Int J Adv Manuf Technol (2009) 45:1007–1016
DOI 10.1007/s00170-009-2032-9

ORIGINAL ARTICLE

Calculating a near time-optimal jerk-constrained trajectory along a specified smooth path

Jan Mattmüller · Damian Gisler

Received: 17 November 2008 / Accepted: 24 March 2009 / Published online: 19 April 2009

Abstract This article presents a near time-optimal and jerk-constrained trajectory planner. The presented work is an extension to the "proximate time-optimal algorithm" (Pardo-Castellote and Cannon, IEEE Int Conf Robot Autom 2:1539–1546, 1996), which is used to determine smooth and near time-optimal path-constrained trajectories, to problems where not only the velocity and the acceleration but also the jerk are explicitly constrained. It is shown that the constraint on the jerk translates into limits for the curvature of the phase-space velocity. As high-speed motion systems become more and more accurate, the trend goes clearly towards jerk-constrained trajectory calculation to avoid large deviations from the planned trajectory during the complete move. The proposed algorithm is nonperturbative and its calculation time is linear with respect to the length of the path.

Keywords Path planner · Motion control · Jerk-constrained · Trajectory · Time-optimal

1 Introduction

A modern micromanipulator is able to perform complex moves over a distance from start to end point of several millimeters with a path following accuracy in the micrometer range. Dozens of motion sequences in an arbitrary dimensional space, defined by the motion axes, are performed per second. Start and end position of those are often determined by camera systems only milliseconds before the move has to start.

The fact that the placement accuracy of a modern micromanipulator has to be in the submicron range has, of course, implications on the control system of the manipulator. One possibility to decrease the deviation of the manipulator from the planned path during its moves is to consider the motor-constraints for the calculation of the trajectory. Even though this may sound trivial, its implementation is less straightforward than expected.

The goal of this work was to develop and implement an algorithm that is able to compute a near time-optimal trajectory for an almost arbitrary smooth path. This paper presents an extension of the "proximate time-optimal algorithm," which was presented in [6]. One of the advantages of this algorithm is its computational efficiency, which is based on the fact that the method is nonperturbative. The proximate time-optimal algorithm had already been successfully applied to problems with constrained velocity and acceleration, but not yet to systems that are also explicitly subjected to a limited jerk. This paper describes how this modification can be incorporated in the mentioned algorithm and shows that it leads to excellent results.

2 Problem statement

Given a specified smooth geometric path and a manipulator moving along this path, the problem consists in finding a trajectory that minimizes the necessary time

J. Mattmüller (✉) · D. Gisler
Oerlikon Assembly Equipment AG,
Hinterbergstrasse 32, 6330 Cham, Switzerland
e-mail: janma@edpnet.be

Present Address:
J. Mattmüller
ICOS VISION SYSTEMS NV, Research Park Haasrode
Zone 1, Esperantolaan 8, 3001 Leuven, Belgium

along the path without violating the given constraints of the motor that drives the manipulator. This is a classical robotic trajectory/motion-planning problem, namely, a time parameterization of a geometric path (see Section 3). Assuming the path to be given as $\mathbf{x_p}(s)$, whereas $\mathbf{x_p} = (x, y, z)$ denotes a vector in the three-dimensional space with parameterization $s \in [s_0, s_f]$, the goal is now to find a function $s(t)$ that minimizes the time

$$T = \int_{s_0}^{s_f} \frac{1}{\dot{s}(s)} ds \tag{1}$$

with $\dot{s} = \frac{ds}{dt}$ denoting the time derivative of s (see [1, 8]).

A lot of work has already been done in this field [2, 6, 10], and multiple solutions exist for the case with constrained velocity v and acceleration a.

$$\mathbf{v}_{\min} \leq \frac{d\mathbf{x_p}}{dt}(t) \leq \mathbf{v}_{\max} \quad \forall t \tag{2}$$

$$\mathbf{a}_{\min} \leq \frac{d^2\mathbf{x_p}}{dt^2}(t) \leq \mathbf{a}_{\max} \quad \forall t \tag{3}$$

In this work, a further constraint is considered, which increases the complexity of the solution to the problem noticeably: A specific maximum and minimum jerk must not be violated.

$$\mathbf{j}_{\min} \leq \frac{d^3\mathbf{x_p}}{dt^3}(t) \leq \mathbf{j}_{\max} \quad \forall t \tag{4}$$

For example (see also Section 5):

$$\mathbf{v}_{\max} = (0.8, 0.8, 0.8)\ m/s = -\mathbf{v}_{\min} \tag{5}$$

$$\mathbf{a}_{\max} = (100, 100, 400)\ m/s^2 = -\mathbf{a}_{\min} \tag{6}$$

$$\mathbf{j}_{\max} = (80{,}000, 80{,}000, 40{,}000)\ m/s^3 = -\mathbf{j}_{\min} \tag{7}$$

The reasons and benefits of applying a finite jerk as an additional constraint are manifold:

- To minimize the deviation from the planned path
- To minimize the excitation of vibrations in general
- To ensure that the motor is able to provide the requested current fast enough

The constraint in jerk is intrinsic to many actuator systems, especially when they are based on voice-coil motors. As the jerk is the change of acceleration per time, it corresponds to the change of current per time if disturbances and internal losses are negligible. It is evident that the current cannot reach a certain level immediately. Therefore, it is important to take the constraint on the jerk into account especially when the maximum tolerated deviation from the planned path is very small.

In addition to the higher complexity due to the jerk-constraint, the algorithm should be rather efficient in order to be useful for a fully programmable and vision-guided micromanipulator. In the best case, the computing would be real-time, which means, in this case, that a smooth path of a few millimeters should be computable in a few milliseconds.

3 Existing solutions

As mentioned above, the problem is a classical robotic problem, and many solutions have been proposed to solve it. Here, only a very brief overview over some publications that dealt with the decoupled approach is given. In the decoupled approach, the geometric path is assumed to be fixed. Only the velocity with which the path is followed is optimized but not the path itself. Dubowsky et al. [2] and Shin and McKay [7] showed how the time-optimal problem can be solved. Shin and McKay [8] and Constantinescu and Croft [1] then extended the solution to problems with jerk constraints, but in contrast to this work, they used a perturbative approach and the flexible tolerance method, respectively. All solutions to the time-optimal problem turned out to require massive computational effort. Slotine and Young [9] and Ecker et al. [3], among others, worked on the efficiency of the proposed algorithms. However, the needed computing time and power are not in favor of this approach. For CNC machining applications, [4] and [5] use somewhat similar approaches in which the feedrate is constructed out of a set of several different basic segments. Nam and Yang [5] used a recursive approach, which is not attractive for our application. Lin et al. [4] worked on a problem that was very similar to the one discussed here. The major shortcoming of their approach is the fact that the feedrate is, to a certain extent, limited by the shape of the basic profiles, while our algorithm is more general and, thus, closer to the constraints defined by the machine and the trajectory. This might not be critical for machining, where the feedrate is typically 50 mm/s, but it becomes a serious penalty for high-speed motion systems where accelerations are 100 times higher than in machining. In addition, it seems difficult to adapt [4] to include different acceleration and jerk constraints for the different axes, which is a requirement for high-speed motion systems.

More promising as a starting point for our purpose is [6], which used the so-called proximate time-optimal algorithm, which is shown to consume very little computing time to solve the problem. This approach will be adapted in order to consider constraints not only on the velocity and the acceleration but also on the jerk.

4 Algorithm

4.1 The phase space $(s, \dot{s})$

The proximate time-optimal approach and most of the other proposed methods transform the problem from the three-dimensional "normal space" to the phase space. There are two major advantages in doing so:

1. Not only the starting point $(s_0, \dot{s}_0)$, but also the end point of the move $(s_f, \dot{s}_f)$, and thus, the boundaries for the integration, are fixed and known in advance. Alternatively, the end point of the move in the normal space $(\mathbf{x}(t_f), t_f)$ is not known in advance because t_f is precisely a result of the algorithm.
2. Instead of having to find an optimum in the four-dimensional $(\mathbf{x}, t)$ space, one can work in the two-dimensional phase space.

The phase space $(s, \dot{s})$ is, thus, a very useful representation. The parameter s describes the motion along the path. It starts at s_0, normally set equal to 0, and ends at s_f, which corresponds to the length of the total path. The parameter $\dot{s}$ describes the velocity along the path. The goal is therefore to find a curve in $(s, \dot{s})$ starting typically at $(0, 0)$ and ending at $(s_f, \dot{s}_f)$, which, somewhat simplified, follows the highest possible values for $\dot{s}$ while respecting the given constraints.

Because the near time-optimal phase space velocity curve will be constructed in $(s, \dot{s})$, the gradient m in the phase space is needed. It is defined as long as the manipulator is in motion $(\dot{s} \neq 0)$:

$$m(s, \dot{s}) := \frac{d\dot{s}}{ds} = \frac{d}{ds}\frac{ds}{dt} = \frac{dt}{ds} \cdot \frac{d}{dt}\frac{ds}{dt} = \frac{\ddot{s}}{\dot{s}} \tag{8}$$

It can be interpreted as the acceleration along the curve divided by the velocity along the curve.

4.2 Constraints

As mentioned in Section 1, the constraints for velocity $\mathbf{v}$, acceleration $\mathbf{a}$, and jerk $\mathbf{j}$ are as follows:

$$\mathbf{v}_{\min} \leq \frac{d\mathbf{x_p}}{dt} \leq \mathbf{v}_{\max} \tag{9}$$

$$\mathbf{a}_{\min} \leq \frac{d^2\mathbf{x_p}}{dt^2} \leq \mathbf{a}_{\max} \tag{10}$$

$$\mathbf{j}_{\min} \leq \frac{d^3\mathbf{x_p}}{dt^3} \leq \mathbf{j}_{\max} \tag{11}$$

In order to be useful in the phase space, those constraints need to be transformed.

4.2.1 The velocity

The above inequality (Eq. 9) is not very useful because $\mathbf{x_p}(t)$ is unknown in the real system and it is, in fact, the solution to the whole problem. However, the velocity can be transformed as follows:

$$\frac{d\mathbf{x_p}}{dt} = \frac{d\mathbf{x_p}}{ds} \cdot \frac{ds}{dt} = \mathbf{x_p}' \cdot \dot{s} \tag{12}$$

This, on the other hand, is useful, because $\mathbf{x_p}'$ can be easily calculated a priori: $\mathbf{x_p}(s)$ is part of the definition of the problem. This, together with Eq. 9, leads to conditions for $\dot{s}$ (one per axis and side of Eq. 9), which, in case of a three-dimensional space, leads to:

$$\dot{s}_{\text{lim,negvel},1}(s), \dot{s}_{\text{lim,posvel},1}(s), \ldots$$
$$\dot{s}_{\text{lim,negvel},2}(s), \dot{s}_{\text{lim,posvel},2}(s), \ldots$$
$$\dot{s}_{\text{lim,negvel},3}(s), \dot{s}_{\text{lim,posvel},3}(s)$$

with

$$\dot{s}_{\text{lim,negvel},1}(s) = \frac{v_{1,\min}}{x_1'} \tag{13}$$

and the other constraints, respectively.

At each position s, those constraints form an upper and lower limit for the phase space velocity $\dot{s}$

$$\dot{s}_{\max}(s) = \text{Min}\left(\dot{s}_{\text{lim,posvel},1}(s), \ldots, \dot{s}_{\text{lim,posvel},3}(s)\right) \tag{14}$$

$$\dot{s}_{\min}(s) = \text{Max}\left(\dot{s}_{\text{lim,negvel},1}(s), \ldots, \dot{s}_{\text{lim,negvel},3}(s)\right) \tag{15}$$

4.2.2 The acceleration

The acceleration along the path in the position–time–space $\frac{d^2\mathbf{x_p}}{dt^2}$ will be transformed to the parameter space in analogy to the velocity:

$$\frac{d^2\mathbf{x_p}}{dt^2} = \frac{d^2\mathbf{x_p}}{ds^2} \cdot \left(\frac{ds}{dt}\right)^2 + \frac{d\mathbf{x_p}}{ds} \cdot \frac{d^2s}{dt^2} \equiv \mathbf{x_p}'' \cdot \dot{s}^2 + \mathbf{x_p}' \cdot \ddot{s} \tag{16}$$

This, again, is of use because $\mathbf{x_p}''$ can also easily be calculated a priori. In order to visualize the constraints it is helpful to have a look at it in the $(\dot{s}, \ddot{s})$ space. In this space, for a certain value of s, Eq. 10 corresponds to a pair of parabolas per dimension, which confine the allowed values of $(\dot{s}, \ddot{s})$. The borders of this area define

$$\ddot{s}_{\text{min,acc},1}(s, \dot{s}), \ddot{s}_{\text{max,acc},1}(s, \dot{s}), \ldots$$
$$\ddot{s}_{\text{min,acc},2}(s, \dot{s}), \ddot{s}_{\text{max,acc},2}(s, \dot{s}), \ldots$$
$$\ddot{s}_{\text{min,acc},3}(s, \dot{s}), \ddot{s}_{\text{max,acc},3}(s, \dot{s}),$$

with

$$\ddot{s}_{\min,\mathrm{acc},1}(s,\dot{s}) = \frac{a_{1,\min} - x_1'' \cdot \dot{s}^2}{x_1'} \tag{17}$$

and the other constraints, respectively.

At each position $(s, \dot{s})$, those constraints form an upper and a lower boundary for the phase space acceleration $\ddot{s}$

$$\ddot{s}_{\max}(s,\dot{s}) = \mathrm{Min}(\ddot{s}_{\max,\mathrm{acc},1}(s,\dot{s}), \ldots, \ddot{s}_{\max,\mathrm{acc},3}(s,\dot{s})) \tag{18}$$

$$\ddot{s}_{\min}(s,\dot{s}) = \mathrm{Max}(\ddot{s}_{\min,\mathrm{acc},1}(s,\dot{s}), \ldots, \ddot{s}_{\min,\mathrm{acc},3}(s,\dot{s})) \tag{19}$$

4.2.3 The jerk

Finally, also the jerk along the path $\frac{d^3\mathbf{x_p}}{dt^3}$ has to be transformed:

$$\begin{aligned}\frac{d^3\mathbf{x_p}}{dt^3} &= \frac{d^3\mathbf{x_p}}{ds^3}\cdot\left(\frac{ds}{dt}\right)^3 + 3\frac{d^2\mathbf{x_p}}{ds^2}\cdot\frac{ds}{dt}\cdot\frac{d^2s}{dt^2} + \frac{d\mathbf{x_p}}{ds}\cdot\frac{d^3s}{dt^3}\\ &\equiv \mathbf{x_p}'''\cdot\dot{s}^3 + 3\mathbf{x_p}''\cdot\dot{s}\cdot\ddot{s} + \mathbf{x_p}'\cdot\dddot{s}\end{aligned} \tag{20}$$

It can be shown that

$$\dddot{s} = \dot{s}\cdot\left(\frac{d\dot{s}}{ds}\right)^2 + \dot{s}^2\cdot\frac{d^2\dot{s}}{ds^2} \tag{21}$$

As long as $\dot{s} \neq 0$

$$\frac{\dddot{s}}{\dot{s}^2} = \frac{m^2}{\dot{s}} + \frac{d^2\dot{s}}{ds^2} \tag{22}$$

This leads with Eqs. 11 and 20 to

$$\frac{\mathbf{j}_{\min}}{\dot{s}^2} \leq \mathbf{x_p}'''\cdot\dot{s} + 3\mathbf{x_p}''\cdot\frac{\ddot{s}}{\dot{s}} + \mathbf{x_p}'\cdot\frac{\dddot{s}}{\dot{s}^2} \leq \frac{\mathbf{j}_{\max}}{\dot{s}^2} \tag{23}$$

$$\begin{aligned}\frac{\mathbf{j}_{\min}}{\dot{s}^2} &\leq \mathbf{x_p}(s)'''\cdot\dot{s} + 3\mathbf{x_p}(s)''\cdot m(s,\dot{s})\\ &\quad + \mathbf{x_p}(s)'\cdot\frac{m(s,\dot{s})^2}{\dot{s}} + \mathbf{x_p}(s)'\cdot\frac{d^2\dot{s}}{ds^2} \leq \frac{\mathbf{j}_{\max}}{\dot{s}^2}\end{aligned} \tag{24}$$

This is the main equation of this work and can now be used to construct the time-optimal path without violating a jerk constraint. It was demonstrated that the velocity constraint results in an upper and lower limit for the phase space velocity and the acceleration constraint results in a maximum and minimum phase space velocity gradient. Finally, Eq. 24 shows that the jerk constraint leads to a maximum and minimum curvature $\frac{d^2\dot{s}}{ds^2}$ of the phase space velocity. At each point in the phase space, this equation allows us to transform the jerk constraints to calculate the maximum and minimum allowed curvature.

4.3 Step-by-step calculation of $\dot{s}(s)$

In order to construct the time-optimal phase space velocity, the trajectory is discretized along its path. Moving along the trajectory, the above elaborated formulas and constraints are applied at certain discrete positions. The increment $\Delta s = s_n - s_{n-1}$ depends on the available time for calculation, the processing power, the cycle-time of the motion controller, and the tolerable deviation from the trajectory. The discrete equations below hold only for the limes $\Delta s \to 0$. The idea is to start at $(s, \dot{s}) = (s_0, \dot{s}_0)$ and to move step-by-step to s_1, s_2, and so on, determining at each step the maximum allowed $\dot{s}_1 \equiv \dot{s}(s_1), \dot{s}_2 \equiv \dot{s}(s_2), \ldots$ according to the criteria elaborated in Section 2. As shown, the velocity constraints introduce an upper boundary for $\dot{s}$. The acceleration constraints were transformed into a maximum gradient for each step:

$$\begin{aligned}\dot{s}_n \equiv \dot{s}(s_n) &= \lim_{\Delta s\to 0}\dot{s}_{n-1} + (s_n - s_{n-1})\cdot m(s_{n-1},\dot{s}_{n-1})\\ &= \lim_{\Delta s\to 0}\dot{s}_{n-1} + (s_n - s_{n-1})\cdot m_{n-1}\end{aligned} \tag{25}$$

$$m_{\max,\mathrm{acc}}(s,\dot{s}) = \frac{\ddot{s}_{\max}(s,\dot{s})}{\dot{s}(s)} \tag{26}$$

Additionally, the jerk constraints lead to upper and lower boundaries for $\frac{d^2\dot{s}}{ds^2}$. This term describes the curvature of $\dot{s}(s)$. It is, thus, helpful to state that the curvature at the position s_n is

$$\frac{d^2\dot{s}}{ds^2} = \lim_{\Delta s\to 0}\frac{\dot{s}_{n+1} - 2\dot{s}_n + \dot{s}_{n-1}}{\Delta s^2} \tag{27}$$

Inserting Eqs. 26 and 27 into Eq. 24 leads readily to conditions for

$$\begin{aligned}&\dot{s}_{n+1,\max}\\ &\quad = f(\mathbf{j}_{\min}, \mathbf{j}_{\max}, \dot{s}_n, \dot{s}_{n-1}, ds, m(s_n,\dot{s}_n), \mathbf{x_p}', \mathbf{x_p}'', \mathbf{x_p}''')\end{aligned} \tag{28}$$

where $\mathbf{x_p}', \mathbf{x_p}'', \mathbf{x_p}'''$ each depend on s_n.

This last equation contains all constraints. The constraints on the jerk are transformed into the curvature $\mathbf{x_p}''(s_n)$ and $\mathbf{j}_{\min}, \mathbf{j}_{\max}$. The constraints on the acceleration manifest themselves in $m(s_n, \dot{s}_n)$ and $\mathbf{x_p}''(s_n)$. For each step, $\dot{s}_{n+1,\max}$ is chosen so that none of the conditions are violated.

The same procedure can be applied at the same time in the reverse direction, starting at $(s, \dot{s}) = (s_f, \dot{s}_f)$ and moving step-by-step to $s_{f-1}, s_{f-2}, \ldots$. In this case, be aware that the maximum allowed phase space-velocity $\dot{s}(s)$ is still a boundary that must not be violated, but for the step-by-step calculation, the minimum allowed gradient and curvature have to be used in order to construct the time-optimal phase space velocity curve.

Depending on the path and the constraints, this method will lead to one of three possible situations:

- Meet point
- Split point
- Forbidden point

which will be explained in the following sections. Meet and split points also occur without jerk constraints and have already been treated in [6]. However, the forbidden points can only occur in problems where the acceleration and the jerk are constrained, and their treatment is, thus, a result of this work. The flowchart in Fig. 1 summarizes the described process.

4.4 Meet points

Assuming that the actuator has to move only a relatively short distance, the phase space velocity will not come close to its limits. The phase space acceleration ($\ddot{s}$) and jerk ($\dddot{s}$) and, thus, the gradient and the curvature in the phase space are the only constraints. In this case, the phase space velocity can be constructed without problems. As it is being constructed simultaneously from the beginning and the end, the two parts will intersect at some point. The resulting phase space velocity reflects at that moment the fastest possible way to move along the given path without violating the constraints. The only point where the jerk constraints are violated is the point where the two parts meet, the so-called meet point. At that point, the gradient changes abruptly from its allowed maximum value at that point to its minimum value. Therefore, the jerk, which is correlated to the curvature, is violated at that point (see Fig. 2).

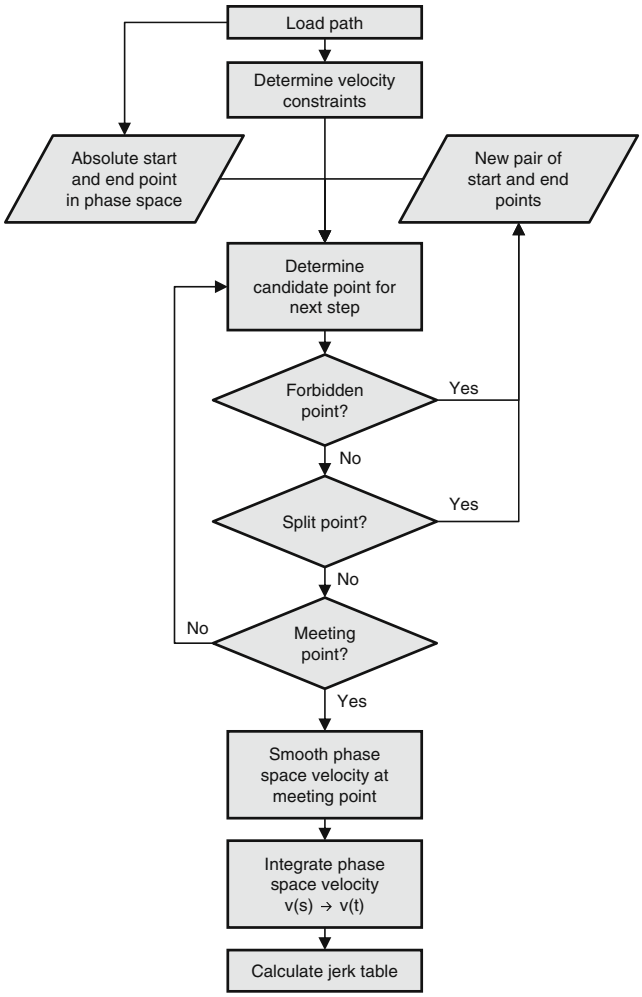

Fig. 1 Flowchart of the algorithm

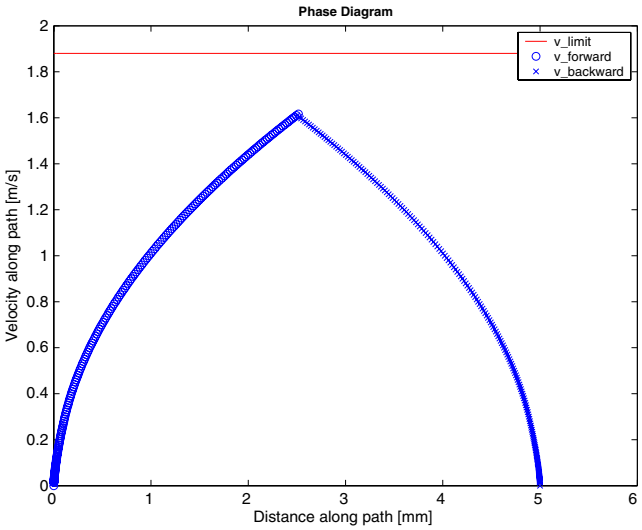

Fig. 2 Typical meetpoint for a simple straight path

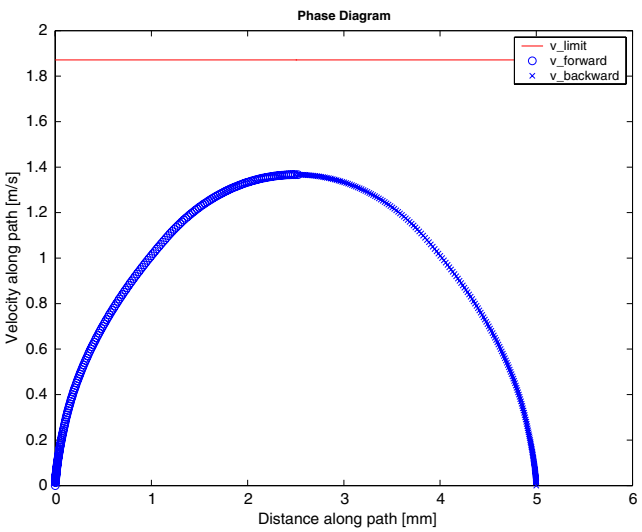

Fig. 3 Typical phase space curve without jerk violation at the meetpoint

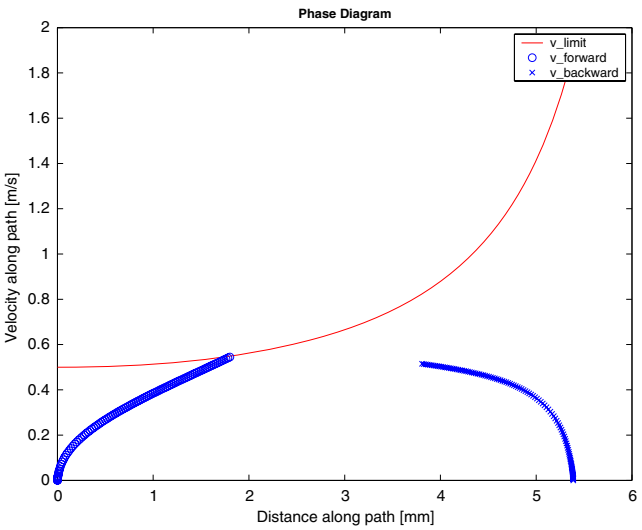

Fig. 4 Example for a case where the constructed phase space velocity intersects with the maximum phase space velocity

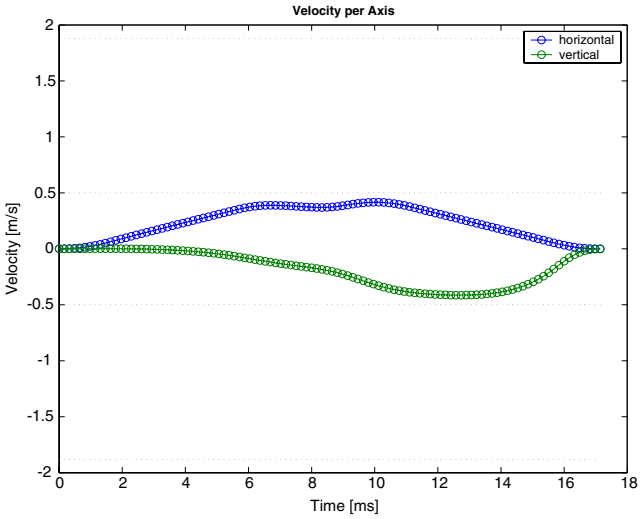

Fig. 6 It can be seen that the constraint on the maximum velocity $v_{\max} = 0.5$ m/s is not violated

In order to solve that problem, the following procedure is proposed: Go a few steps back on both sides and reconstruct the phase space velocity again, but this time with the minimum allowed gradient and curvature for the forward part and maximum allowed gradient and curvature for the backward part. If the phase space jerk is still violated at the meet point, go further back and restart again. Do that until the left and the right sides have a common tangent. This tangent serves as a connection between the two parts (see Fig. 3).

4.5 Split points

There are different cases where it is helpful to split up the construction of the phase space velocity. If the maximum phase space velocity has a local minimum somewhere, it can become impossible for the two parts to intersect because they would both first intersect with the limit of the phase space velocity.

In such a case, an additional starting point for the forward and backward construction of the phase space velocity is introduced. The idea is that the final solution of the phase space velocity will need to go below

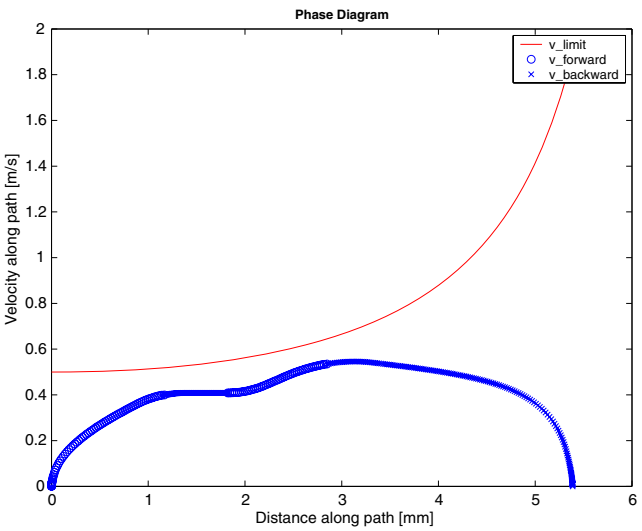

Fig. 5 A split point has been introduced, which splits the original problem into two separate problems that were solved independently

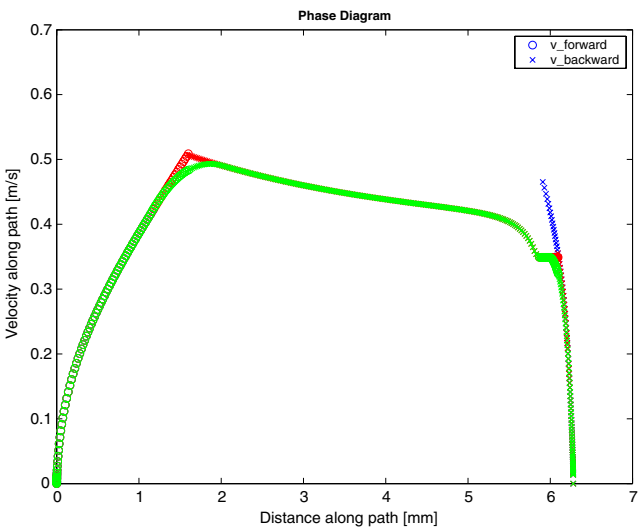

Fig. 7 Example for a case where the constructed phase space velocity reached a forbidden point (on the *right side* at around $s = 5.9$ mm, at the end of the blue curve)

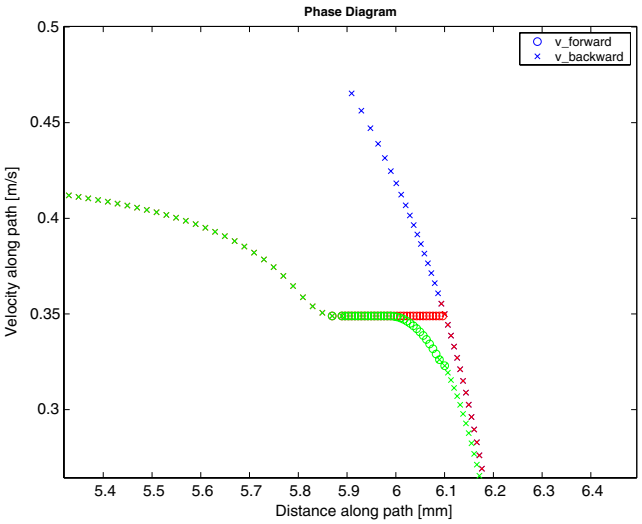

Fig. 8 If a forbidden point is reached, an artificial split point is introduced, which splits the problem into a left and a right part. *Blue*: original construction of phase space velocity, *red*: after introduction of a split-point, *green*: final phase space velocity after having taken into account the jerk constraint

the minimum of the limitation. Therefore, a point $(s_{\text{split}}, \dot{s}_{\text{split}})$ can be defined directly below the minimum and carry on the construction from there. To the left, the backward construction is carried on toward s_0, and to the right, the forward construction is carried on toward s_f. In this way, the split point is transformed into a start/end point and the original problem is separated into two new problems, which can be treated independently of each other.

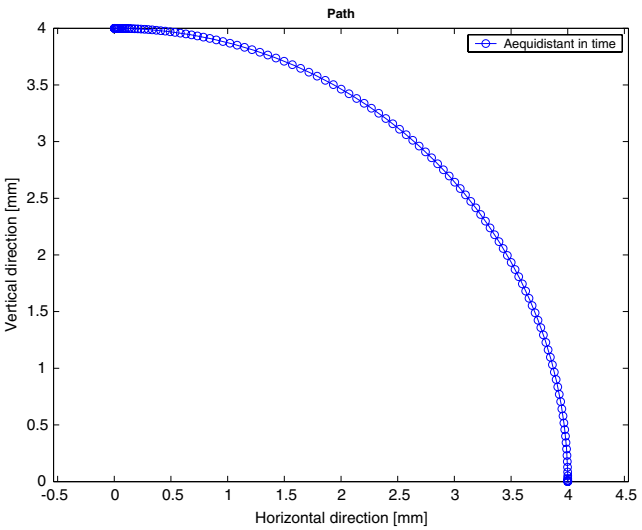

Fig. 9 This is the corresponding trajectory with the final parameterization (constant time-steps)

Table 1 Motor-constraints which were used for the example

	Horizontal axis	Vertical axis
Maximum velocity [m/s]	0.8	0.8
Maximum acceleration [m/s^2]	100	400
Maximum jerk [m/s^3]	80,000	400,000

Even if the limit of the phase space velocity does not have a local minimum, it is possible that the constructed phase space velocity intersects with the limit of the phase space velocity, which makes an introduction of a split point useful also. The split point transforms the original problem with potential violation of the phase space velocity into two parts. One of them definitely has no violation, while the other one remains to be solved (see Figs. 4, 5, and 6).

4.6 Forbidden points

There is one additional case that needs special attention and treatment. It is possible that the combination of acceleration and jerk constraints leads to a situation in which the next step of the construction is not possible anymore. It is possible that there is a position s, denoted s_{viol}, where $\dot{s}_{\min,\text{jerk}}(s) > \dot{s}_{\max,\text{acc}}(s)$ or $\dot{s}_{\max,\text{jerk}}(s) < \dot{s}_{\min,\text{acc}}(s)$. In this case, the phase space velocity has to be artificially lowered to ensure that a solution can be found. Again, at the position s_{viol}, a new start/end point is defined that lies a certain distance below the originally constructed phase space velocity. Thus, the forbidden point is transformed into a starting point and the original problem is once more split into two separate problems, which can be treated independently of each other (see Figs. 7, 8, and 9).

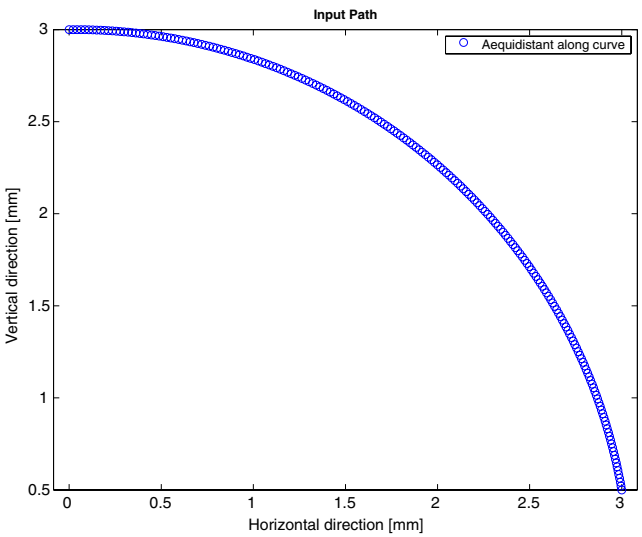

Fig. 10 Input path for the calculation $\mathbf{x}(\mathbf{s})$

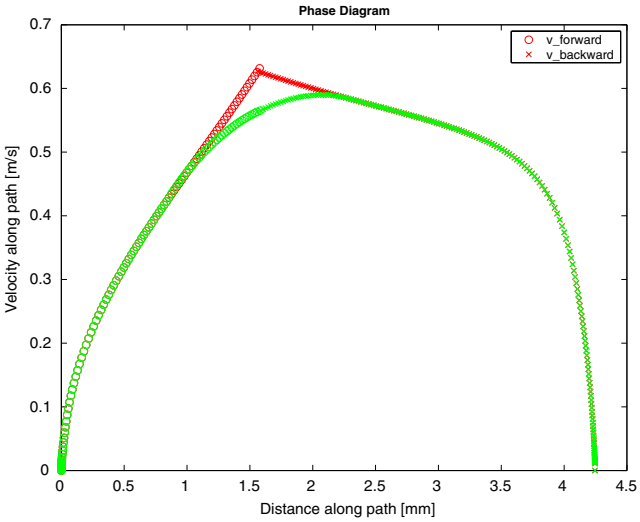

Fig. 11 Construction of the phase space velocity $\dot{s}(s)$, taking into account the jerk constraints

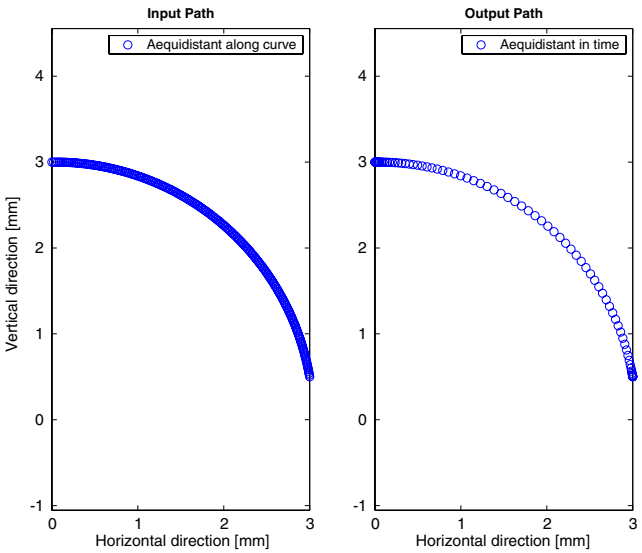

Fig. 13 Calculation of the final result $\mathbf{x}(\mathbf{t})$

4.7 Integration

At this point, the phase space velocity that corresponds to the velocity along the curve is constructed. $\dot{s}(s)$ is known. The next step is now to integrate over its inverse in order to find $t(s)$.

$$t(s^*) = \int_0^{s^*} \frac{1}{\dot{s}(s)} ds \tag{29}$$

If $t(s)$ is known, $s(t)$ can be calculated, which gives the solution to our problem, because now, $\mathbf{x}_p(s)$ is known and allows us to calculate the position for each time step via $\mathbf{x}_p(t) = \mathbf{x}_p(s(t))$. A spline interpolation method was chosen for the implementation, which was used for the example presented in the next section.

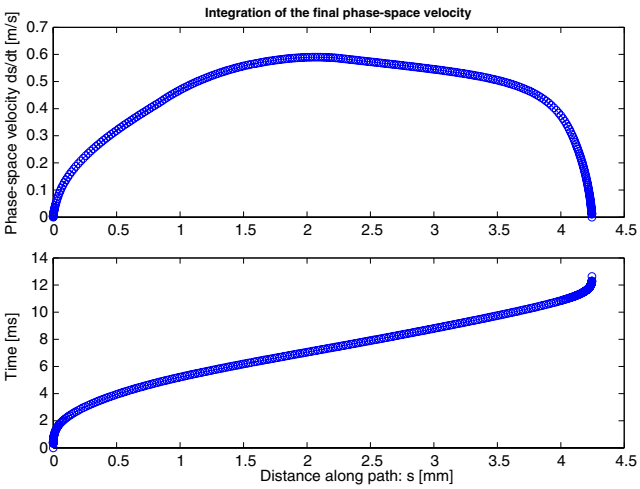

Fig. 12 Integration of the phase space velocity

5 Example

Finally, the complete calculation process will be illustrated with a two-axis motion and constraints as defined in Table 1. The figures in this section illustrate the calculations in one example. Figure 10 shows the input to the problem: the path that has to be followed as fast as possible. Figure 11 shows the construction of the phase space velocity via forward and backward integration in the phase space. The maximum allowed phase space velocity depends mainly on the velocity constraints for the motor. Based on the resulting phase space velocity, which can be seen in the upper part of Fig. 12, the final moving distance per time-step can be

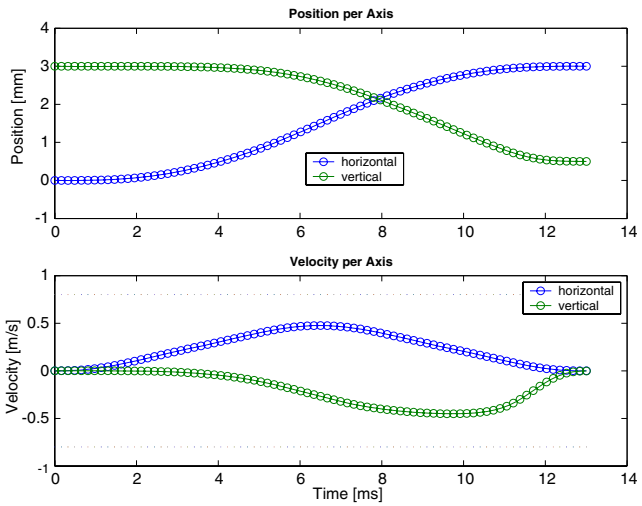

Fig. 14 Verification of the result: The horizontal and vertical velocities do not violate the constraints

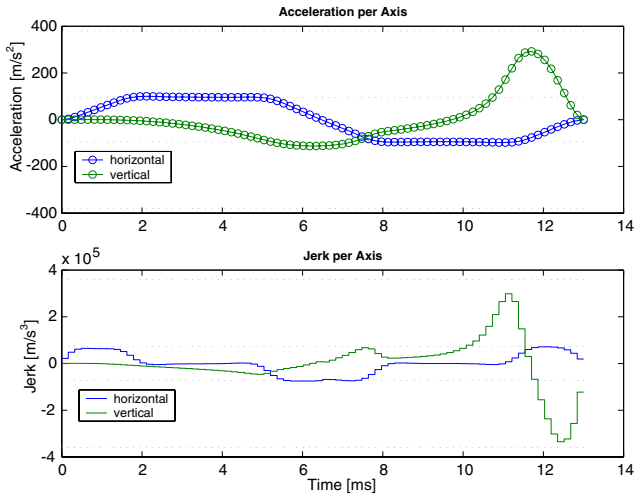

Fig. 15 Verification of the result: acceleration and jerk. It can be seen that neither the limits for the acceleration nor those for the jerk are violated and that, during the whole trajectory, there is always one constraint on one axis at its limit

calculated via integration. The lower part of Fig. 12 shows the duration of the move $t(s)$ as a function of the position along the path.

Based on $t(s)$ (lower part of Fig. 12) and $\mathbf{x}(\mathbf{s})$ (left part of Fig. 13), $\mathbf{x}(\mathbf{t})$ can be calculated (right part of Fig. 13). This is the main result of the whole calculation. Based on $\mathbf{x}(t)$, the position as a function of the time, the jerk, acceleration, and velocity are readily calculated for verification. This is shown in Fig. 14, where the position as a function of the time (top) and the resulting velocity (bottom) are shown separately for the horizontal and the vertical axes. Likewise, Fig. 15 shows the resulting acceleration (top) and the jerk (bottom). It can be seen that the acceleration follows its boundaries (central dotted lines for the horizontal axis, outer dotted lines for the vertical axis) for almost the complete move and that the jerk never violates its constraints.

Table 2 compares the obtained result with what the standard proximate time-optimal algorithm without jerk constraints would give. It can be seen that the presented enhancement of the proximate time-optimal algorithm substantially reduces the maximum jerks on the system while only minimally affecting the total move time.

Table 2 Comparison of the results obtained with and without consideration of the jerk constraints

	Without jerk constraints	With jerk constraints	Change
Move time	12.8 ms	13.0 ms	+1.6%
Max. horizontal jerk	508,830 m/s^3	75,208 m/s^3	−85.2%
Max. vertical jerk	2,470,800 m/s^3	335,330 m/s^3	−86.4%

6 Conclusion and outlook

It has been shown that the "proximate time-optimal algorithm" can be extended to take into account constraints on the jerk. This allows us to calculate near time-optimal jerk-constrained trajectories along three times differentiable paths in a nonperturbative way. It could be shown that the jerks on the system can be substantially reduced while almost maintaining the total move time. This allows us to considerably reduce the vibration in the overall system with a minimal speed penalty. The presented method can, for example, be applied to fast modern micromanipulators to calculate the near-optimal acceleration and jerk profiles. As the systems become faster and the required accuracy higher, it is more and more important to take the jerk constraints into account a priori.

As the trajectories also become more complex, it will become important to optimize the algorithm even further. In this paper, all calculations, considerations, and examples are based on a time-continuous approach. In a real application with a micromanipulator, whose motion controller has a certain cycle-frequency, it turns out that the time-discrete approach results in equations that can be processed faster and, thus, allows us to realize the computation of the paths for a fast modern micromanipulator without an overall time penalty for the system.

Acknowledgements This work was done during my employment at Oerlikon Assembly Equipment AG and would not have been possible without the courtesy of both Oerlikon and Icos Vision Systems NV. I am especially indebted to Marit Seidel; Daniel Bolliger; and, of course, Francisca Scharpé, who were always very supportive.

References

1. Constantinescu D, Croft EA (2000) Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. J Robot Syst 17(5):233–249
2. Dubowsky S, Bobrow JE, Gibson JS (1985) Time-optimal control of robotic manipulators along specified paths. Int J Rob Res 4(3):3–17
3. Ecker JG, Kupferschmid M, Marin SP (1994) Performance of several optimization methods on robot trajectory planning problems. SIAM J Sci Comput 15(6):1401–1412
4. Lin M-T, Tsai M-S, Yau H-T (2007) Development of a dynamics-based NURBS interpolator with real-time look-ahead algorithm. Int J Mach Tools Manuf 47(15):2246–2262

5. Nam S-H, Yang M-Y (2004) A study on a generalized parametric interpolator with real-time jerk-limited acceleration. Comput Aided Des 36(1):27–36
6. Pardo-Castellote G, Cannon Jr RH (1996) Proximate time-optimal algorithm for on-line path parameterisation and modification. IEEE Int Conf Robot Autom 2:1539–1546
7. Shin KG, McKay ND (1985) Minimum-time control of robotic manipulators with geometric path-constraints. IEEE Trans Automat Contr 30(6):531–541
8. Shin KG, McKay ND (1986) Minimum-time trajectory planning for industrial robots with general torque constraints. IEEE Int Conf Robot Autom 1:412–415
9. Slotine J-JE, Yang HS (1989) Improving the efficiency of time-optimal path-following algorithms. IEEE Trans Robot Autom 5(1):118–124
10. Timar SD, Farouki RT, Smith TS, Boyadjieff CL (2005) Algorithms for time-optimal control of CNC machines along curved tool paths. Robot Comput Integr Manuf 21(1):37–53