ are shown for a simple one-dimensional case. Note that the B-spline is defined by adopting a sampling period T_s, that generally coincides with the sampling time of the overall control system, while $\boldsymbol{p}(k)$ is a piecewise constant sequence, in which the generic value $\boldsymbol{p}_j$ is maintained for $T = N \cdot T_s$ seconds. Moreover, as shown in Fig. 2, while the spline evaluation is performed online, its definition (i.e. the computation of the control points) is made offline. Aim of this work is to remove this limitation of the trajectory generator, by means of an additional filter that computes the control points from the sequence of desired via-points to be interpolated. In this way, the proposed trajectory generator can be used in those applications where the the points to be reached are not known a priori but are provided runtime. Since cubic splines are considered, the generator behaves like a filter that produces position profiles with continuous velocities and accelerations. Moreover, by exploiting the features of the so-called smoothing splines [1], a filter is devised that allows to made the curve arbitrarily smooth by acting on a free parameters.

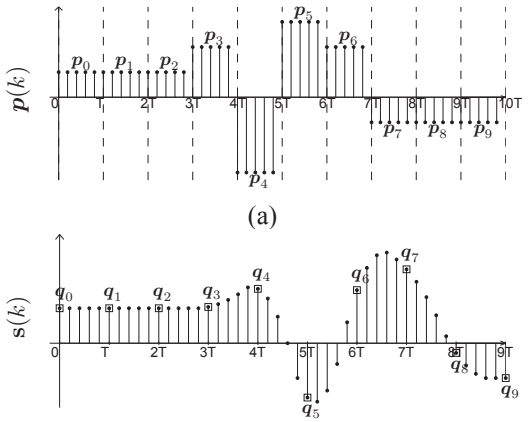

Fig. 1. Piecewise constant sequence $\boldsymbol{p}(k)$ (a) generating the discrete-time spline $s(k)$ that interpolates the given points $\boldsymbol{q}_j$ (b).

L. Biagiotti is with the Department of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia, Via Vignolese 905, 41100 Modena, Italy, e-mail: luigi.biagiotti@unimore.it.

C. Melchiorri is with the Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi", University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy, e-mail: claudio.melchiorri@unibo.it.

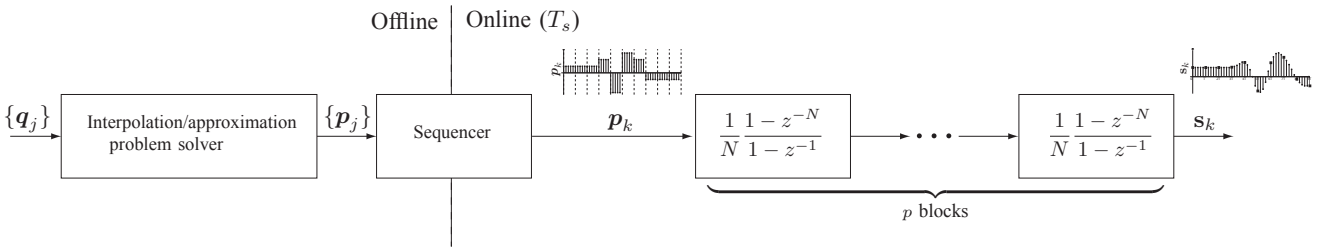

Fig. 2. Structure of the filter for B-spline trajectories planning proposed in [6].

II. Online computation of cubic B-splines control points

In [6], it has been shown that the control points $\boldsymbol{p}_k$ defining a cubic B-spline interpolating a set of via-points $\boldsymbol{q}_k$, $k = 0, \ldots, l$ in uniformly spaced time instants $t_k = k \cdot T$, can be obtained by solving the linear system obtained by stacking the equations

$$\mathbf{s}_s(kT) = \frac{1}{6}\boldsymbol{p}_{k-1} + \frac{4}{6}\boldsymbol{p}_k + \frac{1}{6}\boldsymbol{p}_{k+1} = \boldsymbol{q}_k, \quad k = 0, \ldots, l \quad (1)$$

for all the given points, i.e.

$$\begin{bmatrix} 4 & 1 & 0 & & & \cdots & 0 \\ 1 & 4 & 1 & 0 & & \cdots & 0 \\ \vdots & & & \ddots & & & \vdots \\ 0 & & \cdots & & 0\ 1\ 4\ 1 & & \\ 0 & & \cdots & & 0\ 1\ 4 & & \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \vdots \\ \boldsymbol{p}_{l-2} \\ \boldsymbol{p}_{l-1} \end{bmatrix} = \begin{bmatrix} 6\boldsymbol{q}_1 - \boldsymbol{q}_0 \\ 6\boldsymbol{q}_2 \\ \vdots \\ 6\boldsymbol{q}_{l-2} \\ 6\boldsymbol{q}_{l-1} - \boldsymbol{q}_l \end{bmatrix} \quad (2)$$

being the first and the last control point coincident with initial and final via-point. Although a solution of (2) can be calculated in a very efficient manner, being the system matrix tridiagonal, it is clear that such a solution can only be found once all the via-points $\boldsymbol{q}_k$ are known. However, when the via-points are given progressively, it may be desirable that control points are calculated runtime by approximating, if possible, the ideal solution. To this purpose, it is worth noticing that relationship (1) between control points and via-points can be rewritten in terms of Z-transforms as

$$\frac{P(z)}{Q(z)} = \frac{6}{z + 4 + z^{-1}} = H^3(z). \quad (3)$$

Unfortunately, equation (3), that represents a simple linear filter, cannot directly been used for control points calculation since the poles $z_{1,2} = -2 \pm \sqrt{3}$ do not lie both within the unit circle, and consequently the filter $H^3(z)$ results unstable. However, it is worth noticing that, if α denotes the stable pole, i.e. $\alpha = -2 + \sqrt{3}$, the instable one is given by its reciprocal, that is $1/\alpha = -2 - \sqrt{3}$ and the filter's transfer function can be factorized into a product of complementary causal and anticausal terms, i.e.

$$H^3(z) = H_c^3(z) \cdot H_c^3(z^{-1}) \quad (4)$$

with

$$H_c^3(z) = \frac{1}{1 - \alpha\, z^{-1}}.$$

Note that $H^3(s)$ represents a so-called zero-phase (non-causal) filter, whose output can be computed by applying the stable filter $H_c^3(z)$ to the input sequence, then time reversing the filtered data, applying again the filter $H_c^3(z)$ and time reversing one more time the output. Unfortunately, also this procedure can be performed when the input data are completely known and it is therefore suitable only for offline filtering. An expression alternative to (4), that can lead to an online implementation of $H_3(z)$ consists of a summation of simple partial fractions, i.e.

$$\begin{aligned} H^3(z) &= \frac{1-\alpha}{1+\alpha}\left(\frac{1}{1-\alpha\, z^{-1}} + \frac{1}{1-\alpha\, z} - 1\right) \quad (5) \\ &\quad \frac{1-\alpha}{1+\alpha}\big(H_c^3(z) + H_c^3(z^{-1}) - 1\big). \end{aligned}$$

Equation (5), properly converted in a finite-difference equation, can be adopted for efficient (offline) control points calculations, see [7], while the algorithm for online control points computation can be deduced by approximating (5) with a FIR filter [8]. The most straightforward methods for implementing such a filter is to truncate the ideal impulse response by windowing. By anti-transforming (5) the impulse response results

$$\begin{aligned} h^3(k) &= \frac{1-\alpha}{1+\alpha}\left(\alpha^k\, u(k) + \alpha^{-k}\, u(-k) - \delta_0(k)\right) \\ &= \frac{1-\alpha}{1+\alpha}\,\alpha^{|k|} \quad (6) \end{aligned}$$

where

$$\delta_0(k) = \begin{cases} 1 & \text{if } k = 0 \\ 0, & \text{if } k \neq 0 \end{cases}, \qquad u(k) = \begin{cases} 1 & \text{if } k \geq 0 \\ 0, & \text{if } k < 0 \end{cases}$$

are the discrete unit impulse and unit step sequence respectively. In Tab. I the numerical values of the sequence $h^3(k)$, for $|k| = 0, \ldots, 5$ are reported (consider the case $\lambda = 0$). Note that the magnitude of the samples, that decays exponentially, goes to zero very quickly as $|k|$ grows, therefore the FIR filter with (truncated) impulse response

$$h_{\text{T}}^3(k) = \begin{cases} h^3(k) & k = -M, \ldots, M \\ 0 & \text{otherwise,} \end{cases} \quad (7)$$

provides an excellent approximation of $H^3(z)$ also for small values of M.

In order to guarantee a unit static gain, as in the original filter $H^3(z)$, the samples of the impulse response $h_{\text{T}}^3(k)$, that are the coefficients of the FIR filter, are normalized by $\sum_{n=-M}^{M} h_{\text{T}}^3(n)$. Therefore, the final expression of the FIR

$\lvert k\rvert$	0	1	2	3	4	5
$\lambda = 0$	1.7338	-0.4646	0.1245	-0.0334	0.0089	-0.0024
$\lambda = 1/144$	1.5310	-0.3062	0.0462	-0.0062	0.0008	-0.0001
$\lambda = 1/24$	1.0952	0.0000	-0.0499	-0.0000	0.0023	0.0000
$\lambda = 1/10$	0.8478	0.1385	-0.0450	-0.0193	0.0003	0.0016
$\lambda = 1$	0.4018	0.2424	0.0841	0.0041	-0.0174	-0.0140
$\lambda = 10$	0.1952	0.1666	0.1183	0.0714	0.0350	0.0112
$\lambda = 100$	0.1252	0.1191	0.1056	0.0886	0.0706	0.0535

TABLE I

NUMERICAL VALUES OF THE IMPULSE RESPONSE $h^3_\lambda(k)$ OF SMOOTHING B-SPLINE FILTERS FOR DIFFERENT λ.

filter for control points calculation becomes

$$H^3_{\text{FIR}}(z) = \frac{\sum_{n=-M}^{M} h^3_{\text{T}}(n)\ z^{-n}}{\sum_{n=-M}^{M} h^3_{\text{T}}(n)} = \sum_{n=-M}^{M} h^3_{\text{FIR}}(n)\ z^{-n} \qquad (8)$$

Note that that $H^3_{\text{FIR}}(z)$ still represents a non-causal system, but with a simple time-shift it is possible to make it feasible. As a matter of fact $H^3_{\text{FIR}}(z)$ can be rewritten as

$$H^3_{\text{FIR}}(z) = z^M \sum_{n=0}^{2M} h^3_{\text{FIR}}(n-M)\ z^{-n} = \frac{P(z)}{Q(z)}$$

and therefore

$$z^{-M}P(z) = \sum_{n=0}^{2M} h^3_{\text{FIR}}(n-M)\ z^{-n}Q(z)$$

that corresponds to the finite-difference equation

$$\boldsymbol{p}(k-M) = \sum_{n=0}^{2M} h^3_{\text{FIR}}(n-M)\ \boldsymbol{q}(k-n). \qquad (9)$$

From (9) it comes out that at a generic time instant k, the FIR filter $H^3_{\text{FIR}}(z)$ fed with the last $2M+1$ via-points $\boldsymbol{q}$ provides the value of the control point $\boldsymbol{p}$ at time $k-M$. The choice of M is therefore quite critical, since it must be the result of a tradeoff between the precision in control points calculations the and the delay introduced by the filter.

III. SMOOTHING B-SPLINE FILTER

In many robotics applications, it is not required that the given via-points are exactly crossed but it is preferable a trajectory with lower curvature and therefore lower acceleration. In these cases, smoothing B-splines are the ideal tool [1], being the curves $s(\tau)$ that, in the discrete time-instants $\tau = k\,T$ minimize

$$L := \sum_{k=0}^{l} |s(kT) - q_k|^2 + \lambda \int_0^{lT} \left|\frac{d^2 s(\tau)}{d\tau^2}\right|^2 d\tau \qquad (10)$$

where λ is a parameter which can be freely chosen in order to control their smoothness. In the case of uniform B-splines, the minimum value of the cost function (10) can be found by solving the linear system

$$(\boldsymbol{A} + 6\lambda \boldsymbol{C}^T\boldsymbol{C})\boldsymbol{P} = 6\boldsymbol{Q} \qquad (11)$$

with the matrices

$$\boldsymbol{A} = \begin{bmatrix} 4 & 1 & & 0 \\ 1 & 4 & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & 4 \end{bmatrix},\ \boldsymbol{C} = \begin{bmatrix} 1 & -2 & 1 & & & 0 \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ 0 & & & 1 & -2 & 1 \end{bmatrix}$$

of proper dimensions, and the vectors of via-points $\boldsymbol{Q} = [\boldsymbol{q}_0, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_l]^T$ and that of the (unknown) control points $\boldsymbol{P} = [\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_l]^T$, see [1], [9]. Similarly to standard cubic B-splines, it is possible to find a relationship between via-points and control points in term of the Z-transforms [9], i.e.

$$\frac{P(z)}{Q(z)} = \frac{6}{z + 4 + z^{-1} + 6\lambda(z^2 - 4z + 6 - 4z^{-1} + z^{-2})}, \qquad (12)$$

and like in case of Cubic B-spline filter $H^3(z)$ the poles of the filter in (12), denoted by $H^3_\lambda(z)$, occur in pairs $(p_i,\ 1/p_i)$ because of the symmetry of the coefficients of the denominator's polynomial. As a consequence, the transfer function $H^3_\lambda(z)$ can be decomposed again into the product of a causal and an anti-causal filter

$$H^3_\lambda(z) = H^3_{\lambda,c}(z) \cdot H^3_{\lambda,c}(z^{-1}) \qquad (13)$$

and the methods followed for obtaining cubic B-spline filters can be used with no changes also in case of smoothing B-splines. Unfortunately, in this case the solution is complicated by the fact that the poles of $H^3_\lambda(z)$ are not constant but depends on the parameter λ. In Fig. 3 the position of the poles in the complex plane are shown as a function of λ. In particular, in Fig. 3(b) the two stable poles of $H^3_\lambda(z)$ are reported. Note that for $\lambda \to 0$ a pole tends to $\alpha = -2 + \sqrt{3}$ (which characterizes standard cubic splines), while the other one goes to the origin of Z-plane. Moreover, since the poles of $H^3_\lambda(z)$ may be real (distinct or coincident) or complex conjugate, different situations arise in the computation of its impulse response, that is the basic tool for FIR filter approximation. Accordingly, it is necessary to find the expression of $h^3_\lambda(k)$ as a function of the parameter λ:

- $\lambda = 0$. The smoothing filter degenerates into the standard cubic spline filter.
- $0 < \lambda \leq 1/144$. By considering the auxiliary variable $u = \left(z^{\frac{1}{2}} - z^{-\frac{1}{2}}\right)^2$, the two real poles enclosed within the unit circle can be easily computed as

  $$\alpha_i = 1/2\left(2 + u_i + \sqrt{4u_i + u_i^2}\right), \qquad i = 1, 2$$

  with

  $$u_i = \frac{-1 \pm \sqrt{1 - 144\,\lambda}}{12\,\lambda}, \qquad i = 1, 2.$$

  The filter can be rewritten in terms of partial fraction as

  $$H^3_\lambda(z) = c_1\left(\frac{1}{1-\alpha_1\,z} + \frac{1}{1-\alpha_1\,z^{-1}} - 1\right) - c_2\left(\frac{1}{1-\alpha_2\,z} + \frac{1}{1-\alpha_2\,z^{-1}} - 1\right) \qquad (14)$$

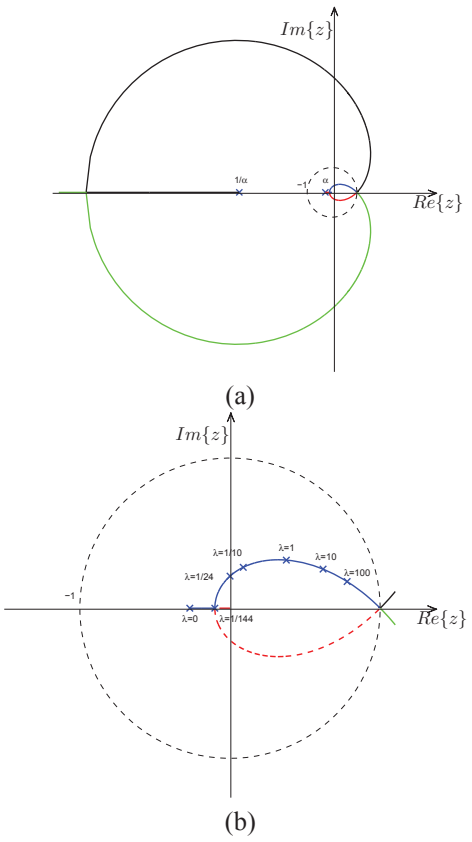

Fig. 3. Position of the poles of $H^3_\lambda(z)$ as a function of the parameters λ (a) and magnification of the area included within the unit circle (b).

where

$$c_1 = \frac{\alpha_1(-1+\alpha_1)(-1+\alpha_2)^2}{(\alpha_1-\alpha_2)(-1+\alpha_1\,\alpha_2)(1+\alpha_1)}$$

$$c_2 = \frac{\alpha_2(-1+\alpha_1)^2(-1+\alpha_2)}{(\alpha_1-\alpha_2)(-1+\alpha_1\,\alpha_2)(1+\alpha_2)}.$$

By anti-transforming (14), the impulse response results

$$h^3_\lambda(k) = c_1\,\alpha_1^{|k|} - c_2\,\alpha_2^{|k|}. \tag{15}$$

- $\lambda = 1/144$. The filter, characterized by two stable coincident real poles $\alpha_{1,2} = -5+2\sqrt{6} = \alpha$, can be written as

$$H^3_\lambda(z) = d_1\left(\frac{1}{1-\alpha\,z} + \frac{1}{1-\alpha\,z^{-1}} - 1\right) + d_2\left(\frac{\alpha\,z}{(1-\alpha\,z)^2} + \frac{\alpha\,z^{-1}}{(1-\alpha\,z^{-1})^2}\right) \tag{16}$$

with

$$d_1 = \frac{(1-\alpha)(1+\alpha^2)}{(1+\alpha)^3}$$

$$d_2 = \frac{(1-\alpha)^2}{(1+\alpha)^2}.$$

Therefore, the impulse response is

$$h^3_\lambda(k) = d_1\,\alpha^{|k|} + d_2\,|k|\,\alpha^{|k|}. \tag{17}$$

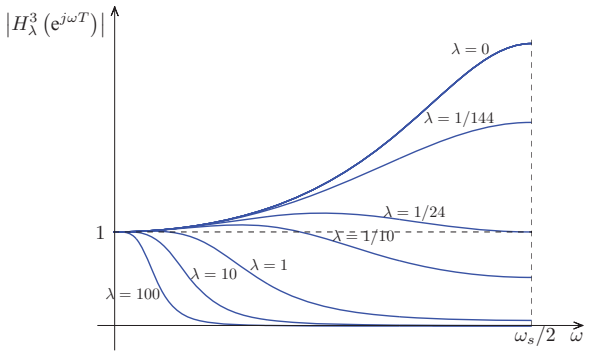

Fig. 4. Frequency response of the smoothing B-spline filter $H^3_\lambda(z)$, for different values of λ.

- $\lambda > 1/144$. The filter has two stable complex conjugate poles $\alpha_{1,2} = \rho\,e^{\pm j\omega}$. In this case, the expression of $h^3_\lambda(k)$ can be found in [10], [9] and is reported only for the sake of completeness. The magnitude of the poles can be computed as

$$\rho = \sqrt{\frac{1}{2}(y - \sqrt{y^2-4})}$$

with

$$y = \frac{1}{6\lambda}\left(12\lambda + 2 + \sqrt{3+144\lambda}\right),$$

while the phase is

$$\omega = \begin{cases} \pi - \arctan\sqrt{\frac{144\lambda-1}{v}} & \lambda < \frac{1}{24} \\ \frac{\pi}{2} & \lambda = \frac{1}{24} \\ \arctan\sqrt{\frac{144\lambda-1}{v}} & \lambda > \frac{1}{24} \end{cases}$$

where

$$v = 1 - 96\lambda + 24\,\lambda\sqrt{3+144\,\lambda}.$$

The anti-transform of $H^3_\lambda(z)$ is

$$h^3_\lambda(k) = a_1\,\rho^{|k|}\Big(\cos(\omega\,|k|) + a_2\,\sin(\omega\,|k|)\Big)$$

with the coefficients

$$a_1 = \frac{(1-2\,\rho\,\cos(\omega)+\rho^2)\,(1+\rho^2)}{(1+2\,\rho\,\cos(\omega)+\rho^2)\,(1-\rho^2)}$$

$$a_2 = \frac{1-\rho^2}{1+\rho^2}\,\frac{1}{\tan(\omega)}$$

In Tab. I the numerical values of the sequence $h^3_\lambda(k)$, for $|k| = 0,\ldots,5$ are reported for some noticeable values of λ. Note that for growing values of λ, the magnitude of the central samples tends to decrease and, in general, the effect of high values of λ is to equalize the magnitude of all the samples (see for instance the case of $\lambda = 100$).
Once the sequence $h^3_\lambda(k)$ is available, it is possible to compute the approximating FIR filter according to (8) and the same remarks reported for standard cubic B-splines can be extended to smoothing B-splines. Moreover, since the coefficients of the filter $H^3_{\lambda,\mathrm{FIR}}(z)$ tend to become similar in magnitude for increasing value of λ, the influence of the

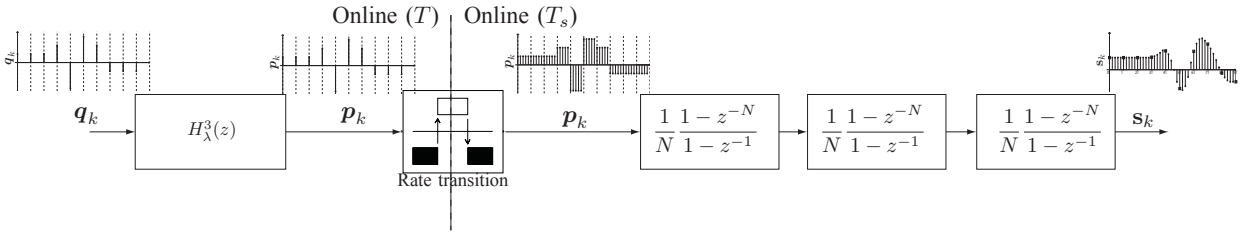

Fig. 5. Structure of the filter for online B-spline trajectories planning.

k-th via-point on the k-th control points decreases, while the effects of adjacent via-points augment. For this reason, $H^3_{\lambda,\text{FIR}}(z)$ behaves like a filter that reduces the variations in the control points. The same conclusions can be deduced by taking into account the frequency response of $H^3_\lambda(z)$, i.e.

$$H^3_\lambda\left(\mathrm{e}^{j\omega T}\right) = \frac{6}{6 - 4\sin^2\left(\pi\frac{\omega}{\omega_s}\right) + 96\lambda\sin^4\left(\pi\frac{\omega}{\omega_s}\right)}.$$

In Fig. 4 the magnitude of $H^3_\lambda\left(\mathrm{e}^{j\omega T}\right)$ is shown for different values of λ. It is clear that for high values of this parameter, the filter behaves like a low-pass filter, whose pass-band can be made arbitrarily narrow. This properties can be helpful, when the via-points are affected by some kind of noise. On the other hand, it is worth noticing that high values of λ imply large interpolation errors.

IV. General Structure of the B-spline planner/filter

In Fig. 5 the structure of the proposed trajectory generator is illustrated. The filter, working completely online, is composed by two main elements

1) a FIR filter $H^3_{\lambda,\text{FIR}}(z)$ of order $2M+1$ that computes the control points from desired control points;
2) a cascade of 3 moving average filters, since cubic B-splines have been considered.

The former element is computed with a sample time T, multiple of the basic sample period T_s ($T = N \cdot T_s$), that represents the time-distance among the points to be interpolated/approximated. The average filters are implemented with a period T_s, and they have an impulse response of length equal to T, being of order N. Between the two elements, it is necessary a rate transition from T to T_s, that maintains the value $\boldsymbol{p}$ for T seconds.

The trajectory generator shown in Figure is suitable for scalar (one-dimensional) trajectories. In order to take into account trajectories in a d-dimensional space ($d = 2, 3$), it is necessary to replicate the filter d times, one for each component. The desired B-spline is obtained if the sequences of the components of via-points $\boldsymbol{q}_k$, provided as input, are synchronized.

From a computational point of view, the FIR filter $H^3_{\lambda,\text{FIR}}(z)$ requires $2M+1$ multiplications and $2M$ additions (each T seconds), while the evaluation of each moving average filter needs two additions and one multiplication. In a sample time T_s, The planning of a d-dimensional B-spline with the proposed method requires at most $(2M+4)\times d$ multiplications and $(2M+6)\times d$ additions, where M is usually a number rather small (between 3 and 8). Therefore the trajectory generator, completely based on FIR filters, is characterized by a remarkable computational efficiency.

The parameter M plays a key role in the filter behavior since it determines the degree of approximation of the generated trajectory with respect to the ideal spline. On the other hand, it is worth to noticing that $H^3_{\lambda,\text{FIR}}(z)$ introduces a delay equal to $M \cdot T$ seconds. For this reason, it is not convenient to take into account values of M too large.

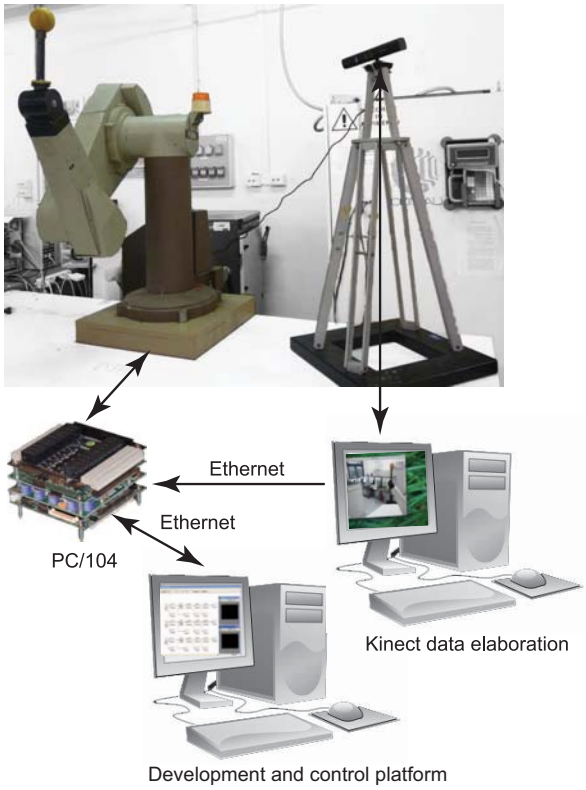

Fig. 6. Experimental setup for trajectory planner validation.

V. Application of the filter to a telemanipulation task

In order to experimentally validate the proposed trajectory planner, a setup that includes a Puma 560 robot manipulator and a low cost 3-D camera system Microsoft Kinect has been arranged, see Fig. 6. The goal of this system is to allow the user to directly control the robot by moving a target object

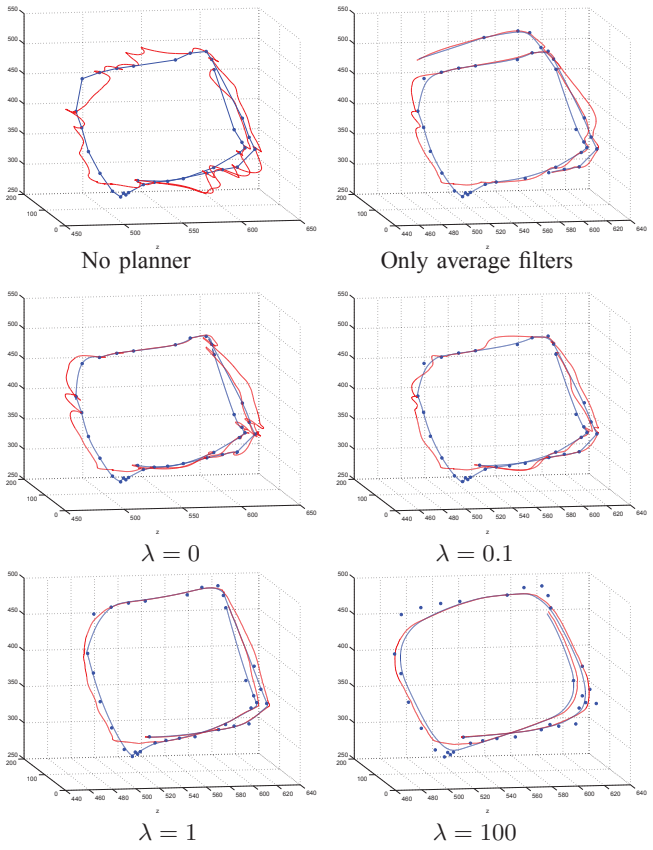

Fig. 7. Workspace position trajectory with different configurations of the trajectory planner (thick line) and robot position (thin line).

in its workspace. Since the focus of the experiment is to test the benefit that the trajectory generator produces, the task consists of simply tracking a ball. The robot controller, that has been completely customized, is based on PC/104 platform running RTAI-linux (for the details refer to [11]) with a sample time of $T_s = 0.001$s, while a standard pc is used to extract the position of the ball from the data coming from the Kinect. The coordinates of this position, expressed in the reference frame of the robot, are sent via ethernet to the robot controller, with a rate of approximatively 10 points per second. Therefore T is equal to 0.1 s. Because of the heavy elaborations of the Kinect data and of the transmission via ethernet it is not possible to guarantee that a new position is always available in T seconds. In this case the old value is maintained, but it is worth noticing the the trajectory planner is implemented in hard real-time.

In order to compare the effects of the trajectory planner for different configurations, the same set of data, collected from the Kinect, has been used for different experiments, but the points have treated as if the task was performed online. In Fig. 7 a segment of trajectory is reported. In the first plot the points are directly provided to the robot controller without any filter. When a new input is given, the control of the robot tries to reach the point as fast as possible and the consequence is a continuous start-and-stop motion with very high errors and large vibrations. If the chain of three FIR filter is inserted between Kinect and robot controller, the behavior of the robot becomes considerably smoother. This is due to the fact that, the chain of filter, without $H^3_{\lambda,\mathrm{FIR}}(z)$, is a special type of smoothing B-spline (with $H^3_{\lambda,\mathrm{FIR}}(z) = 1$) that does not cross the desired via-points but it is continuous in velocity and accelerations. As shown in Fig. 7, the use of $H^3_{\lambda,\mathrm{FIR}}(z)$, with $\lambda = 0$ allows to cross the points without appreciable errors. The use of $H^3_{\lambda,\mathrm{FIR}}(z)$ with higher values of λ contributes to make the trajectory smoother and smoother. For $\lambda = 100$ the precision of the trajectory in points interpolation is rather low but the the tracking error decreases as well.

VI. Conclusions

In this paper, a novel technique based on B-splines that allows to online generating trajectories in the 3-D space has been presented. The planner is completely composed by FIR filters: a first FIR filter computes the control points from the sequence of desired via-points, while a chain of three moving average filters is used to evaluate the cubic B-spline defined by these points. The generator combines the characteristics of FIR filters (simple structure and computational efficiency) with those of B-spline trajectories (smoothness and minimum curvature). In particular, since the so-called smoothing B-splines have been considered, the filter allows to find a tradeoff between the possibility of exactly crossing the given via-points and the smoothness of the resulting trajectory. A simple teleoperation task with a Puma 560 industrial manipulator has been used for experimentally validating the proposed method.

References

[1] L. Biagiotti and C. Melchiorri, *Trajectory Planning for Automatic Machines and Robots*, 1st ed. Heidelberg, Germany: Springer, 2008.

[2] C.-S. Lin, P.-R. Chang, and J. Luh, "Formulation and optimization of cubic polynomial joint trajectories for industrial robots," *IEEE Transaction on Automatic Control*, vol. 28, no. 12, pp. 1066–1074, 1983.

[3] B. Cao, G. Dodds, and G. Irwin, "Constrained time-efficient and smooth cubic spline trajectory generation for industrial robots," *Proceedings of IEE Conference on Control Theory and Applications*, vol. 144, pp. 467–475, 1997.

[4] A. Piazzi and A. Visioli, "Global minimum-jerk trajectory planning of robot manipulator," *IEEE Transaction on Industrial Electronics*, vol. 47, no. 1, pp. 140–149, 2000.

[5] L. Biagiotti and C. Melchiorri, "Input shaping via b-spline filters for 3-d trajectory planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, California, September 25-30 2011.

[6] ——, "B-spline based filters for multi-point trajectories planning," in *2010 IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, May 3-8 2010.

[7] T. Lim and M. Macleod, "On-line interpolation using spline functions," *Signal Processing Letters, IEEE*, vol. 3, no. 5, pp. 144–146, May.

[8] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-time signal processing (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1999.

[9] G. Feng, "Data smoothing by cubic spline filters," *Signal Processing, IEEE Transactions on*, vol. 46, no. 10, pp. 2790–2796, Oct.

[10] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part II-efficient design and applications," *IEEE Transaction on Signal Processing*, vol. 41(2), pp. 834–848, 1993.

[11] G. Palli, L. Biagiotti, and C. Melchiorri, "An open source distributed platform for the control of the puma 560 manipulator," in *Proc. 9th Real Time Linux Workshop*, Linz (AUT), 2007, pp. 205–210.