REAL-TIME TRAJECTORY MODIFICATION BASED ON BÉZIER SHAPE DEFORMATION

L. Hilario[1], N. Montés[1], M. C. Mora[2] and A. Falcó[1]
[1]*Cardenal Herrera CEU University, C/San Bartolomé 55 46115, Alfara del Patriarca, Spain*
[2]*Mechanical Engineering and Construction Department, Universitat Jaume I, Castellón, Spain*

Keywords: Bézier, Deformation, Constrained optimization, Trajectory, Path planning, Artificial potential fields.

Abstract: This paper presents a new technique for flexible path planning based on the deformation of a Bézier curve through a field of vectors. This new technique is called Bézier Shape Deformation (BSD). This deformation is computed with a constrained optimization method (Lagrange Multipliers Theorem). The main advantage of this method is how the solution is obtained. A linear system is solved to achieve the result. As a consequence, the deformed curve is computed in a few milliseconds where the linear system can be solved offline if the Bézier curve order is maintained constant during the movement of the robot. This method allows the use of these trajectories in dynamic environments where the computational cost is critical. This technique can be combined with any collision avoidance algorithm that produces a field of vectors. In particular, it is appropriate for artificial potential field methods. At the end of the paper, the presented methodology is combined with an artificial potential fields algorithm recently proposed, the Potential Field Projection method (PFP). This method is based on the combination of the classical Potential Fields method and the multi-rate Kalman filter estimation and takes into account the uncertainties on locations, the future trajectory of the robot and the obstacles and the multi-rate information supplied by sensors. As shown in the simulation results, flexible trajectories for collision avoidance are generated with smooth curves.

1 INTRODUCTION

A robot can be defined as a machine that should be able to collect information as well as to interact in a natural way with the surrounding environment. The key idea is how to obtain a proper trajectory that must be smooth and must interact with the environment in order to guarantee that the path is free of collisions in real-time.

Collision avoidance is one of the main goals in the research carried out in industrial applications. In this sense, parametric curves are an option to represent these trajectories. The properties of parametric curves produce smooth paths. The most commonly used in robotics are B-Splines, see for instance (Connors and Elkaim, 2007), NURBS, (Alleoti and Caselli, 2005), RBC, (Montes et al., 2008) and Bézier.

The main difference between them 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 objects in CAD programs. However, in mobile robotics real-time applications are required and, for this reason, more researchers work with Bézier curves.

Moreover, (Nagatani et. al, 2001) used a path planning algorithm based on Bézier curves considering the minimum radius of curvature of vehicle. (Hwang et. al, 2003) presented a new interfacing method using a touchpad/screen to control a mobile robot, capable of real-time dynamic obstacle avoidance as well. They developed two algorithms: a significant points extraction algorithm for noisy input data and an on-line piecewise cubic Bézier curves trajectory generation algorithm for a mobile robot. (Skrjanc and Klancar, 2007) developed a new cooperative collision avoidance method for multiple nonholonomic robots with constraints and known start and goal velocities based on Bernstein-Bézier curves. The minimization problem used is an inequality optimization problem. The objective function minimizes the sum of all absolute maximal times subjected to the distances between the robots. (Choi et. al, 2008-2009) presented two path planning algorithms based on

Hilario L., Montés N., Mora M. and Falcó A..
REAL-TIME TRAJECTORY MODIFICATION BASED ON BÉZIER SHAPE DEFORMATION .
DOI: 10.5220/0003086002430248
In *Proceedings of the International Conference on Evolutionary Computation* (ICEC-2010), pages 243-248
ISBN: 978-989-8425-31-7

Bézier curves for autonomous vehicles with waypoints and corridor constraints. In this paper a mathematical problem of constraint optimization is used. The cost function is the curvature of the Bézier curve to obtain resulting paths with larger radii of curvature. (Lizarraga and Elklaim, 2008) used Bézier curves for generating spatially deconflicted paths for multiple UAVs (unmanned aerial vehicles). The critical issue addressed is that of guaranteeing that all the paths lie inside a predefined airspace volume. Bézier curves represent a natural tool for meeting this requirement (convex hull's property). The path generation problem is formulated as a constrained optimization problem over a finite optimization set and solved using standard MATLAB optimization tools. In this case, the cost function penalizes excessive lengths; the vehicles must use the shortest possible path. The constraint of the problem is the distance between the multiple vehicles, as the generated paths must have a minimum separation among them. The problem is non-linear and the solution is obtained by numerical techniques.

Through these works we can conclude that an ideal solution is having a mathematical tool to obtain the deformation of the predefined trajectory, obtaining the modified curve in real time. In the literature, there is a similar research topic where all of the customary parametric curves are used, see (Meek et. al, 2003) for NURBS shape modification and (Fowler and Bartels, 1993) for B-Spline shape modification.

In (Xu et. al, 2002) the Bézier shape modification by constrained optimization based on the discrete coefficient norm is discussed. The problem of parametric curves shape modification by constrained optimization was proposed recently. (Wu and Xia, 2005) developed a new technique to modify a Bézier curve by minimizing the changes of the shape. This result was used by (Montés et. al, 2008) and it was improved and developed for Liquid Composite Moulding Processes. This is our previous work, defined as Bézier Shape Deformation (BSD). This method computes the modification of the shape defining a constraint optimization problem solved by the Lagrange Multipliers theorem. In this theorem a cost function is defined to minimize the distance between two curves and a set of constraints are used to obtain our aim. The principal constraint is how the modified curve passes through the "Target Point". The rest of constraints are necessary to obtain a smooth curve, for example, continuity and derivability on the full curve.The advantage of this method is the possibility to add as many different constraints as necessary to obtain the best solution.

The current paper proposes a new path planning algorithm that has the ability to interact with the environment. This fact is able due to the combination of a proposed mathematical tool, called BSD, and an algorithm that gives repulsive forces. In the present work we use a very recent work in this field, (Mora et al., 2007).

This paper is organized as follows: Section 2 defines the Bézier curve and its useful properties for path planning. Section 3 develops the new technique called Bézier Shape Deformation (BSD). In Section 4, simulation results are given. Finally, Section 5 provides the conclusions and future work.

2 DEFINITIONS AND PRELIMINARY NOTATION

The Bézier curve of degree n, given (n+1) control points, $\mathbf{P}_i$, is defined to be

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

where t is a normalized time variable, $0\leq t/T_{max}\leq 1$, and $B_{n,i}(t)$, are called the Bernstein polynomials or Bernstein Basis functions of degree n.

$$B_{i,n}(t)=\begin{cases}\binom{n}{i}(1-t)^{n-i}t^{i};0\leq i\leq n\\ 0,\text{ otherwise}\end{cases} \tag{2}$$

Bézier curves have useful properties for path planning because: always pass through the first and the last control points, lie within the convex hull of the control points, they are tangent to the vector of the difference $\mathbf{P}_1$-$\mathbf{P}_0$ at the start point and the vector of the difference $\mathbf{P}_n$-$\mathbf{P}_{n-1}$, they can be translated and rotated by performing these properties on the control points and in any case a smooth curve is guaranteed.

The displacement of every control point is denoted as: $\underline{\underline{\boldsymbol{\varepsilon}}}=[\boldsymbol{\varepsilon}_0\cdots\boldsymbol{\varepsilon}_n]$. The Bézier curve obtained modifying its controls points, $S_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t))$, is defined to be

$$\mathbf{S}_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t)):=\sum_{i=0}^{n}(\mathbf{P}_i+\boldsymbol{\varepsilon}_i)\cdot B_{n,i}(t) \tag{3}$$

3 BÉZIER SHAPE DEFORMATION IN MOBILE ROBOTS AND OBSTACLES.

To deform a given Bézier curve, we only need to change the control points, and then it is necessary to compute the perturbation ε_i of every control point. For that, a constraint optimization problem is proposed. Then, it is necessary to define a cost function to optimize and a set of constraints to obtain the desirable solution.

The cost function to optimize is defined as follows;

$$\min_{\underline{\boldsymbol{\varepsilon}}} \int_0^1 \left\| S_{\boldsymbol{\varepsilon}}(\boldsymbol{\alpha}(t)) - \boldsymbol{\alpha}(t) \right\|_2^2 dt = \min_{\underline{\boldsymbol{\varepsilon}}} \int_0^1 \left\| \sum_{i=0}^{n} \boldsymbol{\varepsilon}_i \cdot B_{n,i}(t) \right\|_2^2 dt = \\ = \min_{\underline{\boldsymbol{\varepsilon}}} \int_0^1 \left(\sum_{i=0}^{n} \boldsymbol{\varepsilon}_i \cdot B_{n,i}(t) \right)^2 dt \tag{4}$$

This function minimizes the changes of the shape minimizing the distance between both curves (see Figure 1).

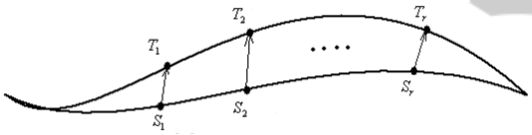


Figure 1: The deformation of a Bézier curve.

A Bézier curve constructed using a large number of control points is numerically unstable (Skrjanc and Klancar, 2007). For this undesirable property it is necessary to concatenate two or more Bézier curves of second order to represent the full trajectory joining the initial and final positions of the mobile robot. The order of the Bézier curve must not be third or higher because it produces a loop or a cusp depending of the geometrical location of the control points.

$$\min_{\underline{\boldsymbol{\varepsilon}}^{(1)} \cdots \underline{\boldsymbol{\varepsilon}}^{(k)}} \int_0^1 \sum_{l=1}^{k} \left\| \sum_{i=0}^{n} \boldsymbol{\varepsilon}_i^{(l)} \cdot B_{n,i}(t) \right\|_2^2 dt = \min_{\underline{\boldsymbol{\varepsilon}}^{(1)}, \cdots \underline{\boldsymbol{\varepsilon}}^{(k)}} \sum_{l=1}^{k} \int_0^1 \left(\sum_{i=0}^{n_l} \boldsymbol{\varepsilon}_i^{(l)} B_{n_l,i}(t) \right)^2 dt \tag{5}$$

Our goal is to develop a navigation algorithm that satisfies a set of constraints:

1.-The modified Bézier curve $\mathbf{S}_{\boldsymbol{\varepsilon}}(\alpha(t))$ passes through the target point **T**. Thus, the curve satisfies the following constraint:

$$\mathbf{T}_1^{(l)} - S_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_1^{(l)}\right)\right) = 0, \cdots, \mathbf{T}_k^{(l)} - S_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_k^{(l)}\right)\right) = 0 \tag{6}$$

2.-To maintain the derivative property of the curve, derivative restrictions in the start and end points of the resulting concatenated curve are imposed to the constrained optimization method,

$$\begin{aligned} \boldsymbol{\alpha}_1'(0) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1'(0)\right) = 0 \Rightarrow n_1 \cdot \left(\boldsymbol{\varepsilon}_1^{(1)} - \boldsymbol{\varepsilon}_0^{(1)}\right) = 0, \\ \boldsymbol{\alpha}_k'(1) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1'(1)\right) = 0 \Rightarrow n_k \cdot \left(\boldsymbol{\varepsilon}_{n_k}^{(k)} - \boldsymbol{\varepsilon}_{n_k-1}^{(k)}\right) = 0 \end{aligned} \tag{7}$$

3.- Continuity has to be imposed in the joined points of the concatenated curves,

$$\mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1(1)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_2(0)\right) = 0, \cdots, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-2}(1)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-1}(0)\right) = 0 \tag{8}$$

The evaluation of these equations is,

$$\mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l(1)\right) = \mathbf{p}_{n_l}^{(l)} + \boldsymbol{\varepsilon}_{n_l}^{(l)}, \cdots, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{l+1}(0)\right) = \mathbf{p}_0^{(l+1)} + \boldsymbol{\varepsilon}_0^{(l+1)} \tag{9}$$

4.- Derivability is also required,

$$\mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1'(1)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_2'(0)\right) = 0, \cdots, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-2}'(1)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{k-1}'(0)\right) = 0 \tag{10}$$

$$\begin{aligned} \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l'(1)\right) = n_l \cdot \left[\left(\mathbf{p}_{n_l}^{(l)} - \mathbf{p}_{n_l-1}^{(l)}\right) + \left(\boldsymbol{\varepsilon}_{n_l}^{(l)} - \boldsymbol{\varepsilon}_{n_l-1}^{(l)}\right)\right] \\ \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{l+1}'(0)\right) = n_{l+1} \cdot \left[\left(\mathbf{p}_1^{(l+1)} - \mathbf{p}_0^{(l+1)}\right) + \left(\boldsymbol{\varepsilon}_1^{(l+1)} - \boldsymbol{\varepsilon}_0^{(l+1)}\right)\right] \end{aligned} \tag{11}$$

The Lagrange multipliers theorem is applied to solve the constrained optimization problem. The constraints are added to the cost function resulting in the Lagrange function defined as,

$$L(\boldsymbol{\varepsilon}_i) = g_1 + r_1 + r_2 + r_3 + r_4 \tag{12}$$

The cost function is:

$$g_1 = \int_0^1 \sum_{l=1}^{k} \left\| \sum_{i=0}^{n} \boldsymbol{\varepsilon}_i^{(l)} B_{n_l,i}(t) \right\|_2^2 dt \tag{13}$$

The constraints are included in the Lagrange function as follows,

$$r_1 = \sum_{l=1}^{k} \sum_{j=1}^{r_l} \left\langle \boldsymbol{\lambda}, \mathbf{T}_j^{(l)} - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l\left(t_j^{(l)}\right)\right) \right\rangle \tag{14}$$

$$r_2 = \left\langle \boldsymbol{\lambda}, \boldsymbol{\alpha}_1'(0) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_1'(0)\right) \right\rangle + \left\langle \boldsymbol{\lambda}, \boldsymbol{\alpha}_k'(1) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_k'(1)\right) \right\rangle \tag{15}$$

$$r_3 = \sum_{l=1}^{k-1} \left\langle \boldsymbol{\lambda}, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_l(1)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}_{l+1}(0)\right) \right\rangle \tag{16}$$

$$r_4 = \sum_{l=1}^{k-1} \left\langle \boldsymbol{\lambda}, \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}'_l(1)\right) - \mathbf{S}_{\boldsymbol{\varepsilon}}\left(\boldsymbol{\alpha}'_{l+1}(0)\right) \right\rangle \tag{17}$$

This function depends on these variables,

$$L = L\left(\boldsymbol{\varepsilon}^{(1)},\ldots,\boldsymbol{\varepsilon}^{(k)},\boldsymbol{\lambda}\right) \tag{18}$$

The number of the constraints depends on the number of the concatenated Bézier curves,

$$\sum_{l=1}^{k} r_l + 1 + 1 + (k-1) + (k-1) \tag{19}$$

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

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{\varepsilon}^{(l)}} = 0 \\ \dfrac{\partial L}{\partial \boldsymbol{\lambda}} = 0 \end{cases} \tag{20}$$

In this way, a system of linear equations is obtained,

$$\mathbf{A}\cdot\mathbf{X} = \mathbf{b}$$

The characteristics of this system are:

1.- The variable vector is a column vector,

$$\mathbf{X} = \begin{bmatrix} \boldsymbol{\varepsilon}^{(1)} \\ \cdots \\ \boldsymbol{\varepsilon}^{(k)} \\ \boldsymbol{\lambda} \end{bmatrix} \tag{21}$$

2.- The independent vector is a column vector too,

$$\mathbf{b}^T = \left[\mathbf{0}, \mathbf{v}^{(1)} \cdots \mathbf{v}^{(k)}, \mathbf{0}, \mathbf{C}\right] \tag{22}$$

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

$$\mathbf{v}^{(l)} = \left[\left(\mathbf{T}_1^{(l)} - \mathbf{S}_1^{(l)}\right)\cdots\left(\mathbf{T}_{r_l}^{(l)} - \mathbf{S}_{r_l}^{(l)}\right)\right] \tag{23}$$

$$\mathbf{C} = \left[\mathbf{C}_{0,(2,1)}, \mathbf{C}_{1,(2,1)}, \cdots, \mathbf{C}_{0,(k,k-1)}, \mathbf{C}_{1,(k,k-1)}\right] \tag{24}$$

$$\begin{aligned} \mathbf{C}_{0,(l,(l-1))} &= \mathbf{p}_0^{(l)} - \mathbf{p}_{n_l}^{(l-1)} \\ \mathbf{C}_{1,(l,(l-1))} &= n_{l-1}\cdot\left(\mathbf{p}_{n_{l-1}}^{(l-1)} - \mathbf{p}_{n_l}^{(l-1)}\right) + n_l\cdot\left(\mathbf{p}_1^{(l)} - \mathbf{p}_0^{(l)}\right) \end{aligned} \tag{25}$$

3.- The matrix of the system is defined in (26). It is a square matrix defined as a block matrix. Every block is defined as a part of the cost function or the constraints. This form allows adding new blocks that belong to other constraints.

$$\mathbf{A} = \begin{pmatrix} \mathbf{A_{11}} & \mathbf{A_{12}} & \mathbf{A_{13}} & \mathbf{A_{14}} \\ \mathbf{A_{21}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A_{31}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{A_{41}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{26}$$

The solution of this system is obtained as,

$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{b} \tag{27}$$

The most important advantage of the proposed method is that the solution is not an approximation. It is an accurate solution. Another important thing is its low computational cost. The deformation of the initial robot path can be done in real time because the problem to solve is linear and the inverse matrix $\mathbf{A}^{-1}$ can be computed in advance.

In Figure 2 it is shown a numerical result of the BSD algorithm. The modified trajectory is computed in 0.23 ms in a Pentium IV 2.4 Ghz.

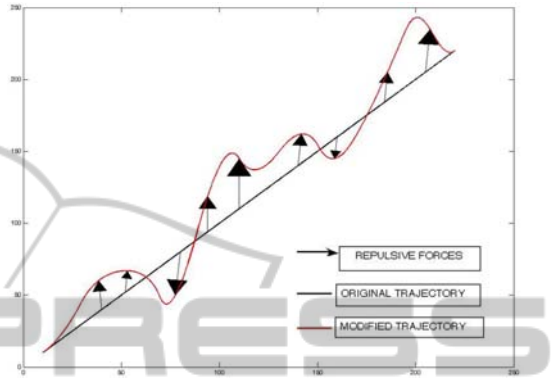


Figure 2: Numerical Result of the BSD with ten Bézier curves of second order.

4 SIMULATION RESULTS

The technique explained above has been evaluated through a predictive path planning method described in (Mora et. al, 2007) and (Mora and Tornero, 2007), the Potential Field Projection method (PFP). This method is based on the combination of the classical Potential Fields method (Khatib, 1986) and the multi-rate Kalman filter estimation (Tornero et. al., 1999), (Pizá, 2003) and takes into account the uncertainties on locations, the future trajectory of the robot and the obstacles and the multi-rate information supplied by sensors.

Dynamic models of moving objects are essential in the estimation procedure. In this case, particle kinematic models, described in (Mora et. al, 2007), have been used for the robot and the obstacles.

Predicted future positions and uncertainties are obtained from the prediction equations of the multi-rate Kalman filter (28) for every object in the environment, where $\hat{\mathbf{x}} \in \Re^n$ is the state estimation vector, $\mathbf{A} \in \Re^{nxn}$, $\mathbf{B} \in \Re^{nxm}$ and $\mathbf{C} \in \Re^{pxn}$ are the state space matrices for linear systems, $\mathbf{P} \in \Re^{nxn}$ is the error estimation variance matrix, $\mathbf{K} \in \Re^{nxp}$ is the Kalman gain and $\mathbf{Q} \in \Re^{nxn}$ and $\mathbf{R} \in \Re^{pxn}$ are, respectively, the process noise and the measurement noise covariance matrices.

$$\begin{aligned}
\hat{\mathbf{x}}_{k+1/k} &= \mathbf{A}\cdot\hat{\mathbf{x}}_{k/k} + \mathbf{B}\cdot\mathbf{u}_k \\
\mathbf{P}_{k+1/k} &= \mathbf{A}\cdot\mathbf{P}_{k/k}\cdot\mathbf{A}^{\mathrm{T}} + \mathbf{Q} \\
\mathbf{K}_k &= \mathbf{P}_{k/k-1}\cdot\mathbf{C}^{\mathrm{T}}\cdot\left[\mathbf{C}\cdot\mathbf{P}_{k/k-1}\cdot\mathbf{C}^{\mathrm{T}} + \mathbf{R}\right]^{-1}\cdot\boldsymbol{\Delta}_k \\
\hat{\mathbf{x}}_{k/k} &= \hat{\mathbf{x}}_{k/k-1} + \mathbf{K}_k\cdot\left[\mathbf{z}_k - \mathbf{C}\cdot\hat{\mathbf{x}}_{k/k-1}\right] \\
\mathbf{P}_{k/k} &= \mathbf{P}_{k/k-1} - \mathbf{K}_k\cdot\mathbf{C}\cdot\mathbf{P}_{k/k-1}
\end{aligned} \tag{28}$$

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. Predicted future positions are derived imposing zero Δ matrices for future instants, as measurements are not available.

Regarding to the robot, the sequence of predicted positions is considered as the Start points, $\boldsymbol{S_i}$, of the reference trajectory for the Bézier Shape Deformation method explained above. Every Start point is located in each second order Bézier curve. Regarding to the obstacles, their future trajectories and associated uncertainties are considering as restricted areas for path planning.

These predicted trajectories are used in the generation of the potential field $U(\mathbf{q},i)=U_{att}(\mathbf{q},i)+U_{rep}(\mathbf{q},i)$, defined in (Mora et. al., 2007) even in the instants of time without measurements of the environment ($i>0$).

The potential field generates a force in every prediction instant i, $\mathbf{F}(\mathbf{q},i)=-\nabla U(\mathbf{q},i)$, that modifies the initial prediction depending on the location of the goal and the future trajectories of the obstacles. In fact, these set of forces are transformed into displacements taking into account a particle dynamic model. These displacements are the Target points, $\boldsymbol{T_i}$, defining the necessary vector, $\mathbf{v}$, for the BSD.

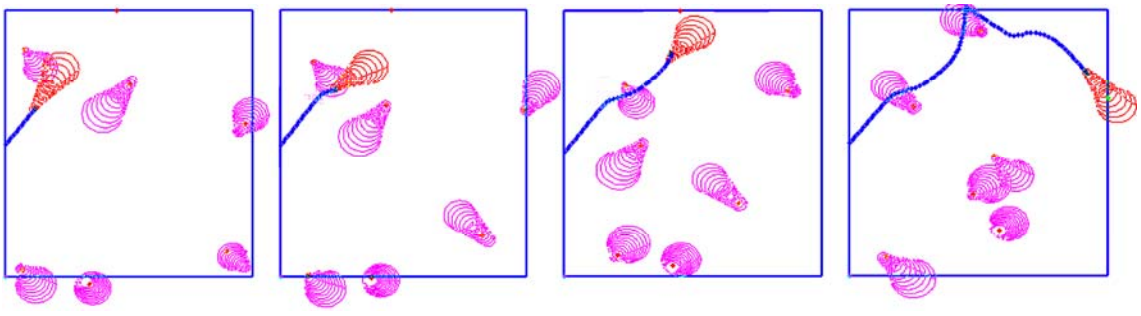
With the BSD algorithm we are able to compute a new Bézier curve in real-time that modifies the preliminary predicted trajectory. This new curve guarantees the smoothness and the continuity.

A simulation application has been implemented in Matlab in order to demonstrate the statements above. Figure 3 represents a 2D four-sided scenario with six mobile obstacles and a mobile robot. The uncertainty ellipses associated to each predicted position within the prediction horizon are also depicted for the robot (in red) and the obstacles (in pink). Initially, the robot and the obstacles follow linear trajectories (given by the kinematic model) going from side to side of the environment.

When the obstacles come close to the robot, it starts a smooth avoiding manoeuvre, based on the Bézier Shape Deformation method, which modifies its initial trajectory and guides the robot to the goal without collision. In Figure 3a the robot is following a straight line trajectory when a future collision is detected. The initial trajectory is modified in real-time using the BSD method and the resulting trajectory is shown in Figure 3b and Figure 3c. Finally, Figure 3d shows another avoidance manoeuvre the modified trajectory.

5 CONCLUSIONS AND FUTURE WORK

This paper presents a new technique, called Bézier Shape Deformation (BSD), for a flexible path planning based on the deformation of a Bézier curve through vectors. These parametric curves represent in a proper way an optimized path satisfying a set of constraints at the same time. The main advantage of this method is the solution obtained; it is an accurate solution of a linear system and the computational cost is low. It computes a new trajectory in real-time avoiding the obstacles in the environment. These obstacles generate repulsive forces; these vectors are introduced in the BSD algorithm. The method used to solve it is the Lagrange theorem. The simulation results show the successful resulting trajectory.



a) b) c) d)

Figure 3: Simulation application - A robot follows a straight line avoiding the mobile obstacles using the BSD method.

Future work lines are: the problem definition can add as many constraints as necessary. The matrix is defined with blocks, and it implies only to change or to include a new block in the matrix. It is important to work with the curvature or the total length of the path because in a vehicle robot there is a maximum curvature that the vehicle can follow and the path length affects the total travel time. Both functions could be including in the definition problem as a cost function or as a constraint.

Finally, another future research is how to extend it to 3D space. In this way, the trajectory of an UAV or a robot arm could be simulated.

ACKNOWLEDGEMENTS

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

REFERENCES

Aleotti, J., Caselli, S., 2005. Trajectory Clustering and Stochastic Approximation for Robot Programming by Demonstration. In IROS 2005, *IEEE Int. Conf. On Intelligent Robots and Systems*, pp. 1029 - 1034.

Aleotti, J., Caselli, S., 2005. Trajectory Reconstruction with NURBS Curves for Robot Programming by Demostration. *In CIRA 2005, IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, pp. 73-78.

Choi, J., Elkaim, G. 2008. Bézier Curves for Trajectory Guidance. In WCECS 2008, *World Congress on Engineering and Computer Science*, pp. 625-630.

Choi, J., Curry, R., Elkaim, G., 2008. Path Planning based on Bézier Curve for Autonomous Ground Vehicles. In *WCECS 2008, World Congress on Engineering and Computer Science*, pp. 158 - 166

Choi, J., Curry, R., Elkaim, G. 2009. Smooth Path Generation Based on Bézier Curves for Autonomous Vehicles. In WCECS 2009, *World Congress on Engineering & Computer Science Vol II*, pp.668-673.

Connors, J., Elkaim, G. 2007. Analysis of a Spline Based, Obstacle Avoiding Path Planning Algorithm. In *VTC2007, IEEE Vehicle Technology Conference*, pp. 2565-2569

Fowler, B., Bartels, R., 1993. Constrained-based curve manipulation. *IEEE Computer Graphics and Applications* 13, 5, pp. 43-49.

Hwang, J. H., Arkin, R. C., Kwon, D. S., 2003. Mobile Robots at your fingertip: Bézier curve on-line trajectory generation for supervisory control. In IROS 2003, *Int. Conf. on Int. Robots and Systems*, pp.1444-1449.

Khatib, O., 1986. Real-time obstacle avoidance for manipulators and mobile robots. In *The International Journal of Robotics Research*, 5, 1, pp. 90-98.

Lizarraga, M., Elklaim, G., 2008. Spatially Deconflicted Path Generation for Multiple UAVs in a Bounded Airspace. In PLANS 2008, ION/IEEE Position, Location and Navigation Symposium, pp. 1213 - 1218.

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

Montés, N., Sánchez, F., Hilario, L., Falcó, A. 2008. Flow Numerical Computation through Bézier Shape Deformation for LCM process simulation methods. In *Int. Journal of Material Forming*, 1, pp. 919-922.

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.

Mora, M. C., Pizá, R., Tornero, J., 2007. Multirate obstacle tracking and path planning for intelligent vehicles. In IV'07, *IEEE Int.Vehicles Symposium*, pp. 172-177.

Mora, M. C., Tornero, J., 2007. Planificación de movimientos mediante la propagación de campos potenciales artificiales. In CIBIM8, *8º Congreso Iberoamericano de Ingeniería Mecánica*, e-book.

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

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

Wu, Q. B., Xia, F. H., 2005. Shape modification of Bézier curves by constrained optimization. In *Journal of Zhejiang University-Science* A, pp 124-127.

Skrjanc, I., Klancar, G., 2007. Cooperative Collision Avoidance between Multiple Robots Based on Bézier Curves.In ITI 2007, 29th Int. Conf. on Information Technology Interfaces, pp. 451-456.

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

Xu, L., Chen, Y. J., Hu, N., 2002. Shape modification of Bézier curves by constrined optimization. In *Journal of software* (china), 13, 6, pp. 1069-1074.