Energy Optimization in Motion Planning of a Two-Link Manipulator using Bernstein Polynomials

Roman Stephan, Paolo Mercorelli
Control and Drive Systems Lab
Leuphana University of Lueneburg
Universitaetsallee 1, D-21335 Lueneburg, Germany
roman-stephan@gmx.net, mercorelli@uni.leuphana.de

Květoslav Belda
Dept. of Adaptive Systems, The Czech Academy of Sciences
Institute of Information Theory and Automation
Pod Vodárenskou věží 4, 18200 Prague, Czech Republic
belda@utia.cas.cz

***Abstract*—In this paper the mechanical system of a two-link manipulator will be optimized in terms of energy consumption using the framework of Bernstein polynomials. Considering specific definition of the cost function, the required torque is minimized such a way to move the system from one given position to another as well as to guarantee a smooth trajectory of the end effector between these positions. The unique property of Bernstein polynomials having an underlying symmetry within the individual basis functions is the motivation for using them in this environment. The results of the paper show a lower energy consumption due to its symmetric properties, compared to power based polynomials, and a smooth trajectory considering an indirect motion determined by a third, intermediate point.**

***Index Terms*—Bernstein polynomials, Bézier curve, Optimization, Two-Link manipulator**

I. Introduction

One of the most important aspects in the design of a reference trajectory, that should be tracked by a closed loop controller system, is represented by an optimum with respect to the power consumption. In this sense, a technique aimed at minimizing power consumption is a suitable technique to be applied. The model of the considered system represents a constraint for the optimization problem. In [1] a constrained optimization problem is formulated for a valve motion tracking problem. A solution then is approximated by exploiting local flatness and physical properties of the system. Different approaches which are considered as general methods in the area of control of electromagnetic drives can be seen in [2]–[4]. The two-link manipulator which is being modeled in [5] serves as the basis of the following equations of motion. The reason for this particular manipulator in Cartesian space is because of the versatile usage in different application areas. A 7$^{\text{th}}$ degree Bézier curve is then being used for modeling the initial conditions. Apart from that this simulation has no further restrictions. The symmetry of Bernstein polynomials is the reason for a more optimal trajectory compared to normal polynomials. In the optimization itself a numerical approximation is necessary due to the complexity of the dynamical system therefore the Simpson Integration is being used due to its calculable error. This paper is structured as follows. First, in Section II the equations of motion are explained briefly, then the necessary Bernstein polynomial framework will be derived as well as the needed information about the kind of optimization in the following algorithm. The overview of the optimization is given in Section III. Section IV deals with the results. Discussion and conclusion closes this paper in Section V.

978-1-7281-8609-2/21/$31.00 ©2021 IEEE

Main Nomenclature

q_{11}, q_{12}:	Angle of the manipulator
m_1, m_2:	Mass of link 1 and link 2
G_1, G_2:	Gravitational acceleration
l_1, l_2:	length of link 1 and link 2
l_{c1}:	length to center of gravity of link 1
l_{c2}:	length to center of gravity of link 2
τ_1, τ_2:	Needed torque
C:	Bézier curve
C':	1$^{\text{st}}$ derivative of the Bézier curve
C'':	2$^{\text{nd}}$ derivative of the Bézier curve
$B_{i,n}$:	Bernstein polynomial n-th degree
Cf:	Cost function of the simulation
$\theta_1; \theta_2$:	Angular position

II. Background

A. Robotics

The manipulator,used in the following optimization, consists of two arms in serial arrangement with two rotational joints. This doesn't include the end-effector itself, which will not be simulated. Point *P* in Fig. 1 represents the end-effector interacting with the environment and that is able to pick up pieces or to use tools for machining of work pieces. Therefore it is assumed that the end point of the second link is the Tool Center Point (TCP) representing the end-point which is being moved throughout the optimization. The joints between both links are mass-less and have no length for a more compact simulation, they are ideal joints. In addition, both joints have no restraints in maximal or minimal angular position, velocity or acceleration. The reachable workspace is set to allow the positive *x* and *y* coordinates [6] only.

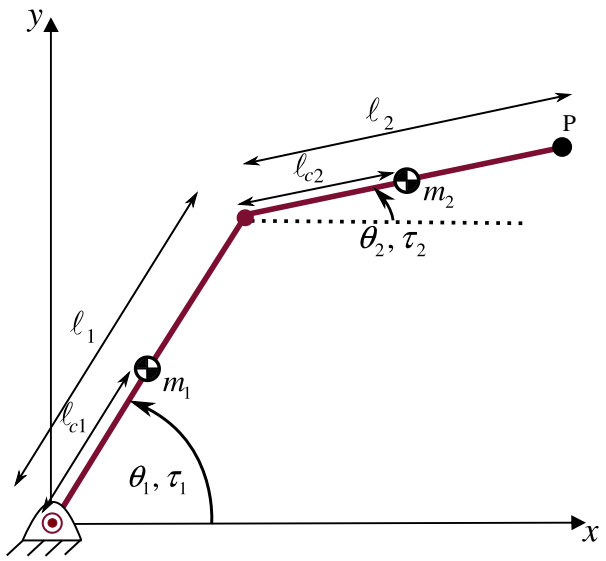

Fig. 1. Graphical representation of the two link manipulator

The Fig. 1 shows schematically the two link manipulator in Cartesian work space. The degrees of freedom are limited to *2*, able to rotate around θ_1 and θ_2 only. This implies that this robot has only two independent position variables in relation to the coordinate system [7]. The following kinematic equations will be derived from the geometric properties. After the understanding of how the two link manipulator is modeled it is now important to describe it not only as a static model but in terms of motion, dynamics. The inverse kinematics necessary for obtaining the joint angles is defined by Eq. (1) and (2). The number of solutions depends on the number of joints [8], in this case for every position of the end effector two solutions can be calculated. Taking the equations of motion further described into consideration, the solutions have an impact on the needed energy due to the gravitational forces. The following figure Fig. 2 shows these possible solutions in a graphical way. Further, the angles q_1 and q_2 are shown.

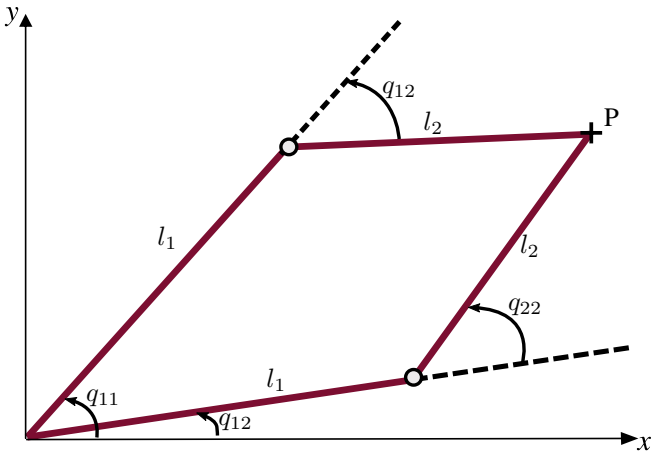

Fig. 2. Possible Solutions for the inverse kinematics

The mathematical solution is written below. Note that either Eq. (1) or Eq. (2) can be used, not a combination of them. The solution for the upper combination is

$$q_{11} = \arctan\frac{y}{x} + \arctan\frac{l_2 \sin q_2}{l_1 + l_2 \cos q_2} \tag{1a}$$

$$q_{21} = -\arccos\frac{x^2 + y^2 - l_1^2 - l_2^2}{2\, l_1\, l_2} \tag{1b}$$

and for the lower combination

$$q_{12} = \arctan\frac{y}{x} - \arctan\frac{l_2 \sin q_2}{l_1 + l_2 \cos q_2} \tag{2a}$$

$$q_{22} = \arccos\frac{x^2 + y^2 - l_1^2 - l_2^2}{2\, l_1\, l_2}. \tag{2b}$$

B. Equations of motion

The following equations describe the model in terms of dynamical components which are derived according to [5]. Now, it is possible to calculate the needed torque for link 1 and link 2, τ_1 and τ_2, for a given set of angular positions θ_1, θ_2. This model includes also the effects of gravitation.

$$H_{11} = m_1\ell_{c1}^2 + m_2\left(\ell_1^2 + \ell_{c2}^2 + 2\ell_1\ell_{c2}\cos\theta_2\right) \tag{3a}$$

$$H_{22} = m_2\ell_{c2}^2 \tag{3b}$$

$$H_{12} = m_2\left(\ell_{c2}^2 + \ell_1\ell_{c2}\cos\theta_2\right) \tag{3c}$$

$$H_{21} = H_{12} \tag{3d}$$

$$h = m_2\ell_1\ell_{c2}\sin\theta_2 \tag{3e}$$

$$G_1 = m_1\ell_{c1}g\cos\theta_1 + m_2g\left\{\ell_{c2}\cos\left(\theta_1 + \theta_2\right) + \ell_1\cos\theta_1\right\} \tag{3f}$$

$$G_2 = m_2g\ell_{c2}\cos\left(\theta_1 + \theta_2\right) \tag{3g}$$

with $l_{c1} = \frac{l_1}{2}$ and $l_{c2} = \frac{l_2}{2}$. The needed torque is calculated as followed.

$$\tau_1 = H_{11}\ddot{\theta}_1 + H_{12}\ddot{\theta}_2 - h\dot{\theta}_2^2 - 2h\dot{\theta}_1\dot{\theta}_2 + G_1 \tag{4}$$

$$\tau_2 = H_{22}\ddot{\theta}_2 + H_{21}\ddot{\theta}_1 + h\dot{\theta}_1^2 + G_2. \tag{5}$$

Now the needed equations for torques (inverse dynamic model) of the two link manipulator are defined. In the next section Bernstein polynomials will be discussed. Then, the mathematical optimization will follow.

C. Bernstein polynomials and Bézier curve

As defined in [9] the Bézier curve is a parameterized sum of polynomials called Bernstein polynomials. Usually, these polynomials are used for modeling free-form curves e.g. in computer aided design. This approach will use the symmetry of the Bernstein polynomials depicted in Fig. 3. They are defined as

$$C(u) = \sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i \quad 0 \le u \le 1 \tag{6}$$

where $B_{i,n}(u)$ is the Bernstein polynomial *n*-th degree which is defined as

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}u^i(1-u)^{n-i}. \tag{7}$$

The form is mathematically equivalent to the power based one but has some unique properties: [9].

1. Partition of unity: $\sum_{i=0}^{n} B_{i,n}(u) = 1$ for all $0 \leq u \leq 1$;
2. $B_{0,n}(0) = B_{n,n}(1) = 1$;
3. $B_{i,n}$ attains exactly one maximum on the interval [0,1], that is, at $u = i/n$;
4. For any *n*, the set of polynomials $B_{i,n}$ is symmetric with respect to $u = 1/2$;
5. Endpoint interpolation: $C(0) = \mathbf{P}_0$ and $C(1) = \mathbf{P}_n$.

Partition of unity means that no matter how many Bernstein polynomials are being used for describing a curve, the sum of these polynomials is always 1. Especially the symmetry is important for the following optimization.
To give a better understanding on how the symmetry characterizes the Bernstein polynomials, the following figure shows the polynomials of the quadratic Bézier curve:
$B_{0,2} = (1-u)^2$, $B_{1,2} = 2u(1-u)$ and $B_{2,2} = u^2$.

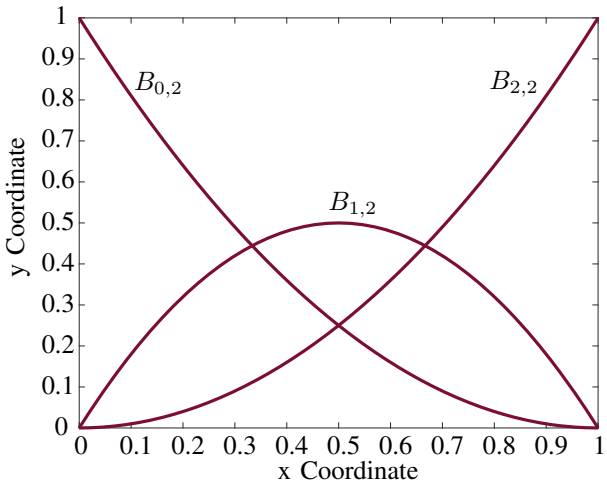

Fig. 3. Bernstein polynomials of quadratic Bézier curve

To determine the first and second derivative of a Bézier curve the following equations are necessary.

$$C^{'}(u) = n \sum_{i=0}^{n-1} B_{i,n-1}(u)(\mathbf{P_{i+1}} - \mathbf{P_i}) \tag{8}$$

with $B_{-1,n-1}(u) = B_{n,n-1}(u) = 0$. And for the second derivative

$$C^{''}(u) = n(n-1) \sum_{i=0}^{n-2} B_{i,n-2}(u)(\mathbf{P_{i+2}} - 2\mathbf{P_{i+1}} + \mathbf{P_i}). \tag{9}$$

Finally the used, 7th degree Bézier curve and its derivatives are shown in Eq. (10, 11) and Eq. (12).

$$\begin{aligned} C(u) = {} & (1-u)^7\mathbf{P0} + 7u(1-u)^6\mathbf{P1} + 21u^2(1-u)^5\mathbf{P2} \\ & + 35u^3(1-u)^4\mathbf{P3} + 35u^4(1-u)^3\mathbf{P4} \\ & + 21u^5(1-u)^2\mathbf{P5} + 7u^6(1-u)\mathbf{P6} + u^7\mathbf{P7} \end{aligned} \tag{10}$$

$$\begin{aligned} C^{'}(u) = {} & 7(1-u)^6(\mathbf{P1} - \mathbf{P0}) + 42u(1-u)^5(\mathbf{P2} - \mathbf{P1}) \\ & + 105u^2(1-u)^4(\mathbf{P3} - \mathbf{P2}) + 140u^3(1-u)^3 \\ & (\mathbf{P4} - \mathbf{P3}) + 105u^4(1-u)^2(\mathbf{P5} - \mathbf{P4}) \\ & + 42u^5(1-u)(\mathbf{P6} - \mathbf{P5}) + 7u^6(\mathbf{P7} - \mathbf{P6}) \end{aligned} \tag{11}$$

$$\begin{aligned} C^{''}(u) = {} & 42(1-u)^5(\mathbf{P2} - \mathbf{2P1} + \mathbf{P0}) + 210u(1-u)^4 \\ & (\mathbf{P3} - \mathbf{2P2} + \mathbf{P1}) + 420u^2(1-u)^3 \\ & (\mathbf{P4} - \mathbf{2P3} + \mathbf{P2}) + 420u^3(1-u)^2 \\ & (\mathbf{P5} - \mathbf{2P4} + \mathbf{P3}) + 210u^4(1-u) \\ & (\mathbf{P6} - \mathbf{2P5} + \mathbf{P4}) + 42u^5(\mathbf{P7} - \mathbf{2P6} + \mathbf{P5}). \end{aligned} \tag{12}$$

D. Optimization and cost function

Convex optimization is characterized by finding the smallest value for a vector $x = (x_1, ..., x_n)$ which satisfies all given constraints being represented by the constraint functions f_i. The constants b_i are the limits of the constraint function defined as

$$\begin{aligned} &\text{minimize} \quad f_0(x) \\ &\text{subject to} \quad f_i(x) \leq b_i, \qquad i = 1, ..., m. \end{aligned} \tag{13}$$

If a convex function is being minimized whether it is with or without constraints, this function will only have one minimum. This function, the cost function, includes mathematical properties from the mechanical system which was described earlier. Due to its complexity and the fact that it includes *sine* and *cosine* functions a non convex cost function can be presumed. This means, not only one global minimum is possible, more local minima are existing. Further the question remains if the cost function, which is acquired in the next chapter, even has a global minimum, which would complicate the optimization even more. A local minimum is defined as a point x_* which satisfies the following condition [10]:

$$f(x_*) \leq f(x) \quad \text{for all } x \text{ such that } \|x - x_*\| < \epsilon. \tag{14}$$

Furthermore, a differentiability of the cost function is assumed, not only for the analysis of the results later but for general understanding of the system, derivatives are necessary. When it comes to the minimization of the two-link manipulator, the first, *first-order necessary condition* Eq. (15) is followed by finding the minima, defining the *sufficient condition*. In this paper the mathematical approach is being simplified by inserting 0 for both variables. As a result the corresponding solution reveals, if it is a minimum or a maximum.

$$\nabla f(x_*) = 0. \tag{15}$$

It is now necessary to define a cost function which will be the core of the optimization. Due to the nature of the system given

the energy to be minimized is given via τ_1 and τ_2, which then will be implemented in the cost function

$$Cf = \int_{t_i}^{t_f} (\tau_1 \dot{\theta}_1 + \tau_2 \dot{\theta}_2) dt. \tag{16}$$

This cost function takes into account both torques needed for the movement, states the energy which the manipulator needs to move from the initial point P_0 to the final point P_1 passing through a given intermediate point P_2. It represents the mechanical work (energy) needed to move the whole manipulator from the beginning trough the specification given by points P_0, P_1 and P_2. In terms of dimension this means finding the optimum in units $J\,[= Nm]$.

III. Structure of the Simulation

Bernstein polynomials and normal polynomials are both capable to describe the motion of two-link manipulator. Therefore it is possible so that a simulation using normal polynomials gives a different result than a simulation with Bernstein polynomials. The results may vary due the different structure of the Bernstein polynomials. That's why the following simulation is done once with power based and once with Bernstein polynomials. The sequence of the individual steps is the same as well as the inputs of the system. Only the construct for implementing the angular position varies. The structure of the Matlab code is shown in the following flowchart Fig. 4. Fig. 5 shows scenarios where the manipulator is not capable of reaching both points. This is referred as 'check possibility' in the flowchart.

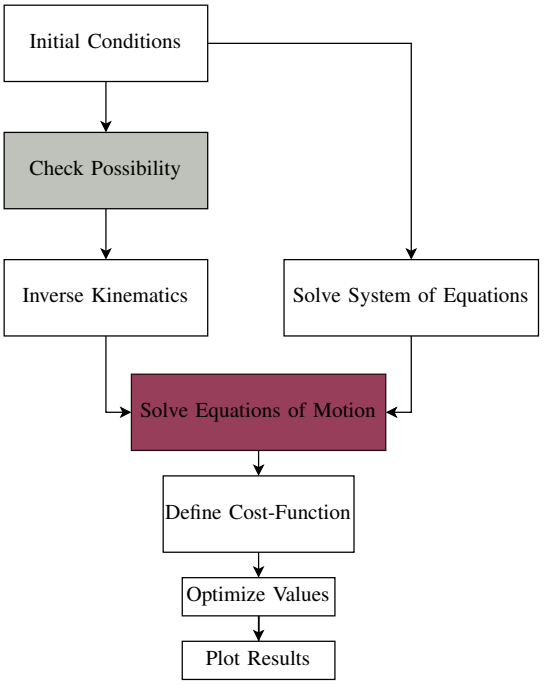

Fig. 4. Flowchart of the algorithm and simulation in Matlab

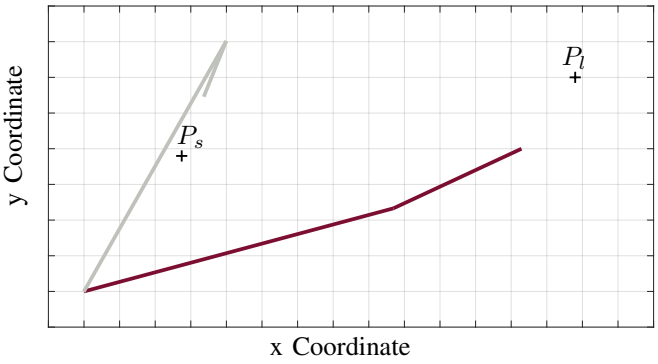

Fig. 5. Scenarios for not reachable points

The coordinates in [m] of the points are $P_0 = [15.7; 2.2]$, $P_1 = [13.1; 6.8]$ and $P_2 = [11.1; 3.5]$. The lengths of the links are $l_1 = 9m$ and $l_2 = 7m$. The masses of the links are $m_1 = 5kg$ and $m_2 = 4kg$. Finally, the gravitation is set to $9.81m\,s^{-2}$. The manipulator has one second to execute the movement. At time $t = 0.6s$, it reaches point P_2. The last setting of the interpolation in the code is sampling (*dT*). Since the integration of the cost function is done numerically, then the degree of fineness can be changed.
For the numerical approximation of the integral the Simpson integration is being used. The cost function itself is complex therefore it is necessary to approximate numerically via

$$S^{(N)}(f) = \frac{h}{6}\Big(f(x_0) + 2\sum_{k=1}^{N-1} f(x_k) + f(x_N) + 4\sum_{k=1}^{N} f\Big(\frac{x_{k-1} + x_k}{2}\Big)\Big). \tag{17}$$

Due to the numerical integration a certain error is being implemented. The error can be reduced approximating with a higher rate.In Fig. 12 a short cutout from the original Matlab code is taken in order to show how the cost function Cf is being optimized.

IV. Results

The following Fig. 6 and Fig. 8 demonstrate the trajectory of the end effector moving from P_0 to P_1 passing through the intermediate Point P_2. The trajectory using the power based polynomials seems to have a more natural curve where as the Bernstein polynomial based one has a little bump at the beginning. In both cases all points have been reached and both trajectories seems to be smooth. Now it is important to look at the needed torque for each link. Fig. 7 shows the needed torque for the first link using polynomial (grey) and Bézier (red) optimization. The horizontal lines represent the approximated absolute sum of the needed torque. Despite the wider interval and dynamic range, the Bézier optimization needs less energy, explicitly $\tau_{Bl1} = 2163.24Nm$ and $\tau_{Pl1} = 2366.11Nm$. The same applies to the second link in Fig. 9, where $\tau_{Bl2} = 275.23Nm$ and $\tau_{Pl2} = 332.44Nm$.

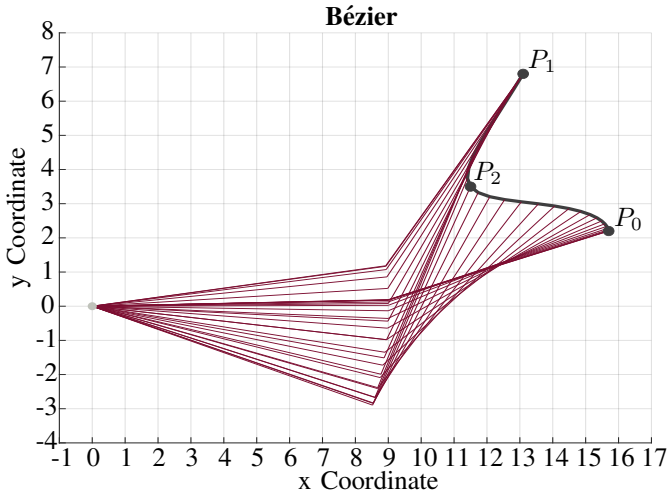

Fig. 6. Graphical representation of optimization with Bézier

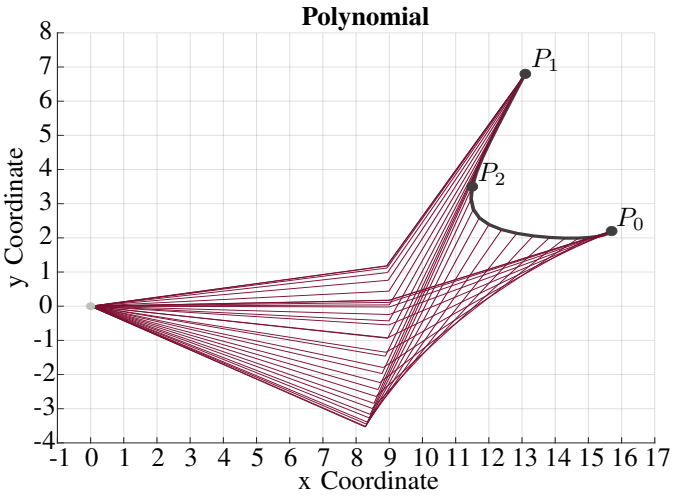

Fig. 8. Graphical representation of optimization with normal polynomials

In both cases less torque is being used for optimizing the given problem witch makes it about 10% more effective with Bernstein polynomials. Comparing the needed torque with the position of the end effector it is possible to see a connection between these graphs. Near $t = 0.6s$ the maximum needed torque for link 1 as well as the minimum needed torque for link 2 are both located giving an indication of the middle point P_2.

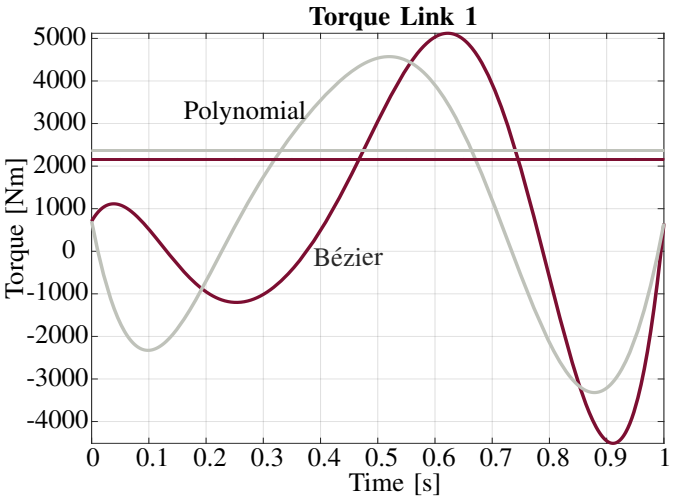

Fig. 7. Needed torque for link 1

The overshoot is lower, especially when looking at the second link and it happens around $t = 0.6s$ which again gives an indication on when the intermediate point is being reached. In this particular scenario not only a more effective optimization is being obtained but a more intuitive model when analyzing its parameters.

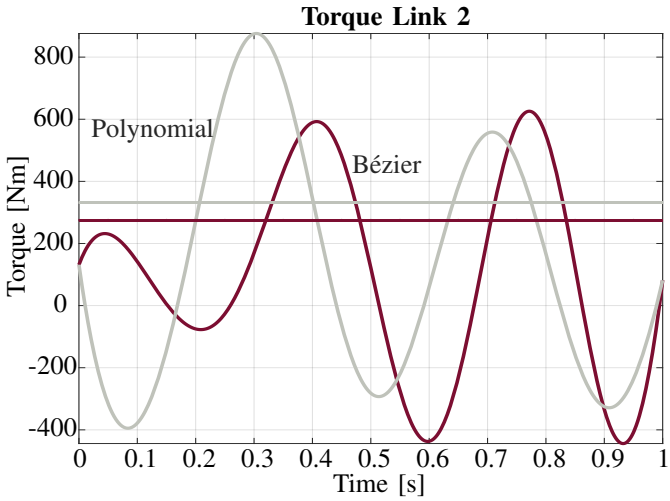

Fig. 9. Needed torque for link 2

Another way of comparing the two optimization methods is looking at the cost-function itself. Again, when comparing these two values the Bézier optimization will reveal a slight advantage in the needed energy which can be seen in Equation (18). Last a distinction is visible in Fig 10. This figure shows the angular position of both links, the upper two lines referring to the first, the lower ones to the second link. While the rough trend remains the same a difference can be seen in the maximum deviation of the desired angular position.

$$Cf_B = 259.48Nm[= J] \quad Cf_P = 306.1Nm \qquad (18)$$

Still important to note is the comparison in acceleration of link 1 shown in Fig. 11. The maximum and minimum value of the acceleration profile using the Bézier optimization is higher but its absolute mean value is lower than the polynomial one. A similar result is obtained comparing the second link. This and the maximum and minimum torque values in Fig. 7 and Fig. 9 show a higher range but smaller mean value reflecting a lower energy consumption.

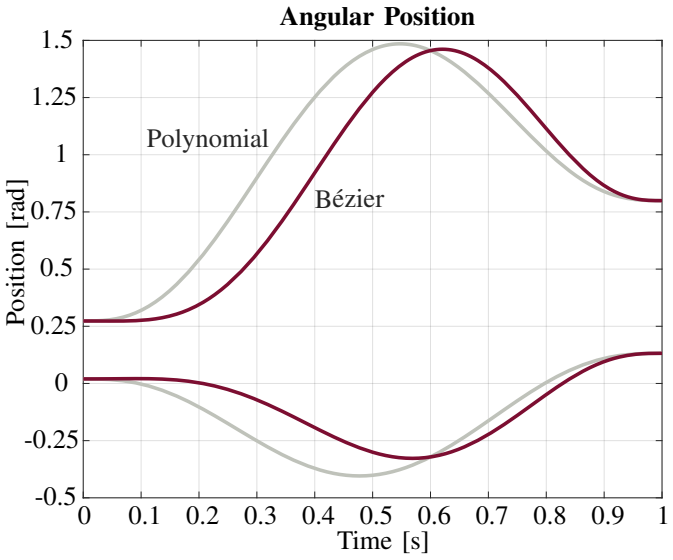

Fig. 10. Angular position compared

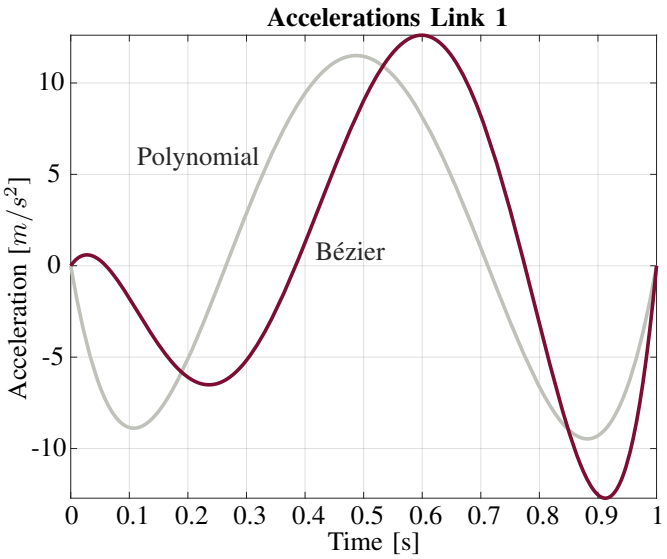

Fig. 11. Acceleration of Link 1

V. Conclusion and Future Work

Comparing the needed torque with the trajectory the refereed indication of the intermediate point can give a better understanding in the peak of needed torque while modeling the trajectory itself. It ca be an indication on when to expect the peak. In accordance with the obtained results, Bernstein polynomials offer interesting applications in the field of control.

In fact, as already observed, because of the symmetric structure of Bernstein polynomial curve parameterization, for the same given constraints, smoother trajectories can be obtained. Smoother trajectories imply small accelerations and thus less energy involved in the motion. If the obtained Bernstein polynomial trajectories are taken as reference trajectories for a controller of a control loop, this offers a real opportunity to improve the performance of the whole controlled system. It can moreover be complemented by some initial path shaping or smoothing as it is presented in [11].

```
Cf=(tau1*Cu1)+(tau2*Cu2);
%Simpson
N=1/dT;
f0=subs(Cf,t,ti);
fN=subs(Cf,t,tf);
I1=ti+dT:dT:tf-dT;
for s1=1:N-1
    T1(s1)=subs(Cf,t,I1(1,s1));
end
fs1=sum(T1);
I2=ti:dT:tf;
for s2=1:N
    T2(s2)=subs(Cf,t,((I2(1,s2)+I2(1,s2+1))/2));
end
fs2=sum(T2);
CApp=(dT/6)*(f0+2*fs1+fN+4*fs2);
GApp=gradient(CApp); GApp==0;
I=subs(GApp,{P3_1,P3_2},{0,0});I=double(I);
min=min(I);
%Take minimum
for i=1:length(I)
    if I(i,1)==min
        Opt=GApp(i,1);
    end
end
Opt==0;Opt=solve(Opt,P3_1,P3_2);
P3_1opt=Opt.P3_1;                P3_2opt=Opt.P3_2;
```

Fig. 12. Overview of the algorithm for optimizing the cost function.

References

[1] A. Fabbrini, A. Garulli, and P. Mercorelli. A trajectory generation algorithm for optimal consumption in electromagnetic actuators. *IEEE Transactions on Control Systems Technology*, 20(4):1025–1032, 2012.

[2] W. Hoffmann, K. Peterson, and A. G. Stefanopoulou. Iterative learning control for soft landing of electromechanical valve actuator in camless engines. *IEEE Transactions on Control Systems Technology*, 11(2):174–184, 2003.

[3] Chun Tai and Tsu-Chin Tsao. Control of an electromechanical actuator for camless engines. In *Proceedings of the 2003 American Control Conference, 2003.*, volume 4, pages 3113–3118 vol.4, 2003.

[4] P. Mercorelli. Trajectory tracking using mpc and a velocity observer for flat actuator systems in automotive applications. In *2008 IEEE International Symposium on Industrial Electronics*, pages 1138–1143, 2008.

[5] S. Oh and K. Kong. Two-degree-of-freedom control of a two-link manipulator in the rotating coordinate system. *IEEE Transactions on Industrial Electronics*, 62(9):5598–5607, 2015.

[6] Wolfgang Weber. *Industrieroboter: Methoden der Steuerung und Regelung*. Hanser, München, 4., aktualisierte auflage edition, 2019.

[7] H. Harry Asada. Introduction to robotics: Lecture notes. Massachusetts Institute of Technology.

[8] John J. Craig. *Introduction to robotics: Mechanics and control*. Pearson Education, Upper Saddle River, N.J., 3rd ed. edition, 2005.

[9] Les A. Piegl and Wayne Tiller. *The NURBS book*. Monographs in visual communications. Springer, Berlin and London, 2nd ed. edition, 1997.

[10] Igor Griva. *Linear and nonlinear optimization, second edition*. Society for Industrial and Applied Mathematics, 2008.

[11] K. Belda. Smoothing and time parametrization of motion trajectories for industrial machining and motion control. In *Proc. of the 16th Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO 2019)*, pages 229–236. SCITEPRESS, 2019.