Mech. Sci., 6, 245–254, 2015
www.mech-sci.net/6/245/2015/
doi:10.5194/ms-6-245-2015

Mechanical Sciences
Open Access

Successive dynamic programming and subsequent spline optimization for smooth time optimal robot path tracking

M. Oberherber, H. Gattringer, and A. Müller

Institute of Robotics, Johannes Kepler University Linz, Altenbergerstr. 69, 4040 Linz, Austria

Correspondence to: M. Oberherber (matthias.oberherber@jku.at)

Received: 28 May 2015 – Revised: 3 September 2015 – Accepted: 1 October 2015 – Published: 26 October 2015

Abstract. The time optimal path tracking for industrial robots regards the problem of generating trajectories that follow predefined end-effector (EE) paths in shortest time possible taking into account kinematic and dynamic constraints. The complicated tasks used in industrial applications lead to very long EE paths. At the same time smooth trajectories are mandatory in order to increase the service life.

The consideration of jerk and torque rate restrictions, necessary to achieve smooth trajectories, causes enormous numerical effort, and increases computation times. This is in particular due to the high number of optimization variables required for long geometric paths. In this paper we propose an approach where the path is split into segments. For each individual segment a smooth time optimal trajectory is determined and represented by a spline. The overall trajectory is then found by assembling these splines to the solution for the whole path. Further we will show that by using splines, the jerks are automatically bounded so that the jerk constraints do not have to be imposed in the optimization, which reduces the computational complexity. We present experimental results for a six-axis industrial robot. The proposed approach provides smooth time optimal trajectories for arbitrary long geometric paths in an efficient way.

1 Introduction

Highly automated production lines with efficient exploitation of the available resources become increasingly important for high wage countries to stay competitive. The utilization of present hardware plays a central role. This can be achieved by an intelligent path planning, using mathematical models of the mechanics to consider technical constraints in an optimization for the determination of optimal motions. It is an established approach to divide the problem into the geometric path planning using a scalar path parameter s and optimization (Bobrow et al., 1985; Pfeiffer and Johanni, 1987). The path planning can for instance be done by combining lines and circles via clothoids. A popular and comfortable way is the definition of the path using polynomials or splines (Geu Flores and Kecskemethy, 2012; Gattringer et al., 2014). In Quang-Cuong (2014) a classification is proposed to divide optimization strategies deriving an optimal trajectory in the 2-D-phase space with coordinates $(s, \dot{s})$ into three families:

1. Dynamic programming (DP): the phase space is divided into a discrete grid. Based on Bellman's optimality principle (Bellman and Dreyfus, 1962) an optimal trajectory is derived on this grid (Shin and McKay, 1985).

2. Numerical integration: the optimal solution for the path parameter time evolution is obtained by integrating maximum and minimum accelerations and finding optimal switching points for this acceleration and deceleration periods in the phase plane (Bobrow et al., 1985; Pfeiffer and Johanni, 1987). Geu Flores and Kecskemethy (2012) used this method to solve the so-called generalized waiter motion problem.

3. Convex optimization: the third method is based on a convex formulation of the optimization problem that can be solved with efficient optimization packages (Verscheure et al., 2009; Ardeshiri et al., 2011).

The tasks to be performed in industrial applications often contain long paths with rough as well as fine contours. Furthermore, the movement should be precise in order to comply with required accuracies on the one hand, and the motor torques should be smooth in order to avoid vibrations and to protect the hardware. Short optimization times are also crucial in order that the saving in execution time is not annihilated by increased calculation times.

To overcome these demands, one may start to optimize the whole path using one approach mentioned above considering kinematic (joint velocity and acceleration) and dynamic (torque) constraints. Smooth trajectories can be achieved by taking jerk or torque rate restrictions into account. In Constantinescu and Croft (2000) and Oberherber et al. (2014) methods to consider them in phase space are presented, while Debrouwere et al. (2013) proposes a sequential convex scheme to solve a nonlinear program. However, for standard six axis industrial robots and long geometric paths the calculation effort is enormous due to the high number of optimization variables and restrictions. This is also caused by the fine discretization, required for a detailed implementation of fine structures of the geometric path. A non-equidistant discretization, where only certain regions of the path are discretized finely, would be a first choice to reduce the size of the problem. But, if the path is very long and contains many finely discretized regions, this approach is not feasible. Another way to reduce the number of optimization variables is a decomposition of the path into segments, performing the optimization for the segments and assembling the solutions to the whole optimal trajectory. In order to achieve a continuous trajectory, terminal conditions have to be defined at the intersection points and regarded in the optimization. We use a DP approach to calculate optimal trajectories for each segment and subsequently for the entire path. For this task we consider restrictions of path velocity, joint velocity and acceleration and also torque limits. Neglecting the jerk restrictions is reflected in a bang-bang behavior of the motor torques.

To obtain smooth robot movements in short optimization times, we approximate the solution provided by the DP approach for each segment of the path with a spline. Subsequently, this approximation is used as initial guess for the optimization of the spline using an active-set solver (Nocedal and Wright, 2006), considering the same restrictions as before. The degree of the splines and the number of control points determine the smoothness of the solution. Since the initial state is near to the global optimum, the calculation times remain low.

As the initial state for the spline optimization is derived by a DP approach, the first part can be assigned to category one. A categorization of the spline optimization is not really possible, since we use an active set solver to optimize the non-convex problem.

The paper is organized as follows: in Sect. 2 the geometric path, describing the task the robot should perform, is defined. Section 3 treats the time optimal path planning problem in the parameter space in a general way, Sect. 4 introduces our solution strategy. Section 5 addresses the used optimization based on DP in detail. An approach to attain smooth trajectories with short calculation times is introduced in Sect. 6. Experimental results, realized on a Stäubli RX130L – a six-axis industrial robot, are presented in Sect. 7.

In this paper we use the abbreviation $\dot{(\,)} = \frac{\mathrm{d}}{\mathrm{d}t}$ for the time derivative of a quantity. The derivative with respect to the path parameter is denoted with $(\,)' = \frac{\mathrm{d}}{\mathrm{d}s}$. Bold lower case letters characterize vectors, while capital letters are used for matrices with the exception of commonly used notations in mechanics. The euclidean norm of the vector $\boldsymbol{x}$ is denoted with $\|\boldsymbol{x}\|$.

2 Geometric path planning

The aim of path planning is the definition of the robot's task. There are several methods to define such a geometric path, like polynomials, combination of lines, circles and clothoids (Müller et al., 2007) or splines (Bobrow, 1988; Gattringer et al., 2014). The latter provide a comfortable way to define the robots motion whether in joint coordinates $\boldsymbol{q}$ or in Cartesian coordinates z_{E}. To consider obstacles in the workspace, it is common to define the geometric path in the Cartesian space as shown in Fig. 1.

A three-dimensional spline curve, describing the end-effector position can be written as

$$\boldsymbol{r}_{\mathrm{E}}(s) = \sum_{l=1}^{n_{\mathrm{D}}} \boldsymbol{d}_l N_l^d(s), \tag{1}$$

with the scalar path parameter $s = [s_{\mathrm{B}}, s_{\mathrm{E}}]$ (s_{B} – begin of the path, s_{E} – end of the path), and the n_{D} control points $\boldsymbol{d}_l$ defining the shape of the curve. These control points can either be defined directly or can be determined using interpolation points $\boldsymbol{p}_l$ as shown in Gattringer et al. (2014). $N_l^d(s)$ denote the B-spline basis functions of degree d. There are different ways to define this basis functions as local or global support functions. For details we refer to De Boor (1978) and Piegl and Tiller (1995). Unit Quaternions $\mathcal{Q} = \left[e_0, \boldsymbol{e}^T\right]^T$ (scalar part e_0 and vector part $\boldsymbol{e}$) are used for the definition of the end-effector orientation $\mathcal{Q}_{\mathrm{E}}$. The evolution of the separate coordinates e_0, e_x, e_y and e_z along the path is again defined via splines with a subsequent normalization. The angular velocity of the end-effector, represented in the end-effector frame, can be calculated to ${}_{\mathrm{E}}\boldsymbol{\omega}_{\mathrm{E}} = 2\,[e_0\dot{\boldsymbol{e}} - \widetilde{\boldsymbol{e}}\dot{\boldsymbol{e}} - \boldsymbol{e}\dot{e}_0]$.

3 Time optimal path tracking

3.1 Problem description

In Sect. 2 the geometric path is defined as a function of the scalar path parameter s. The goal of this section is to find an optimal relationship between the path parameter and time:

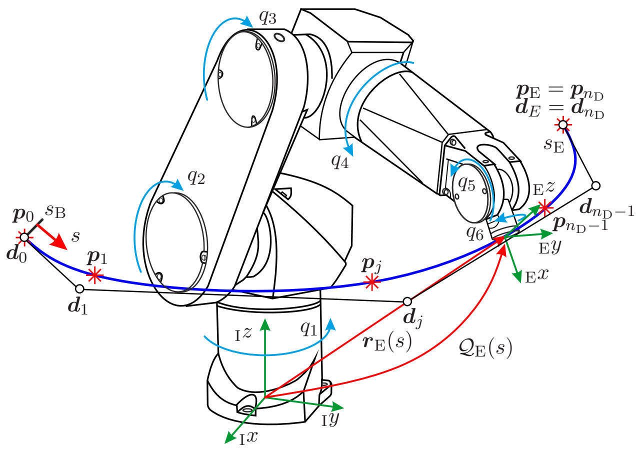

Figure 1. Six-axis industrial robot with geometric path.

$s(t)$. This is accomplished with an optimization considering technical constraints like motor-torque, velocity and acceleration restrictions. The general formulation of the path tracking problem is given by

$$\min_{t_\mathrm{E}\in\mathbb{R}^+,\boldsymbol{\tau}(\cdot)} \int_0^{t_\mathrm{E}} \left(k_1 + k_2 \boldsymbol{\tau}^T \boldsymbol{\tau}\right) \mathrm{d}t, \tag{2}$$

wherein $\boldsymbol{\tau}$ denotes the vector of motor torques as a function of time. With the coefficients k_1 and k_2 a weighting between time and energy optimality can be achieved. Basically we are able to handle this general case, but in this paper we concentrate on time optimal solutions, therefore the factors $k_1 = 1$ and $k_2 = 0$ are used and the torques vanish from the cost function. The cycle time t_E represents the solution of the optimization and is consequently an unknown quantity. A change of the integration variable from t to s leads to

$$t_\mathrm{E} = \int_0^{t_\mathrm{E}} 1\,\mathrm{d}t = \int_{s_\mathrm{B}}^{s_\mathrm{E}} \frac{1}{\dot{s}(s)}\,\mathrm{d}s \tag{3}$$

and the optimization problem can be written as

$$\min_{\dot{s}(\cdot)} \int_{s_\mathrm{B}}^{s_\mathrm{E}} \frac{1}{\dot{s}(s)}\,\mathrm{d}s. \tag{4}$$

3.2 Technical constraints

3.2.1 Process constraints

For the trajectory optimization several technical constraints should be considered. The first one concerns the end-effector velocity $v_\mathrm{E} = \left\| \frac{\mathrm{d}\boldsymbol{r}_\mathrm{E}}{\mathrm{d}t} \right\|$, which is a process related restriction. Such constraints can be found in grinding or welding operations. By using the chain rule

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{\mathrm{d}x}{\mathrm{d}s}\frac{\mathrm{d}s}{\mathrm{d}t} = \frac{\mathrm{d}x}{\mathrm{d}s}\dot{s} = x'\dot{s}, \tag{5}$$

the path velocity follows as

$$v_\mathrm{E} = \left\| \frac{\mathrm{d}\boldsymbol{r}_\mathrm{E}}{\mathrm{d}s}\frac{\mathrm{d}s}{\mathrm{d}t} \right\| = \left\| \boldsymbol{r}'_\mathrm{E}(s) \right\| \dot{s} \tag{6}$$

in the parameter range.

3.2.2 Manipulator constraints

Restrictions imposed by the used hardware concern the motor torque, joint velocity, and joint acceleration. In Sect. 2 the path is determined in Cartesian coordinates whereas these restrictions are defined in joint coordinates. Since the end-effector coordinates are calculated with the forward kinematics $\underline{z}_\mathrm{E} = [\boldsymbol{r}_\mathrm{E}^T\, \mathcal{Q}_\mathrm{E}^T]^T = \boldsymbol{f}(\boldsymbol{q})$ for desired joint positions, the inverse kinematics $\boldsymbol{q}(s) = \boldsymbol{f}^{-1}(\underline{z}_\mathrm{E}(s))$ provides the joint angles for desired end-effector coordinates. There are different ways to solve this locally, but not globally unique problem. We use a numerical approach based on the relation

$$\begin{pmatrix} \boldsymbol{v}_\mathrm{E} \\ \boldsymbol{\omega}_\mathrm{E} \end{pmatrix} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}}\ , \tag{7}$$

with the end-effector velocities $\boldsymbol{v}_\mathrm{E}$, $\boldsymbol{\omega}_\mathrm{E}$ represented in the inertial frame and the Jacobian

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \mathbf{v}_\mathrm{E}}{\partial \dot{\boldsymbol{q}}} \\ \frac{\partial \boldsymbol{\omega}_\mathrm{E}}{\partial \dot{\boldsymbol{q}}} \end{pmatrix}. \tag{8}$$

With the chain rule Eq. (5), the joint velocity and acceleration follow as

$$\dot{\boldsymbol{q}} = \frac{d\boldsymbol{q}}{ds}\dot{s} = \boldsymbol{q}'\dot{s} \tag{9}$$

$$\ddot{\boldsymbol{q}} = \boldsymbol{q}''\dot{s}^2 + \frac{1}{2}\boldsymbol{q}'(\dot{s}^2)'. \tag{10}$$

The end-effector prime quantities are calculated, using

$$z'_\mathrm{E}(s) = \left[\left(\boldsymbol{r}'_\mathrm{E}(s)\right)^T, \left(\boldsymbol{\omega}_\mathrm{E}(s)\right)^T\right]^T,$$

$$z''_\mathrm{E}(s) = \left[\left(\boldsymbol{r}''_\mathrm{E}(s)\right)^T, \left(\boldsymbol{\omega}'_\mathrm{E}(s)\right)^T\right]^T$$

and the Jacobian $\mathbf{J}$

$$\boldsymbol{q}' = \mathbf{J}^{-1} z'_\mathrm{E} \tag{11}$$

$$\boldsymbol{q}'' = \mathbf{J}^{-1}\left[z''_\mathrm{E} - \mathbf{J}'\boldsymbol{q}'\right]. \tag{12}$$

In order to formulate the torque restrictions, a dynamic model of the robot is necessary. It is derived with the help of the Projection Equation (Bremer, 2008) resulting in the Equations of Motion (EoM)

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{\tau}, \tag{13}$$

wherein $\mathbf{M}(\boldsymbol{q})$ is the position dependent, symmetric and positive definite mass matrix. All other generalized forces like

Coriolis-, gravitational-, centrifugal- and friction forces are contained in $\boldsymbol{g}$. The motor torques are represented by $\boldsymbol{\tau}$. As well as the velocity and acceleration restrictions Eqs. (6), (11), (12), also the torque restrictions are required in terms of the path parameter. There exist different ways to express the EoM in terms of the path parameter as those proposed in Johanni (1988) and Geu Flores and Kecskemethy (2012). An analytical formulation is shown in Gattringer et al. (2014), where the parametrized EoM are written as

$$\boldsymbol{\tau} = \boldsymbol{a}(s)(\dot{s}^2)' + \boldsymbol{b}(s)\,\dot{s}^2 + \mathbf{c}(s) + \boldsymbol{d}_{\mathrm{v}}(s)\dot{s}. \tag{14}$$

The coefficients $\boldsymbol{a}$, $\boldsymbol{b}$, $\mathbf{c}$ and $\boldsymbol{d}_{\mathrm{v}}$ of the parametrized EoM follow directly by rewriting and parametrizing the Projection Equation.

By introducing the variable $z = \dot{s}^2$ the optimization problem in parameter space follows as

$$\min_{z(\cdot)} \int_{s_{\mathrm{B}}}^{s_{\mathrm{E}}} \frac{1}{\sqrt{z(s)}}\mathrm{d}s \tag{15}$$

$$s.t.\ \left\|\boldsymbol{r}'_{\mathrm{E}}(s)\right\|\sqrt{z(s)} \le v_{\mathrm{E,max}} \tag{16}$$

$$\dot{\boldsymbol{q}}_{\min} \le \boldsymbol{q}'(s)\sqrt{z(s)} \le \dot{\boldsymbol{q}}_{\max} \tag{17}$$

$$\ddot{\boldsymbol{q}}_{\min} \le \boldsymbol{q}''(s)z(s) + \frac{1}{2}\boldsymbol{q}'(s)z'(s) \le \ddot{\boldsymbol{q}}_{\max} \tag{18}$$

$$\boldsymbol{\tau}_{\min} \le \boldsymbol{a}(s)z'(s) + \boldsymbol{b}(s)z(s) + \mathbf{c}(s) + \boldsymbol{d}_{\mathrm{v}}(s)\sqrt{z(s)} \le \boldsymbol{\tau}_{\max}. \tag{19}$$

The values for the motor torque restrictions $\boldsymbol{\tau}_{\min}$, $\boldsymbol{\tau}_{\max}$ and joint velocity restrictions $\dot{\boldsymbol{q}}_{\min}$, $\dot{\boldsymbol{q}}_{\max}$ can be taken from the data sheets of the motors respectively gears. Generally the lower limits are equal to the negative values of the upper limits: $\boldsymbol{\tau}_{\min} = -\boldsymbol{\tau}_{\max}$ and $\dot{\boldsymbol{q}}_{\min} = -\dot{\boldsymbol{q}}_{\max}$. The same applies to the acceleration restrictions $\ddot{\boldsymbol{q}}_{\min} = -\ddot{\boldsymbol{q}}_{\max}$, which are usually defined in the joint controllers. Depending on the process to be performed, the path velocity limit $v_{\mathrm{E,max}}$ is set to the optimal working speed.

The better the robot model matches the real system, the better the limits can be exploited. Primarily parameters that are necessary to simulate the kinematics, like link lengths or distances between axis can be taken from CAD – data. The parameters for the dynamic model – masses, centers of gravity or inertia tensors can usually only be estimated with CAD models. For a good match of the derived robot model with the real system, a parameter identification as shown in Neubauer et al. (2014) and Swevers et al. (1996) is indispensable.

4 General solution strategy

The requirements of smooth time optimal trajectories for long geometric paths and short computation times were discussed in Sect. 1 as well as our idea to overcome this challenge. A reduction of optimization variables for a particular optimization is achieved by dividing the path into sections and performing the optimization for these segments. To get smooth trajectories in acceptable calculation times, we define the evolution of $z(s)$ in the "phase space" $s \times z$ as a smooth spline curve and optimize its shape with an active-set solver. For this kind of problem a suitable initial guess $z_0(s)$ is crucial but is not always easy to define. In Verscheure et al. (2009) a parabola is used. This approach works reasonably well for zero velocity at the begin and end of the path, but is not guaranteed to work in the general case of desired start and end velocities. The division of the path into segments makes their consideration mandatory to achieve a continuous overall trajectory. For that purpose we propose an elegant way to derive the initial states by using a DP approach considering the terminal conditions and approximate its solution with a spline curve. Since the DP approach provides the global optimum, it yields an excellent approximation. However, the discretization for this algorithm can be chosen coarse in order to save computation time, since it only provides the initial states for a further optimization of the spline curve.

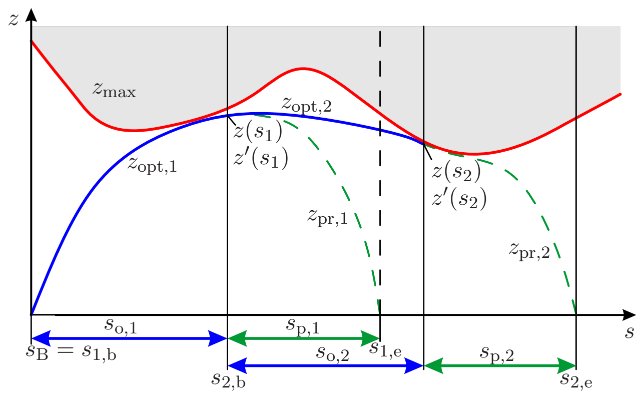

Figure 2. Solution strategy: successive dynamic programming (SDP).

The piecewise optimization procedure is sketched in Fig. 2. It starts with the definition of an optimization s_{o} and a prediction horizon s_{p}. From s_{B} to $s_{\mathrm{o},1} + s_{\mathrm{p},1} = s_{1,\mathrm{e}}$ the optimization is performed and provides an optimal trend, that is split into $z_{\mathrm{opt},1}$ and $z_{\mathrm{pr},1}$. Along the optimization horizon the optimal trajectory is stored and used as a part of the optimal trajectory of the whole path. The prediction horizon is necessary to determine the terminal conditions. At the end of the optimization horizon $z(s_1)$ and $z'(s_1)$ are stored as terminal conditions. Afterwards the horizons are shifted forward by $s_{\mathrm{o},1}$ and the next optimization starts at $s_{2,\mathrm{b}}$ and leads to $s_{2,\mathrm{b}} + s_{\mathrm{o},2} + s_{\mathrm{p},2} = s_{2,\mathrm{e}}$ under consideration of $z(s_1)$, $z'(s_1)$. This procedure is repeated till the end of the path is reached. In further consequence we will call the sequential execution of the algorithm successive dynamic programming (SDP).

The length of the horizons should be chosen carefully. A short optimization horizon in combination with a long prediction horizon leads to unnecessary long calculation times since the solution of the prediction horizon is discarded. Conversely a long optimization horizon and a short prediction

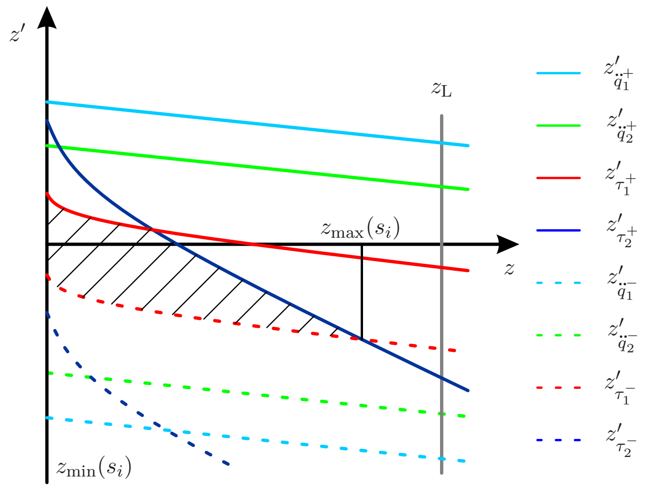

Figure 3. Graphical illustration of technical restrictions in the $z \times z'$ plane.

horizon effects unnecessary deceleration phases. A general choice can not be proposed, rather they have to be chosen problem dependent.

5 Dynamic programming approach to determine an initial solution

5.1 Graphical illustration of restrictions

For a fix point s_i on the path the restrictions in Eqs. (16)–(19) can be graphically illustrated in the $z \times z'$ plane. This is exemplary shown in Fig. 3 for $k = 2$ degrees of freedom and with the assumption that the lower restrictions are equal to the negative upper restriction values.

The velocity restrictions can be squared and combined to

$$z_{\mathrm{L}} = \min\left[\frac{\dot{q}^2_{k,\max}}{q'^2_k}, \frac{v^2_{\mathrm{E,\,max}}}{\left\|\boldsymbol{r}'_{\mathrm{E}}\right\|^2}\right] \tag{20}$$

representing a vertical line in the $z \times z'$ plane. For each joint the acceleration limits follow to

$$z'_{\ddot{q}^{\pm}_k} = \frac{2\left(\pm\ddot{q}_{k,\max} - q''_k z\right)}{q'_k} \tag{21}$$

and thus to linear functions in the $z \times z'$ plane. Due to the consideration of viscous friction (coefficient $\boldsymbol{d}_{\mathrm{v}}$) each joints torque restriction follows to

$$z'_{\tau^{\pm}_k} = \frac{b_k}{a_k}z + \frac{c_k}{a_k} + \frac{d_{\mathrm{v},k}}{a_k}\sqrt{z} \pm \frac{\tau_{k,\max}}{a_k} \tag{22}$$

and thus to nonlinear functions in the $z \times z'$ plane.

The set of curves provide a feasible region (shaded in Fig. 3) for the states. Significant points are given by the minimum and maximum feasible values of z, denoted with $z_{\min}(s_i)$ and $z_{\max}(s_i)$. In Fig. 3 $z_{\min}(s_i)$ is equal to zero. Since a numerical method is used to derive the feasible region it is also possible to consider values $z_{\min}(s_i) > 0$, as they occur for example at the so-called waiter motion problem in Geu Flores and Kecskemethy (2012). $z_{\max}(s_i)$ can either be given by the intersection point of the lowest upper with the highest lower restriction or can concur with the velocity restriction z_{L}. Within the range $z = z_{\min}\ldots z_{\max}(s_i)$ the feasible region defines the minimum $\underline{z}'(z)$ and maximum $\overline{z}'(z)$ allowed gradients.

5.2 Discretizing the problem

The DP approach is based on Bellman's optimality principle. "[...] An optimal policy has the property that whatever the initial state and the initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision (Bellman and Dreyfus, 1962) [...]".

With a process running backwards, these decisions can be picked up and optimized with respect to a desired optimality. For that purpose the first step is to discretize the whole path into n segments $s = s_{\mathrm{B}}\ldots s_i \ldots s_{\mathrm{E}}$ with a discretization step size $\Delta s = \frac{s_{\mathrm{E}} - s_{\mathrm{B}}}{n}$ and $i = 0\ldots n$. Then the optimization horizon s_{o} with n_{o} discrete points and prediction horizon s_{p} with n_{p} discrete points are chosen as integral multiples of Δs. In the following only one segment $s = s_{\mathrm{b}}\ldots s_{\mathrm{e}}$ is considered, where the start point is denoted with s_{b} and the endpoint with s_{e}. Afterwards we evaluate the minimum $z_{\min}(s_i)$ and maximum admissible values $z_{\max}(s_i)$ at all discrete points as shown in Fig. 3 resulting in the limiting curves $z_{\min}(s)$ and $z_{\max}(s)$, that provide the base of operations. Further z is discretized with the step size $\Delta z = \frac{z_{\max}(s_i) - z_{\min}(s_i)}{m}$ into m pieces in the range $z = z_{\min}(s_i)\ldots z_{\min}(s_i) + j\,\Delta z \ldots z_{\max}(s_i)$ with $j = 0\ldots m$. The treatment of the path sections leads to a discrete $(n_{\mathrm{o}} + n_{\mathrm{p}}) \times m$ grid instead of a $n \times m$ grid $(n_{\mathrm{o}} + n_{\mathrm{p}} < n)$ in the phase plane. Within this grid also the cost functional from Eq. (15) has to be discretized to

$$W = \sum_{i=n_{\mathrm{b}}}^{n_{\mathrm{e}}} \frac{\Delta s}{\sqrt{z_i}} \tag{23}$$

with $n_{\mathrm{b}} = s_{\mathrm{b}}/\Delta s$ and $n_{\mathrm{e}} = s_{\mathrm{e}}/\Delta s$.

5.3 Successive dynamic programming

The backwards running process starts at the paths end by initializing the cost function W. Starting from a desired end velocity represented by z_{e} the gradients to all velocity points $z_{e-1,j}$ on the path point s_{e-1} are calculated with

$$z'_{e-1,j} = \frac{z_{e-1,j} - z_{\mathrm{e}}}{\Delta s}. \tag{24}$$

Subsequently they are compared to the minimum $\underline{z}'_{e-1,j}$ and maximum $\overline{z}'_{e-1,j}$ allowed gradients at these points, provided

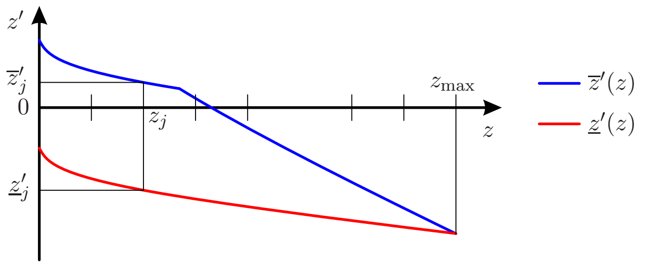

Figure 4. Feasible region in the $z \times z'$ plane with max $\overline{z}'_j$ and min $\underline{z}'_j$ at z_j.

by Fig. 4. The cost function is initialized with $W_{e-1,j} = \frac{\Delta s}{\sqrt{z_j}}$ at points with valid gradients (green solid in Fig. 5) and with $W_{e-1,j} \rightarrow \infty$ for points with invalid gradients (green dashed). With the cost function on hand the proceeding can be continued for the path points $s_{e-2} \ldots s_{\mathrm{b}}$. At these points the cost function has to be considered in the calculation of the optimal gradients. This is accomplished by determining the minimum and maximum allowed gradients $\underline{z}'_{i,j}$ and $\overline{z}'_{i,j}$ using Fig. 4 of the actual point $z_{i,j}$ and calculating the covered range $[\underline{z}_{i,j}, \overline{z}_{i,j}]$ at the following path point s_{i+1}. Within this range along the z axis the location of the cost functions minimum $z^*_{i,j}$ has to be found. A popular procedure is a minimum search based on the golden ratio. However, we take advantage of its special shape for purely time optimal trajectories $W = \frac{\Delta s}{\sqrt{z}}$. A closer examination shows that the minimum is located at the highest value of z which leads to a feasible value different to ∞. This step is reflected in a strongly reduced computation time. With $z^*_{i,j}$ the optimal gradient follows to $z'_{i,j} = \frac{z^*_{i,j} - z_{i,j}}{\Delta s}$ and the cost function is adapted to $W_{i,j} = W_{i+1,j} + \frac{\Delta s}{\sqrt{z_{i,j}}}$ or $W_{i,j} \rightarrow \infty$ if it is out of range. The whole procedure has to be executed for every discrete point at the remaining $n_{\mathrm{e}} - n_{\mathrm{b}} - 2 \times m$ grid. After completion of the optimal gradient determination, the optimal evolution of $z(s)$ can be calculated iteratively with $z_{\mathrm{opt},i+1} = z_{\mathrm{opt},i} + z'_{i,j} \Delta s$ (blue in Fig. 5), starting with a desired start value $z_{\mathrm{opt}}(s_{\mathrm{b}}) = z_{\mathrm{b}}$ and a valid gradient $z'(s_{\mathrm{b}}) = z'_{\mathrm{b}}$ provided by the terminal conditions.

6 Spline based smoothing of the initial solution

6.1 Local spline approximation

The solution provided by the DP approach in Sect. 5 leads to a bang-bang – behavior in the motor torques, resulting in heavy stress for the actuators and the mechanics. In Oberherber et al. (2014) an approach is proposed, that considers torque derivative and joint jerk restrictions in the optimization. This extension of the DP algorithm results in long calculation times and is thus not feasible for long paths. For this reason, we propose a different way in this paper to obtain

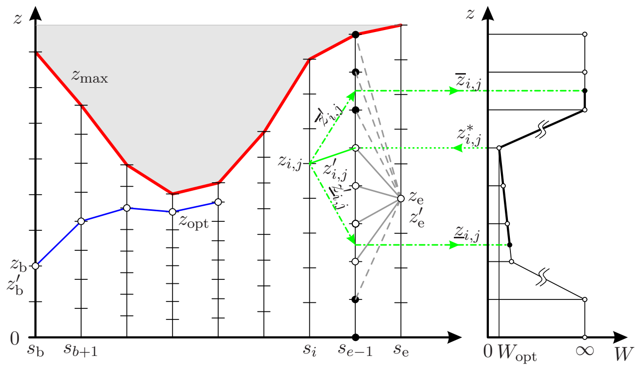

Figure 5. Dynamic programming algorithm.

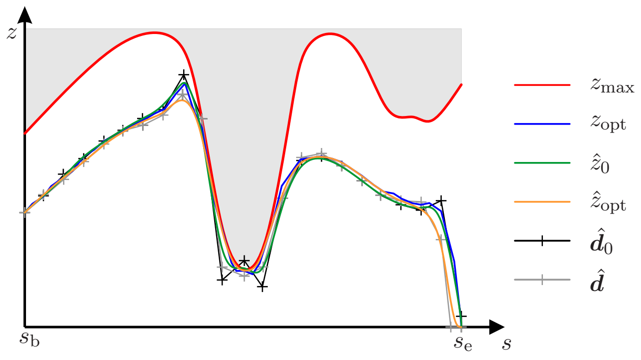

Figure 6. Spline approximation and optimal solution.

smooth trajectories. We approximate the optimal evolution of $z_{\mathrm{opt}}(s)$ derived by the dynamic programming algorithm, with a spline curve.

In a first step the trend of $z(s)$ is expressed as a spline

$$\hat{z}_0(s) = \sum_{l=1}^{\hat{n}_{\mathrm{D}}} N_l^d(s) \hat{d}_{0,l} \tag{25}$$

whose $\hat{n}_{\mathrm{D}}$ control points $\hat{\boldsymbol{d}}_0$ follow from a least squares approximation

$$\min_{\hat{\boldsymbol{d}}_0 \in \mathbb{R}^{\hat{n}_{\mathrm{D}}}} \sum_{i=n_{\mathrm{b}}}^{n_{\mathrm{e}}} \left\| \hat{z}_0(s_i) - z_{\mathrm{opt}}(s_i) \right\|^2 \tag{26}$$

minimizing the error between optimal and approximated trend at the discrete points $s = s_{\mathrm{b}}, s_{\mathrm{b}} + \Delta s \ldots s_i, \ldots s_{\mathrm{e}}$ of the DP algorithm. The spline $\hat{z}_0(s)$ is discretized to the originally demanded fine discretization $\Delta \hat{s} = \frac{s_{\mathrm{E}} - s_{\mathrm{B}}}{\hat{n}}$ with $\hat{n} > n$.

6.2 Ensuring consistency

6.2.1 Optimization problem

Since the approximation does not respect any restrictions, local violations are the consequence. Therefore an optimization of the control points, using $\hat{\boldsymbol{d}}_0$ as initial states, is performed in order to satisfy the restrictions at all discrete points. The optimization problem that has to be solved to fulfill the restrictions is

$$\min_{\hat{\boldsymbol{d}}\in\mathbb{R}^{\hat{n}_\mathrm{D}}} \sum_{i=\hat{n}_\mathrm{b}}^{\hat{n}_\mathrm{e}} \frac{\Delta\hat{s}}{\sqrt{\hat{z}_i}} \tag{27}$$

$$\text{s.t.}\ \left\|\boldsymbol{r}'_\mathrm{E}\right\|\sqrt{\hat{z}} \leq v_{\mathrm{E,max}} \tag{28}$$

$$\dot{\boldsymbol{q}}_{\min} \leq \boldsymbol{q}'\sqrt{\hat{z}} \leq \dot{\boldsymbol{q}}_{\max} \tag{29}$$

$$\ddot{\boldsymbol{q}}_{\min} \leq \boldsymbol{q}''\hat{z} + \frac{1}{2}\boldsymbol{q}'\hat{z}' \leq \ddot{\boldsymbol{q}}_{\max} \tag{30}$$

$$\boldsymbol{\tau}_{\min} \leq \boldsymbol{a}\hat{z}' + \boldsymbol{b}\hat{z} + \mathbf{c} + \boldsymbol{d}_\mathrm{v}\sqrt{\hat{z}} \leq \boldsymbol{\tau}_{\max} \tag{31}$$

with $\hat{n}_\mathrm{b} = s_\mathrm{b}/\Delta\hat{s}$ and $\hat{n}_\mathrm{e} = s_\mathrm{e}/\Delta\hat{s}$. Figure 6 shows the SDP solution z_opt and the approximation $\hat{z}_0(s)$ with the control points $\boldsymbol{d}_0$ used as initial states for the optimization. The smooth time optimal trajectory $\hat{z}_\mathrm{opt}(s)$ follows from the optimal control points $\hat{\boldsymbol{d}}$ provided by the optimization Eqs. (27)–(31).

6.2.2 Gradients

For a fine discretization, the restriction check at every discrete point is computationally expensive. A significant calculation time reduction can be achieved by providing analytical expressions for the gradients of the cost functional and restrictions with respect to the optimization variables. There are actually software tools to compute gradient and Hessians using automatic differentiation. Nevertheless, as the restrictions and the objective are given as analytical functions of the optimization variables, the gradients can easily be calculated analytically. By inserting the spline $\hat{z}_\mathrm{opt}(s) = \sum_{l=1}^{\hat{n}_\mathrm{D}} N_l^d(s)\hat{d}_l$ into the discrete cost functional

$$W = \sum_{i=\hat{n}_\mathrm{b}}^{\hat{n}_\mathrm{e}} \frac{\Delta\hat{s}}{\sqrt{\hat{z}_{\mathrm{opt},i}}}, \tag{32}$$

its gradient regarding the optimization variables $\hat{d}_l$ follows to

$$\frac{\partial W}{\partial \hat{d}_l} = \sum_{i=\hat{n}_\mathrm{b}}^{\hat{n}_\mathrm{e}} \frac{-\frac{1}{2}\Delta\hat{s}N_{l,i}^d}{\left(\hat{z}_{\mathrm{opt},i}\right)^{\frac{3}{2}}}. \tag{33}$$

Inserting the spline and its derivative with respect to the path parameter $\hat{z}'_\mathrm{opt}(s) = \sum_{l=1}^{\hat{n}_\mathrm{D}} N_l^{d'}(s)\hat{d}_l$ into the restrictions, their gradients regarding the optimization variables $\hat{d}_l$ follow to

$$\frac{\partial v_\mathrm{E}}{\partial \hat{d}_l} = \frac{1}{2}\left\|\boldsymbol{r}'_\mathrm{E}\right\|\frac{N_l^d}{\sqrt{\hat{z}_\mathrm{opt}}} \tag{34}$$

$$\frac{\partial \dot{\boldsymbol{q}}}{\partial \hat{d}_l} = \frac{1}{2}\boldsymbol{q}'\frac{N_l^d}{\sqrt{\hat{z}_\mathrm{opt}}} \tag{35}$$

$$\frac{\partial \ddot{\boldsymbol{q}}}{\partial \hat{d}_l} = \boldsymbol{q}''N_l^d + \frac{1}{2}\boldsymbol{q}'N_l^{d'} \tag{36}$$

$$\frac{\partial \boldsymbol{\tau}}{\partial \hat{d}_l} = \boldsymbol{a}N_l^{d'} + \boldsymbol{b}N_l^d + \frac{1}{2}\boldsymbol{d}_\mathrm{v}\frac{N_l^d}{\sqrt{\hat{z}_\mathrm{opt}}}. \tag{37}$$

6.2.3 Terminal conditions

To achieve a continuous trajectory a consideration of the terminal conditions for the spline optimization is necessary, as with the DP algorithm. For this purpose the first two control points have to be calculated separately. The first control point is defined by the terminal condition for $z(s_\mathrm{b})$

$$d_1 = z(s_\mathrm{b}), \tag{38}$$

while the second control point follows to

$$d_2 = z'(s_\mathrm{b}) - d_1\frac{N_1^{d'}(s_\mathrm{b})}{N_2^{d'}(s_\mathrm{b})} \tag{39}$$

for a transition gradient $z'(s_\mathrm{b})$ and the derivatives of the first two basis functions $N_1^{d'}$ and $N_2^{d'}$ with respect to the path parameter If also transition conditions $z(s_\mathrm{e}) = z_\mathrm{e}$ and $z'(s_\mathrm{e}) = z'_\mathrm{e}$ at the end of the path are required, the same procedure also works for the last and second last control point.

The definition of $z(s)$ as spline entails a further advantage namely an easy way to achieve a smooth start and stop. Jerky accelerations at the beginning and decelerations at the end of the path lead to end-effector vibrations which are problematical especial for elastic systems since they need a long time to settle to the desired endpoint. The definition of $z'(s_\mathrm{b}) = 0$ leads to a smooth start while $z'(s_\mathrm{e}) = 0$ provides a smooth stop.

7 Results

The experiments are realized with a Stäbli RX130L, a six-axis industrial robot. It is controlled by a Bernecker und Rainer system with PD controller and torque feed forward control for each joint. We use a spline curve of degree $d = 4$ in form of our institute logo (a robin), shown in Fig. 7, as geometric path. It is about $l \approx 7.8$ m long and is discretized into $\hat{n} = 2000$ pieces to represent even the fine contours. The end-effector orientation is held constant equal to the initial orientation, so that an observer directly faces the robots endpoint.

For the calculation of the initial solution for the spline optimization with the SDP algorithm the velocity discretization amounts $m = 300$, while the path is discretized into $n = 250$ in the range $s_\mathrm{B} = 0 \ldots s_\mathrm{E} = 1$. As horizons $s_\mathrm{o} = 0.2$ (50 segments) and $s_\mathrm{p} = 0.08$ (20 segments) are defined which lead to a subdivision of the total path into five segments.

Figure 8 shows the phase plane $s \times z$ with the liming curve $z_{\max}$, the optimal evolution z_opt provided by the SDP approach and the smooth optimal trend $\hat{z}_\mathrm{opt}$. This spline curve of degree $d = 4$ contains $\hat{n}_\mathrm{D} = 40$ control points for each path segment.

A comparison of the motor torques, discontinuous (provided by the SDP approach) and smooth, is given by Fig. 9. The basic behavior looks very similar, with the exception of

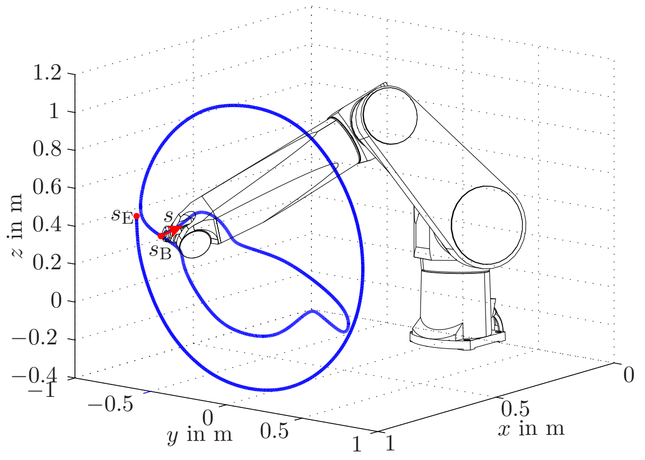

Figure 7. Geometric path – logo of the Institute of Robotics at the JKU Linz.

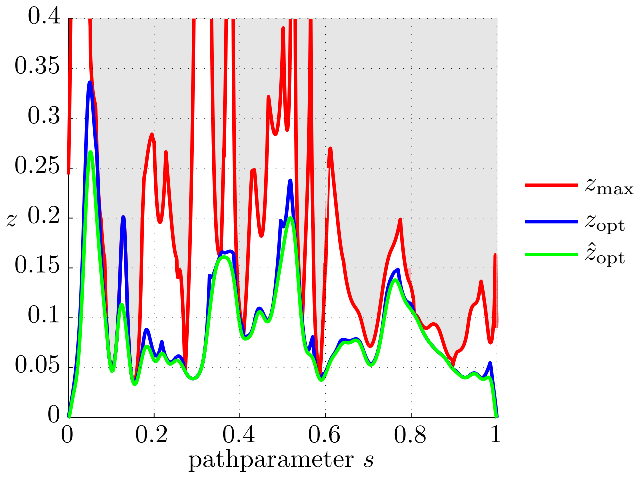

Figure 8. Limiting curve and optimal evolution of the SDP approach and for the smooth solution.

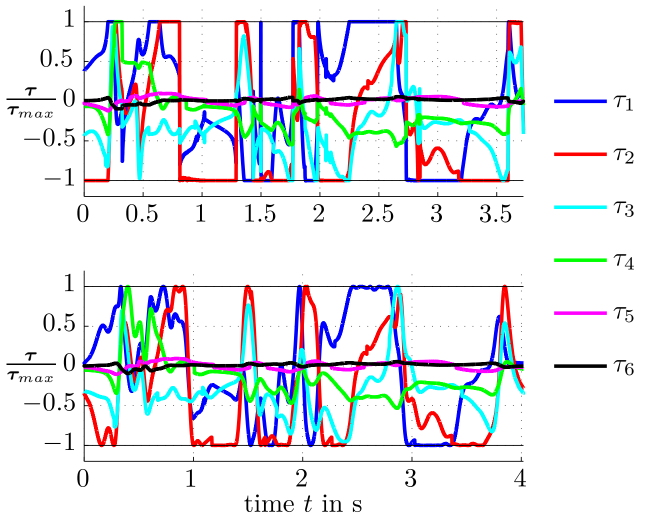

Figure 9. Motor torques, above: torque trends for the SDP solution, below: torque trends for the smooth solution.

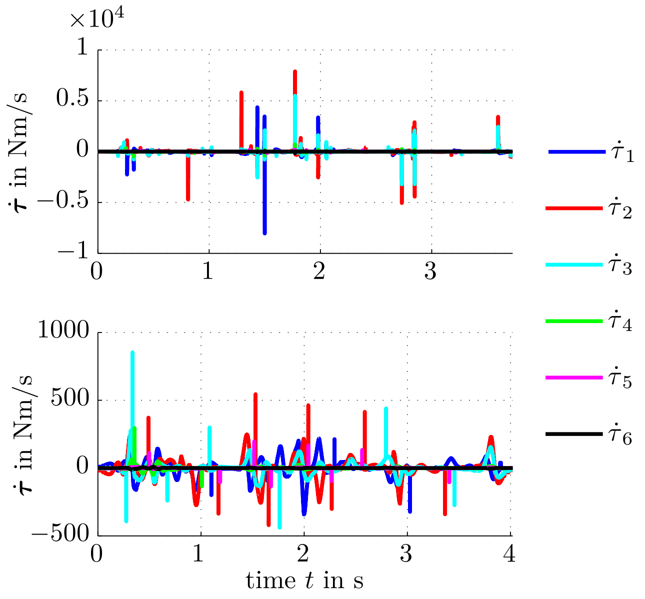

Figure 10. Motor torque rates, above: torque rate trends for the SDP solution, below: torque rate trends for the smooth solution.

a smooth start and stop, where the torques run into the static torques. The difference becomes clearer by looking at the torque rates in Fig. 10 and the joint jerks in Fig. 11. Figure 10 shows, that the torque rates of the discontinuous trajectory are approximately ten times higher then the torque rates of the smooth solution with $\hat{n}_D = 40$ control points for each section.

The smoothness of the trajectory contrasts with an increasing execution time from $t_E \approx 3.72$ s for the bang bang solution to $t_E \approx 4.02$ s for the smooth solution with $\hat{n}_D = 40$ control points for each section. This smooth trajectory was successfully implemented in simulations as well as on the real system, a video clip of the implementation is available on https://youtu.be/c5jllkLE4oU. A reduction of the number of control points to $\hat{n}_D = 25$ leads to a smoother solution, but increases the execution time to $t_E \approx 4.22$ s. Figure 11 shows the joint jerks for the different cases. One can see, that the reduction of the number of control points increases the execution time but reduces the joint jerks.

For validation purposes we implemented an optimization with hard jerk constraints $\dddot{\boldsymbol{q}}_{\max} = 3000\,\frac{\text{rad}}{\text{s}^3}$, considered in the spline optimization. This optimization converges only for a low number of control points up to $\hat{n}_D = 25$. The calculation time rises to $t_{\text{CPU}} \approx 65$ s, while the execution time $t_E \approx 4.13$ s is nearly the same as without jerk restrictions.

Finally we tried to calculate a time optimal trajectory for the whole path in one go, with $\hat{n}_D = 100$ control points. For that purpose we implemented two different approaches to derive an initial guess. A coarse discretized DP approach along

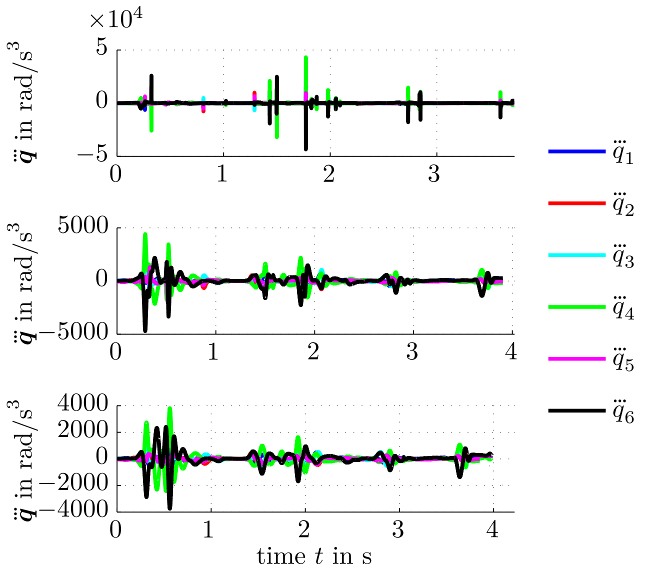

Figure 11. Joint jerks, above: trend of joint jerks for the SDP approach, middle: trend of joint jerks for a smooth solution with $\hat{n}_D = 40$ control points for each section, below: trend of joint jerks for a smooth solution with $\hat{n}_D = 25$ control points for each section.

the whole path, and inspired by Verscheure et al. (2009), a parabola trend of the control points. For that purpose the locations of the control points $\hat{\boldsymbol{d}}$ are defined as

$$\hat{d}_{k+1} = -\min(z_{\max}) \frac{4k\left(-\hat{n}_D + k + 1\right)}{\left(\hat{n}_D - 1\right)^2}, \tag{40}$$

with $k = 1, \ldots, \hat{n}_D$ and $\hat{d}_0 = 0$. The factor $\min(z_{\max})$ ensures, that no velocity restriction is violated by the initial guess. Both approaches require a coarse discretization of the path ($\hat{n} = 1250$) to achieve a convergence of the spline optimization. Despite the coarse discretization, the calculation times increase clearly. The slightly smaller execution times t_E can be attributed to the coarser discretization.

The results of the different methods are listed in Table 1. In this table, SO is the abbreviation for spline optimization and g indicates the usage of analytical gradients. Jerk suggests the consideration of hard joint jerk restrictions in the spline optimization. The last two lines of Table 1 show the results for the optimization in one go with the DP and parabola approach for the initial guess.

The spline optimization was done with the active-set algorithm of the Matlab optimizer fmincon. With the time t_{CPU} we indicate the computation time on a standard PC with a CPU clock of 2.83 GHz. The results in Table 1, clearly indicate the improvements of the presented approach compared to an approach with jerk constraints. Nevertheless, the calculation times are significantly higher compared to the execution times. An improvement could for example be achieved by implementing the optimization not in MATLAB, but in a C-based optimization toolbox.

Table 1. Trajectory execution times t_E and calculation times t_{CPU} for the different methods.

Method	t_{CPU}	t_E	$\hat{n}_D$	$\hat{n}$
	(s)			
SDP	5	3.72	–	2000
SDP and SO	120	4.02	40	2000
SDP and SO and g	25	4.02	40	2000
SDP and SO and g	20	4.22	25	2000
SDP and SO and g and jerk	65	4.13	25	2000
DP and SO and g	58	4.01	100	1250
parabola and SO and g	101	4.01	100	1250

8 Conclusions

This paper presents an approach to derive smooth time optimal trajectories for arbitrary long geometric paths. The main idea is to split the path into sections, to calculate optimal trajectories using terminal conditions, and to assemble the solutions for the individual segments. To achieve smooth trajectories in acceptable calculation times we propose a spline optimization in the phase space. The problem of convenient initial states for the optimization is solved with a DP approach in which terminal conditions can be considered in an easy way. With the spline optimization it is also simple to achieve a smooth start and stop of the robot. The presented approach may be interesting for robot manufacturers which already have algorithms for the path tracking problem and want to extend them to achieve smooth trajectories. Experiments to show the proper functionality of the method are realized on a six-axis industrial robot. An extension of the algorithm to consider jerk and torque rate restrictions in the optimization will be part of future work.

Acknowledgements. This work has been supported by the Austrian COMET-K2 program of the Linz Center of Mechatronics (LCM), and was funded by the Austrian federal government and the federal state of Upper Austria.

Edited by: M. Hofbaur
Reviewed by: two anonymous referees

References

Ardeshiri, T., Norlöf, M., Löfberg, J., and Hansson, A.: Convex optimization approach for time optimal path tracking of robots with speed dependent constraints, in: Proceedings of the 18th IFAC Congress, Milano, Italy, 28 August–2 September 2011, 14648–14653, 2011.

Bellman, R. E. and Dreyfus, S. E.: Applied Dynamic Programming, Princeton University Press, Princeton, New Jersey, USA, 1962.

Bobrow, J. E.: Optimal Robot Path Planning Using the Minimum-Time Criterion, IEEE Journal of Robotics and Automation, 4, 443–450, 1988.

Bobrow, J. E., Dubowsky, S., and Gibson, J. S.: Time-optimal control of robotic manipulators along specified paths, Int. J. Robot. Res., 4, 3–17, 1985.

Bremer, H.: Elastic Multibody Dynamics: A Direct Ritz Approach, Springer Netherlands, Dordrecht, Netherlands, 2008.

Constantinescu, D. and Croft, E. A.: Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, J. Robot. Syst., 17, 233–249, 2000.

De Boor, C.: A practical guide to splines, Springer Verlag New York, New York, USA, 1978.

Debrouwere, F., Van Loock, W., Pipeleers, G., Tran Dinh, Q., Diehl, M., De Schutter, J., and Severs, J.: Time-Optimal Path Following for Robots with Trajectory Jerk Constraints using Sequential Quadratic Programming, in: Proceedings of IEEE ICRA 2013, 6–10 May 2013, Karlsruhe, Germany, 1916–1921, 2013.

Gattringer, H., Oberherber, M., and Springer, K.: Extending continuous path trajectories to point-to-point trajectories by varying intermediate points, International Journal of Mechanics and Control, 15, 35–43, 2014.

Geu Flores, F. and Kecskemethy, A.: Time-Optimal Path Planning Along Specified Trajectories, in: Multibody System Dynamics, Robotics and Control, edited by: Gattringer, H. and Gerstmayr, J., Springer Vienna, Wien, Austria, 1–16, 2012.

Johanni, R.: Optimale Bahnplanung bei Industrierobotern, VDI Verlag, Düsseldorf, Germany, 1988.

Müller, B., Deutscher, J., and Grodde, S.: Continuous Curvature Trajectory Design and Feedforward Control of Parking a Car, IEEE T. Contr. Syst. T., 15, 541–553, 2007.

Neubauer, M., Gattringer, H., and Bremer, H.: A persistent method for parameter identification of a seven-axes manipulator, Robotica, 1–14, 2014.

Nocedal, J. and Wright, S. J.: Numerical Optimization, 2nd Edn., Springer-Verlag New York, New York, USA, 2006.

Oberherber, M., Gattriger, H., and Springer, K.: Time Optimal Path Planning for Industrial Robots: A Dynamic Programming Approach Considering Torque Derivative and Jerk Constraints, in: Proceedings in Applied Mathematics and Mechanics, 14, 75–76, 2014.

Pfeiffer, F. and Johanni, R.: A Concept for Manipulator Trajectory Planning, IEEE Journal of Robotics and Automation, 3, 115–123, 1987.

Piegl, L. A. and Tiller, W.: The NURBS Book, 2, Springer-Verlag Berlin Heidelberg, Berlin, Germany, 1995.

Quang-Cuong, P.: A General, Fast, and Robust Implementation of the Time-Optimal Path Parameterization Algorithm, IEEE T. Robot., 30, 1533–1540, 2014.

Shin, K. and McKay, N.: Minimum-Time Control of Robotic Manipulators with Geometric Path Constraints, IEEE T. Automat. Control, 30, 531–541, 1985.

Swevers, J., Ganseman, C., De Schutter, J., and Van Brussel, H.: Experimental robot identification using optimised periodic trajectories, Mech. Syst. Signal Pr., 10, 561–577, 1996.

Verscheure, D., Diehl, M., De Schutter, J., and Swevers, J.: Recursive Log-barrier Method for On-line Timeoptimal Robot Path Tracking, in: Proceedings of 2009 American Control Conference, Hyatt Regency Riverfront, 10–12 June 2009, St. Louis, MO, USA, 4134–4140, 2009.