HOW TO ASSIGN TIME-OPTIMAL TRAJECTORIES TO PARALLEL ROBOTS - AN ADAPTIVE JERK-LIMITED APPROACH

Ingo Pietsch
Robert Bosch GmbH, Stuttgart, Germany
e-mail: ingo.pietsch@de.bosch.com

Carlos Bier
Institut für Werkzeugmaschinen und Fertigungstechnik, TU Braunschweig, Langer Kamp 19b, Germany
e-mail: c.bier@tu-bs.de

Oliver Becker
Robert Bosch GmbH, Stuttgart, Germany
e-mail: oliver.becker@de.bosch.com

Jürgen Hesselbach
Institut für Werkzeugmaschinen und Fertigungstechnik, TU Braunschweig, Langer Kamp 19b, Germany
e-mail: j.hesselbach@tu-bs.de

Abstract. *This paper provides a new algorithm for time-optimal adaptive trajectory planning for parallel robots. The approach enhances existing solutions by including the parameter information obtained for adaptive control schemes in the trajectory planning process. Furthermore, an explicit limitation of the path jerk is realized. A new algorithm to identify reachable points of the maximum velocity curve (MVC), which leads to smooth trajectories, is derived. Experimental results using the plane parallel robot* PORTYS *demonstrate the high performance of the proposed algorithm.*

Keywords*: Machine Tools, Trajectory Planning, Parallel Kinematics*

1. Introduction

Time-optimal trajectory planning is an effective way to increase the productivity of robotic manipulators for a variety of tasks especially in the areas of handling and assembly. A time-optimal trajectory describes the maximal velocity and acceleration for each pose along a prespecified path, where the dynamic properties of the robot, limited drive forces or torques as well as different task-specific boundary conditions are met.

A basic algorithm for time-optimal trajectory planning has been developed almost simultaneously by the groups of Boborow, Dubowsky and Gibson (1985), Shin and McKay (1985) and Pfeiffer and Johanni (1986). The basic idea of these algorithms is to describe the robot dynamics in dependency on a path description in parameterized form. Thus, an optimization problem independent of the degrees of freedom (DOF) of the robot is obtained. It has been shown, that an intuitive solution of this problem can be found, eventually based on Pontryagins maximum principle. The approaches differ e.g. in the way reachable points on the maximum velocity curve (MVC) are determined. The MVC describes the maximal velocity for each point on the path which could be reached if no dependencies between position, velocity and acceleration along the path would occur. Thus, it defines the boundary velocity for each trajectory and can only be reached at certain ("reachable") points. The main drawback of these approaches is the generation of a hard switching between minimal and maximal acceleration on the trajectory causing increasing waste of the drives and robot mechanics.

Pledel and Bestaoui (1995) propose to include the drive dynamics in the trajectory generation using a cost functional, which weights travelling time as well as drive energy in order to achieve a "softer" trajectory characteristic. A similar approach has been developed by Shiller (1996). The difficulty of these approaches is the determination of the cost functional. An implicit jerk limitation is achieved by the algorithm of Constantinescu and Croft (2001) by limiting the drive force rate, leading to reduced strain, improved tracking accuracy and speed. In contrast, the proposed algorithm limits the trajectory jerk explicitly while the drive force rate is implicitly limited. So far, all algorithms can be used for serial as well as for parallel robots. Generally the ratio of payload to moved robot mass of serial robot manipulators amounts typically 1/10 (Hirzinger *et al.*, 2001). Therefore the influence of changing payloads on the dynamic robot behavior is small. On the other hand high structural stiffness of parallel robots allows the construction of light-weight links and joints. As a result, the ratio of payload to moved robot mass decreases. Therefore, payload changes significantly influence the dynamic behavior of the robot.

This changing dynamic behavior has to be taken into account for control as well as for the generation of time-optimal trajectories in order to exploit the full dynamic robot capabilities. Therefore, an algorithm for time-optimal trajectory planning is derived which can be combined with adaptive control approaches. The algorithm is formulated in parameter-linear form, allowing the direct use of parameter estimates of the adaption laws of standard adaptive control

schemes as e.g. the approaches of Craig (1988) or Slotine and Li (1989). Furthermore, the path jerk can explicitly be prespecified and a new algorithm for finding reachable points on the MVC is presented. The performance of the proposed approach is evaluated using the plane parallel robot PORTYS (Hesselbach, Pietsch and Budde, 2001) (Fig. 1).

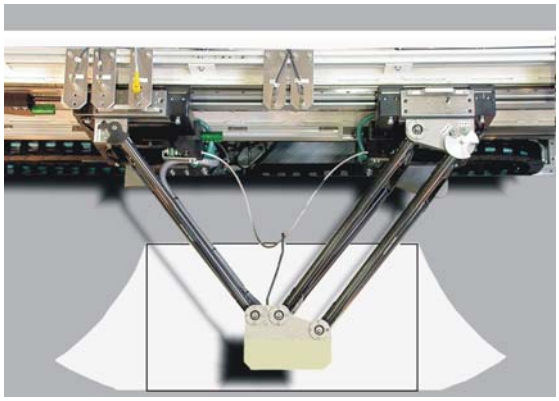


Figure1. Plane high-speed parallel robot PORTYS

2. Adaptive time-optimal trajectory planning

The aim of time-optimal trajectory planning, as it is commonly understood in this context, is the determination of the maximum velocity profile along a given path that complies with all given dynamic and kinematic robot constraints. The problem can be described mathematically as an optimal control problem aiming to minimize the traversal time

$$t_e = \int_{s_0=s(t=0)}^{s_e=s(t=t_e)} \frac{1}{v_p} ds_p \rightarrow min \tag{1}$$

with the strictly increasing path parameter s_p and the path velocity v_p underlying the following constraints

$$\frac{ds_p}{dt} = v_p, \quad \frac{dv_p}{dt} = a_p, \quad \frac{da_p}{dt} = j_p, \tag{2}$$

where a_p and j_p are path acceleration and path jerk respectively. $\mathbf{q} = \mathbf{q}(s_p)$ forms the path constraint in joint-coordinates. The drive torque limitations are given as $\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau}(s_p, v_p, a_p) \leq \boldsymbol{\tau}_{\max}$.

The drive velocity contraints are formulated as (3a), the path velocity limitation as (3b), where forward motion along the path is assumed and the jerk limitation as (3c).

$$\text{(a)}\ \mathbf{v}_q^{\min} \leq \mathbf{v}_q = \dot{\mathbf{q}} \leq \mathbf{v}_q^{\max}, \qquad \text{(b)}\ 0 \leq v_p \leq v_{path}^{\max}, \qquad \text{(c)}\ j_p^{\min} \leq j_p \leq j_p^{\max}. \tag{3}$$

The admissible torques can be obtained from the robot dynamics. Dynamic modelling of parallel robots is a complex task because of additional kinematic constraints imposed by closure conditions of the kinematic loops (Dasgupta and Mruthyunjaya, 1998). The robot dynamics can always be expressed in parameter-linear form as $\boldsymbol{\tau} = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\boldsymbol{\pi}$, with the regressor matrix $\mathbf{Y} \in \mathbb{R}^{l \times n}$ and the dynamic parameter vector $\boldsymbol{\pi} \in \mathbb{R}^n$, where n determines the DOF of the robot (Sciavicco and Siciliano, 2000). The usage of the parameter-linear form has two advantages: Firstly, the dynamic parameter estimates of adaptive control algorithms can directly be used to improve the results of the trajectory planning process. Secondly, it needs less computational time (Hesselbach *et al.*, 2003). From (1) it is obvious that the optimal control problem can be solved in the (s_p, v_p)-state-space by finding the maximum admissible value of v_p for each s_p. By inserting the parameterized path $\mathbf{q} = \mathbf{q}(s_p)$ and its time derivatives as a function of the path parameter s_p in the robot dynamics, the dimension of the optimization problem can be reduced from $2n$ to 2 for rigid robots (Dahl, 1992). In order to determine the admissible acceleration from the robot dynamics, $\boldsymbol{\tau} = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\boldsymbol{\pi}$, can be reformulated as

$$\boldsymbol{\tau} = \left(\mathbf{Y}_a\left(\mathbf{q}(s_p), \mathbf{q}'(s_p)\right)a_p + \mathbf{Y}_{sv}\left(\mathbf{q}(s_p), \mathbf{q}'(s_p), \mathbf{q}''(s_p), v_p\right)\right)\boldsymbol{\pi}, \tag{4}$$

where $\dot{\mathbf{q}}(s_p) = \mathbf{q}'(s_p)\dot{s}_p$, $\ddot{\mathbf{q}}(s_p) = \mathbf{q}''(s_p)\dot{s}_p^2 + \mathbf{q}'(s_p)a_p$ have been used. With (4) the maximum and minimum dynamically allowed acceleration for each drive i can be calculated analytically as:

$$a_{p\max}^{i\ dyn}(s_p,v_p)=\begin{cases}\frac{(\tau_{\max}^i-\mathbf{Y}_{sv}^i(s_p,v_p)\boldsymbol{\pi})}{\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}} & \mathbf{Y}_a^i(s_p)\boldsymbol{\pi}>0\\ \frac{(\tau_{\min}^i-\mathbf{Y}_{sv}^i(s_p,v_p)\boldsymbol{\pi})}{\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}} & \mathbf{Y}_a^i(s_p)\boldsymbol{\pi}<0\\ \infty & \mathbf{Y}_a^i(s_p)\boldsymbol{\pi}=0\end{cases},\quad a_{p\min}^{i\ dyn}(s_p,v_p)=\begin{cases}\frac{(\tau_{\min}^i-\mathbf{Y}_{sv}^i(s_p,v_p)\boldsymbol{\pi})}{\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}} & \mathbf{Y}_a^i(s_p)\boldsymbol{\pi}>0\\ \frac{(\tau_{\max}^i-\mathbf{Y}_{sv}^i(s_p,v_p)\boldsymbol{\pi}_i)}{\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}} & \mathbf{Y}_a^i(s_p)\boldsymbol{\pi}<0.\\ -\infty & \mathbf{Y}_a^i(s_p)\boldsymbol{\pi}=0\end{cases}\tag{5}$$

The drive, which is maximal stressed, is responsible for the acceleration limit $a_{p\max}^{dyn}(s_p,v_p)=\min_i\left(a_{p\max}^{i\ dyn}(s_p,v_p)\right)$, $a_{p\min}^{dyn}(s_p,v_p)=\max_i\left(a_{p\min}^{i\ dyn}(s_p,v_p)\right)$. The acceleration is also limited by the maximum jerk: $a_{p\max}^{jerk}(t)=\int j_{p\max}dt$, $a_{p\min}^{jerk}(t)=\int j_{p\min}dt$.

Thus, the extreme path accelerations yield

$$a_{p\max}=\min\left(a_{p\max}^{dyn}(s_p,v_p),a_{p\max}^{jerk}(t)\right),\qquad a_{p\min}=\max\left(a_{p\max}^{dyn}(s_p,v_p),a_{p\min}^{jerk}(t)\right).\tag{6}$$

As a result of the different constraints, for each path parameter s_p an upper limit for the reachable velocity $v_p^{\max}$ can be calculated which cannot be exceeded without violating at least one constraint. The set of all $v_p^{\max}$ for $[s_0,s_e]$ determines the maximum velocity curve (MVC). Given a pair of state-space variables $(s_p,\ v_p)$ the constraints (3) can easily be checked:

$$\text{(a)}\ \mathbf{v}_q=\mathbf{q}'(s_p)v_p\le\mathbf{v}_{q_{\max}},\qquad\text{(b)}\ v_p\le v_{p\max},\qquad\text{(c)}\ j_{p\min}\le j_p=\frac{d^2v_p}{dt^2}\le j_{p_{\max}}.\tag{7}$$

With the following considerations the maximum velocity that depends on the drive torque constraints $\boldsymbol{\tau}_{\min}\le\boldsymbol{\tau}(s_p,v_p,a_p)\le\boldsymbol{\tau}_{\max}$ can be calculated. The boundary condition for the drive torque constraints is $\mathbf{a}_{\min}=\mathbf{a}_{\max}$. With regard to (4), if $\mathbf{Y}_a(s_p)\boldsymbol{\pi}\neq\mathbf{0}$, this case only occurs if $a_{p\max}^{dyn}=a_{p\min}^{dyn}$. Introducing the abbreviations $\mathbf{Y}_{sv}(s_p,v_p)\boldsymbol{\pi}=\left(\mathbf{a}_1(s_p)v_p^2+\mathbf{a}_2(s_p)\right)\boldsymbol{\pi}$ the last condition can be checked analytically using (5), thus leading towards

$$v_{p\max}^{k\ dyn}=\sqrt{\frac{\mathbf{Y}_a^i\boldsymbol{\pi}\left(\tau_{\min}^j-\mathbf{a}_2^j\boldsymbol{\pi}\right)+\mathbf{Y}_a^j\boldsymbol{\pi}\left(\mathbf{a}_2^i\boldsymbol{\pi}-\tau_{\max}^i\right)}{\boldsymbol{\pi}^T\mathbf{Y}_a^{iT}\mathbf{a}_1^j\boldsymbol{\pi}-\boldsymbol{\pi}^T\mathbf{Y}_a^j\mathbf{a}_1^i\boldsymbol{\pi}}},\quad i\neq j\wedge\left(\left(\ \mathbf{Y}_a^i\boldsymbol{\pi}>0\wedge\ \mathbf{Y}_a^j\boldsymbol{\pi}>0\right)\vee\left(\mathbf{Y}_a^i\boldsymbol{\pi}<0\wedge\ \mathbf{Y}_a^j\boldsymbol{\pi}<0\right)\right)\tag{8}$$

$$v_{p\max}^{k\ dyn}=\sqrt{\frac{\mathbf{Y}_a^i\boldsymbol{\pi}\left(\tau_{\max}^j-\mathbf{a}_2^j\boldsymbol{\pi}\right)+\mathbf{Y}_a^j\boldsymbol{\pi}\left(\mathbf{a}_2^i\boldsymbol{\pi}-\tau_{\max}^i\right)}{\boldsymbol{\pi}^T\mathbf{Y}_a^{iT}\mathbf{a}_1^j\boldsymbol{\pi}-\boldsymbol{\pi}^T\mathbf{Y}_a^j\mathbf{a}_1^i\boldsymbol{\pi}}},\quad i\neq j\wedge\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}>0\wedge\mathbf{Y}_a^j(s_p)\boldsymbol{\pi}<0,\tag{9}$$

$$v_{p\max}^{k\ dyn}=\sqrt{\frac{\mathbf{Y}_a^i\boldsymbol{\pi}\left(\tau_{\min}^j-\mathbf{a}_2^j\boldsymbol{\pi}\right)+\mathbf{Y}_a^j\boldsymbol{\pi}\left(\mathbf{a}_2^i\boldsymbol{\pi}-\tau_{\min}^i\right)}{\boldsymbol{\pi}^T\mathbf{Y}_a^{iT}\mathbf{a}_1^j\boldsymbol{\pi}-\boldsymbol{\pi}^T\mathbf{Y}_a^j\mathbf{a}_1^i\boldsymbol{\pi}}},\qquad i\neq j\wedge\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}<0\wedge\mathbf{Y}_a^j(s_p)\boldsymbol{\pi}>0.\tag{10}$$

If at least one element of $\mathbf{Y}_a(s_p)\boldsymbol{\pi}$ equals zero, $v_{p\max}^{k\ dyn}$ can be calculated according to

$$v_{p\max}^{k\ dyn}=\sqrt{\frac{\tau_{\max}^i-\mathbf{a}_1^i(s_p)}{\mathbf{a}_2^i(s_p)\boldsymbol{\pi}}},\quad v_{p\max}^{k\ dyn}=\sqrt{\frac{\tau_{\min}^i-\mathbf{a}_1^i(s_p)}{\mathbf{a}_2^i(s_p)\boldsymbol{\pi}}}.\tag{11}$$

The minimum value of $v_{p\max}^{k\ dyn}(s_p)$ yields the maximum admissible velocity $v_{p\max}^{dyn}(s_p)=\min_k v_{p\max}^{k\ dyn}(s_p),\ \forall v_{p\max}^{k\ dyn}\in\mathbb{R}$.

In general, a trajectory can reach the MVC at single points or on single intervals only because of its dynamic constraints (2). Reachable MVC points are characterized by one of the four following conditions:

a) Points, where the MVC results from constraints (7) and the acceleration is below the limits defined by (6).
b) Tangency points where the slope of the MVC is equal to the slope of the trajectory. The condition can be checked by detecting a sign change of

$$\kappa(s_p)=v_p'(s_p)-v_{b\max}'(s_p)=\frac{a_{p\max}(s_p)}{v_{p\max}(s_p)}-\frac{dv_{p\max}(s_p)}{ds_p}.\tag{12}$$

c) Points, where $\mathbf{Y}_a^i(s_p)\boldsymbol{\pi}=\mathbf{0}$.
d) Discontinuity points of the MVC.

Thus, the algorithm can be broken down into eight steps. It is formulated in a way suited for direct implementation:

Algorithm 1: Jerk-limited, time-optimal trajectory planning

A1.1. Integrate forward from s_0 with maximum acceleration according to (6) until either $s_p \geq s_p(t_e) = s_e$ or the trajectory exceeds the MVC. In the latter case set $s_1 = s_p$. Go on with step A1.2.

A1.2. Integrate backward from s_e with minimum acceleration until either the curve intersects the forward trajectory or vanishes at the MVC. In the first case go to A1.8, in the second case set $s_2 = s_p$ and go on with A1.3.

A1.3. Calculate the MVC once and all reachable points on the MVC between s_1 and s_2. Continue with A1.4.

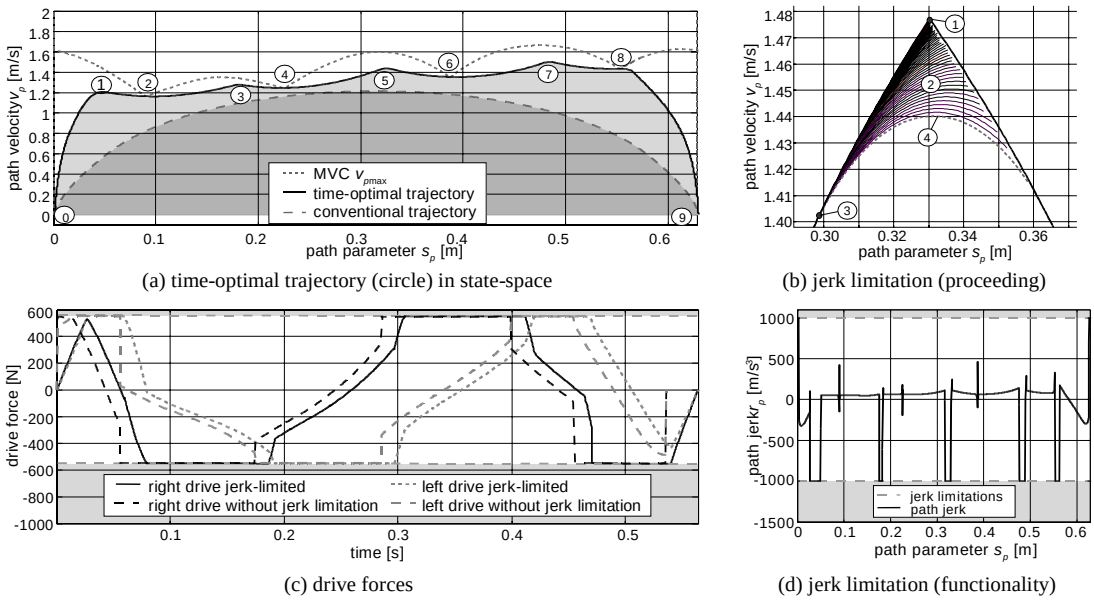
A1.4. Proceed from $s_p = s_1$ on the MVC as long as the points on the MVC are reachable or until the backward trajectory is intersected. In the first case attach the new trajectory segment to the forward trajectory and set s_3 the last reachable point. If $s_3 \neq s_1$ continue with A1.7, else with A1.5.

A1.5. Go to the next reachable point on the MVC and set this point s_1. If no reachable point is left, the algorithm fails, otherwise proceed with A1.6.

A1.6. Integrate backward from s_1 until the forward trajectory is met and connect the new trajectory segment with the forward trajectory. Continue with A1.7. Otherwise go back to A1.5. If $s_1 = s_2$ the trajectory is closed. Proceed with A1.8.

A1.7. Integrate forward from s_1 until the backward trajectory is reached or the trajectory vanishes at the MVC. In the first case go to A1.8, in the latter case attach the new trajectory segment to the forward trajectory, set s_1 the last point of the new forward trajectory and go back to A1.4.

A1.8. Search for points s_i on the trajectory, where the jerk limitation is violated. This case can occur at intersection points of different algorithm steps, i.e. if trajectory segments from forward and backward integration meet or at intersection points of trajectory segments calculated either with forward or backward integration with trajectory segments of former MVC segments (A1.4). Divide the trajectory at s_i in one front segment with $s_0 \leq s_p \leq s_i$ and one back segment with $s_i \leq s_p \leq s_e$. From $s_s = s_i$ go stepwise backwards on the front trajectory segment, start a new trajectory segment considering the jerk limitation starting at this point. Repeat this procedure until the new trajectory segment meets the back trajectory segment tangentially or $s_s = s_0$. In the first case the jerk limitation for this point was successful, and the new trajectory segment can be inserted between the actual s_s and the meeting point at the backward trajectory segment. In the latter case the jerk limit cannot be adhered. Thus, either the jerk limitation is relaxed until the algorithm succeeds or the algorithm fails and stops.



(a) time-optimal trajectory (circle) in state-space

(b) jerk limitation (proceeding)

(c) drive forces

(d) jerk limitation (functionality)

Figure 2. Time-optimal, jerk-limited and smoothed trajectory for a circular path

Figure 2a demonstrates the proceeding. Step A1.1 - forward integration - is active between point 0 and point 1. Between point 9 and point 8 the algorithm integrates backwards in step A1.2. In the third step A1.3 the MVC is calculated. Step A1.4 leads to $s_3 = s_1$, and the algorithm identifies the next reachable point on the MVC at point 2 in step A1.5. From point 2 integrate backward and forward in step A1.6 and A1.7, respectively. Again skipping step 4, point 4 is found in step 5. Go on by integrating backward and forward in steps A1.6 and A1.7 and so on until the trajectory is closed at point 8. Finally, the jerk limitation algorithm of step A1.8 is active at points 1, 3, 5, 7 and 8.

The jerk-limiting algorithm is illustrated in Fig. 2b and 2d. Figure 2b depicts how the jerk at point 1 is limited by an iterative search for a jerk-limited trajectory in area 2. It is found within trajectory segment 4. Figure 2d proves that the

algorithm works properly and the jerk is limited to the prescribed value of 1000 m/s^3 in this example. Figure 2c shows the drive forces with and without jerk limitation. The traversal time of the jerk limited trajectory is slightly increasing, but the difference is small compared to the increasing settling time at the desired position if the jerk is not limited. The reachable points on the MVC are identified using an improved algorithm, that allows to determine reachable points with a prespecified accuracy and adequate computational effort. Previous approaches from Boborow, Dubowsky and Gibson (1985) and Shin and McKay (1985) suffer from high computational burden or low accuracy, respectively.

Algorithm 2: Identification of reachable MVC points for smooth trajectories

A2.1 A forward or backward trajectory section (FTS or BTS) is calculated according to step A1.6 or A1.7 in algorithm 1. The start value for a reachable point $x_e = \left[s_p^e \quad v_p^e \right]$ is determined using (12). The maximum value of the numerical deviation between the true and the numerically determined reachable point $\Delta^{\lim} v_p \left(s_p^e \right) > 0$ is defined.

a. Backward integration: The intersection between FTS and the recent BTS serves as starting point $x_s = \left[s_p^s \quad v_p^s \right]$ for the smoothing algorithm.

b. Forward integration: If the FTS intersects the BTS, the intersection point is the starting point x_s. If the FTS exceeds the MVC, this intersection point serves as starting point x_s.

A2.2 Defining the starting conditions for the iterative smoothing algorithm:

a. Backward integration: $i = i+1$. Beginning from x_s integrate forward until $s_p \geq s_p^e$. Compute $\Delta v_p^i \left(s_p^e \right) = v_p \left(s_p^e \right) - v_p^e \left(s_p^e \right)$. If $\Delta v_p^i > \Delta^{\lim} v_p$, x_s was too high in the $s_p - v_p$ plane. Thus, the next lower point on the FTS is defined as x_s^{new}. If $\operatorname{sgn}\left(\Delta v_p^i \right) = \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$ set $x_s = x_s^{new}$ and repeat A2.2. If $\operatorname{sgn}\left(\Delta v_p^i \right) \neq \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$, the starting point is between x_s^{new} and x_s. In this case $k > 2$ equidistant points $x_s^k(j), \quad j = 1 \ldots k, \quad x_s^k(1) = x_s^{new}, \quad x_s^k(k+1) = x_s$ are inserted in the trajectory segment between x_s^{new} and x_s. Go on with step A2.3. If $\Delta v_p^i < -\Delta^{\lim} v_p$, x_s was too low in the $s_p - v_p$ plane. Thus, the next higher point on the FTS is defined as x_s^{new}. If $\operatorname{sgn}\left(\Delta v_p^i \right) = \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$ set, $x_s = x_s^{new}$ and repeat A2.2. In this case $k > 2$ equidistant points $x_s^k(j), \quad j = 1 \ldots k, \quad x_s^k(1) = x_s, \quad x_s^k(k+1) = x_s^{new}$ are inserted in the trajectory segment between x_s^{new} and x_s. Go on with step A2.3. If $-\Delta^{\lim} v_p \leq v_p^i \leq \Delta^{\lim} v_p$ a sufficiently smooth trajectory is found.

b. Forward integration: $i = i+1$. Beginning from x_s integrate forward until $s_p \leq s_p^e$ and compute $\Delta v_p \left(s_p^e \right) - v_p^e \left(s_p^e \right)$. If $\Delta v_p^i > \Delta^{\lim} v_p$ x_s was too high in the $s_p - v_p$ plane. If the BTS in A1.7 was intersected, choose the next lower point on the BTS as x_s^{new}; otherwise define $x_s^{new} = \left[s_p^s \quad v_p^{s-1} \right]$, where v_p^{s-1} is the velocity of the next lower point on the FTS. If $\operatorname{sgn}\left(\Delta v_p^i \right) = \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$, $x_s = x_s^{new}$ and repeat A2.2. If $\operatorname{sgn}\left(\Delta v_p^i \right) \neq \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$, the starting point is between x_s^{new} and x_s. In this case $k > 2$ equidistant points $x_s^k(j), \quad j = 1 \ldots k, \quad x_s^k(1) = x_s^{new}, \quad x_s^k(k+1) = x_s$ are inserted in the trajectory segment between x_s^{new} and x_s. Go on with A2.3.

If $\Delta v_p^i < -\Delta^{\lim} v_p$ x_s was too low in the $s_p - v_p$ plane. If the BTS in A1.7 was intersected, choose the next higher point on the BTS as x_s^{new}; otherwise define $x_s^{new} = \left[s_p^s \quad v_p^{s+1} \right]$, where v_p^{s+1} is the velocity of the virtual next higher point on the FTS, obtaine by extrapolation. If $\operatorname{sgn}\left(\Delta v_p^i \right) = \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$, $x_s = x_s^{new}$ and repeat A2.2. If $\operatorname{sgn}\left(\Delta v_p^i \right) \neq \operatorname{sgn}\left(\Delta v_p^{i-1} \right)$, the starting point is between x_s^{new} and x_s. In this case $k > 2$ equidistant points $x_s^k(j), \quad j = 1 \ldots k, \quad x_s^k(1) = x_s, \quad x_s^k(k+1) = x_s^{new}$ are inserted in the trajectory segment between x_s and x_s^{new}. Go on with A2.3. If $-\Delta^{\lim} v_p \leq v_p^i \leq \Delta^{\lim} v_p$ a sufficiently smooth trajectory is found.

A2.3. Iterative search for a smooth trajectory:

a. Backward integration: $i = i+1$ and $j = j+1$. Starting from $x_s^k(j)$ integrate forward until $s_p \geq s_p^e$. Compute $\Delta v_p \left(s_p^e \right) - v_p^e \left(s_p^e \right)$. If $\Delta v_p^i < -\Delta^{\lim} v_p$ repeat A2.3. If $\Delta v_p^i > \Delta^{\lim} v_p$, $k > 2$ equidistant points $x_s^k(j), \quad j = 1 \ldots k$, $x_s^k(1) = x_s^{new}, \quad x_s^k(k+1) = x_s$ are inserted in the trajectory segment between x_s^{new} and x_s, set $j = 1$, repeat A2.3. If $-\Delta^{\lim} v_p \leq v_p^i \leq \Delta^{\lim} v_p$ a sufficiently smooth trajectory is found.

b. Forward integration: $i = i+1$ and $j = j+1$. Starting from $x_s^k(j)$ integrate backward until $s_p \leq s_p^e$. Compute $\Delta v_p \left(s_p^e \right) - v_p^e \left(s_p^e \right)$. If $\Delta v_p^i < -\Delta^{\lim} v_p$ repeat A2.3. If $\Delta v_p^i > \Delta^{\lim} v_p$, $k > 2$ equidistant points $x_s^k(j), \quad j = 1 \ldots k$, $x_s^k(1) = x_s^{new}, \quad x_s^k(k+1) = x_s$ are inserted in the trajectory segment between x_s^{new} and x_s, set $j = 1$, repeat A2.3. If $-\Delta^{\lim} v_p \leq v_p^i \leq \Delta^{\lim} v_p$ a sufficiently smooth trajectory is found.

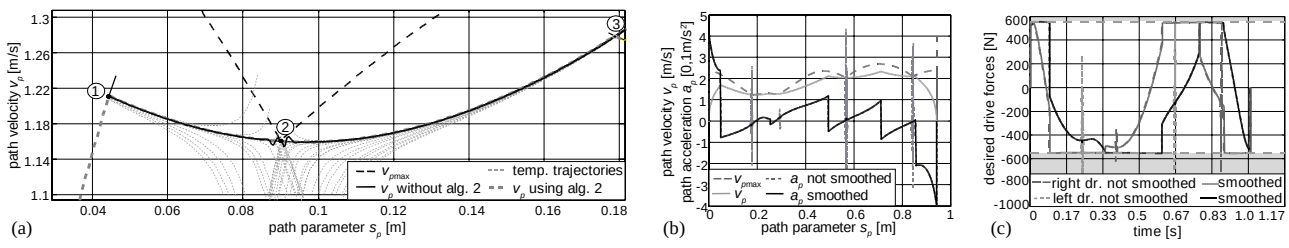


Figure 3. Procceding and functionality of algorithm 2: Smoothing of discontinuities caused by numerical inaccuracies due to finite sample time during calculation of reachable points on the MVC

In Fig. 3a the proceeding of algorithm 2 is depicted. The reachable point 2 on the MVC is firstly determined using the analytic algorithm of Shin and McKay (1985), according to a sign change of equation (12). Because of the finite sample time (0.0005 s for this example), the reachable point 2 is detected with a slight deviation from its true value. According to steps A1.6 and A1.7 it defines the starting conditions for the adjacent backward and forward integration, respectively. These inaccuracies of the integration starting conditions cause the oscillations in the vicinity of point 2. These oscillations result in strong peaks in the acceleration (Fig. 3b) and in the drive forces (Fig. 3c) accordingly. Using the intersection points 1 and 3 of the (oscillatory) trajectory sections with the BTS and FTS, as starting points for steps A2.2 and A2.3 for an iterative accuracy improvement of the computationof the reachable MVC point leads to a smooth trajectory as shown in Fig. 3a. Thus, also the undesired peaks in the path acceleration signal and the drive forces signal are removed, as depicted in Fig. 3b and 3c, respectively.

3. Experimental Results

The robot PORTYS (Fig. 1) is used to evaluate the performance of the proposed approach for adaptive time-optimal path planning. In Tab. 1 some important robot parameters are given:

Table 1. Parameters of the PORTYS robot.

Parameter	Value
moved robot mass	29.2 kg
max. payload	10.0 kg
max. drive force	550.0 N
max. drive velocity	6.0 m/s

Again, it is noteworthy, that the ratio of payload to moved robot mass is high (1/3), thus payload changes are expected to influence significantly the robot dynamic behavior.

Two different test paths are used for the experiments: a circular (Fig. 4a and 4b) and a triangular path (Fig. 4c and 4d). The results of the adaptive, time-optimal trajectories are compared to the results obtained using conventionally planned trajectories based on a trapezoid acceleration profile. The conventional trajectories are iteratively planned with respect to the maximum payload such that the maximum drive forces are fully exploited but do not exceed the drive force limitations.

An adaptive composite control approach developed by Slotine and Li with bounded-gain forgetting is used to control the PORTYS robot (Slotine and Li, 1989):

$$\boldsymbol{\tau}_d = \hat{\mathbf{B}}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\ddot{\mathbf{q}}_{rs} + \hat{\mathbf{C}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}_r + \hat{\mathbf{g}}(\mathbf{q}) = Y_{rs}(\mathbf{q},\dot{\mathbf{q}},\mathbf{q}_r,\dot{\mathbf{q}}_{rs})\hat{\boldsymbol{\Pi}}, \tag{13}$$

with the estimated mass matrix $\hat{\mathbf{B}} \in \mathbb{R}^{n\times n}$, the matrix of the estimated centrifugal and coriolis terms $\hat{\mathbf{C}} \in \mathbb{R}^{n\times n}$, the gravitational force vector $\hat{\mathbf{g}} \in \mathbb{R}^n$. $\dot{\mathbf{q}}_r = \dot{\mathbf{q}}_d - \boldsymbol{\Lambda}\tilde{\mathbf{q}}$ describes a reference velocity, where $\mathbf{q}_d$ is the desired position, $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ is a weighting matrix and the position deviation is described as $\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}_d$. The parameter estimation is performed using a least-squares approach:

$$\dot{\hat{\boldsymbol{\Pi}}} = -\boldsymbol{\Gamma}(t)\big(\mathbf{Y}_{rs}^T\mathbf{s} + \mathbf{W}^T\mathbf{R}\tilde{\mathbf{y}}(t)\big), \tag{14}$$

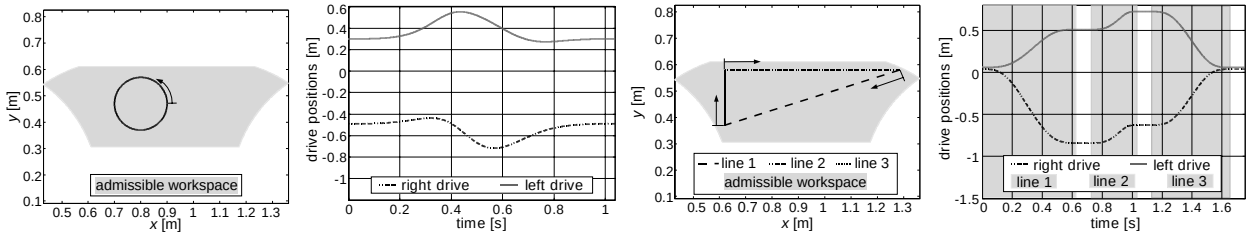
where $\mathbf{s} = \dot{\tilde{\mathbf{q}}} + \boldsymbol{\Lambda}\tilde{\mathbf{q}}$ is the weighted position and velocity error, $\mathbf{W}(\mathbf{q},\dot{\mathbf{q}})$ is a low-pass filtered version of $\mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ (in order to avoid error-prone measurement of $\ddot{\mathbf{q}}$) and $\mathbf{R}$ weights the ratio between $\mathbf{s}$ and the prediction error $\tilde{\mathbf{y}}(t) = \mathbf{W}\hat{\boldsymbol{\Pi}} - \mathbf{W}\boldsymbol{\Pi}$. The time-variant adaption gain is calculated according to

$$\dot{\boldsymbol{\Gamma}}(t) = -\chi(t)\boldsymbol{\Gamma}(t) + \boldsymbol{\Gamma}(t)\mathbf{W}^T(t)\mathbf{W}(t)\boldsymbol{\Gamma}(t), \tag{15}$$

with the forgetting factor χ, which is defined as

$$\chi(t) = \chi_0\left(1 - \frac{\|\boldsymbol{\Gamma}(t)\|}{k_0}\right). \tag{16}$$

k_0 describes an upper bound of $\chi(t)$.



(a) circular path (b) drive positions circular path (c) triangular path (d) drive positions triangular path

Figure 4. Circular and triangular test path

A dSpace DS1103 Rapid-Prototyping System based on a DSP (Motorola PowerPC 604e, 400 MHz) is used as controller hardware. A control sample frequency of 2 kHz could be realized. During the experiments the traversal time t_e as well as the mean tracking accuracy ΔH^5 and drive utilization rate η_u are investigated. We define the mean tracking accuracy in operational space after passing N times the test paths as

$$\Delta H^N = \frac{1}{N}\sum_{t_k=1}^{N} \|\mathbf{p}_d(t) - \mathbf{p}(t)\|,\ \mathbf{p} = [x, y] \tag{17}$$

For the described experiments we chose $N = 5$ in order to compare the tracking error when parameter adaption of the controller is completed. Payloads of $m_e = 0.0$ kg and $m_e = 7.5$ kg are used to show the influence of payload changes on the dynamic behavior of the robot. Both, traversal times and mean tracking error for the two test paths for conventionally planned (conv.) as well as for time-optimal trajectories (t.o., generated using the approach decribed above) are compared in Tab. 2. In the case of the time-optimal trajectories, the payload estimate of the adaptive control law (13)-(16) is used to adapt the trajectory to changing dynamic robot properties. For conventionally planned information can not be used directly to optimize the trajectory. Therefore, in this case for both payloads the same trajectory has been used.

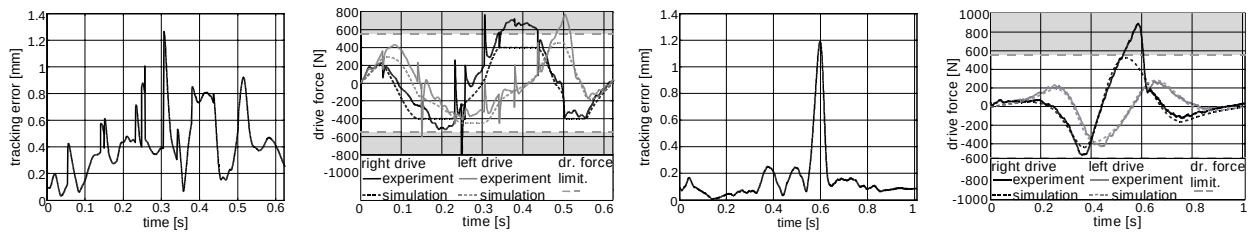
Obviously, the traversal time can significantly be reduced up to 40 % if the time-optimal trajectory is used instead of the conventional trajectory.

Table 2. Traversal times and mean tracking accuracies for conventional and time-optimal trajectories.

Trajectory	m_e [kg]	t_e [s]	ΔH^5 [mm]
circular, conv.	0.0; 7.5	1.04	0.06; 0.08
circular, t.o.	0.0	0.62	0.62
circular, t.o.	7.5	0.68	0.51
triangular, conv.	0.0; 7.5	1.45	0.07; 0.08
triangular, t.o.	0.0	1.22	0.37
triangular, t.o.	7.5	1.28	0.35

This result can be explained according to Fig. 2a using equation (1). The traversal time is inversely proportional to the area beyond the trajectory in state-space formulation. Because the time-optimal trajectory better exploits the drive capabilities especially during acceleration and deceleration phases it is located higher in the $s_p - v_p$-plane, thus covering a larger area, hence reaching lower traversal times. Furthermore, if the payload information is included, the traversal time is further decreased by app. 10 %.

At the same time, the mean tracking error increases. But even if the relative increase is significant, the absolute tracking error remains considerably small. Figure 5a shows the tracking error and the drive forces for the circular path using the time-optimal trajectory without additional payload.



(a) Testpath with time-optimal trajectory (b) testpath with conventional trajectory

Figure 5. Tracking error and drive force for circular testpath with, (a) time-optimal trajectory and no additional payload after five trajectory passes, (b) conventional trajectory and no additional payload after five trajectory passes.

In order to refute the assumption that similar traversal time reductions could be obtained using conventional trajectories which are modified in a way that similar path deviations occur, Figure 5b depicts tracking error and drive forces for a modified conventionally planned trajectory.

Showing a similar maximum tracking error as obtained using the time-optimal trajectory, the drive forces exceeds their limitations clearly, whereas the traversal time is only slightly reduced to 1.01 s.

4. Conclusions

The adaptive time-optimal trajectory planning approach presented in this paper enhances existing algorithms by integrating information of dynamic parameter changes, including a jerk limitation and an improved algorithm to determine reachable MVC points. Because of its adaptive properties it is especially suited to the needs of parallel robots obtaining a high ratio of payload to moved robot mass. Experimental results indicate a significant decrease in the reachable traversal times thus leading to increasing productivity of the robotic system. A drawback of the proposed approach are slightly increasing tracking errors. A possibility to overcome this problem may be the application of a path velocity control algorithm as proposed by Dahl (1992).

5. Acknowledgements

This work is supported by the German Research Foundation DFG within the scope of the Colloborative Research Center 562.

6. References

Boborow, J., Dubowsky, S., Gibson, J., 1985, “Time-Optimal Control of Robotic Manipulators along Specified Paths”, Int. Journal of Robotics Research 4 1985/3, pp. 3-17.

Constantinescu, D., Croft E. A., 2001 “Smooth and Time-Optimal Trajectory Planning for Industrial Robots Along Specified Paths”, Journal of Robotic Systems 17 2001/5, pp. 233-249.

Craig, J. J., 1988, “Adaptive Control of Mechanical Manipulators”, Addison-Weasley.

Dahl, O., 1992, “Path Constraint Robot Control”, PhD Thesis, Departement of Automatic Control, Lund Institute of Technology, Sweden.

Dasgupta, B., Mruthyunjaya, T. S., 1998, “A Newton-Euler Formulation for the Inverse Dynamics of the Stewart Platform Manipulator”, J. of Mechanisms and Machine Theory 33 (1998) 8, pp. 1135-1152.

Hesselbach, J., Pietsch, I., Budde, C., 2001 „ PORTYS – ein neues Maschinenkonzept für Hochgeschwindigkeits-handhabung und Montage”, Z. für wirtschaftlichen Fabrikbetrieb (ZWF) 96 2001/9, pp. 534-538 .

Hesselbach, J., Pietsch, I., Krefft, M., Becker, O., Plitea, N., 2003, “A Generic Formulation of the Dynamics of Plane Parallel Robots for Real-Time Applications. In: CD-Proc. of 12th Int. Workshop on Robotics in Adria-Alpe-Danube Region, Cassino, Italy.

Hirzinger, G., Albu-Schäffer, A., Hähnle, M., Schaefer, I., Sporer, N., 2001 “On a New Generation of Torque Controlled Ligth-Weight Robots”, In: Proc. of IEEE Int. Conf. on Robotics and Automation, Vol. 4, Seoul, Korea, pp. 3356-3363.

Pledel, P., Bestaoui, Y., 1995, “A Comparative Study of Optimal Control Algorithms for Robot Continuous Path Planning”, In: Proc. IEEE Conf. on Decision and Control, New Orleans, USA, 2 , pp. 1009-1010.

Pfeiffer, F., Johanni, R., 1986, “A Concept for Manipulator Trajectory Planning”, In: Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, USA, 3, pp. 1399-1405.

Sciavicco, L., Siciliano, B., 2000, “Modelling and Control of Robot Manipulators”, Springer, London, 2000.

Shiller, Z., 1996 “Time-Energy Optimal Control of Articulated Systems with Geometric Path Constraints”, J. of Dynamic Systems, Measurement and Control 118 , pp. 139-143.

Shin, K.G., McKay, N.D., 1985, “Minimum-Time Control of Robotic Manipulators with Geometric Path Constraints”, IEEE Trans. on Automatic Control 30 1985/6, pp. 531-541.

Slotine, J.E., Li, W., 1989, “Composite Adaptive Control of Robot Manipulators”, Automatica 25, pp. 509-519.