2011 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 25-30, 2011. San Francisco, CA, USA

Real-time Bézier Trajectory Deformation for Potential Fields Planning Methods

L.Hilario, N.Montés, M.C.Mora, A.Falcó

Abstract— **This article presents a new technique for obtaining a flexible trajectory based on the deformation of a Bézier curve through a field of vectors. This new technique is called Bézier Trajectory Deformation (BTD). The trajectory deformation is computed with a constrained optimization method (Lagrange Multipliers Theorem). A linear system is solved to achieve the result. As a consequence, the deformed trajectory is computed in a few milliseconds. In addition, the linear system can be solved offline if the Bézier curve order is maintained constant during the movement of the robot, which is the common case. This technique can be combined with any collision avoidance algorithm that produces a field of vectors. In particular, it has been developed for artificial potential field methods. BTD is combined with a recently proposed PF method, the Potential Field Projection method (PFP). The resulting technique is tested in dynamic environment showing its real-time performance and its efficacy for obstacle avoidance.**

I. Introduction

A robot can be defined as a machine able to collect information as well as to interact with the surrounding environment in a natural way. The simple aim of moving a mobile robot from an initial pose (x_i, y_i, θ_i) to a goal pose (x_g, y_g, θ_g) in an automatic and safe way is a task that involves several research fields: collision avoidance, path planning, sensor fusion, mapping, control systems, etc. All these fields must be combined to obtain a computationally efficient motion planning algorithm.

A lot of researchers consider parametric curves in the construction of trajectories for wheeled robots. It is due to their useful properties for the path generation problem. In fact, they produce inherently smooth paths and, consequently, many path planning techniques use parametric curves to enhance the trajectories produced by the planner. The most commonly used are B-Splines [1, 2], NURBS [3-6], Bézier [7-15] and Rational Bézier Curves, [16, 17]. The main difference between these curves is the complexity of their mathematical definition. While Bézier curves are the simplest ones, B-Splines or NURBS are more complex although they can more accurately represent particular curves or objects. Nevertheless, complexity involves additional computation and, therefore, Bézier curves are generally selected for path smoothing. For instance, in [7] a method for on-line modification of non-holonomic mobile robot trajectories is developed to obey to environmental changes. An initial path is assumed to be provided by a motion planner. It is resampled for obtaining a sequence of bubbles and constructing a bubble band. A path is computed as a concatenation of Reeds and Shepp paths. It is discontinuous at the concatenation points and a Bézier smoothing is used in order to obtain a feasible path for car-like robots.

In other research fields like CAD/CAGD, the shape modification of parametric curves is a recently research topic (see [18] for NURBS and [19] for B-Spline). In fact, the problem of parametric curves shape modification by constrained optimization was recently proposed in [20]. In [21] a new technique was developed to modify a Bézier curve by minimizing the changes of its shape in a fast and exact way. This shape modification is controlled by a field of vectors applied to particular points of the Bézier curve.

The shape modification of parametric curves is an emerging field in path planning and existing methods are still not suitable for navigation. For instance, in [13-15] a path is computed based on a quadratic Bézier curve and its maximum curvature is minimized with boundary constraints. However, obstacles and changes in the environment are not considered.

In this work the Bézier shape modification technique [21] is adapted for its use in path planning for holonomic robots using Potential Fields methods (PF), as they produce a field of vectors that guide the robot to non-collision positions. An exact, fast and efficient mathematical way for computing the deformation of the Bézier curve from forces derived from the Potential Field Projection (PFP) method [23-25] has been developed, named the Bézier Trajectory Deformation (BTD) method.

PF methods are very popular in path planning because of their simplicity and real-time performance [18, 25]. The main idea consists in filling the robot's workspace with an artificial potential field in which the robot is attracted by the goal and repelled by the obstacles. In this research field, a considerable amount of variants and improvements have

Manuscript received March 14, 2011. This work was supported by the project GV/2010/054 of the Generalitat Valenciana.

L. Hilario is with the University CEU Cardenal Herrera, Alfara del Patriarca, Valencia, Spain. (corresponding author to provide phone: +34961369000. Ext: 3952; e-mail: luciah@uch.ceu.es).

N. Montés, is with the University CEU Cardenal Herrera, Alfara del Patriarca, Valencia, Spain (e-mail: nimonsan@uch.ceu.es).

M. C. Mora is with the Mechanical Engineering and Construction Dept., Universitat Jaume I, Castellón, Spain (e-mail: mmora@emc.uji.es).

A. Falco, is with the University CEU Cardenal Herrera, Alfara del Patriarca, Valencia, Spain (e-mail: afalco@uch.ceu.es).

been developed. Here, the Potential Field Projection method (PFP) is used. It is based on the combination of the classical Potential Fields method [22] and the multi-rate Kalman filter estimation [26, 27] and takes into account the object volume, the uncertainties on locations, the future trajectories of the robot and the obstacles and the multi-rate information supplied by sensors.

An initial path is obtained by means of the PFP method, taking into account the attractive forces produced by the goal in the present and future instants of time. The set of repulsive forces computed from the repulsive potential field are provided to the BTD, which modifies the initial Bézier trajectory in order to avoid the obstacles and guiding the robot to the goal by means of a smooth trajectory computed in real-time.

This paper is organized as follows: section II defines the Bézier curve and the translation into trajectory; section III develops the mathematical method to modify a Bézier trajectory from a field of vectors, the Bézier Trajectory Deformation (BTD) method; in Section IV the Potential Field Projection method (PFP) is explained; this method is combined with BTD in section V; simulation results with PFP-BTD are given in section VI and, finally, section VII provides conclusions and future works.

II. Definitions And Preliminary Notation

The Bézier curve of n order, given $(n+1)$ control points, $\mathbf{Q_i}$, is defined as:

$$\boldsymbol{\alpha}(u)=\sum_{i=0}^{n}\mathbf{Q_i}\cdot B_{i,n}(u);\ u\in[0,1] \qquad (1)$$

where u is the intrinsic parameter and $B_{i,n}(u)$ are the Bernstein polynomials or Bernstein Basis functions of n order, given by:

$$B_{i,n}(u)=\begin{cases}\binom{n}{i}(1-u)^{n-i}u^{i};\ 0\le i\le n\\ 0,\ \text{otherwise}\end{cases}$$

The intrinsic parameter, u, is non-dimensional. In order to use a Bézier curve as a trajectory, this intrinsic parameter must be redefined as a time variable, associating each curve position (robot position) with a time instant $t\in[t_0, t_f]$, where t_0 and t_f are the initial and final trajectory instants. For this purpose, the definition of the Bézier curve has to change:

$$\boldsymbol{\alpha}(t)=\sum_{i=0}^{n}\mathbf{Q}_i\cdot B_{i,n}(t);\ \ t\in[t_0,t_f] \qquad (2)$$

$$B_{i,n}(t)=\binom{n}{i}\cdot\left(\frac{t_f-t}{t_f-t_0}\right)^{n-i}\cdot\left(\frac{t-t_0}{t_f-t_0}\right)^{i};i=0,1,..,n$$

An initial Bézier robot trajectory should be modified to avoid the mobile obstacles in the environment. Such an operation can be done moving the control points from its original position to the new one, determined by any obstacle avoidance algorithm. The displacement of a control point i is denoted by $\boldsymbol{\varepsilon}_i$ while $\underline{\underline{\boldsymbol{\varepsilon}}}=[\boldsymbol{\varepsilon}_0\cdots\boldsymbol{\varepsilon}_n]$ is the displacement of the set of control points that define the curve. The new Bézier curve, $\mathbf{S}_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t))$, obtained modifying its controls points is defined as:

$$\mathbf{S}_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t)):=\sum_{i=0}^{n}(\mathbf{Q}_i+\boldsymbol{\varepsilon}_i)\cdot B_{i,n}(t)\ ,\ t\in[t_0, t_f] \qquad (3)$$

III. Bézier Trajectory Deformation (BTD): Mathematical model

To deform a given Bézier curve describing a trajectory, the control points must be changed and the perturbation $\boldsymbol{\varepsilon}_i$ of every control point must be computed. This results in the *Bézier Trajectory Deformation method* (BTD), which is explained in the following paragraphs.

In order to do that, a constraint optimization problem is proposed similar to [21] but with some modification for mobile robot applications. The cost function to optimize is defined as follows:

$$\min_{\underline{\underline{\boldsymbol{\varepsilon}}}}\int_{t_0}^{t_f}\|\mathbf{S}_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t))-\boldsymbol{\alpha}(t)\|_2^2\,dt=\min_{\underline{\underline{\boldsymbol{\varepsilon}}}}\int_{t_0}^{t_f}\left\|\sum_{i=0}^{n}\boldsymbol{\varepsilon}_i\cdot B_{i,n}(t)\right\|_2^2 dt \qquad (4)$$

This function minimizes the changes of the shape minimizing the distance between the original Bézier curve (2) and the modified one (3), as depicted in Fig.1. This cost function is adequate for holonomic mobile robots because it is supposed that the optimal trajectory is the original one computed by the path planning stage.

Nevertheless, the first problem to solve in the BTD for mobile robot applications is that the new curve cannot be constructed using a large number of control points because those kind of Bézier curves are numerically unstable [10]. In addition to this, the order of the Bézier curve must not be higher than two because it can produce loops or cusps depending on the geometrical location of the control points. For that reason, two or more second-order Bézier curves (with three control points each) are concatenated to represent the complete trajectory. Thus, the cost function in (4) must be redefined as in (5) considering a set of k concatenated Bézier curves of order n_l and being $\underline{\boldsymbol{\varepsilon}}^{(l)}$ the perturbation vector for modifying curve l. From now on, the superscript (l) references the corresponding l curve.

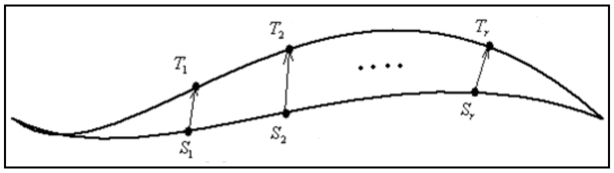

Fig. 1. The deformation of a Bézier curve.

$$\min_{\underline{\underline{\boldsymbol{\varepsilon}}}^{(1)}\cdots\underline{\underline{\boldsymbol{\varepsilon}}}^{(k)}} \int_{t_0^{(l)}}^{t_f^{(l)}} \sum_{l=1}^{k} \left\| \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l(t)\right) - \boldsymbol{\alpha}_l(t) \right\|_2^2 dt = \min_{\underline{\underline{\boldsymbol{\varepsilon}}}^{(1)}\cdots\underline{\underline{\boldsymbol{\varepsilon}}}^{(k)}} \int_{t_0^{(l)}}^{t_f^{(l)}} \sum_{l=1}^{k} \left\| \sum_{i=0}^{n_l} \varepsilon_i^{(l)} \cdot B_{i,n_l}(t) \right\|_2^2 dt \quad (5)$$

Continuity and derivability constraints must be imposed on the joint points of the concatenated curves. The particular restrictions are:

1. Continuity constraints:

$$\begin{aligned} &\mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1\left(t_f^{(1)}\right)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_2\left(t_0^{(2)}\right)\right) = 0, \ldots \\ &\ldots, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-2}\left(t_f^{(k-2)}\right)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-1}\left(t_0^{(k-1)}\right)\right) = 0 \end{aligned} \quad (6)$$

where $\boldsymbol{\alpha}_i$ is the i-th Bézier curve.

2. The required constraints to guarantee derivability are specified in (7), where $\boldsymbol{\alpha}_i^{'}$ is the first derivative of the i-th Bézier curve.

$$\begin{aligned} &\mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1^{'}\left(t_f^{(1)}\right)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_2^{'}\left(t_0^{(2)}\right)\right) = 0, \ldots \\ &\ldots, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-2}^{'}\left(t_f^{(k-2)}\right)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-1}^{'}\left(t_0^{(k-1)}\right)\right) = 0 \end{aligned} \quad (7)$$

3. Derivative restrictions in the start and end points of the resulting concatenated curve must be imposed to maintain the start and end pose of the mobile robot. These constraints result in equations (8).

$$\boldsymbol{\alpha}_1{'}\left(t_0^{(1)}\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1^{'}\left(t_0^{(1)}\right)\right) = 0 \qquad \boldsymbol{\alpha}_k{'}\left(t_f^{(k)}\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_k^{'}\left(t_f^{(k)}\right)\right) = 0 \quad (8)$$

4. As in [21], a field of vectors connect the *Start points* (points on the curve), $\mathbf{S}_i$, i=1,..,l with the *Target points* (points on the modified curve), $\mathbf{T}_i$. The modified Bézier curve $\mathbf{S}_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t))$ is designed to pass through the target points. This constraint of the BTD is given by equations (9).

$$\mathbf{T}_1^{(l)} - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_1^{(l)}\right)\right) = 0, \ldots, \mathbf{T}_k^{(l)} - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_k^{(l)}\right)\right) = 0;\ l = 1, \cdots, k \quad (9)$$

Due to the curve order and continuity restrictions necessary in the BTD for mobile robot applications, only one deformation vector can be applied to each concatenated Bézier curve. Thus, we will have as many vectors as concatenated curves.

The Lagrange Multipliers theorem has been applied to solve the constrained optimization problem (5). The Lagrange function $L(\boldsymbol{\varepsilon}_i)$ is defined in (10), where the cost function g_1 is given by (11) and the constraints r_i, i=1,…,4, have been formulated before by means of expressions (6), (7), (8) and (9).

$$L(\boldsymbol{\varepsilon}_i) = g_1 + r_1 + r_2 + r_3 + r_4 \quad (10)$$

$$g_1 = \int_{t_0^{(l)}}^{t_f^{(l)}} \sum_{l=1}^{k} \left\| \sum_{i=0}^{n} \boldsymbol{\varepsilon}_i^{(l)} B_{i,n_l}(t) \right\|_2^2 dt \quad (11)$$

The constraints previously defined are included in the Lagrange function in the following form:

$$\begin{aligned} r_1 &= \sum_{l=1}^{k} \sum_{j=1}^{r_l} \left\langle \boldsymbol{\lambda}, \mathbf{T}_j^{(l)} - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_j^{(l)}\right)\right) \right\rangle \\ r_2 &= \left\langle \boldsymbol{\lambda}, \boldsymbol{\alpha}_1^{'}\left(t_0^{(1)}\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1^{'}\left(t_0^{(1)}\right)\right) \right\rangle + \left\langle \boldsymbol{\lambda}, \boldsymbol{\alpha}_k^{'}\left(t_f^{(k)}\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_k^{'}\left(t_f^{(k)}\right)\right) \right\rangle \\ r_3 &= \sum_{l=1}^{k-1} \left\langle \boldsymbol{\lambda}, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_f^{(l)}\right)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{l+1}\left(t_0^{(l+1)}\right)\right) \right\rangle \\ r_4 &= \sum_{l=1}^{k-1} \left\langle \boldsymbol{\lambda}, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}{'}_l\left(t_f^{(l)}\right)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}{'}_{l+1}\left(t_0^{(l+1)}\right)\right) \right\rangle \end{aligned} \quad (12)$$

where λ is the Lagrange multipliers vector.

Thus, the L function depends on:

$$L = L\left(\boldsymbol{\varepsilon}^{(1)}, \ldots, \boldsymbol{\varepsilon}^{(k)}, \boldsymbol{\lambda}\right)$$

The problem is solved making zero the partial derivatives of the Lagrange function:

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{\varepsilon}^{(l)}} = 0; l = 1, \cdots, k \\ \dfrac{\partial L}{\partial \boldsymbol{\lambda}} = 0 \end{cases}$$

A linear system of equations is obtained. The matrix formulation of this system is defined as $\mathbf{D} \cdot \mathrm{X} = \mathbf{b}$, where:

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_{11} & \mathbf{D}_{12} & \mathbf{D}_{13} & \mathbf{D}_{14} \\ \mathbf{D}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}_{41} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \in M_{q \times q}$$

$$q = \left(\sum_{i=1}^{k} n_i + k\right) + \left(\sum_{i=1}^{k} r_i\right) + (2) + (2k-2)$$

$$\mathbf{X}^{\mathrm{T}} = \left[\boldsymbol{\varepsilon}^{(1)}, \cdots, \boldsymbol{\varepsilon}^{(k)}, \boldsymbol{\lambda}\right]$$

$$\mathbf{b}^{\mathrm{T}} = \left[\mathbf{0}, \mathbf{v}^{(1)} \cdots \mathbf{v}^{(k)}, \mathbf{0}, \mathbf{H}\right]$$

The terms of $\mathbf{b}^{\mathrm{T}}$ are defined as,

$$\begin{aligned} &\mathbf{v}^{(l)} = \left[\left(\mathbf{T}_1^{(l)} - \mathbf{S}_1^{(l)}\right) \cdots \left(\mathbf{T}_{r_l}^{(l)} - \mathbf{S}_{r_l}^{(l)}\right)\right] \\ &\mathbf{H} = \left[\mathbf{H}_{0,(2,1)}, \mathbf{H}_{1,(2,1)}, \cdots, \mathbf{H}_{0,(k,k-1)}, \mathbf{H}_{1,(k,k-1)}\right] \\ &\mathbf{H}_{0,(i,(i-1))} = \mathbf{Q}_0^{(i)} - \mathbf{Q}_{n_i}^{(i-1)} \\ &\mathbf{H}_{1,(l,(l-1))} = n_{l-1} \cdot \left(\mathbf{Q}_{n_{l-1}-1}^{(l-1)} - \mathbf{Q}_{n_l}^{(l-1)}\right) + n_l \cdot \left(\mathbf{Q}_1^{(l)} - \mathbf{Q}_0^{(l)}\right) \end{aligned}$$

The matrix **D** is square and defined as a block matrix. The first row is computed from the partials of the Lagrangian function and every block depends on the cost function or a specific constraint. The second, third and fourth rows are the necessary constraints defined on the problem. This matrix is invertible and, therefore, the solution of the system can be obtained as X = $\mathbf{D}^{-1} \cdot \mathbf{b}$. This fact guarantees that the solution obtained is not an approximation. It is an exact solution. Another important issue is the low computational cost involved in this calculation. The deformation of the initial robot path can be done in real time because the problem to solve is a linear system and the inverse matrix

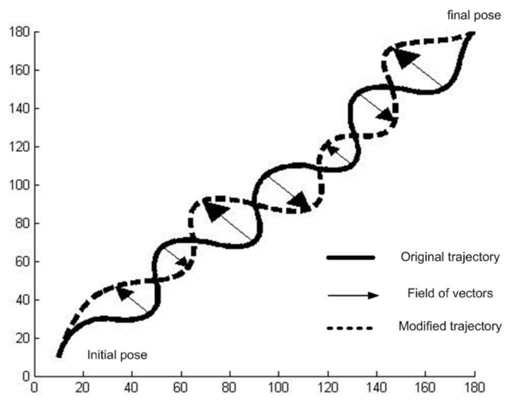

Fig.2. BTD result with eight Bézier curves.

$\mathbf{D}^{-1}$ can be computed in advance if the number of second-order Bézier curves are maintained invariable.

An example of the BTD algorithm is shown in Fig. 2. A curved initial trajectory is deformed by means of a field of vectors. Eight Bézier curves are used to join the initial and final poses. The robot orientation in these poses does not change in the modified trajectory. The computational time of the BTD algorithm is 0.23ms in a Pentium IV 2.4 Ghz (without including the $\mathbf{D}^{-1}$ computation, which was obtained off-line once the prediction horizon was established).

IV. Potential Field Projection (PFP): Generating a Field of Vectors

The technique explained above requires an algorithm that generates an initial trajectory and a field of vectors to modify it in case obstacles are detected, ensuring that the deformed trajectory is collision-free. In the present study, the BTD technique is evaluated through a predictive PF path planning method described in [23-25], the *Potential Field Projection* method. This method is based on the combination of the classical Potential Fields method [22] and the multi-rate Kalman filter estimation [26, 27] and takes into account the uncertainties on locations, the future trajectories of the robot and the obstacles and the multi-rate information supplied by sensors. The particularities of this approach are described in this section.

This method generates a predicted trajectory during a prediction horizon T_h for the robot taking into account its kinematic model and the attractive forces produced by a potential field in the present (j=0) and future instants of time (j>0). Also the obstacles trajectories are predicted and repulsive forces are generated to be applied on the robot for obstacle avoidance.

Here, second-order particle kinematic models (13) have been used for modeling the robot and the obstacles' motion, where T is the control or estimation period, the state vector $\mathbf{x}\in\Re^4$ in the j-th instant ($j\in[0,\ldots,T_h/T]$) is composed of position x, y and velocity v_x, v_y in XY coordinates and the

$$\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{j+1} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_j + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \cdot \begin{bmatrix} a_x \\ a_y \end{bmatrix}_j$$
$$\begin{bmatrix} x \\ y \end{bmatrix}_j = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_j \tag{13}$$

control input is the object acceleration a_x, a_y. The model (13) responds to the linear state space representation (14), where $\mathbf{u}\in\Re^2$ is the input vector, $\mathbf{z}\in\Re^2$ is the measurement vector, $\mathbf{w}\in\Re^4$ is the process noise, $\mathbf{v}\in\Re^2$ is measurement noise and $\mathbf{A}\in\Re^{4x4}$, $\mathbf{B}\in\Re^{4x2}$ and $\mathbf{C}\in\Re^{2x4}$ are the state space matrices for linear systems.

$$\begin{aligned} \mathbf{x}_{j+1} &= \mathbf{A}\cdot\mathbf{x}_j + \mathbf{B}\cdot\mathbf{u}_j + \mathbf{w}_j \\ \mathbf{z}_j &= \mathbf{C}\cdot\mathbf{x}_j + \mathbf{v}_j \end{aligned} \tag{14}$$

Predicted future positions and uncertainties are obtained from the prediction equations of the multi-rate Kalman filter (15) for every object in the environment, where $\hat{\mathbf{x}}\in\Re^4$ is the state estimation vector, $\mathbf{P}\in\Re^{4x4}$ is the error estimation variance matrix, $\mathbf{K}\in\Re^{4x2}$ is the Kalman gain and $\mathbf{Q}\in\Re^{4x4}$ and $\mathbf{R}\in\Re^{2x4}$ are, respectively, the process noise and the measurement noise covariance matrices.

$$\begin{aligned} \hat{\mathbf{x}}_{j+1/j} &= \mathbf{A}\cdot\hat{\mathbf{x}}_{j/j} + \mathbf{B}\cdot\mathbf{u}_j \\ \mathbf{P}_{j+1/j} &= \mathbf{A}\cdot\mathbf{P}_{j/j}\cdot\mathbf{A}^{\mathrm{T}} + \mathbf{Q} \\ \mathbf{K}_j &= \mathbf{P}_{j/j-1}\cdot\mathbf{C}^{\mathrm{T}}\cdot\left[\mathbf{C}\cdot\mathbf{P}_{j/j-1}\cdot\mathbf{C}^{\mathrm{T}} + \mathbf{R}\right]^{-1}\cdot\mathbf{\Delta}_j \\ \hat{\mathbf{x}}_{j/j} &= \hat{\mathbf{x}}_{j/j-1} + \mathbf{K}_j\cdot\left[\mathbf{z}_j - \mathbf{C}\cdot\hat{\mathbf{x}}_{j/j-1}\right] \\ \mathbf{P}_{j/j} &= \mathbf{P}_{j/j-1} - \mathbf{K}_j\cdot\mathbf{C}\cdot\mathbf{P}_{j/j-1} \end{aligned} \tag{15}$$

The delta function Δ modifies the expression of the Kalman gain indicating the presence (unit Δ matrix) or the absence (zero Δ matrix) of measurements in one particular estimation instant j. Predicted future positions are derived imposing zero Δ matrices for future instants, as measurements are not available.

The predicted positions and their uncertainties are used in the generation of a potential field $U_j(\mathbf{x}) = U_{att,j}(\mathbf{x}) + U_{rep,j}(\mathbf{x})$ that guides the robot to the goal avoiding the obstacles in the environment. It is composed of an attractive component $U_{att,j}(\mathbf{x})$ affected by the goal and a repulsive component $U_{rep,j}(\mathbf{x})$ generated by the detected obstacles and their uncertainties, that are considered restricted areas for path planning. Both are defined in [25] even in the instants of time without measurements of the environment (j>0). These potential fields generate forces in every prediction instant j.

On the one hand, the set of attractive forces $\mathbf{F}_{att,j}(\mathbf{x}) = -\nabla U_{att,j}(\mathbf{x})$ are transformed into accelerations by a dynamic particle model and, this way, are considered as control inputs in the prediction cycle of the multi-rate Kalman filter (15), since the attractive forces have the structure of a PD controller. Thus, a set of predicted positions is obtained that would lead the robot toward the goal if there were no obstacles.

On the other hand, this prediction is modified taking into account the set of repulsive forces $\mathbf{F}_{rep,j}(\mathbf{x}) = -\nabla U_{rep,j}(\mathbf{x})$ by means of the BTD method. In fact, the sequence of predicted positions is considered as the start points $\mathbf{S}_i$ of the reference trajectory for the BTD explained above. Then, the set of repulsive forces are transformed into displacements by means of a particle dynamic model, and constitute the perturbation vector of the deformed Bézier curve. These displacements affect the shape of the initial parametric trajectory, they are used in the computation of the target points $\mathbf{T}_\mathrm{i}$ of the deformed trajectory.

V. BTD- PFP: Combining BTD With a PF Method

In order to illustrate the BTD-PFP fusion, an example is used here. This example corresponds to eight predicted positions (start points) along the prediction horizon T_h, with a set of repulsive vectors associated to every predicted point. The time span used in this prediction is 14 s (T_h=14s), that is, 2 s between each position point (T=2s).

The set of vectors associated to the i predicted points must be equal to the number of concatenated Bézier curves in BTD, i.e., k. Then i=k. In this example the number of predicted robot positions is $i = 8$, so there are k=8 Bézier curves to be computed.

In this example the predicted trajectory is a straight line. Therefore, the control points for the Bézier cuves are equally distributed along the predicted trajectory. This means that each vector will be located in the middle of each concatenated Bézier curve, except the first and last ones, which will be located at the initial and end points, respectively. In general, the start point must be located in the current instant point of the BTD. In Fig. 3, the straight line is the predicted trajectory of the robot. The field of vectors is plotted on the positions of the predicted trajectory. The discontinuous line is the modified trajectory.

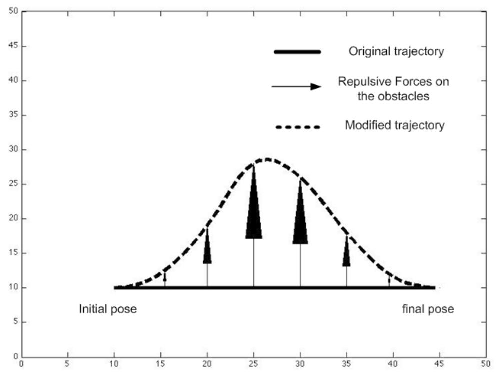

Fig. 3. Modification of eight concatenated Bézier curves.

VI. Simulation Results

A simulation application has been implemented in Matlab to demonstrate the integration of the BTD method with a PF method such as PFP. Also a video is provided.

The environment shown in the left sequence of images in Fig. 4 represents a 2D four-sided scenario with five mobile obstacles (in yellow) and a mobile robot (in blue). The obstacles follow linear trajectories (given by their kinematic model) going from side to side of the environment, simulating a rebound in a pool table. The circles surrounding the robot and the obstacles are the uncertainties of their predicted trajectories.

When the obstacles come close to the robot, it starts a smooth avoiding maneuver based on the BTD, which modifies its initial trajectory (given by the PFP) and guides the robot to the goal without collision. If no obstacles are detected, BTD also transforms the predicted trajectory into a Bézier trajectory, more appropriate for robot navigation.

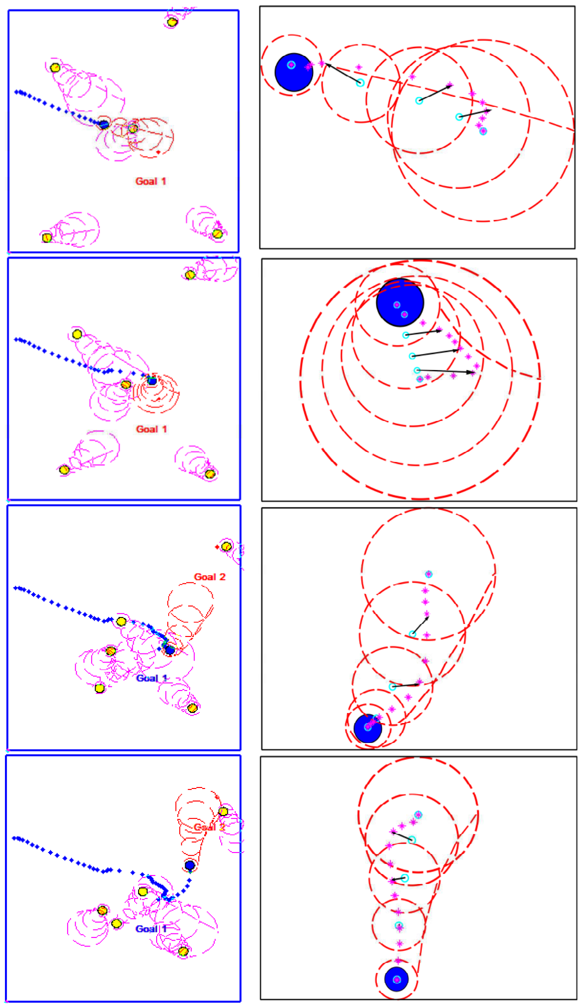

Fig. 4. Snapshots of the trajectory (left images) obtained by the BTD-PFP algorithm in an environment with 5 mobile obstacles. Right images show detailed views of robot trajectory for the corresponding left images.

Images depict the trajectory followed by the robot in order to reach goals 1 and 2 avoiding the obstacles in the environment in different instants of the simulation. In the right sequence of images shown in Fig. 4 we can observe details of the specific instants of the robot trajectory shown in the corresponding left picture. We can see that, when a future collision is detected, the initial trajectory is modified in real-time using the BTD method and that the resulting trajectory successfully performs the avoiding maneuver. The complete simulation lasts 25s and the mean execution time of the BTD-PFP algorithm throughout the simulation is 11ms, which is appropriate for real-time performance.

VII. Conclusions And Future Works

This article presents a novel technique, the Bézier Trajectory Deformation (BTD), for computing a flexible trajectory based on the deformation of a Bézier curve through a field of vectors. The trajectory deformation is computed as a constrained optimization problem solved with the Lagrange Multipliers Theorem and a linear system is obtained to achieve the deformation. As a consequence, the deformed trajectory is computed with low computational cost. This technique has been combined with a PF method, the Potential Field Projection method (PFP), which computes a local trajectory for the robot taking into account the future trajectories for the robot and the obstacles, their uncertainties and the multi-rate information supplied by sensors. The PFP-BTD technique is tested by simulation in a 2D dynamic environment with mobile obstacles.

Our immediately future work is related to the introduction of other constrains in the definition problem, like the path length or the curvature for non-holonomic cases. We also plan to implement this method on a real platform as well as compare it with other techniques.

Acknowledgment

This research work is financially supported by the project GV/2010/054 of the Generalitat Valenciana.

References

[1] K. Komoriya and K. Tanie, Trajectory Design and Control of a Wheel-Type Mobile Robot Using B-Spline Curve, Int. Workshop on Intelligence Robots and Systems, pp. 398-405, 1989.

[2] B. Vazquez, G.J. Humberto Sossa A. and Juan L. Diaz de Leon S, Auto Guided Vehicle Control Using Expanded Time B-Spline, Int. Conf. on Systems, Man, and Cybernetics, pp. 2786-2791, 1994.

[3] N. Tatematsu and K. Ohnishi, Tracking motion of mobile robot for moving target using NURBS curve, In Int'l Conf. on Industrial Technology, pages 245-249, San Francisco, CA, December 2003.

[4] J. Aleotti, S. Caselli, Trajectory Clustering and Stochastic Approximation for Robot Programming by Demonstration, 2005.

[5] J. Aleotti, S. Caselli, Trajectory Reconstruction with NURBS Curves for Robot Programming by Demonstration, In IEEE Int'l Symp, on Computational Intelligence in Robotics and Automation, June 2005.

[6] Jacopo Aleotti, Stefano Caselli, Grasp Recognition in Virtual Reality for Robot Pregrasp Planning by Demonstration, in IEEE Intl. Conf. on Robotics and Automation, Orlando, FL, May 2006.

[7] M. Khatib, H. Jaouni, R. Chatila, J. P. Laumond, Dynamic Path Modification for Car-Like Nonholonomic Mobile Robots, Proceedings of the 1997 IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, April 1997.

[8] K. Nagatani, Y.Iwai, and Y.Tanaka, Sensor Based Navigation for car-like mobile robots using Generalized Voronoi Graph, Int. Conf. on Intelligent Robots and Systems, pp. 1017-1022, 2001.

[9] J-H. Hwang, R. C.Arkin, D-S. Know, Mobile Robots at your fingertip: Bézier Curve on-line trajectory generation for supervisory control, Proc. IEEE/RSJ Int. Conf. on Int. Robots and System, 2003.

[10] Igor Skrjanc, Gregor Klancar, Cooperative Collision Avoidance between Multiple Robots Based on Bézier Curves, ITI 2007 29th (International Conference on Information Technology Interfaces), June 25-28, 2007, Cavtat, Croatia.

[11] M. Lizarraga, G.Elklaim, Spatially Deconflicted Path Generation for Multiple UAVs in a Bounded Airspace, ION/IEEE Position, Location and Navigation Symposium, ION/IEEE PLANS 2008, Monterey, CA, May 5-8, 2008.

[12] I.Skrjanc, G.Klancar. Optimal cooperative collision avoidance between multiple robots based on Bernstein-Bézier curves. Robotics and Autonomous systems. Vol 58, Is 1, pp1-9, 2010

[13] J. Choi, G.Elkaim, Bézier Curves for Trajectory Guidance, World Congress on Engineering and Computer Science, WCECS 2008, San Francisco, CA, Oct. 22-24, 2008.

[14] J. Choi, R.Curry, G.Elkaim, Smooth Path Generation Based on Bézier Curves for Autonomous Vehicles, WCECS (World Congress on Engineering and Computer Science 2009 Vol II), October 20-22, 2009, San Francisco, USA.

[15] J. Choi, R. Curr, G.Elkaim, Path Planning based on Bézier Curve for Autonomous Ground Vehicles, in Proceedings of the Advances in Electrical and Electronics Engineering-IAENG Special Edition of the World Congress on Engineering and Computer Science 2008 (WCECS 2008).

[16] N. Montés, N., Herráez, A., Armesto, L., Tornero, J., 2008. Real-time clothoid approximation by Rational Bézier curves. In ICRA 2008, IEEE Int. Conf. on Robotics and Automation, pp.2246 – 2251.

[17] N.Montés, M.C.Mora, J.Tornero. Trajectory generation based on Rational Bézier curves as clothoids. IEEE Intelligent Vehicles Symposium 2007, p.p. 505-510

[18] D.S.Meek, B.H.Ong, D.J.Walton, Constrained interpolation with rational cubics, Computer Aided Geometric Design. Vol.20, pp. 253-275, 2003.

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

[20] L.Xu, Y.J. Chen, N. Hu, Shape modification of Bézier curves by constrained optimization, Journal of software (China), Vol. 13(6), pp.1069-1074. 2002.

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

[22] O. Khatib, 1986. Real-time obstacle avoidance for manipulators and mobile robots. In The International Journal of Robotics Research, vol. 5, nº 1, pp. 90-98.

[23] M.C. Mora, R. Pizá, J. Tornero, J., 2007. Multirate obstacle tracking and path planning for intelligent vehicles. In IEEE Intelligent Vehicles Symposium, pp. 172-177.

[24] M.C. Mora, J. Tornero, 2007. Planificación de movimientos mediante la propagación de campos potenciales artificiales. In 8º Congreso Iberoamericano de Ingeniería Mecánica, e-book.

[25] M.C. Mora and J. Tornero, "Path planning and trajectory generation using multi-rate predictive artificial potential fields," in *Proc. IEEE/RSJ IROS,* pp. 2990-2995, 2008.

[26] J. Tornero, J. Salt, P. Albertos. 1999. LQ Optimal Control for Multirate Sampled Data Systems. In 14th IFAC World Congress, pp. 211-216.

[27] R. Pizá, 2003. Modelado del entorno y localización de robots móviles autónomos mediante técnicas de muestreo no convencional, PhD Thesis, Universidad Politécnica de Valencia.