A matrix-based optimization algorithm for shape deformation using parametric curves*

L. Hilario[(1)] A. Falcó[(1)] N. Montés[(2)]

M. C. Mora[(3)]

[(1)]Departamento de Ciencias, Físicas, Matemáticas y de la Computación
Universidad CEU Cardenal Herrera
San Bartolomé 55, 46115 Alfara del Patriarca (Valencia), Spain.

[(2)]Departamento de Ingeniería de la Edificación y Producción Industrial
Universidad CEU Cardenal Herrera
San Bartolomé 55 46115 Alfara del Patriarca (Valencia), Spain.

[(3)]Departamento de Ingeniería Mecánica y Construcción
Universitat Jaume I
Avd. Vicent Sos Baynat s/n 12071 Castellón, Spain

Abstract

In this paper we propose a tensor based description of the Bézier Shape Deformation (BSD) algorithm, denoted T-BSD. The BSD algorithm is a well-known technique, based on the deformation of a Bézier curve through a field of vectors. A critical point in the use of real-time applications is the cost in computational time. Recently, the use of tensors in numerical methods has been increasing because they drastically reduce computational costs. Our formulation based in tensors T-BSD provides an efficient reformulation of the BSD algorithm. More precisely, the evolution of the execution time with respect to the number of curves of the BSD algorithm is an exponentially increasing curve. As the numerical experiments shown, the T-BSD algorithm transforms this evolution into a linear one. This fact allows to compute the deformation of a Bézier with a much lower computational cost.

*Keywords: Tensor product, Bézier Curves, Parametric Curve Deformation

1 Introduction

One of the most important facts in engineering applications is the cost in computational time. A critical point appearing in this context is related to real-time processes. In consequence, one of the main goals is to develop algorithms that reduce, as much as possible, the execution time of existing real-time algorithms.

Lately, interest in numerical methods that make use of tensors has increased because they drastically reduce computational costs. It is particularly useful for high-dimensional spaces where one must pay attention to the numerical cost (in time and storage).

A first family of applications using tensor decompositions concerns the extraction of information from complex data. It has been used in many areas such as psychometrics [13, 5], chemometrics [2], analysis of turbulent flows [3], image analysis and pattern recognition [14], data mining... Another family of applications concerns the compression of complex data (for storage or transmission), also introduced in many areas such as signal processing [10] or computer vision [16]. A survey of tensor decompositions in multilinear algebra and an overview of possible applications can be found in the review paper [9]. In the above applications, the aim is to compress the best as possible the information. The use of tensor product approximations is also receiving a growing interest in numerical analysis for the solution of problems defined in high-dimensional tensor spaces, such as PDEs arising in stochastic calculus [1, 4, 7] (e.g., Fokker-Planck equation), stochastic parametric PDEs arising in uncertainty quantification with spectral approaches [11, 6, 12], and quantum chemistry (cf., e.g., [15]). Details can be found in [8].

On the other hand we recall that parametric curves are extensively used in Computer Aided Geometric Design (CAGD). The different engineering applications that exist are due to the useful mathematical properties of this kind of curves. The most common parametric curves in these applications are, among others, Bézier, B-Splines, NURBS and Rational Bézier, and every one of them has many special properties. A recently research topic in the CAGD framework is the study of shape deformations in parametric curves. There are different ways to compute these deformations depending on the parametric curve under consideration (see [24, 25, 26, 27, 28, 29] for NURBS and [20, 21, 22, 23] for B-Splines). In particular, in [31] a deformation of a Bézier was introduced by means of a constrained optimization problem related to a discrete coefficient norm. Later in [32] a new technique to deform the shape of a Bézier curve was introduced. This technique was improved including a set of concatenated Bézier curves and more constraints to the optimization problem. This improvement was applied in the numerical simulation of Liquid Composite Moulding Processes [33] and also in the path planning problem in mobile robotics [41, 42].

The main goal of this paper is to introduce tensor calculus in order to improve the procedure described in [33, 41, 42]. The tensor reformulation of the algorihtm is called T-BSD. As a result, the computational cost is reduced to obtain a suitable real-time performance.

This paper is organized as follows. In Section 2 we give preliminary definitions and results about tensors, also some useful properties are introduced. In Section 3 the algorithm for shape deformation using a basis of parametric curve is developed. In Section 4 the T-BSD algorithm using a set of Bézier curves concatenated is defined . In Section 5 the comparative between BSD and T-BSD algorithm is shown. In Section 6 two applications of the T-BSD algorithm are described. Finally, in Section 7 we provide some conclusions about the present

work.

2 Definitions and preliminary results

First of all we introduce some of the notation used in this paper,(see [17], [18] or [19] for more details). We denote the set of $(n \times m)$-matrices by $\mathbb{R}^{n\times m}$, and the transpose of a matrix A is denoted A^T. By $\langle \mathbf{x}, \mathbf{y}\rangle$ we denote the usual Euclidean inner product given by $\mathbf{x}^T\,\mathbf{y} = \mathbf{y}^T\,\mathbf{x}$ and its corresponding 2-norm, $\|x\|_2 = \langle \mathbf{x}, \mathbf{x}\rangle^{1/2}$. The matrix I_n is the $(n \times n)$-identity matrix and when the dimension is clear from the context, we simply denote it by I.

Now, we recall the definition and some properties of the Kronecker product. The Kronecker product of $A \in \mathbb{R}^{n_1'\times n_1}$ and $B \in \mathbb{R}^{n_2'\times n_2}$, written $A \otimes B$, is the tensor algebraic operation defined as

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n_1'}B \\ a_{21}B & a_{22}B & \cdots & a_{2n_1'}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{n_11}B & a_{n_12}B & \cdots & a_{n_1n_1'}B \end{bmatrix} \in \mathbb{R}^{n_1'n_2'\times n_1n_2}.$$

Also, the Kronecker product of two matrices $A \in \mathbb{R}^{n_1'\times n_1}$ and $B \in \mathbb{R}^{n_2'\times n_2}$, can be defined as $A \otimes B \in \mathbb{R}^{n_1'n_2'\times n_1n_2}$, where

$$(A \otimes B)_{(j_1-1)n_2'+j_2;(i_1-1)n_2+i_2} = A_{j_1;i_1}B_{j_2;i_2}.$$

Finally, we list some of the well-know properties of the Kronecker product.

(T1) $A \otimes (B \otimes C) = (A \otimes B) \otimes C$.

(T2) $(A + B) \otimes (C + D) = (A \otimes C) + (B \otimes C) + (A \otimes D) + (B \otimes D)$.

(T3) If $A + B$ and $C + D$ exist, $AB \otimes CD = (A \otimes C)(B \otimes D)$.

(T4) If A and B are non-singular, $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$.

(T5) If $(A \otimes B)^T = A^T \otimes B^T$.

(T6) If A and B are banded, then $A \otimes B$ is banded.

(T7) If A and B are symmetric, then $A \otimes B$ is symmetric.

(T8) If A and B are definite positive, then $A \otimes B$ is definite positive.

Let $A = [\mathbf{A}_1 \cdots \mathbf{A}_n]$ be an $m \times n$ matrix where $\mathbf{A}_j$ is its j-th column vector. Then $\operatorname{vec} A$ is the $mn \times 1$ vector

$$\operatorname{vec} A = \begin{bmatrix} \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_n \end{bmatrix}.$$

Thus the vec operator transforms a matrix into a vector by stacking the columns of the matrix one underneath the other. Notice that $\operatorname{vec} A = \operatorname{vec} B$ does not imply $A = B$, unless A and B are matrices of the same order. The following properties are useful:

(V1) $\operatorname{vec} \mathbf{u}^T = \operatorname{vec} \mathbf{u} = \mathbf{u}$, for any column vector $\mathbf{u}$.

(V2) $\operatorname{vec} \mathbf{u}\mathbf{v}^T = \mathbf{v} \otimes \mathbf{u}$, for any two column vectors $\mathbf{u}$ and $\mathbf{v}$ (not necessarily of the same order).

(V3) Let A, B and C be three matrices such that the matrix product ABC is defined. Then,

$$\operatorname{vec} ABC = (C^T \otimes A)\operatorname{vec} B. \tag{1}$$

Definition 1. *Let $F : \mathbb{R}^{n\times q} \longrightarrow \mathbb{R}^{m\times p}$ be a differentiable function. The Jacobian matrix of F at X is the $mp \times nq$ matrix*

$$DF(X) = \frac{\partial \operatorname{vec} F(X)}{\partial (\operatorname{vec} X)^T}.$$

Clearly, $DF(X)$ is a straightforward matrix generalization of the traditional definition of the Jacobian matrix and all properties of Jacobian matrices are preserved. Thus, the above definition reduces the study of functions of matrices to the study of vector functions of vectors, since it allows $F(X)$ and X only in their vectorized forms $\operatorname{vec} F$ and $\operatorname{vec} X$. The next properties will be useful (see Chapter 9 in [18])

(P1) Assume $\mathbf{y} = f(X) = X\,\mathbf{u}$, such that $\mathbf{u}$ is a vector of constants, here $f : \mathbb{R}^{n\times m} \longrightarrow \mathbb{R}^n$. Then the Jacobian matrix is $D(X\mathbf{u}) = \mathbf{u}^T \otimes I \in \mathbb{R}^{n\times nm} \cong \mathbb{R}^{1\times m} \otimes \mathbb{R}^{n\times n}$.

(P2) Assume $y = f(\mathbf{x}) = \mathbf{x}^T\,\mathbf{x}$, here $f : \mathbb{R}^n \longrightarrow \mathbb{R}$. then $D(\mathbf{x}^T\,\mathbf{x}) = 2\mathbf{x}^T \in \mathbb{R}^{1\times n}$

(P3) Let $F : \mathbb{R}^{n\times m} \longrightarrow \mathbb{R}^{p\times q}$ be defined as $F(X) = AXB$ where $A \in \mathbb{R}^{p\times n}$ and $B \in \mathbb{R}^{m\times q}$ are matrices of constants. Then

$$DF(X) = B^T \otimes A. \tag{2}$$

Theorem 1 (chain rule)**.** *Let S be a subset of $\mathbb{R}^{n\times q}$ and assume that $F : S \longrightarrow \mathbb{R}^{m\times p}$ is differentiable at an interior point C of S. Let T be a subset of $\mathbb{R}^{m\times p}$ such that $F(X) \in T$ for all $X \in S$, and assume that $G : T \longrightarrow \mathbb{R}^{r\times s}$ is differentiable at an interior point $B = F(C) \in T$. Then the composite function $H : S \longrightarrow \mathbb{R}^{r\times s}$ defined by $H(X) = G(F(X))$ is differentiable at C, and*

$$DH(C) = (DG(B))(DF(C)).$$

The following theorem will be useful.

Theorem 2. *Let A be a $(p+q)\times(p+q)$-matrix such that*

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ -A_{1,2}^T & 0 \end{bmatrix} \text{ where } A_{1,1} \in \mathbb{R}^{p\times p} \text{ and } A_{1,2} \in \mathbb{R}^{p\times q}.$$

Assume that $A_{1,1}$ is non-singular, that is, invertible and $\operatorname{rank} A_{1,2} = q$. *Then A es non-singular and*

$$A^{-1} = \begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{bmatrix},$$

where $X_{2,2} = (A_{1,2}^T A_{1,1} A_{1,2})^{-1}, X_{2,1} = X_{2,2} A_{1,2}^T A_{1,1}^{-1}, X_{1,1} = A_{1,1}^{-1} - A_{1,1}^{-1} A_{1,2} X_{2,1}$ and $X_{1,2} = -A_{1,1}^{-1} A_{1,2} X_{2,2}$.

Proof. First, observe that the rank $A = p + q$ because rank $A_{1,1} = p$ and rank $A_{1,2} = q$. To compute A^{-1} we need to solve the matrix linear system

$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ -A_{1,2}^T & 0 \end{bmatrix} \begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{bmatrix} = I_{p+q},$$

that is,

$$A_{1,1}X_{1,1} + A_{1,2}X_{2,1} = I_p, \tag{3}$$
$$A_{1,1}X_{1,2} + A_{1,2}X_{2,2} = 0_{p\times q}, \tag{4}$$
$$-A_{1,2}^T X_{1,1} = 0_{p\times q}, \tag{5}$$
$$-A_{1,2}^T X_{1,2} = I_q. \tag{6}$$

Since $A_{1,1}$ is invertible, from (3) and (5) we obtain

$$A_{1,2}^T A_{1,1}^{-1} A_{2,1} X_{1,2} = A_{1,2}^T A_{1,1}^{-1}$$

and hence

$$X_{2,1} = (A_{1,2}^T A_{1,1} A_{1,2})^{-1} A_{1,2}^T A_{1,1}^{-1},$$

because rank $(A_{1,2}^T A_{1,1} A_{1,2}) = q$. In a similar way, using (4) and (6) we have

$$-I_q + A_{1,2}^T A_{1,1}^{-1} A_{1,2} X_{2,2} = 0_{q\times q}$$

and then

$$X_{2,2} = (A_{1,2}^T A_{1,1}^{-1} A_{1,2})^{-1}.$$

To end the proof observe that (3) and (4) are equivalent to

$$X_{1,1} = A_{1,1}^{-1} - A_{1,1}^{-1} A_{1,2} X_{2,1},$$
$$X_{1,2} = -A_{1,1}^{-1} A_{1,2} X_{2,2},$$

respectively. □

3 A matrix-based optimization algorithm for shape deformation using a basis of parametric curves

Our main goal is to fit a parametric curve through a field of vectors by using a basis of parametric curves. There are different techniques to obtain this, in particular an approximation based on the deformation of a Bézier curve is proposed in [33, 41, 42]. The aim is the reduction of the cost in computational time of this algorithm because this is a critical point in real-time applications.

We will consider for each fixed $n \geq 1$ a finite dimensional basis $\{B_{0,n}, \ldots, B_{n,n}\} \subset L^2[0,1]$ and $\Omega \subset \mathbb{R}^2$ a compact and convex set. Now, assume that for each time $t \in (0, t_{end}]$ we have a matrix of Target Points

$$T_r(t) = \begin{bmatrix} \mathbf{T}_r^1(t) & \cdots & \mathbf{T}_r^r(t) \end{bmatrix} \in \mathbb{R}^{2\times r}$$

where $T_r^j(t) \in \Omega, 1 \leq j \leq r$. Our main goal is construct a map from $[0, t_{end}]$ to $\mathcal{C}^1([0,1]; \mathbb{R}^2)$ given by

$$t \mapsto \boldsymbol{\alpha}_t^n(u) = \sum_{i=0}^{n} \mathbf{P}_i(t) B_{i,n}(u); \quad u \in [0,1], \tag{7}$$

where

$$P_n(t) = \left[\begin{array}{ccc} \mathbf{P}_n^0(t) & \cdots & \mathbf{P}_n^n(t) \end{array} \right] \in \mathbb{R}^{2\times(n+1)},$$

for each fixed t is a finite set of control points, $P_n(0)$ is previously known and

$$\{\mathbf{T}_r^1(t), \cdots, \mathbf{T}_r^r(t)\} \subset \boldsymbol{\alpha}_t^n([0,1]),$$

for each $t \in (0, t_{end}]$. Observe that we can write (7) in a equivalent matrix form,

$$\boldsymbol{\alpha}_t^n(u) = P_n(t)\, \mathbf{B}_n(u); u \in [0,1] \tag{8}$$

where

$$\mathbf{B}_n(u) = \left[\begin{array}{ccc} B_{0,n}(u) & \cdots & B_{n,n}(u) \end{array} \right]^T \in \mathbb{R}^{(n+1)\times 1}. \tag{9}$$

Since $\boldsymbol{\alpha}_t^n(u) \in \mathbb{R}^2$ we can write its standard euclidean norm as

$$\|\boldsymbol{\alpha}_t^n(u)\|_2^2 = \mathbf{B}_n(u)^T\, P_n(t)^T\, P_n(t)\, \mathbf{B}_n(u), \tag{10}$$

then, for each fixed t, the energy of the u-parametrized curve $\boldsymbol{\alpha}_t^n$ in $C([0,1], \mathbb{R}^2)$ can be given by its $L^2([0,1], \mathbb{R}^2)$-norm, that is,

$$\|\boldsymbol{\alpha}_t^n\|_{\Delta_2} = \left(\int_0^1 \|\boldsymbol{\alpha}_t^n(u)\|_2^2\, du \right)^{1/2} = \left(\int_0^1 \mathbf{B}_n(u)^T\, P_n(t)^T\, P_n(t)\, \mathbf{B}_n(u)\, du \right)^{1/2}. \tag{11}$$

Assume that for some $t \in [0, t_{end})$ we previously know $\boldsymbol{\alpha}_t^n$ as $\boldsymbol{\alpha}_t^n(u) = P_n(t)\, \mathbf{B}_n(u)$. Then it moves in a given small interval of time Δt to a parametric curve $\boldsymbol{\alpha}_{t+\Delta t}^n$, by using a set of perturbations for each control point, namely

$$X_n(t+\Delta t) = \left[\begin{array}{ccc} \mathbf{X}_n^0(t+\Delta t) & \cdots & \mathbf{X}_n^n(t+\Delta t) \end{array} \right] \in \mathbb{R}^{2\times(n+1)}, \tag{12}$$

The resultant parametric curve $\boldsymbol{\alpha}_{t+\Delta t}^n$ will be given by,

$$\boldsymbol{\alpha}_{t+\Delta t}^n(u) = P_n(t+\Delta t)\, \mathbf{B}_n(u); \quad u \in [0,1]. \tag{13}$$

where

$$P_n(t+\Delta t) := P_n(t) + X_n(t+\Delta t). \tag{14}$$

To compute $X_n(t+\Delta t)$ we will use the following least action principle: the curve minimize the energy to move from $\boldsymbol{\alpha}_t^n$ to $\boldsymbol{\alpha}_{t+\Delta t}^n$ and it has to pass through the set of Target Points $\{\mathbf{T}_r^1(t+\Delta t), \ldots, \mathbf{T}_r^r(t+\Delta t)\}$ for a given set of parameter values, namely

$$0 = u_1^r < u_2^r < \cdots < u_{r-1}^r < u_r^r = 1.$$

More precisely, we would like to find $\boldsymbol{\alpha}_{t+\Delta t}^n$ such that

$$\begin{aligned} &\min \|\boldsymbol{\alpha}^n_{t+\Delta t} - \boldsymbol{\alpha}^n_t\|^2_{\Delta_2} \\ &\text{s. t. } \ \boldsymbol{\alpha}^n_{t+\Delta t}(u^r_j) = \mathbf{T}^j_r(t+\Delta t) \ \text{ for } 1 \le j \le r \ \text{ and } \ r \le n-1. \end{aligned} \tag{15}$$

In order to write (15) in a equivalent matrix form we introduce the following notation. Let

$$B^r_n = \left[\begin{array}{ccc} \mathbf{B}_n(u^r_1) & \cdots & \mathbf{B}_n(u^r_r) \end{array} \right] \in \mathbb{R}^{(n+1)\times r},$$

where we assume that

$$\operatorname{rank} B^r_n = r = \min\{n+1, r\}, \tag{16}$$

holds for the set of parameter values $\{u^r_1, u^r_2, \ldots, u^r_{r-1}, u^r_r\}$. Finally, we consider the matrix function $\Phi_n : \mathbb{R}^{2\times(n+1)} \to \mathbb{R}$, defined by

$$\Phi_n(X_n(t+\Delta t)) = \int_0^1 \mathbf{B}_n(u)^T \, X_n(t+\Delta t)^T X_n(t+\Delta t) \, \mathbf{B}_n(u) \, du. \tag{17}$$

Then the miminization problem (15) can be written in a matrix form as:

$$\min_{X_n(t+\Delta t)\in\mathbb{R}^{2\times(n+1)}} \Phi_n(X_n(t+\Delta t)) \tag{18}$$

$$\text{s. t. } \ (P_n(t) + X_n(t+\Delta t))B^r_n = T_r(t+\Delta t). \tag{19}$$

By using the vec operator in (19) and the property V3 defined in the equation (1), we obtain a useful equivalent formulation written as

$$((B^r_n)^T \otimes I_2) \operatorname{vec} X_n(t+\Delta t) = \operatorname{vec} T_r(t+\Delta t) - \operatorname{vec}(P_n(t)B^r_n). \tag{20}$$

Note that the set of constrains of this problem 19 is linear where $(B^r_n)^T \otimes I_2 \in \mathbb{R}^{2r\times 2(n+1)}$. In consequence, the map Φ_n is defined over a convex set. Thus, by proving the convexity of Φ_n, each stationary point of Φ_n over the constrained set will give us an absolute minimum. In particular, the following proposition give us the first and the second derivative of Φ_n.

Proposition 1. *The following statements hold:*

(a) $D\Phi_n(X_n(t+\Delta t)) = 2\displaystyle\int_0^1 (X_n(t+\Delta t)\,\mathbf{B}_n(u))^T(\mathbf{B}_n(u)^T \otimes I_2)\, du, \in \mathbb{R}^{1\times 2(n+1)}$.

(b) $(D\Phi_n(X_n(t+\Delta t)))^T = 2\left(\displaystyle\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u) \otimes I_2)\, du\right) \operatorname{vec} X_n(t+\Delta t)$.

(c) $D^2\Phi_n(X_n(t+\Delta t)) = 2\displaystyle\int_0^1 (\mathbf{B}_n(u)\mathbf{B}_n(u)^T \otimes I_2)\, du = 2\left(\int_0^1 \mathbf{B}_n(u)\mathbf{B}_n(u)^T\, du\right) \otimes I_2$.
Moreover, $D^2\Phi_n(X_n(t+\Delta t)) \in \mathbb{R}^{2(n+1)\times 2(n+1)}$ *is a definite positive symmetric matrix and hence* Φ_n *is a convex function over each convex set* Ω.

Proof. First, we observe that

$$D\Phi_n(X_n(t+\Delta t)) = \int_0^1 D\left(\mathbf{B}_n(u)^T X_n(t+\Delta t)^T X_n(t+\Delta t)\,\mathbf{B}_n(u)\right)\,du. \tag{21}$$

Let us consider $\mathbf{y}_n = F(X_n(t+\Delta t)) = X_n(t+\Delta t)\mathbf{B}_n(u)$ and $G(\mathbf{y_n}) = \mathbf{y}_n^T\mathbf{y}_n$. Then, $DF(X_n(t+\Delta t)) = \mathbf{B}_n(u)^T \otimes I_2$ and $DG(\mathbf{y_n}) = 2\mathbf{y}_n^T$. Thus, by using Theorem 1 we obtain that

$$\begin{aligned} D\left(\mathbf{B}_n(u)^T X_n(t+\Delta t)^T X_n(t+\Delta t)\,\mathbf{B}_n(u)\right) &= 2\mathbf{y}_n^T\left(\mathbf{B}_n(u)^T \otimes I_2\right) \\ &= 2\left(X_n(t+\Delta t)\mathbf{B}_n(u)\right)^T\left(\mathbf{B}_n(u)^T \otimes I_2\right), \end{aligned}$$

and this follows statement (a). By using the fact that,

$$(D\Phi_n(X_n(t+\Delta t)))^T = 2\int_0^1 (\mathbf{B}_n(u)\otimes I_2)(X_n(t+\Delta t)\,\mathbf{B}_n(u))\,du \in \mathbb{R}^{2(n+1)\times 1} \tag{22}$$

and taking the vec operator we obtain that,

$$(D\Phi_n(X_n(t+\Delta t)))^T = 2\left(\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u)\otimes I_2)\,du\right)\operatorname{vec} X_n(t+\Delta t), \tag{23}$$

an it follows (b). To prove (c) note that

$$D^2\left(\mathbf{B}_n(u)^T X_n(t+\Delta t)^T X_n(t+\Delta t)\,\mathbf{B}_n(u)\right) = 2\left(\mathbf{B}_n(u)^T \otimes I_2\right)^T\left(\mathbf{B}_n(u)^T \otimes I_2\right). \tag{24}$$

Since $(\mathbf{B}_n(u)\mathbf{B}_n(u)^T \otimes I_2)$ is definite positive, we obtain that $D^2\Phi_n(X_n(t+\Delta t))$ is also definite positive for all $X_n(t+\Delta t) \in \mathbb{R}^{2\times(n+1)}$. □

Moreover, we have the following lemma.

Lemma 1. *The matrix* $\displaystyle\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u)\otimes I_2)\,du$ *is invertible.*

Proof. Observe that

$$\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u)\otimes I_2)\,du =$$

$$= \begin{bmatrix} \int_0^1 B_{0,n}(u)B_{0,n}(u)du & \cdots & \int_0^1 B_{n,n}(u)B_{0,n}(u)du \\ \int_0^1 B_{0,n}(u)B_{1,n}(u)du & \cdots & \int_0^1 B_{n,n}(u)B_{1,n}(u)du \\ \vdots & \ddots & \vdots \\ \int_0^1 B_{0,n}(u)B_{n,n}(u)du & \cdots & \int_0^1 B_{n,n}(u)B_{n,n}(u)du \end{bmatrix} \otimes I_2.$$

Since $\displaystyle\int_0^1 B_{i,n}(u)B_{j,n}(u)du = \langle B_{i,n}, B_{j,n}\rangle_{L^2([0,1];\mathbb{R})}$ we have that

$$G(B_{0,n},\ldots,B_{n,n}) = \begin{bmatrix} \int_0^1 B_{0,n}(u)B_{0,n}(u)du & \cdots & \int_0^1 B_{n,n}(u)B_{0,n}(u)du \\ \int_0^1 B_{0,n}(u)B_{1,n}(u)du & \cdots & \int_0^1 B_{n,n}(u)B_{1,n}(u)du \\ \vdots & \ddots & \vdots \\ \int_0^1 B_{0,n}(u)B_{n,n}(u)du & \cdots & \int_0^1 B_{n,n}(u)B_{n,n}(u)du \end{bmatrix}$$

is the Gramian matrix of the basis $\{B_{0,n}, \ldots, B_{n,n}\}$. From Lemma 7.5 of [30] we have that $G(B_{0,n}, \ldots, B_{n,n})$ is a non-singular matrix and hence

$$\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u) \otimes I_2)\, du = G(B_{0,n}, \ldots, B_{n,n}) \otimes I_2$$

in invertible. □

In order to characterize the minimum of (18) and (20) we construct the associate lagrangian

$$\begin{aligned}\mathcal{L}(X_n(t+\Delta t), \boldsymbol{\mu}) =& \Phi_n(X_n(t+\Delta t)) - \boldsymbol{\mu}^T[(B_n^r)^T \otimes I_2) \operatorname{vec} X_n(t+\Delta t) - \\ & - \operatorname{vec} T_r(t+\Delta t) + \operatorname{vec}(P_n(t) B_n^r)],\end{aligned}$$

where $\boldsymbol{\mu} = [\mu_1 \cdots \mu_{2r}]^T \in \mathbb{R}^{2r}$. The first order optimality conditions are (20) and

$$D\Phi_n(X_n(t+\Delta t)) - \boldsymbol{\mu}^T((B_n^r)^T \otimes I_2) = 0, \tag{25}$$

which is obtained by using Proposition 1 (a) and the property (P3) defined by the equation 2. The equation (25) is equivalent to

$$(D\Phi_n(X_n(t+\Delta t)))^T - (I_2 \otimes B_n^r)\boldsymbol{\mu} = \mathbf{0}. \tag{26}$$

Proposition 1 (b) allows us to write the first order optimality conditions as:

$$\begin{aligned} & 2\left(\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u) \otimes I_2)\, du\right) \operatorname{vec} X_n(t+\Delta t) - (I_2 \otimes B_n^r)\boldsymbol{\mu} = \mathbf{0} \\ & ((B_n^r)^T \otimes I_2) \operatorname{vec} X_n(t+\Delta t) = \operatorname{vec} T_r(t+\Delta t) - \operatorname{vec}(P_n(t) B_n^r), \end{aligned} \tag{27}$$

that we can write in matrix form as

$$A\mathbf{z}(t+\Delta t) = \mathbf{f}(t), \tag{28}$$

where

$$A = \begin{bmatrix} 2\left(\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u) \otimes I_2)\, du\right) & -(I_2 \otimes B_n^r) \\ ((B_n^r)^T \otimes I_2) & 0 \end{bmatrix} \in \mathbb{R}^{(2(n+1)+2r)\times(2(n+1)+2r)},$$

$$\mathbf{z}(t+\Delta t) = \begin{bmatrix} \operatorname{vec} X_n(t+\Delta t) \\ \boldsymbol{\mu} \end{bmatrix} \text{ and } \mathbf{f}(t) = \begin{bmatrix} 0 \\ \operatorname{vec} T_r(t+\Delta t) - \operatorname{vec}(P_n(t) B_n^r) \end{bmatrix},$$

which are in $\mathbb{R}^{(2(n+1)+2r)}$.

Proposition 2. *Assume that* $\operatorname{rank} B_n^r = r$. *Then the matrix A is invertible.*

Proof. First at all we remark that

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ -A_{1,2}^T & 0 \end{bmatrix} \in \mathbb{R}^{2(n+1)+2r \times 2(n+1)+2r},$$

where

$$A_{11} = 2\left(\int_0^1 (\mathbf{B}_n(u)^T \otimes \mathbf{B}_n(u) \otimes I_2)\, du\right) \in \mathbb{R}^{2(n+1)\times 2(n+1)},$$

and

$$A_{1,2} = -(I_2 \otimes B_n^r) = \begin{bmatrix} -B_n^r & 0 \\ 0 & -B_n^r \end{bmatrix} \in \mathbb{R}^{2(n+1)\times 2r}.$$

From Lemma 1 we known that $A_{1,1}$ is a non-singular matrix. Since $\operatorname{rank} A_{1,2} = 2r$ because $\operatorname{rank} B_n^r = r$, from Theorem 2 the proposition follows. □

Now, the algorithm is the following.

1. Construct the matrix A.
2. Consider $\Delta t = \dfrac{t_{end}}{N-1}$ for a fixed $N \geq 2$, and write $t_j = (j-1)\Delta t$ for $1 \leq j \leq N$.
3. Obtain the initial control points $P_n(t_1)$ and a sample of Target Points $\{T_r(t_1), \ldots, T_r(t_N)\} \subset \Omega \subset \mathbb{R}^{2\times r}$.
4. For $j = 1$ to N
   (a) Compute $\mathbf{z}(t_{j+1})$ as the solution of the linear system $A\mathbf{z}(t_{j+1}) = \mathbf{f}(t_j)$;
   (b) Obtain $X_n(t_{j+1})$ from $\mathbf{z}(t_{j+1})$;
   (c) Compute the new control points $P_n(t_{j+1}) = P_n(t_j) + X_n(t_{j+1})$;

4 A matrix-based optimization algorithm for Bézier Shape Deformation

Now, we consider that the curve $\boldsymbol{\alpha}_t \in \mathcal{C}([0,1];\Omega)$ is now described by a finite set of concatenated parametrized Bézier curves $\boldsymbol{\alpha}_t^{n_1}, \ldots, \boldsymbol{\alpha}_t^{n_k}$ constructed with basis functions of dimensions $n_1, \ldots, n_k$, respectively. By using Section 3, each of these curves can be written as

$$\boldsymbol{\alpha}_t^{n_i}(u) = P_{n_i}(t)\, \mathbf{B}_{n_i}(u); \quad u \in [0,1]; \quad 1 \leq i \leq k \tag{29}$$

where,

$$P_{n_i}(t) = \begin{bmatrix} \mathbf{P}_{n_i}^0(t) & \cdots & \mathbf{P}_{n_i}^{n_i}(t) \end{bmatrix} \in \mathbb{R}^{2\times(n_i+1)}. \tag{30}$$

and

$$\mathbf{B}_{n_i}(u) = \begin{bmatrix} B_{0,n_i}(u) & \cdots & B_{n_i,n_i}(u) \end{bmatrix}^T \in \mathbb{R}^{(n_i+1)\times 1}. \tag{31}$$

We assume that

$$B_{i,n_j}(u) = \binom{n_j}{i} u^i (1-u)^{n_j-i}, \quad i = 0, \ldots n_j$$

are the Bernstein basis polynomials of degree n_j for $1 \leq j \leq k$. Let us consider for $t \in (0, t_{end}]$ and each $1 \leq i \leq k$ a set of r_i-target points

$$T_{r_i}(t) = \left[\begin{array}{ccc} \mathbf{T}^1_{r_i}(t) & \cdots & \mathbf{T}^{r_i}_{r_i}(t) \end{array} \right] \in \mathbb{R}^{2\times r_i}, \tag{32}$$

where $\mathbf{T}^j_{r_i}(t) \in \Omega$ for $1 \leq j \leq r_i$, and

$$\{\mathbf{T}^1_{r_i}(t), \ldots, \mathbf{T}^{r_i}_{r_i}(t)\} \subset \boldsymbol{\alpha}^{n_i}_t([0,1]) \tag{33}$$

for all $t \in (0, t_{end}]$. Moreover, $P_{n_i}(0)$ is previously known.

In a similar way as in Section 3, we assume that $\boldsymbol{\alpha}_t$, described by $\{\boldsymbol{\alpha}^{n_i}_t\}^k_{i=1}$, is given. Then we would like to construct $\boldsymbol{\alpha}_{t+\Delta t}$ from $\{\boldsymbol{\alpha}^{n_i}_{t+\Delta t}\}^k_{i=1}$, as follows. Consider

$$\boldsymbol{\alpha}^{n_i}_{t+\Delta t}(u) = P_{n_i}(t+\Delta t)\, \mathbf{B}_{n_i}(u); \quad u \in [0,1] \tag{34}$$

where,

$$P_{n_i}(t+\Delta t) := P_{n_i}(t) + X_{n_i}(t+\Delta t) \tag{35}$$

and

$$X_{n_i}(t+\Delta t) = \left[\begin{array}{ccc} \mathbf{X}^0_{n_i}(t+\Delta t) & \cdots & \mathbf{X}^{n_i}_{n_i}(t+\Delta t) \end{array} \right] \in \mathbb{R}^{2\times(n_i+1)}, \tag{36}$$

for each $1 \leq i \leq k-1$.

Since (33) holds, for each $1 \leq i \leq k$ we will consider

$$0 = u^{r_i}_1 < u^{r_i}_2 \cdots < u^{r_i-1}_{r_i-1} < u^{r_i}_{r_i} = 1$$

and the matrix

$$B^{r_i}_{n_i} = \left[\begin{array}{ccc} \mathbf{B}_{n_i}(u^{r_i}_1) & \cdots & \mathbf{B}_{n_i}(u^{r_i}_{r_i}) \end{array} \right] \in \mathbb{R}^{(n_i+1)\times r_i}. \tag{37}$$

Since $\boldsymbol{\alpha}^{n_i}_{t+\Delta t}(u^{r_i}_j) = \mathbf{T}^j_{r_i}(t+\Delta t)$, for $1 \leq j \leq r_i$ and $1 \leq i \leq k$, we have

$$(P_{n_i}(t) + X_{n_i}(t+\Delta t))B^{r_i}_{n_i} = T_{r_i}(t+\Delta t) \text{ for } 1 \leq i \leq k. \tag{38}$$

The continuity of $\boldsymbol{\alpha}_t$ given by $\boldsymbol{\alpha}^{n_i}_t(1^-) = \boldsymbol{\alpha}^{n_{i+1}}_t(0^+)$ for $1 \leq i \leq k-1$, implies that

$$\mathbf{P}^{n_i}_{n_i}(t) = \mathbf{P}^0_{n_{i+1}}(t) \tag{39}$$

holds for $1 \leq i \leq k-1$. Since $\boldsymbol{\alpha}_{t+\Delta t} \in \mathcal{C}([0,1];\Omega)$, from $\boldsymbol{\alpha}^{n_i}_{t+\Delta t}(1^-) = \boldsymbol{\alpha}^{n_{i+1}}_{t+\Delta t}(0^+)$ we have

$$\mathbf{X}^{n_i}_{n_i}(t+\Delta t) = \mathbf{X}^0_{n_{i+1}}(t+\Delta t), \tag{40}$$

for $1 \leq i \leq k-1$. Assume that $\boldsymbol{\alpha}^{n_1}_t(0), \boldsymbol{\alpha}^{n_k}_t(1)$ belong to the boundary of Ω, denoted by $\partial\Omega$, and that

$$\frac{d}{du}\, \boldsymbol{\alpha}^{n_1}_t(u)|_{u=0^+} = \mathbf{V}_1(t), \quad \frac{d}{du}\, \boldsymbol{\alpha}^{n_k}_t(u)|_{u=1^-} = \mathbf{V}_k(t),$$

are given data for all t. This equality and the fact that B_{i,n_j}, for $0 \leq i \leq n_j$ and $1 \leq j \leq k$, are Bernstein polinomials, implies

$$n_1(\mathbf{P}^1_{n_1}(t) - \mathbf{P}^0_{n_1}(t)) = \mathbf{V}_1(t), \quad n_k(\mathbf{P}^{n_k}_{n_k}(t) - \mathbf{P}^{n_{k-1}}_{n_k}(t)) = \mathbf{V}_k(t). \tag{41}$$

In a similar way, since

$$\frac{d}{du}\,\boldsymbol{\alpha}_{t+\Delta t}^{n_1}(u)\big|_{u=0^+} = \mathbf{V}_1(t+\Delta t), \quad \frac{d}{du}\,\boldsymbol{\alpha}_{t+\Delta t}^{n_k}(u)\big|_{u=1^-} = \mathbf{V}_k(t+\Delta t),$$

and (41) hold we obtain

$$\begin{aligned} n_1(\mathbf{X}_{n_1}^1(t+\Delta t) - \mathbf{X}_{n_1}^0(t+\Delta t)) &= \mathbf{V}_1(t+\Delta t) - \mathbf{V}_1(t), \quad (42)\\ n_k(\mathbf{X}_{n_k}^{n_k}(t+\Delta t) - \mathbf{X}_{n_k}^{n_k-1}(t+\Delta t)) &= \mathbf{V}_k(t+\Delta t) - \mathbf{V}_k(t). \quad (43)\end{aligned}$$

To obtain a differentiability condition, that is $\boldsymbol{\alpha}_t \in \mathcal{C}^1([0,1];\Omega)$, we assume

$$\frac{d}{du}\,\boldsymbol{\alpha}_t^{n_i}(u)|_{u=1^-} = \frac{d}{du}\,\boldsymbol{\alpha}_t^{n_{i+1}}(u)|_{u=0^+}\,. \quad (44)$$

and then

$$n_i(\mathbf{P}_{n_i}^{n_i}(t) - \mathbf{P}_{n_i}^{n_i-1}(t)) = n_{i+1}(\mathbf{P}_{n_{i+1}}^1(t) - \mathbf{P}_{n_{i+1}}^0(t)) \quad (45)$$

holds for $1 \le i \le k-1$. In a similar way, if we assume that $\boldsymbol{\alpha}_{t+\Delta t} \in \mathcal{C}^1([0,1];\Omega)$ then,

$$n_i(\mathbf{X}_{n_i}^{n_i}(t+\Delta t) - \mathbf{X}_{n_i}^{n_i-1}(t+\Delta t)) = n_{i+1}(\mathbf{X}_{n_{i+1}}^1(t+\Delta t) - \mathbf{X}_{n_{i+1}}^0(t+\Delta t)) \quad (46)$$

holds for $1 \le i \le k-1$.

To conclude, we would like to compute $X_{n_i}(t+\Delta t) \in \mathbb{R}^{2\times(n_i+1)}$ for $1 \le i \le k$ satisfying

$$\begin{aligned} &\min_{(X_{n_1}(t+\Delta t),\ldots,X_{n_k}(t+\Delta t))} && \textstyle\sum_{i=1}^k \Phi_{n_i}(X_{n_i}(t+\Delta t))\\ \text{s. t.}\quad &(P_{n_i}(t) + X_{n_i}(t+\Delta t))B_{n_i}^{r_i} && = T_{r_i}(t+\Delta t),\ 1 \le i \le k,\\ &\mathbf{X}_{n_i}^{n_i}(t+\Delta t) && = \mathbf{X}_{n_{i+1}}^0(t+\Delta t),\ 1 \le i \le k-1,\\ &n_1(\mathbf{X}_{n_1}^1(t+\Delta t) - \mathbf{X}_{n_1}^0(t+\Delta t)) && = \mathbf{V}_1(t+\Delta t) - \mathbf{V}_1(t),\\ &n_k(\mathbf{X}_{n_k}^{n_k}(t+\Delta t) - \mathbf{X}_{n_k}^{n_k-1}(t+\Delta t)) && = \mathbf{V}_k(t+\Delta t) - \mathbf{V}_k(t),\\ &n_i(\mathbf{X}_{n_i}^{n_i}(t+\Delta t) - \mathbf{X}_{n_i}^{n_i-1}(t+\Delta t)) && = n_{i+1}(\mathbf{X}_{n_{i+1}}^1(t+\Delta t) - \mathbf{X}_{n_{i+1}}^0(t+\Delta t)),\ 1 \le i \le k-1, \end{aligned} \quad (47)$$

Introduce the matrix function

$$\Phi(X_{n_1}(t+\Delta t),\ldots,X_{n_k}(t+\Delta t)) := \sum_{i=1}^k \Phi_{n_i}(X_{n_i}(t+\Delta t)),$$

which is a linear combination with positive coefficients of convex functions. In consequence it is also a convex function over each convex set $\Omega \subset (\mathbb{R}^{2\times(n_1+1)} \times \cdots \times \mathbb{R}^{2\times(n_k+1)})$. Moreover,

$$D\Phi(X_{n_1}(t+\Delta t),\ldots,X_{n_k}(t+\Delta t)) = \left[\begin{array}{ccc} D\Phi_{n_1}(X_{n_1}(t+\Delta t)) & \cdots & D\Phi_{n_k}(X_{n_k}(t+\Delta t)) \end{array}\right],$$

where $D\Phi(X_{n_1}(t+\Delta t),\ldots,X_{n_k}(t+\Delta t)) \in \mathbb{R}^{1\times 2\sum_{i=1}^k (n_i+1)}$ and

$$D^2\Phi(X_{n_1}(t+\Delta t),\ldots,X_{n_k}(t+\Delta t)) = \operatorname{diag}\left(D^2\Phi_{n_1}(X_{n_1}(t+\Delta t)),\ldots,D^2\Phi_{n_k}(X_{n_k}(t+\Delta t))\right).$$

By using Proposition 1, we see that $D^2\Phi(X_{n_1}(t+\Delta t),\ldots,X_{n_k}(t+\Delta t))$ is a definite positive matrix. Now, we would like to write (47) in a more compact notation. To this end we use the following four block matrices. For $1 \le i \le k$ we define

$$R_{n_i} = \left[\begin{array}{ccccc} 0 & \cdots & 0 & 0 & I_2 \end{array}\right] \in \mathbb{R}^{2\times 2(n_i+1)},$$

$$R^*_{n_i} = \left[\begin{array}{cccccc} 0 & \cdots & 0 & -I_2 & I_2 \end{array}\right] \in \mathbb{R}^{2\times 2(n_i+1)},$$

$$L_{n_i} = \left[\begin{array}{cccccc} I_2 & 0 & 0 & \cdots & 0 \end{array}\right] \in \mathbb{R}^{2\times 2(n_i+1)}$$

and

$$L^*_{n_i} = \left[\begin{array}{cccccc} -I_2 & I_2 & 0 & \cdots & 0 \end{array}\right] \in \mathbb{R}^{2\times 2(n_i+1)}.$$

Finally, we denote by

$$0_{n_i} = \left[\begin{array}{cccccc} 0 & 0 & 0 & \cdots & 0 \end{array}\right] \in \mathbb{R}^{2\times 2(n_i+1)},$$

here and for all the above matrices 0 denotes the square matrix

$$\left[\begin{array}{cc} 0 & 0 \\ 0 & 0 \end{array}\right].$$

Then by using the above matrices and the vec operator we can write the set of constrains in (47) as

$$\begin{array}{lcl} ((B^{r_i}_{n_i})^T \otimes I_2)\mathrm{vec}\, X_{n_i}(t+\Delta t) & = & \mathrm{vec}\, T_{r_i}(t) - \mathrm{vec}\,(P_{n_i}(t)B^{r_i}_{n_i}),\ 1 \le i \le k, \\ R_{n_i}\mathrm{vec}\, X_{n_i}(t+\Delta t) & = & L_{n_{i+1}}\mathrm{vec}\, X_{n_{i+1}}(t+\Delta t),\ 1 \le i \le k-1, \\ n_1 L^*_{n_1}\mathrm{vec}\, X_{n_1}(t+\Delta t) & = & \mathbf{V}_1(t+\Delta t) - \mathbf{V}_1(t), \\ n_k R^*_{n_k}\mathrm{vec}\, X_{n_k}(t+\Delta t) & = & \mathbf{V}_k(t+\Delta t) - \mathbf{V}_k(t), \\ n_i R^*_{n_i}\mathrm{vec} X_{n_i}(t+\Delta t) & = & n_{i+1}L^*_{n_{i+1}}\mathrm{vec}\, X_{n_{i+1}}(t+\Delta t),\ 1 \le i \le k-1. \end{array} \tag{48}$$

Now, the Lagrangian function associated to (47) is

$$\begin{array}{rl} & \mathcal{L}(X_{n_1}(t+\Delta t), \ldots, X_{n_k}(t+\Delta t), \boldsymbol{\lambda}^{r_1}_1, \ldots, \boldsymbol{\lambda}^{r_k}_k, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{2k}) \\ = & \sum_{i=1}^k \Phi_{n_i}(X_{n_i}(t+\Delta t)) \\ & - \sum_{i=1}^k (\boldsymbol{\lambda}^{r_i}_i)^T \left[((B^{r_i}_{n_i})^T \otimes I_2)\mathrm{vec}\, X_{n_i}(t+\Delta t) - \mathrm{vec}\, T_{r_i}(t) + \mathrm{vec}\,(P_{n_i}(t)B^{r_i}_{n_i})\right] \\ & - \sum_{i=1}^{k-1} \boldsymbol{\mu}^T_i \left[R_{n_i}\mathrm{vec}\, X_{n_i}(t+\Delta t) - L_{n_{i+1}}\mathrm{vec}\, X_{n_{i+1}}(t+\Delta t)\right] \\ & - \boldsymbol{\mu}^T_k \left[n_1 L^*_{n_1}\mathrm{vec}\, X_{n_1}(t+\Delta t) - \mathbf{V}_1(t+\Delta t) + \mathbf{V}_1(t)\right] \\ & - \boldsymbol{\mu}^T_{k+1} \left[n_k R^*_{n_k}\mathrm{vec}\, X_{n_k} - \mathbf{V}_k(t+\Delta t) + \mathbf{V}_k(t)\right] \\ & - \sum_{i=1}^{k-1} \boldsymbol{\mu}^T_{i+1+k} \left[n_i R^*_{n_i}\mathrm{vec} X_{n_i}(t+\Delta t) - n_{i+1}L^*_{n_{i+1}}\mathrm{vec}\, X_{n_{i+1}}(t+\Delta t)\right], \end{array} \tag{49}$$

where,

$$\boldsymbol{\lambda}^{r_i}_i = \left[\begin{array}{c} \lambda^1_i \\ \vdots \\ \lambda^{2r_i}_i \end{array}\right] \in \mathbb{R}^{2r_i}, \tag{50}$$

for $1 \leq i \leq k$, and

$$\boldsymbol{\mu}_j = \begin{bmatrix} \mu_j^1 \\ \mu_j^2 \end{bmatrix} \in \mathbb{R}^2, \tag{51}$$

for $1 \leq j \leq 2k$. The first order optimality conditions are given by (48),

$$D\Phi_{n_1}(\, X_{n_1}(t+\Delta t)) - (\boldsymbol{\lambda}_1^{r_1})^T((B_{n_1}^{r_1})^T \otimes I_2) - \boldsymbol{\mu}_1^T R_{n_1} - \boldsymbol{\mu}_k^T n_1 L_{n_1}^* - \boldsymbol{\mu}_{k+2}^T n_1 R_{n_1}^* = 0, \tag{52}$$

$$D\Phi_{n_i}(X_{n_i}(t+\Delta t)) - (\boldsymbol{\lambda}_i^{r_i})^T((B_{n_i}^{r_i})^T \otimes I_2) - \boldsymbol{\mu}_i^T R_{n_i} + \boldsymbol{\mu}_{i-1}^T L_{n_i} - \boldsymbol{\mu}_{k+1+i}^T n_i R_{n_i}^* + \boldsymbol{\mu}_{k+i}^T n_i L_{n_i}^* = 0, \tag{53}$$

for $2 \leq i \leq k-1$, and

$$D\Phi_{n_k}(X_{n_k}(t+\Delta t)) - (\boldsymbol{\lambda}_k^{r_k})^T((B_{n_k}^{r_k})^T \otimes I_2) + \boldsymbol{\mu}_{k-1}^T L_{n_k} - \boldsymbol{\mu}_{k+1}^T n_k R_{n_k}^* + \boldsymbol{\mu}_{2k}^T n_k L_{n_k}^* = 0. \tag{54}$$

Thus, the first order optimality conditions with respect to the $\operatorname{vec} X_{n_i}(t+\Delta t)$-variables (52)–(54) can be written respectively as,

$$\begin{aligned} 0 &= 2\left(\int_0^1 (\mathbf{B}_{n_1}(u)^T \otimes \mathbf{B}_{n_1}(u) \otimes I_2)\, du\right) \operatorname{vec} X_{n_1}(t+\Delta t) - (I_2 \otimes B_{n_1}^{r_1})\boldsymbol{\lambda}_1^{r_1} \\ &\quad - R_{n_1}^T \boldsymbol{\mu}_1 - n_1 (L_{n_1}^*)^T \boldsymbol{\mu}_k - n_1 (R_{n_1}^*)^T \boldsymbol{\mu}_{k+2}, \end{aligned} \tag{55}$$

$$\begin{aligned} 0 &= 2\left(\int_0^1 (\mathbf{B}_{n_i}(u)^T \otimes \mathbf{B}_{n_i}(u) \otimes I_2)\, du\right) \operatorname{vec} X_{n_i}(t+\Delta t) - (I_2 \otimes B_{n_i}^{r_i})\boldsymbol{\lambda}_i^{r_i} \\ &\quad + L_{n_i}^T \boldsymbol{\mu}_{i-1} - R_{n_i}^T \boldsymbol{\mu}_i + n_i (L_{n_i}^*)^T \boldsymbol{\mu}_{k+i} - n_i (R_{n_i}^*)^T \boldsymbol{\mu}_{k+i+1}, \end{aligned} \tag{56}$$

for $2 \leq i \leq k-1$.

$$\begin{aligned} 0 &= 2\left(\int_0^1 (\mathbf{B}_{n_k}(u)^T \otimes \mathbf{B}_{n_k}(u) \otimes I_2)\, du\right) \operatorname{vec} X_{n_k}(t+\Delta t) - (I_2 \otimes B_{n_k}^{r_k})\boldsymbol{\lambda}_k^{r_k} \\ &\quad + L_{n_k}^T \boldsymbol{\mu}_{k-1} - n_k (R_{n_k}^*)^T \boldsymbol{\mu}_{k+1} + n_k (L_{n_k}^*)^T \boldsymbol{\mu}_{2k}. \end{aligned} \tag{57}$$

Finally, we conclude that the solution of the minimization program (47) can be computed by means the following linear system,

$$A\mathbf{z}(t+\Delta t) = \mathbf{f}(t) \tag{58}$$

where A is a block matrix given by

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & 0 \end{bmatrix}.$$

The block matrices are

$$A_{1,1} = \operatorname{diag}(Z_{n_1}, \ldots, Z_{n_k})$$

where,

$$Z_{n_i} = 2\int_0^1 (\mathbf{B}_{n_i}(u)^T \otimes \mathbf{B}_{n_i}(u) \otimes I_2) du \in \mathbb{R}^{2(n_i+1)\times 2(n_i+1)}$$

for $i = 1, 2, \ldots, k$,

$$A_{1,2} = [B_1 \quad B_2 \quad B_3 \quad B_4] \text{ and } A_{2,1} = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix}.$$

where

$$B_1 = \text{diag}\,(-(I_2 \otimes B_{n_1}^{r_1}), \ldots, -(I_2 \otimes B_{n_k}^{r_k})),$$

$$C_1 = \text{diag}\,((B_{n_1}^{r_1})^T \otimes I_2, \ldots, (B_{n_k}^{r_k})^T \otimes I_2),$$

$$B_2 = \begin{bmatrix} -R_{n_1}^T & 0_{n_1}^T & 0_{n_1}^T & 0_{n_1}^T & \cdots & 0_{n_1}^T & 0_{n_1}^T \\ L_{n_2}^T & -R_{n_2}^T & 0_{n_2}^T & 0_{n_2}^T & \cdots & 0_{n_2}^T & 0_{n_2}^T \\ 0_{n_3}^T & L_{n_3}^T & -R_{n_3}^T & 0_{n_3}^T & \cdots & 0_{n_3}^T & 0_{n_3}^T \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_{n_{k-1}}^T & 0_{n_{k-1}}^T & 0_{n_{k-1}}^T & 0_{n_{k-1}}^T & \cdots & L_{n_{k-1}}^T & -R_{n_{k-1}}^T \\ 0_{n_k}^T & 0_{n_k}^T & 0_{n_k}^T & 0_{n_k}^T & \cdots & 0_{n_k}^T & L_{n_k}^T \end{bmatrix},$$

$$C_2 = \begin{bmatrix} R_{n_1} & -L_{n_2} & 0_{n_3} & \cdots & 0_{n_{k-2}} & 0_{n_{k-1}} & 0_{n_k} \\ 0_{n_1} & R_{n_2} & -L_{n_3} & \cdots & 0_{n_{k-2}} & 0_{n_{k-1}} & 0_{n_k} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0_{n_1} & 0_{n_2} & 0_{n_3} & \cdots & R_{n_{k-2}} & -L_{n_{k-1}} & 0_{n_k} \\ 0_{n_1} & 0_{n_2} & 0_{n_3} & \cdots & 0_{n_k-2} & R_{n_{k-1}} & -L_{n_k} \end{bmatrix},$$

$$B_3 = \begin{bmatrix} -n_1(L_{n_1}^*)^T & 0_{n_1}^T \\ 0_{n_2}^T & 0_{n_2}^T \\ \vdots & \vdots \\ 0_{n_{k-1}}^T & 0_{n_{k-1}}^T \\ 0_{n_k}^T & -n_k(R_{n_k}^*)^T \end{bmatrix},$$

$$C_3 = \begin{bmatrix} n_1 L_{n_1}^* & 0_{n_2} & 0_{n_3} & \cdots & 0_{n_{k-1}} & 0_{n_k} \\ 0_{n_1} & 0_{n_2} & 0_{n_3} & \cdots & 0_{n_{k-1}} & n_k R_{n_k}^* \end{bmatrix}$$

$$B_4 = \begin{bmatrix} -n_1(R_{n_1}^*)^T & 0_{n_1}^T & 0_{n_1}^T & \cdots & 0_{n_1}^T & 0_{n_1}^T & 0_{n_1}^T \\ n_2(L_{n_2}^*)^T & -n_2(R_{n_2}^*)^T & 0_{n_2}^T & \cdots & 0_{n_2}^T & 0_{n_2}^T & 0_{n_2}^T \\ 0_{n_3}^T & n_3(L_{n_3}^*)^T & -n_3(R_{n_3}^*)^T & \cdots & 0_{n_3}^T & 0_{n_3}^T & 0_{n_3}^T \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0_{n_{k-1}}^T & 0_{n_{k-1}}^T & 0_{n_{k-1}}^T & \cdots & 0_{n_{k-1}}^T & n_{k-1}(L_{n_{k-1}}^*)^T & -n_{k-1}(R_{n_{k-1}}^*)^T \\ 0_{n_k}^T & 0_{n_k}^T & 0_{n_k}^T & \cdots & 0_{n_k}^T & 0_{n_k}^T & n_k(L_{n_k}^*)^T \end{bmatrix},$$

and

$$C_4 = \begin{bmatrix} n_1 R_{n_1}^* & -n_2 L_{n_2}^* & 0_{n_3} & \cdots & 0_{n_{k-2}} & 0_{n_{k-1}} & 0_{n_k} \\ 0_{n_1} & n_2 R_{n_2}^* & -n_3 L_{n_3}^* & \cdots & 0_{n_{k-2}} & 0_{n_{k-1}} & 0_{n_k} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0_{n_1} & 0_{n_2} & 0_{n_3} & \cdots & n_{k-2} R_{n_{k-2}}^* & -n_{k-1} L_{n_{k-1}}^* & 0_{n_k} \\ 0_{n_1} & 0_{n_2} & 0_{n_3} & \cdots & 0_{n_{k-2}} & n_{k-1} R_{n_{k-1}}^* & -n_k L_{n_k}^* \end{bmatrix}.$$

We point out the dimension of the block matrices:

$$A_{1,1} \in \mathbb{R}^{2\sum_{i=1}^k (n_i+1) \times 2\sum_{i=1}^k (n_i+1)},\ B_1 \in \mathbb{R}^{2\sum_{i=1}^k (n_i+1) \times 2\sum_{i=1}^k r_i},\ B_2 \in \mathbb{R}^{2\sum_{i=1}^k (n_i+1) \times 2(k-1)},$$
$$B_3 \in \mathbb{R}^{2\sum_{i=1}^k (n_i+1) \times 4},\ B_4 \in \mathbb{R}^{2\sum_{i=1}^k (n_i+1) \times 2(k-1)},\ C_1 \in \mathbb{R}^{2\sum_{i=1}^k r_i \times 2\sum_{i=1}^k (n_i+1)},$$
$$C_2 \in \mathbb{R}^{2(k-1) \times 2\sum_{i=1}^k (n_i+1)},\ C_3 \in \mathbb{R}^{4 \times 2\sum_{i=1}^k (n_i+1)} \text{ and } C_4 \in \mathbb{R}^{2(k-1) \times 2\sum_{i=1}^k (n_i+1)},$$

and hence $A \in \mathbb{R}^{p \times p}$ for

$$p = 2\sum_{i=1}^k (n_i+1) + 2\sum_{i=1}^k r_i + 2(k-1) + 4 + 2(k-1) = 2\sum_{i=1}^k (n_i + r_i) + 6k. \tag{59}$$

Since,

$$C_j = -B_j^T \text{ for } j = 1, 2, 3, 4, \tag{60}$$

we can write

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ -A_{1,2}^T & 0 \end{bmatrix}.$$

Finally, we have

$$\mathbf{z}(t+\Delta t) = \begin{bmatrix} \operatorname{vec} X_{n_1}(t+\Delta t) \\ \vdots \\ \operatorname{vec} X_{n_k}(t+\Delta t) \\ \boldsymbol{\lambda}_1^{r_1} \\ \vdots \\ \boldsymbol{\lambda}_k^{r_k} \\ \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_{2k} \end{bmatrix} \in \mathbb{R}^{p \times 1} \text{ and } \mathbf{f}(t) = \begin{bmatrix} \mathbf{V}_1(t+\Delta t) - \mathbf{V}_1(t) \\ 0 \\ \vdots \\ 0 \\ \mathbf{V}_k(t+\Delta t) - \mathbf{V}_k(t) \\ \operatorname{vec} T_{r_1}(t) - \operatorname{vec} P_{n_1}(t) B_{n_1}^{r_1} \\ \vdots \\ \operatorname{vec} T_{r_k}(t) - \operatorname{vec} P_{n_k}(t) B_{n_k}^{r_k} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{R}^{p \times 1}.$$

Next, we will show that A is a non-singular matrix and hence there exists a unique minimum for our problem.

Proposition 3. *Assume that* $\operatorname{rank} B_{n_j}^{r_j} = r_j$ *for* $1 \le j \le k$. *Then the matrix* A *is invertible.*

Proof. First, observe that

$$A_{1,1} = \operatorname{diag}(Z_{n_1}, \ldots, Z_{n_k}).$$

By Proposition 1 the matrix Z_{n_j} is invertible for $1 \leq j \leq k$ and, in consequence, $\operatorname{rank} A_{1,1} = \sum_{j=1}^{k} 2(n_j + 1)$. Thanks to Theorem 2 the proposition follows if

$$\operatorname{rank} A_{1,2} = \operatorname{rank}(-A_{1,2}^T) = \sum_{j=1}^{k} 2r_j + 2k,$$

holds. Observe that $k < \sum_{j=1}^{k} r_j < \sum_{j=1}^{k}(n_j + 1)$. Let us denote by $\{\mathbf{e}_1^{(i)}, \ldots, \mathbf{e}_{n_i+1}^{(i)}\}$ the canonical basis of $\mathbb{R}^{n_i+1}$ for $1 \leq i \leq k$, and observe that

$$R_{n_i} = \begin{bmatrix} 0 & \cdots & 0 & 0 & I_2 \end{bmatrix} = (I_2 \otimes \mathbf{e}_{n_i+1}^{(i)})^T,$$

$$R_{n_i}^* = \begin{bmatrix} 0 & \cdots & 0 & -I_2 & I_2 \end{bmatrix} = (I_2 \otimes (\mathbf{e}_{n_i+1}^{(i)} - \mathbf{e}_{n_i}^{(i)}))^T,$$

$$L_{n_i} = \begin{bmatrix} I_2 & 0 & 0 & \cdots & 0 \end{bmatrix} = (I_2 \otimes \mathbf{e}_1^{(i)})^T$$

and

$$L_{n_i}^* = \begin{bmatrix} -I_2 & I_2 & 0 & \cdots & 0 \end{bmatrix} = (I_2 \otimes (\mathbf{e}_2^{(i)} - \mathbf{e}_1^{(i)}))^T,$$

By inspection it is possible to see that the subspace spanned by the rows of the $\sum_{j=1}^{k} 2r_j + 2k \times \sum_{j=1}^{k} 2(n_j + 1)$-matrix $(-A_{1,2}^T)$, that is,

$$\begin{bmatrix}
(B_{n_1}^{r_1})^T \otimes I_2 & 0 & 0 & \cdots & 0 & 0 \\
0 & (B_{n_2}^{r_1})^T \otimes I_2 & 0 & \cdots & 0 & 0 \\
0 & 0 & (B_{n_3}^{r_2})^T \otimes I_2 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & (B_{n_{k-1}}^{r_{k-1}})^T \otimes I_2 & 0 \\
0 & 0 & 0 & \cdots & 0 & (B_{n_k}^{r_k})^T \otimes I_2 \\
- & - & - & - & - & - \\
(\mathbf{e}_{n_1+1}^{(1)})^T \otimes I_2 & -(\mathbf{e}_1^{(2)})^T \otimes I_2 & 0 & \cdots & 0 & 0 \\
0 & (\mathbf{e}_{n_2+1}^{(2)})^T \otimes I_2 & -(\mathbf{e}_1^{(3)})^T \otimes I_2 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & -(\mathbf{e}_{n_{k-1}}^{(k-1)})^T \otimes I_2 & 0 \\
0 & 0 & 0 & \cdots & (\mathbf{e}_{n_{k-1}+1}^{(k-1)})^T \otimes I_2 & -(\mathbf{e}_1^{(k)})^T \otimes I_2 \\
- & - & - & - & - & - \\
n_1((\mathbf{e}_2^{(1)} - \mathbf{e}_1^{(1)}))^T \otimes I_2 & 0 & 0 & \cdots & 0 & \\
0 & 0 & 0 & \cdots & 0 & n_k((\mathbf{e}_{n_k+1}^{(k)} - \mathbf{e}_{n_k}^{(k)}))^T \otimes I_2 \\
- & - & - & - & - & - \\
n_1((\mathbf{e}_{n_1+1}^{(1)} - \mathbf{e}_{n_1}^{(1)}))^T \otimes I_2 & -n_2(\mathbf{e}_2^{(2)} - \mathbf{e}_1^{(2)})^T \otimes I_2 & 0 & \cdots & 0 & 0 \\
0 & n_2(\mathbf{e}_{n_2+1}^{(2)} - \mathbf{e}_{n_2}^{(2)}))^T \otimes I_2 & -n_3(\mathbf{e}_2^{(3)} - \mathbf{e}_1^{(3)})^T \otimes I_2 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & -n_{k-1}(\mathbf{e}_2^{(k-1)} - \mathbf{e}_1^{(k-1)})^T \otimes I_2 & 0 \\
0 & 0 & 0 & \cdots & n_{k-1}(\mathbf{e}_{n_{k-1}+1}^{(k-1)} - \mathbf{e}_{n_{k-1}}^{(k-1)}))^T \otimes I_2 & -n_k(\mathbf{e}_2^{(k)} - \mathbf{e}_1^{(k)})^T \otimes I_2
\end{bmatrix}$$

has dimension equal to $\sum_{j=1}^{k} 2r_j + 2k$ and the proposition follows. □

5 Comparing BSD with T-BSD

The BSD has been applied and published in [33, 41, 42]. The BSD algorithm computes the deformation of a continuous Bézier curve. In [33], it was used to improve the numerical simulation of Liquide Composite Moulding processes. In this case the BSD was used to represent as a continuous curve and update the information of the resin's flow front when the mould is filling.

Later, in [41, 42], the algorithm was applied to mobile robots to obtain a new technique for flexible path planning based on the deformation of a Bézier curve through a field of forces (vectors). The focus of this research has been the generation of a smooth collision-free trajectory for an holonomic mobile robot.

With T-BSD the reduction of the computational cost of the BSD algorithm is achieved. Figure 1 shows the evolution in computational time required for the calculation of the deformation of Bézier curves (composed of different number of Bézier curves) with the BSD and T-BSD methods. It is clear that, as the number of curves increases, BSD grows exponentially, whereas T-BSD grows linearly.

This algorithm can be used in real-time. In fact, whereas the BSD algorithm can use up to 70 quadratic curves for the computation of the modified Bézier curve in one second (see Figure 1), its reformulated algorithm T-BSD is able to use up to 170 curves within the same period of time, which highly increases the accuracy of the modified Bézier curve when it is compared to the BSD method. The use of tensors reduces the calculation time.

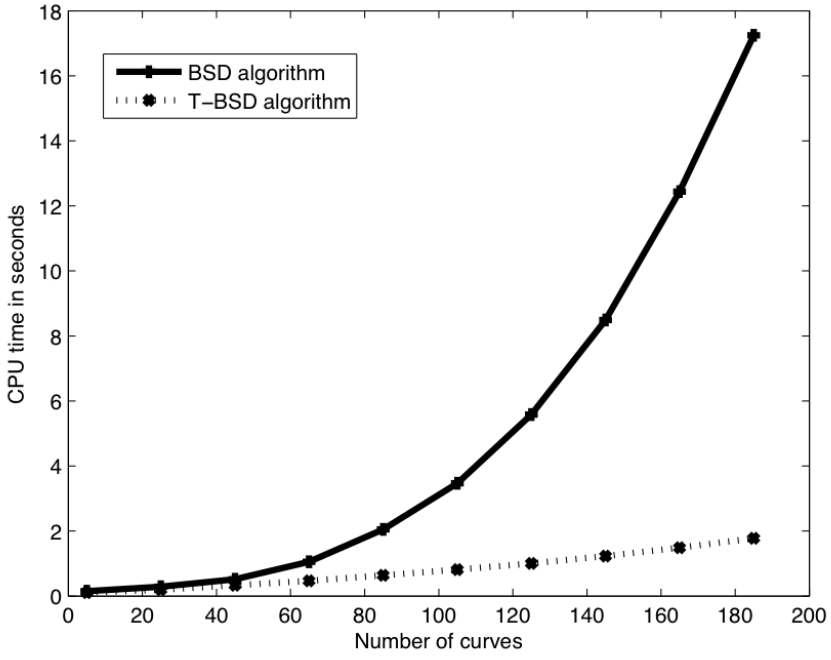

Figure 1: The comparison of the computational cost of BSD and T-BSD methods

This comparison has been computed using a PC with a 3.06 GHz Intel Core i3 and 4GB RAM.

6 Simulation Results

The algorithm T-BSD has been applied in Liquid Composite Moulding (LCM) processes and Mobile Robots.

In LCM processes, see Figure 2, the resin's flow front is an important tool to take decisions during the mould filling. This flow front has been computed and updated using T-BSD. It is represented with a Bézier curve and updated through a field of vectors. In this case, these vectors are the velocity vectors obtained solving the flow kinematics with Finite Element Methods (FEM), see [34]. The parametrization of the flow front permits a continuous numerical formulation using a Bézier, avoiding approximation techniques.

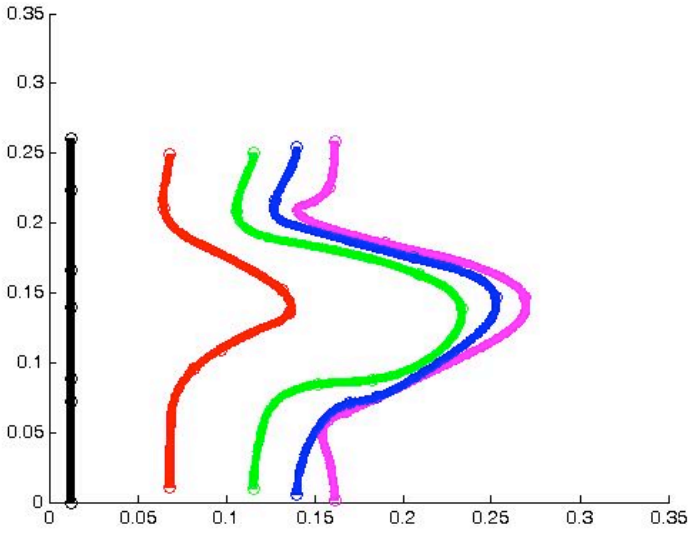

Figure 2: Particle Age evolution through T-BSD+FEM.

In Mobile Robots, see Figure 3, the idea is to obtain a flexible Trajectory for a Mobile Robot free of collisions. This flexible Trajectory is based on the deformation of a Bézier curve through a field of vectors. The field of vectors, in this case forces, is computed with a recently artificial potential field method called Potential Field Projection method (PFP), see [35, 36, 37]. The Initial Trajectory is modified through this field of forces in order to avoid the obstacles and guiding the robot to non-collision positions.

7 Conclusions

This work presents a use of tensors which reduces the computational time of the BSD (Bézier Shape Deformation) algorithm, initially developed within the framework of mobile robotics and liquid composite moulding processes. The result is a tensorial method called T-BSD (Tensor Bézier Shape Deformation) which enjoys the same properties of the former method including a key advantage: its low computational time and real-time performance. This is a critical issue in some engineering applications. For that reason, the reduction of the execution time of the Bézier generation algorithm is such an important goal to achieve.

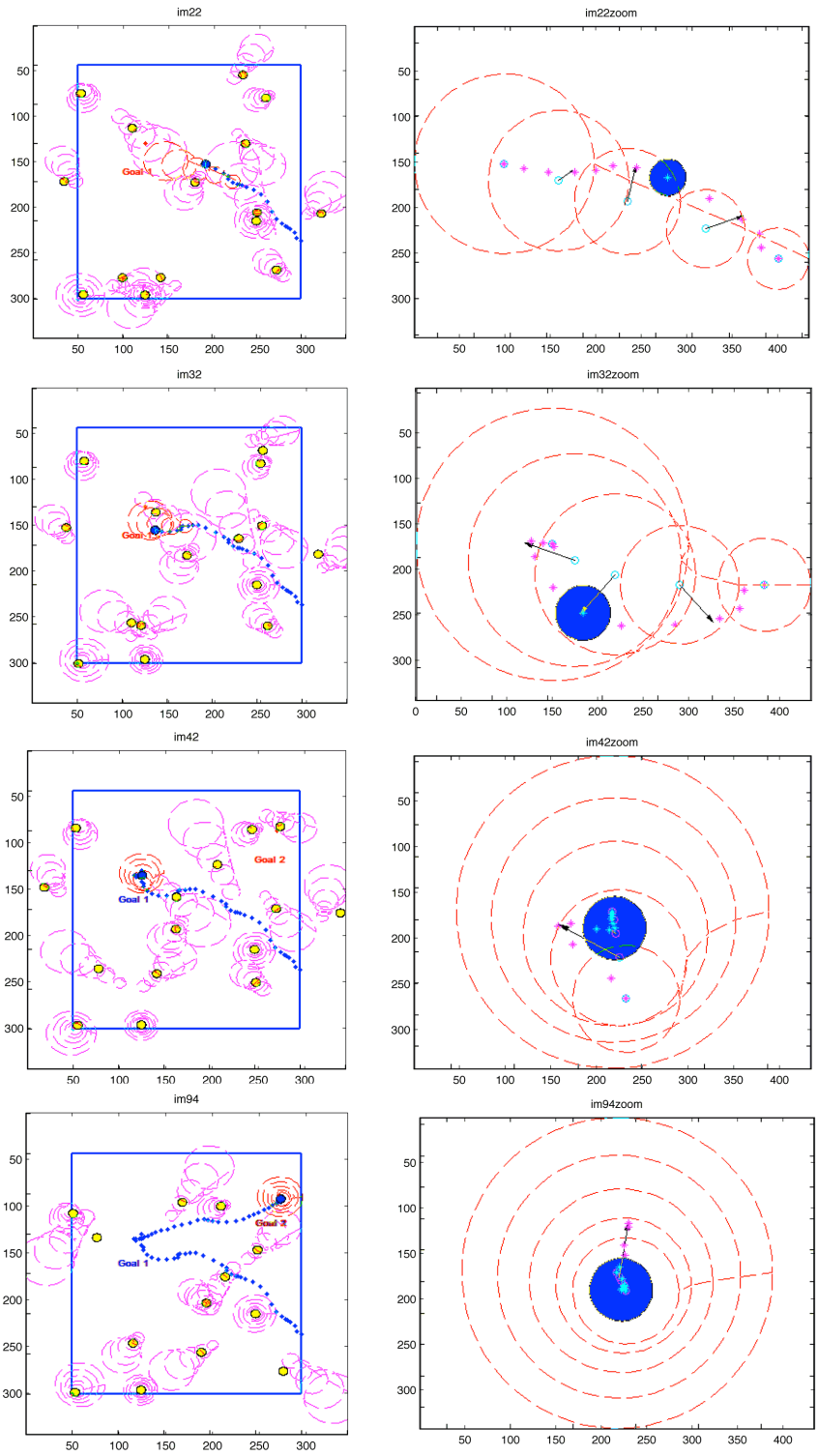

Figure 3: Snapshots of the Trajectory (left images) obtained by the T-BSD+PFP algorithm in an environment with 15 obstacles. Right images show detailed views of robot Trajectory for the corresponding left images.

In this case of the T-BSD algorithm, the calculation costs depend on the number of curves required to compute a new Bézier from a given one. A field of vectors indicate the direction of deformation at predefined points in the initial Bézier. The algorithm performs then the concatenation of a set of curves to obtain the deformed Bézier. Besides, the number of curves is highly related to the accuracy of the modified Bézier. In fact, the more curves are used the better the accuracy of the new Bézier.

As a consequence, the T-BSD algorithm is computed with very low computational time and excellent accuracy. This is a great outcome in a lot of engineering fields.

References

[1] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings: *A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modelling of complex fluids*. Journal of Non-Newtonian Fluid Mechanics, 139 **3** (2006) 153–176.

[2] C.J. Appellof and E.R. Davidson: *Strategies for analyzing data from video fluorometric monitoring of liquid-chromatographic effluents*. Analytical Chemistry, 53 **13** (1981) 2053–2056.

[3] G. Berkooz, P. Holmes, and J. L. Lumley: *The proper orthogonal decomposition in the analysis of turbulent flows*. Annual Review of Fluid Mechanics, **25** (1993) 539–575.

[4] E. Cances, V. Ehrlacher, and T. Lelievre: *Convergence of a greedy algorithm for high-dimensional convex nonlinear problems*. In arXiv arXiv1004.0095v1 [math.FA]), April 2010 (to appear in Math. Models and Methods in Appl. Sciences)

[5] J. D. Carroll and J. J. Chang: *Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart-Young decomposition*. Psychometrika, **35** (1970) 283–319.

[6] A. Doostan and G. Iaccarino: *A least-squares approximation of partial differential equations with high-dimensional random inputs*. J. Comput. Phys., **228** (12) (2009) 4332–4345.

[7] A. Falcó: *Algorithms and numerical methods for high dimensional financial market models*. Revista de Economia Financiera, **20** (2010) 51-68.

[8] W. Hackbusch: *Tensor spaces and numerical tensor calculus.* Monograph in preparation.

[9] T. G. Kolda and B. W. Bader: *Tensor decompositions and applications*. SIAM Rev., **51** (2009) 455–500.

[10] L. De Lathauwer and J. Vandewalle: *Dimensionality reduction in higher-order signal processing and rank−$(r_1, r_2, ..., r_n)$ reduction in multilinear algebra*. Linear Algebra Appl., **391** (2004) 31–55.

[11] A. Nouy: *A generalized spectral decomposition technique to solve a class of linear stochastic partial differential equations*. Computer Methods in Applied Mechanics and Engineering, **96** (45-48) (2007) 4521–4537.

[12] A. Nouy: *Proper generalized decompositions and separated representations for the numerical solution of high dimensional stochastic problems*. Archives of Computational Methods in Engineering, 2010, In press.

[13] L. R. Tucker: *Some mathematical notes on three-mode factor analysis*. Psychometrika, **31** (1966) 279-311.

[14] M. A. O. Vasilescu and D. Terzopoulos: *Multilinear analysis of image ensembles: tensorfaces*. ECCV 2002:Proceedings of the 7th European Conference on Computer Vision, Lecture Notes in Comput. Sci. 2350, 447–460. Springer, 2002.

[15] G. Vidal: *Efficient classical simulation of slightly entangled quantum computations.* Phys. Rev. Letters **91** (2003), 147902.

[16] H. Wang and N. Ahuja: *Compact representation of multidimensional data using tensor rank-one decomposition*. ICPR 2004:Proceedings of the 17th International Conference on Pattern Recognition, volume 1, (2004) 44–47

[17] Graham A. *Kronecker Products and Matrix Calculus with Applications*. John Wiley, 1981.

[18] Magnus J. R. and Neudecker H. . *Matrix Differential Calculus with Applications in Statistics and Econometrics: Third Edition*. John Wiley and Sons, 2007.

[19] Van Loan C.F. *The ubiquitous Kronecker product*. J. Comput. Appl. Math. 123 (2000), pp.85-100

[20] Piegl L. *Modifying of the shape of rational B-Spline part 1: Curves*. Computer Aided Design, 21(8); 509–518.

[21] Opfer, Oberle G.,H.J. *The derivation of cubic splines with obstacles by methods of optimization and optimal control*. Numer.Math.,52,1988;17–31.

[22] Qingbiao Wu, Shuyi Tao. *Shape modification of B-Splines curves via constrained optimization for multitarget points*. World Congress on Computer Science and Information Engineering, 2009.

[23] Fowler B.,Bartels R. *Constrained-based curve manipulation*. IEEE Computer Graphics and Application, Vol.13(5),1993;43–49.

[24] Piegl L. *On NURBS: A Survey*. IEEE Computer Graphics and Applications, 11(1), January 1991;55–71.

[25] Au C.K., Yuen M.M.F. *Unified approach to NURBS curve shape modification*. Computer Aided Design, Vol.27(2).

[26] Sánchez R.J. *A simple technique for NURBS shape modification*. IEEE Computer Graphics and Applications, Vol.17(1),1997;52–59.

[27] Hu S.M., Zhou D.W., Sun J.G. *Shape modification of NURBS curves via constrained optimization*. Proceedings of the CAD/Graphics, 1999;958–962.

[28] Hu S.M., Zhou D.W., Sun J.G. *Modifying the shape of NURBS surfaces with geometric constraints*. Computer Aided Design, Vol.33(2),2001;903–912

[29] Meek D.S.,Ong B.H.,Walton D.J. *Constrained interpolation with rational cubics*. Computer Aided Geometric Design, Vol.20,2003;253–275.

[30] Deusch F., *Best Approximation in Inner Product Spaces* CMS Books in Mathematics, Springer–Verlag (2001).

[31] Xu L.,Chen Y.J, Hu N. *Shape modification of Bézier curves by constrained optimization*. Journal of Software(China), Vol.13(6),2002;1069–1074.

[32] Wu O.B.,Xia F.H. *Shape modification of Bézier curves by constrained optimization*. Journal of Zhejiang University-Science, Springer-Verlag GmbH,2005;124–127.

[33] Montés N.,Sánchez F.,Hilario L.,Falcó A. *Flow Numerical Computation through Bézier Shape Deformation for LCM process simulation methods*. International Journal of Material Forming,Vol.1,2008;919–922.Lyon(France).

[34] Sánchez, F. and Gascon, L.I and Garcia, F.A. and Chinesta, F. *On the incubation Time Computation in Resin Transfer Molding Process simulation*. International Journal of Forming Processes, 2005; 51–67.

[35] Mora M.C., Pizá R., Tornero J. *Multirate obstacle tracking and Path Planning for Intelligent Vehicles*. IEEE Intelligent Vehicles Symposium, 2007;172–177.

[36] Mora M.C., Tornero J. *Planificación de movimientos mediante la propagación de campos potenciales artificiales*. Congreso Iberoamericano de Ingeniería mecánica, Octava Edición.

[37] Mora M.C.,Tornero J. *Path planning and trajectory generation using multi-rate predictive artificial potential-fields*. IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2990-2995, 2008.

[38] Khatib O. *Real-time obstacle avoidance for manipulators and mobile robots*. The International Journal of Robotics Research,Vol.5 Issue 1;90–98.

[39] Tornero J.,Salt J., Albertos P. *LQ Optimal Control for Multirate Sampled Data Systems*. IFAC World Congress, 1999. 14^{th} Edition; 211–216.

[40] Pizá R. *Modelado del entorno y localización de robots móviles autónomos mediante técnicas de muestreo no convencional*. PhD Thesis, Polytechnic University of Valencia.

[41] Hilario L., Montés N., Falcó A., Mora M.C. *Real-Time Trajectory Modification Based on Bézier Shape Deformation*. International Conference on Evolutionary Computation, October 2010. Valencia (Spain)

[42] Hilario L., Montés N., Mora M.C., Falcó A. *Real-Time Trajectory Deformation for Potential Fields Planning Methods*. IEEE/RSJ International Conference on Intelligent Robots and Systems;1567–1572, September 2011. San Francisco, CA, USA