Trajectory planning and feedforward design for electromechanical motion systems version 2

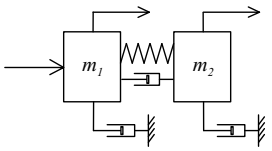
Report nr. DCT 2003-18

Paul Lambrechts
Email: P.F.Lambrechts@tue.nl

April 1, 2003

Abstract

This report considers trajectory planning with constrained dynamics and design of an appropriate feedforward controller for single axis motion control. A motivation is given for using model-based fourth order feedforward with fourth order trajectories. An algorithm is given for calculating higher order trajectories with bounds on all considered derivatives for point to point moves. It is shown that these trajectories are time-optimal in the most relevant cases. All required equations for third and fourth order trajectory planning are explicitly derived. Implementation, discretization and quantization effects are considered. Simulation results show the superior effectiveness of fourth order feedforward in comparison with rigid-body feedforward.
This report differs from DCT 2003-08 in that it contains an improved algorithm for fourth order trajectory planning (total absence of iteration loops).



Contents

1 Introduction **3**

2 Rigid-body feedforward **6**

3 Higher order feedforward **8**

4 Higher order trajectory planning **10**

5 Third order trajectory planning **13**

6 Fourth order trajectory planning **17**
6.1 Main algorithm 17
6.2 Solution of equation 49 23

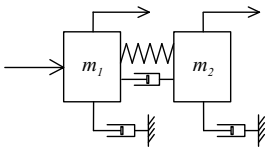
7 Implementation aspects **25**
7.1 Switching times 25
7.2 Synchronization of profiles 26
7.3 Implementation of first order filter 28
7.4 Calculation of reference trajectory 29

8 Simulation results **30**

9 Conclusions **34**

A List of symbols **37**

B The Matlab function MAKE4.m **38**



1 Introduction

Feedforward control is a well known technique for high performance motion control problems as found in industry. It is, for instance, widely applied in robots, pick-and-place units and positioning systems. These systems are often embedded in a factory automation scheme, which provides desired motion tasks to the considered system. Such a motion task can be to perform a motion from a position A to a position B, starting at a time t.

Usually this task is transferred to computer hardware dedicated to the control of the system, leaving the details of planning and execution of the motion to this dedicated motion control computer. The tasks of this dedicated motion controller will then consist of:

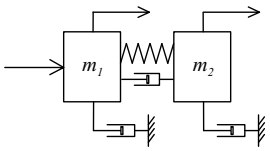
- trajectory planning: the determination of an allowable trajectory for all degrees of freedom, separately or as a multidimensional trajectory, and the calculation of an allowable trajectory for each actuation device;
- feedforward control: the representation of the desired trajectory in appropriate form and the calculation of a feedforward control signal for each actuation device, with the intention to obtain the desired trajectory;
- system compensation: to reduce or remove unwanted behavior like known or measured disturbances and non-linearities;
- feedback control: the processing of available measurements and calculation of a feedback control signal for each actuation device to compensate for unknown disturbances and unmodelled behavior,
- internal checks, diagnostics, safety issues, communication, etc.

To simplify the tasks of the motion controller, the trajectory planning and feedforward control are usually done for each actuating device separately, relying on system compensation and feedback control to deal with interactions and non-linearities. In that case, each actuating device is considered to be acting on a simple object, usually a single mass, moving along a single degree of freedom. The feedforward control problem is then to generate the force required to perform the acceleration of the mass in accordance with the desired trajectory.

Conversely, the desired trajectory should be such that the required force is allowable (in the sense of mechanical load on the system) and can be generated by the actuating device. For obvious reasons this approach is often referred to as ‘mass feedforward’ or ‘rigid-body feedforward’. It allows a simple and practical implementation of both trajectory planning and feedforward control, as the required calculations are straightforward.

The disadvantage of this approach is its dependence on system compensation and feedback control to deal with unmodelled behavior as mentioned before. The resulting problem formulation can be split in two.

1. During execution of the trajectory the position errors are large, such that feedback control actions are considerable. Actual velocity and acceleration (hence: actuator force) may therefore be much larger than planned. This may lead to undesired and even dangerous deviations from the planned trajectory and damage to actuator and system.



2. When arriving at the desired endpoint, the positioning error is large and the dynamical state of the controlled system is not settled. Although the trajectory has finished, it is often necessary to wait for a considerable time before the position error is settled within some given accuracy bounds before subsequent actions or motions are allowed. A practical consequence is the need for of a complex test to determine whether settling has sufficiently occurred. Furthermore, it is a source of time uncertainty that may be undesirable on the factory automation level.

To improve on this, many academic and practical approaches are possible. These can roughly be categorized in three.

1. Trajectory smoothing or shaping: This can be done by simply reducing the acceleration and velocity bounds used for trajectory planning, but also by smoothing or shaping the trajectory and/or application of force (higher order trajectories, S-curves, input shaping, filtering). The result of this can be very good, especially if the dynamical behavior of the motion system is explicitly taken into account. However, it may also lead to a considerable increase in execution time of the trajectory, often without a clear mechanism for finding a time optimal solution. Various examples of this approach can be found in [4, 7, 8, 9, 13, 6].

2. Feedforward control based on plant inversion: This attempts to take the effect of unmodelled behavior into account by either using a more detailed model of the motion system or by learning its behavior based on measurements. An important practical disadvantage is that they do not provide an approach for designing an appropriate trajectory. Various examples of this approach can be found in [3, 5, 10, 11, 12, 15, 16, 17].

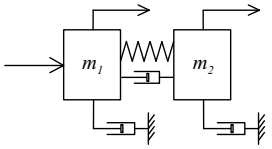
3. Feedback control optimization (possibly aided by system compensation improvement): By improving the feedback controller, the positioning errors can be kept smaller during and at the end of the trajectory. Furthermore, settling will occur in a shorter time. Also in this case the design of an appropriate trajectory is not considered. Obviously, any feedback control design method can be used for this. Some references given above also include a discussion on the effect of feedback control on trajectory following; e.g. see [11, 12, 17].

This report will provide a method for higher order trajectory planning that can be used with all of the approaches given above. It is attempted to give a better understanding of the effect of smoothing, especially when considering a more optimal balancing between time optimality, physical bounds (actuator device and motion system limits) and accuracy.

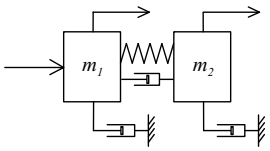
'Fourth order feedforward' will be presented as a clear and well implementable extension of rigid-body feedforward, with also a clear strategy for obtaining the aforementioned optimal balance. The implementation of a fourth order trajectory planner will be set up as a natural extension of second and third order planners to show the potential for practical application. Furthermore, the calculation of the optimal feedforward control signal will be shown.

Next the effect of discrete time implementation will be considered: this includes the planning of a fourth order trajectory in discrete time and the optimal compensation of time delays in the feedforward control signal. Finally, some simulation results are given to further motivate the use of fourth order feedforward.

The next chapter will review rigid-body feedforward, mostly with the purpose of introducing some notation. Next, chapter 3 will consider the extension of mass feedforward to fourth order feedforward, based on an extended model of the motion system. A general algorithm for higher order



trajectory planning will be considered in chapter 4, after which the relevant equations are derived for third order trajectory planning in chapter 5 and for fourth order trajectory planning in chapter 6. Next, implementation aspects are considered in chapter 7, followed by some simulation results in chapter 8, and conclusions in chapter 9.



2 Rigid-body feedforward

The specifics of planning a trajectory and calculating a feedforward signal based on rigid-body feedforward are fairly simple and can be found in many commercially available electromechanical motion control systems. In this chapter a short review is given as an introduction to a standardized approach to third and fourth order feedforward calculations.

Consider the configuration of figure 1 with m denoting the mass of the motion system, F the force supplied by the actuating device, x the position and k a viscous damping term. The

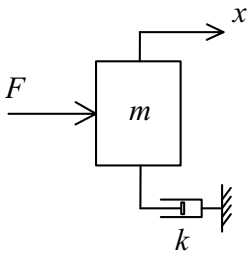


Figure 1: Simple motion system: a single mass.

equation of motion for this simple configuration is of course

$$m\ddot{x} + k\dot{x} = F \tag{1}$$

Now suppose we have a given bound on acceleration $\bar{a}$ (i.e. a bound on F), and we want to perform a motion from the current position A to a position B over a distance we will denote as $\bar{x}$. Then the shortest time within which the motion can be performed is calculated as:

$$\bar{x} = 2 \times \frac{1}{2}\bar{a}t^2 \;\; \Rightarrow \;\; t_{\bar{a}} = \sqrt{\frac{\bar{x}}{\bar{a}}} \;\; \Rightarrow \;\; t_{\bar{x}} = 2t_{\bar{a}} \tag{2}$$

with $t_{\bar{a}}$ denoting the constant acceleration phase duration and $t_{\bar{x}}$ denoting the total trajectory execution time. Hence, this gives rise to a trajectory consisting of a constant maximal acceleration phase followed directly by a constant maximal deceleration phase. Clearly if a bound on velocity, denoted as $\bar{v}$, is taken into account, $t_{\bar{x}}$ can only become larger. We can test whether the velocity bound $\bar{v}$ is violated by calculating the maximal velocity obtained using the minimal time trajectory:

$$\hat{v} := \bar{a} \cdot t_{\bar{a}} \tag{3}$$

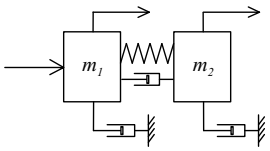
Now if $\hat{v} <= \bar{v}$ we are finished: $t_{\bar{x}} = 2t_{\bar{a}}$ and no constant velocity phase is required. If $\hat{v} > \bar{v}$ we calculate:

$$t_{\bar{a}} = \frac{\bar{v}}{\bar{a}} \;\; \Rightarrow \;\; x_{\bar{a}} := 2 \times \frac{1}{2}\bar{a}t_{\bar{a}}^2 < \bar{x} \tag{4}$$

From this, the constant velocity phase duration $t_{\bar{v}}$ is calculated:

$$t_{\bar{v}} = \frac{(\bar{x} - x_{\bar{a}})}{\bar{v}} \tag{5}$$

We now have: $t_{\bar{x}} = 2t_{\bar{a}} + t_{\bar{v}}$ and $\bar{x} = \bar{a}t_{\bar{a}}^2 + \bar{v}t_{\bar{v}}$. Note that $t_{\bar{v}}$ automatically reverts to zero if the velocity bound is not obtained.



Construction of the acceleration profile a from $t_{\bar{a}}$ and $t_{\bar{v}}$ is straightforward. From this, the desired trajectory can be determined by integrating it once to obtain the velocity profile v, and integrating it twice to obtain the position profile x; see figure 2. As the position profile thus establishes

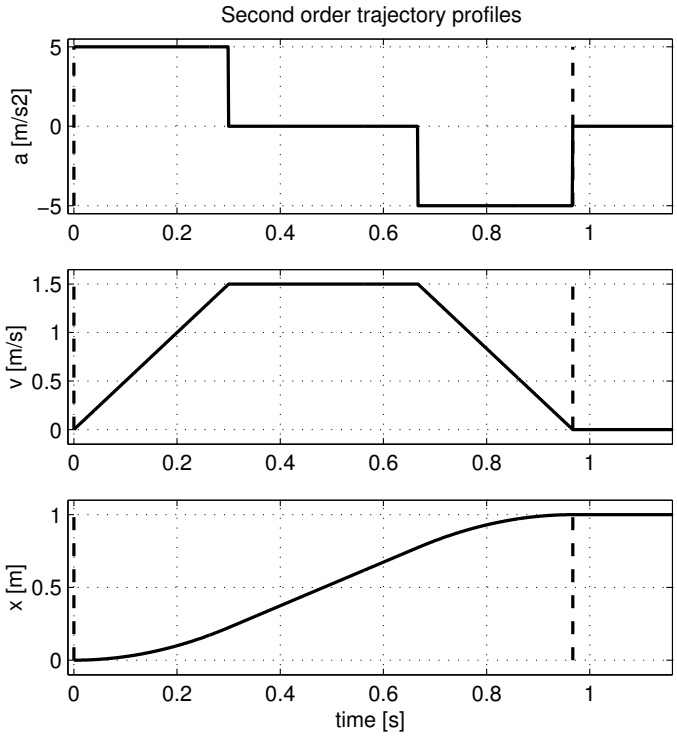


Figure 2: Second order trajectory determination.

the trajectory as a sequence of polynomials in time with a degree of at most two, rigid-body feedforward is also referred to as 'second order feedforward'.

The actual implementation of the trajectory planner and feedforward controller is indicated in figure 3. Note that the feedforward force F is simply calculated from equation 1 and the profiles

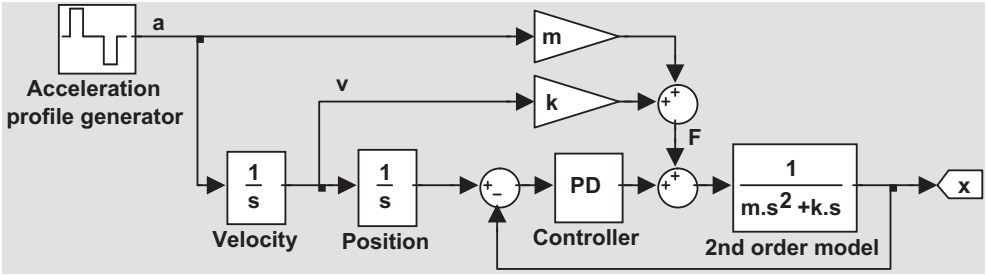
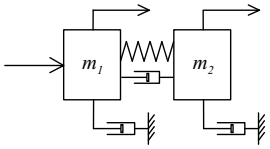


Figure 3: Rigid-body feedforward implementation.

in figure 2 as:

$$F = ma + kv \tag{6}$$



3 Higher order feedforward

The previous chapter shows that mass feedforward is based on a simple single mass model of the motion system. This implies that the performance of rigid-body feedforward is determined by how much the actual motion system deviates from this single mass model.

On the other hand, the performance of the motion system as a whole is also determined by the quality of system compensation and/or feedback control. It can be stated that the success of feedback control is such that in many cases the mass feedforward approach is considered sufficient, given that an appropriate feedback controller is required anyway for disturbance reduction and stabilization.

However, when considering further improvement of motion control system performance, the use of higher order feedforward is often a very effective approach in comparison with improved feedback control. The first effect is that higher order trajectories inherently have a lower energy content at higher frequencies, which results in a lower high frequency content of the error signal, which in turn enables the feedback controller to be more effective. The second effect is that higher order trajectories have less chance of demanding a motion which is physically impossible to perform by the given motion system: e.g. most power amplifiers exhibit a 'rise time' effect, such that it is impossible to produce a step-like change in force.

These effects are commonly referred to as 'smoothing' and result in a decrease of positioning errors during execution of the trajectory and a reduced settling time. The disadvantage of higher order trajectories, i.e. the increase in trajectory execution time, is usually more than compensated by the reduced settling time. Because of this, many high performance motion systems are already equipped with a third order trajectory planner as a direct extension of mass feedforward. In this chapter it will be determined that a fourth order trajectory planner and feedforward calculation gives a significant further improvement.

The main argument for this is that an electromechanical motion control system will usually have some compliance between actuator and load, and that both actuator and load will have a relevant mass. For this reason it is natural to extend the single mass model of figure 1 to the double mass model of figure 4. Here m_1 denotes the mass of the actuator, m_2 the mass of the load, F the force supplied by the actuating device, x_1 the actuator position, x_2 the load position, c the stiffness between the two masses, k_{12} the viscous damping between the two masses, k_1 the viscous damping of the actuator towards ground and k_2 the viscous damping of the load towards ground. The equations of motion for this configuration are:

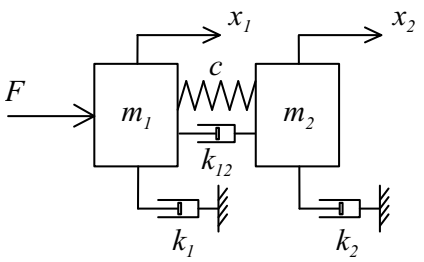
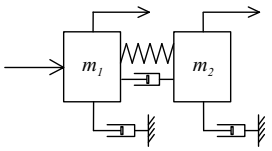


Figure 4: Extended motion system: double mass.



$$\begin{cases} m_1\ddot{x}_1 = -k_1\dot{x}_1 - c(x_1 - x_2) - k_{12}(\dot{x}_1 - \dot{x}_2) + F \\ m_2\ddot{x}_2 = -k_2\dot{x}_2 + c(x_1 - x_2) + k_{12}(\dot{x}_1 - \dot{x}_2) \end{cases} \tag{7}$$

Laplace transformation and substitution then results in the following expression:

$$F = \frac{q_1 s^4 + q_2 s^3 + q_3 s^2 + q_4 s}{k_{12} s + c} \cdot x_2 \tag{8}$$

with:

$$\begin{cases} q_1 = m_1 m_2 \\ q_2 = (m_1 + m_2)k_{12} + m_1 k_2 + m_2 k_1 \\ q_3 = (m_1 + m_2)c + k_1 k_2 + (k_1 + k_2)k_{12} \\ q_4 = (k_1 + k_2)c \end{cases} \tag{9}$$

This implies that if we have planned some fourth order trajectory for x_2, from which we can derive the corresponding profiles for velocity v, acceleration a, jerk $\jmath$ and derivative of jerk d, the feedforward force F can be calculated as:

$$F = \frac{1}{k_{12} s + c} \cdot \{q_1 d + q_2 \jmath + q_3 a + q_4 v\} \tag{10}$$

Analogous to the implementation given in figure 3, this feedforward scheme can readily be implemented as given in figure 5. Note that it is convenient to specify the trajectory by means of the

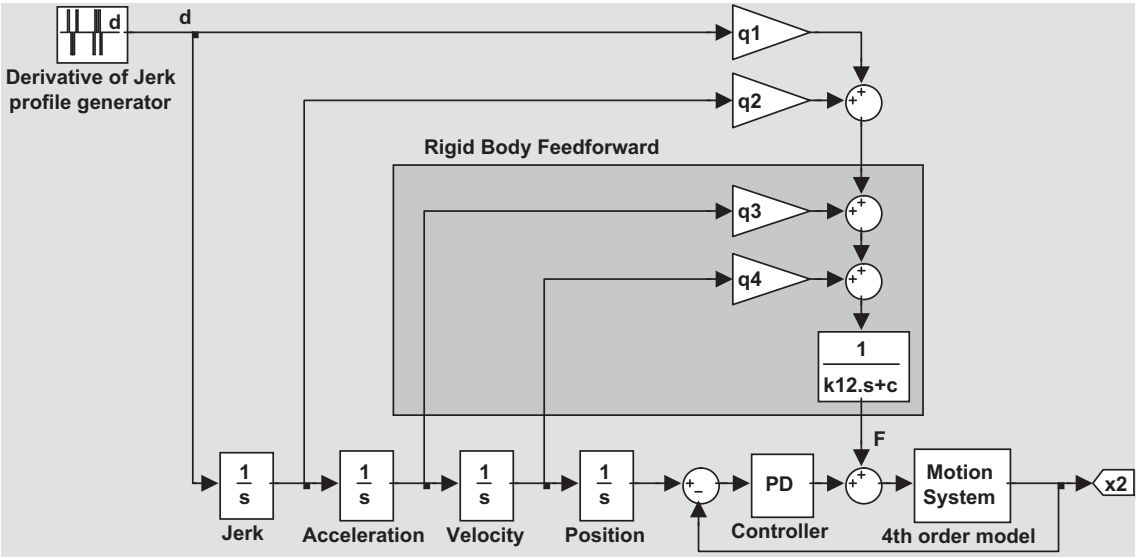
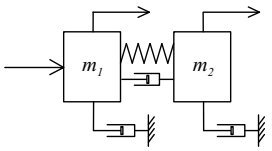


Figure 5: Fourth order feedforward implementation.

derivative of jerk profile d, such that all other profiles can easily be obtained by integration. An algorithm for obtaining d will be the subject of the next chapters.



4 Higher order trajectory planning

Planners for second and third order trajectories are fairly well known in industry and academia and there are many approaches for obtaining a valid solution. Extension to fourth order trajectory planning is however not trivial. In this chapter we will review the main objectives of trajectory planning in general and set up an algorithm for obtaining these objectives more or less irrespective of the order of the resulting trajectory.

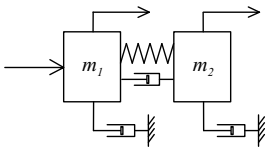
As mentioned before, a trajectory is usually planned for a motion from the current position A to some desired end position B under some boundary constraints. This will also be the basis for the discussion given here, although the resulting algorithm can be adjusted for relevant alternative objectives like for instance speed control instead of position control, scanning motions or speed change operations.

Several further objectives that are of importance for applicability of a trajectory planning algorithm are given below.

- **Timing.** In principle we should strive for time optimality; at least it must be clear what the consequences are for the trajectory execution time when considering the effect of boundary conditions.
- **Actuator effort.** This is usually the basis for the selection of bounds for velocity, acceleration and jerk, although they may also be related to bounds on mechanics or safety issues.
- **Accuracy.** For point to point moves, the planned end position of the trajectory must be equal to the desired end position (within measurement accuracy).
- **Complexity.** Usually trajectory planning is thought of as being done off-line. In practice however, the desired end position often becomes available at the moment the trajectory should start. Hence, the time required for planning the trajectory is lost and should be minimized.
- **Reliability.** The planning algorithm should always come up with a valid solution.
- **Implementation.** Trajectory planning is done in computer hardware, and is therefore subject to discretization and digitization.

Now we argue that these objectives, except the final one, are all met by the simple algorithm given in chapter 2:

1. calculate $t_{\bar{a}} = \sqrt{\frac{\bar{x}}{\bar{a}}}$ (equation 2),
2. calculate $\hat{v} = \bar{a} \cdot t_{\bar{a}}$ (equation 3),
3. test $\hat{v} > \bar{v}$:
   - if true recalculate $t_{\bar{a}} = \frac{\bar{v}}{\bar{a}}$ (equation 4),
   - if false: no action,
4. calculate $x_{\bar{a}} = \bar{a} t_{\bar{a}}^2$ (equation 4),
5. calculate $t_{\bar{v}} = \frac{(\bar{x} - x_{\bar{a}})}{\bar{v}}$ (equation 5), and

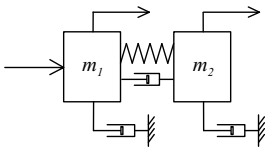


6. finished: $\bar{x} = \bar{a}t_{\bar{a}}^2 + \bar{v}t_{\bar{v}}$ and $t_{\bar{x}} = 2t_{\bar{a}} + t_{\bar{v}}$.

First consider timing; obviously timing is minimal if the bound on v is discarded as is done in step 1. If introducing this bound leads to a reduction of $t_{\bar{a}}$ (verify that equation 4 always leads to a reduction) the result will be that $v = \bar{v}$ is obtained in the shortest possible time, and is continued for as long as possible. Hence the trajectory execution time is always minimal, given the bounds $\bar{a}$ and $\bar{v}$. Next, actuator effort is immediately taken into account by means of the given bounds, accuracy is precise, complexity is low and reliability is high (no iteration loops that can hang up, algorithm only fails if a non-positive bound is specified). The implementation problem is considered later.

Given the advantages of this algorithm, it is desirable to generalize it for planning higher order trajectories. The position displacement $\bar{x}$ and bounds on all derivatives of the trajectory up to the highest order derivative d (indicated as $\bar{d}$) are assumed to be given. Furthermore we will require that all derivatives are equal to zero at the start and end positions.

1. Determine a symmetrical trajectory that has d equal to either $\bar{d}$ or $-\bar{d}$ at all times and obtains displacement $\bar{x}$.
2. Determine $t_{\bar{d}}$: the shortest time that d remains constant (always the first period).
3. Calculate the maximal value of velocity $\hat{v}$ obtained during this trajectory.
4. Test $\hat{v} > \bar{v}$:
   - if true re-calculate $t_{\bar{d}}$ based on $\bar{v}$ ($t_{\bar{d}}$ decreases),
   - if false continue.
5. Calculate the maximal value of acceleration $\hat{a}$ obtained during this trajectory.
6. Test $\hat{a} > \bar{a}$:
   - if true re-calculate $t_{\bar{d}}$ based on $\bar{a}$ ($t_{\bar{d}}$ decreases),
   - if false continue.
7. Continue these tests and possible recalculations until d; the last test is performed on $\jmath$, defined as the highest derivative before d. The resulting $t_{\bar{d}}$ will not be changed anymore.
8. Extend the trajectory symmetrically with periods of constant $\jmath$ whenever $\jmath$ reaches the value $\bar{\jmath}$ or $-\bar{\jmath}$: this must be done such that the required displacement $\bar{x}$ is obtained.
9. Determine $t_{\bar{\jmath}}$: the shortest time that $\jmath$ remains constant.
10. Starting with velocity and ending with the derivative before $\jmath$, calculate the maximal value and re-calculate $t_{\bar{\jmath}}$ if the appropriate bound is violated (each re-calculation can only decrease $t_{\bar{\jmath}}$).
11. The resulting $t_{\bar{\jmath}}$ will not be changed anymore.
12. Extend the trajectory symmetrically with periods of constant next lower derivative; again such that the required displacement $\bar{x}$ is obtained. Determine the associated time interval and do the tests and recalculations resulting in a final value for it.

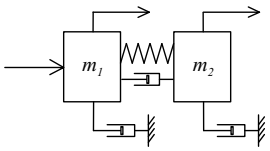


13. Continue until v: the constant speed phase duration $t_{\bar{v}}$ is calculated such that the required displacement $\bar{x}$ is obtained (i.e. the displacement 'left to go' divided by $\bar{v}$).

14. Finished: $t_{\bar{d}}$, $t_{\bar{j}}$, etc. until $t_{\bar{v}}$ completely determine the trajectory.

The properties of the resulting trajectories (in the sense of the objectives given above) appear to depend partly on the order of the trajectory planner. Obviously, the required calculations become more complex with increasing order. Time-optimality is still automatically obtained with third order trajectories, but not necessarily for fourth order trajectories (this can be obtained but costs much complexity at a small gain with respect to a good sub-optimal solution). In principle, higher than fourth order trajectories can also be planned by means of this algorithm, but this is considered impractical due to the large increase in complexity. It is noted here that all calculations for third and fourth order planning can be done analytically with fairly basic mathematical functions, except one calculation for fourth order planning. For this case a simple and numerically stable search algorithm can be used.

The next chapter will provide the calculations for third order trajectory planning; after that fourth order trajectory planning will be considered.



5 Third order trajectory planning

Although chapter 3 shows that fourth order trajectory planning and feedforward is strongly motivated by a model based approach, it may still be useful to apply third order trajectory planning. This chapter will provide the derivations and calculations necessary to 'fill in' the algorithm developed in chapter 4 for third order trajectory planning.

We will assume that a trajectory must be planned over a distance $\bar{x}$, that bounds are given on velocity ($\bar{v}$), acceleration ($\bar{a}$) and jerk ($\bar{\jmath}$), and that the trajectory must be time optimal within these bounds. The bound on jerk can for instance be related to 'rise time' behavior as commonly found in electromechanical actuation systems with a non-ideal power amplifier for generating the actuation force. This results in a bound on the maximal current change per second, which translates to a maximal actuation force change per second and a maximal acceleration change per second: hence a bound on jerk.

The trajectory planning algorithm will be based on the construction of an appropriate jerk profile (instead of the acceleration profile as in rigid-body feedforward) that can be integrated three times to obtain the third order position trajectory. A symmetrical trajectory is completely determined by three time intervals: the constant jerk interval $t_{\bar{\jmath}}$, the constant acceleration interval $t_{\bar{a}}$ and the constant velocity interval $t_{\bar{v}}$. The resulting profiles are given in figure 6. Including the starting time of the trajectory at t_0, there are eight time instances at which the jerk changes. The following relations are clear:

$$\begin{array}{rcl} t_{\bar{\jmath}} & = & t_1 - t_0 = t_3 - t_2 = t_5 - t_4 = t_7 - t_6 \\ t_{\bar{a}} & = & t_2 - t_1 = t_6 - t_5 \\ t_{\bar{v}} & = & t_4 - t_3 \end{array} \tag{11}$$

For step 1 of the algorithm, we will discard the bounds on acceleration and velocity, and only consider the bound on jerk. This implies $t_{\bar{v}} = 0$ and $t_{\bar{a}} = 0$ and it is clear that this will provide us with a lower bound for the trajectory execution time $t_{\bar{x}} = 4 \times t_{\bar{\jmath}}$.

To obtain $t_{\bar{\jmath}}$ (step 2), we need the relation between $t_{\bar{\jmath}}$ and $\bar{x}$ with given $\bar{\jmath}$. For this we make use of the constant value of jerk of $+\bar{\jmath}$ or $-\bar{\jmath}$ during each interval. If we set the time instance of jerk change to zero, it is easily verified that for any time t during the constant jerk interval the third order profiles for acceleration, velocity and position can be expressed as follows:

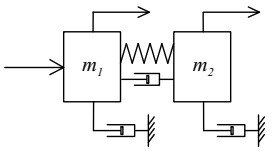
$$\begin{array}{rcccccl} a(t) & = & \jmath_0 t & + & a_0 & & \\ v(t) & = & \frac{1}{2}\jmath_0 t^2 & + & a_0 t & + & v_0 \\ x(t) & = & \frac{1}{6}\jmath_0 t^3 & + & \frac{1}{2}a_0 t^2 & + & v_0 t + x_0 \end{array} \tag{12}$$

Because we assume that the bounds on acceleration and velocity are not violated, we have $t_{\bar{a}} = 0$ and $t_{\bar{v}} = 0$, and consequently from figure 6: $t_2 = t_1 = t_{\bar{\jmath}}$ and $t_4 = t_3$. Hence:

$$\begin{array}{rcl} a(t_2) = a(t_1) & = & \bar{\jmath} t_{\bar{\jmath}} \\ v(t_2) = v(t_1) & = & \frac{1}{2}\bar{\jmath} t_{\bar{\jmath}}^2 \\ x(t_2) = x(t_1) & = & \frac{1}{6}\bar{\jmath} t_{\bar{\jmath}}^3 \end{array} \tag{13}$$

and for the next period in which $\jmath = -\bar{\jmath}$:

$$\begin{array}{rcl} a(t_4) = a(t_3) & = & -\bar{\jmath} t_{\bar{\jmath}} + a(t_2) \\ v(t_4) = v(t_3) & = & -\frac{1}{2}\bar{\jmath} t_{\bar{\jmath}}^2 + a(t_2) t_{\bar{\jmath}} + v(t_2) \\ x(t_4) = x(t_3) & = & -\frac{1}{6}\bar{\jmath} t_{\bar{\jmath}}^3 + a(t_2) t_{\bar{\jmath}}^2 + v(t_2) t_{\bar{\jmath}} + x(t_2) \end{array} \tag{14}$$



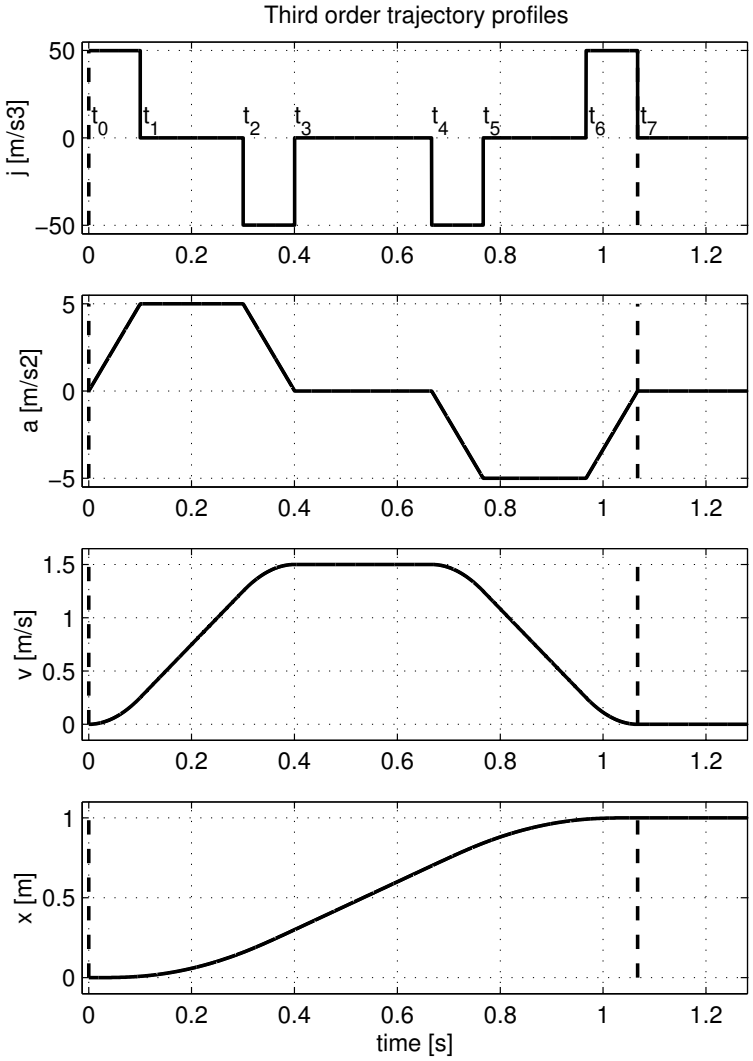


Figure 6: Third order trajectory planning.

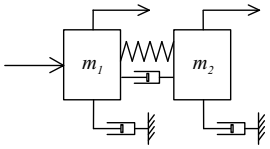
Substitution gives:

$$\begin{aligned} a(t_4) = a(t_3) &= 0 \\ v(t_4) = v(t_3) &= \bar{\jmath} t_{\bar{\jmath}}^2 \\ x(t_4) = x(t_3) &= \bar{\jmath} t_{\bar{\jmath}}^3 \end{aligned} \tag{15}$$

and due to symmetry of the profile: $x(t_7) = 2x(t_3) = 2\bar{\jmath} t_{\bar{\jmath}}^3$. From this we can calculate the minimal required time to execute the trajectory for a distance $\bar{x}$, given the bound on jerk $\bar{\jmath}$ as:

$$t_{\bar{x}} = 4t_{\bar{\jmath}} = 4 \cdot \sqrt[3]{\frac{\bar{x}}{2\bar{\jmath}}} \tag{16}$$

As mentioned before, the result of equation 16 does not take into account whether or not the bounds on acceleration and/or velocity are violated. However, if this would be the case, it is obvious that $t_{\bar{x}}$ must increase, but also that $t_{\bar{\jmath}}$ must decrease.



Step 3 of the algorithm follows from equation 15; the maximum velocity occurs at t_3, such that:

$$\hat{v} = \bar{\jmath} t_{\bar{\jmath}}^2 \tag{17}$$

If $\hat{v} > \bar{v}$ the bound is violated and we must recalculate $t_{\bar{\jmath}}$ as follows:

$$t_{\bar{\jmath}} = \sqrt{\frac{\bar{v}}{\bar{\jmath}}} \tag{18}$$

Note that the resulting $t_{\bar{\jmath}}$ will always be smaller than the result from equation 16, and that consequently also the acceleration (from equation 13) will become smaller. This establishes step 4 of the algorithm.

Step 5 follows from equation 13; the maximum acceleration occurs at t_1, such that:

$$\hat{a} = \bar{\jmath} t_{\bar{\jmath}} \tag{19}$$

If $\hat{a} > \bar{a}$ the bound is violated and we must recalculate $t_{\bar{\jmath}}$ as follows:

$$t_{\bar{\jmath}} = \frac{\bar{a}}{\bar{\jmath}} \tag{20}$$

Again $t_{\bar{\jmath}}$ can only be smaller than previously calculated, such that we now have the guarantee that $t_{\bar{\jmath}}$ is the maximal value for which none of the bounds is violated. At the same time we have that a maximal value of $t_{\bar{\jmath}}$ will lead to a minimal value of $t_{\bar{x}}$ and will therefore be (part of) the time-optimal solution. This establishes step 6 of the algorithm and because a is the highest derivative before $\jmath$, also step 7.

The next thing to do is to calculate $t_{\bar{a}}$: also this value should be maximal for time-optimal performance, while at the same time it must be such that no bounds are violated. To do the necessary calculations, assume that $t_{\bar{a}} > 0$, such that we now have $t_2 > t_1$ (from figure 6). Equation 13 can then be extended with the help of equation 12 and the fact that $\jmath = 0$ during the period from t_1 to t_2:

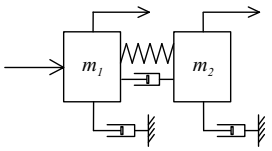
$$\begin{aligned} a(t_2) &= a(t_1) + 0 t_{\bar{a}} = \bar{\jmath} t_{\bar{\jmath}} \\ v(t_2) &= v(t_1) + a(t_1) t_{\bar{a}} = \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}}^2 + \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}} \\ x(t_2) &= x(t_1) + v(t_1) t_{\bar{a}} + \tfrac{1}{2} a(t_1) t_{\bar{a}}^2 \\ &= \tfrac{1}{6} \bar{\jmath} t_{\bar{\jmath}}^3 + \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}}^2 t_{\bar{a}} + \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}}^2 \end{aligned} \tag{21}$$

Again using equation 12 we can then calculate:

$$\begin{aligned} a(t_3) &= a(t_2) - \bar{\jmath} t_{\bar{\jmath}} = 0 \\ v(t_3) &= v(t_2) + a(t_2) t_{\bar{\jmath}} - \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}}^2 \\ &= \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}}^2 + \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}} + \bar{\jmath} t_{\bar{\jmath}}^2 - \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}}^2 \\ &= \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}} + \bar{\jmath} t_{\bar{\jmath}}^2 \\ x(t_3) &= x(t_2) + v(t_2) t_{\bar{\jmath}} + \tfrac{1}{2} a(t_2) t_{\bar{\jmath}}^2 - \tfrac{1}{6} \bar{\jmath} t_{\bar{\jmath}}^3 \\ &= \bar{\jmath} t_{\bar{\jmath}}^3 + \tfrac{3}{2} \bar{\jmath} t_{\bar{\jmath}}^2 t_{\bar{a}} + \tfrac{1}{2} \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}}^2 \end{aligned} \tag{22}$$

Now we assume that the velocity bound is not violated, such that $t_4 = t_3$. Then due to symmetry of the profile we have:

$$x(t_7) = 2x(t_3) = 2\bar{\jmath} t_{\bar{\jmath}}^3 + 3\bar{\jmath} t_{\bar{\jmath}}^2 t_{\bar{a}} + \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}}^2 \tag{23}$$



By setting $\bar{x} = x(t_7)$ we can then solve $t_{\bar{a}}$ from the following:

$$\begin{aligned} (\bar{\jmath} t_{\bar{\jmath}}) \cdot t_{\bar{a}}^2 + (3\bar{\jmath} t_{\bar{\jmath}}^2) \cdot t_{\bar{a}} + (2\bar{\jmath} t_{\bar{\jmath}}^3 - \bar{x}) &= 0 \\ t_{\bar{a}}^2 + (3t_{\bar{\jmath}}) \cdot t_{\bar{a}} + (2t_{\bar{\jmath}}^2 - \tfrac{\bar{x}}{\bar{\jmath} t_{\bar{\jmath}}}) &= 0 \end{aligned} \tag{24}$$

As $t_{\bar{a}}$ must be positive to make sense, the solution is:

$$\begin{aligned} t_{\bar{a}} &= -1\tfrac{1}{2} t_{\bar{\jmath}} + \tfrac{1}{2}\sqrt{9t_{\bar{\jmath}}^2 - 8t_{\bar{\jmath}}^2 + \tfrac{4\bar{x}}{\bar{\jmath} t_{\bar{\jmath}}}} \\ &= -1\tfrac{1}{2} t_{\bar{\jmath}} + \tfrac{1}{2}\sqrt{t_{\bar{\jmath}}^2 + \tfrac{4\bar{x}}{\bar{\jmath} t_{\bar{\jmath}}}} \end{aligned} \tag{25}$$

Note that $t_{\bar{a}} >= 0$ follows immediately from $\bar{x} >= 2\bar{\jmath} t_{\bar{\jmath}}^3$. Hence, we now have determined $t_{\bar{a}}$ under the assumption that the velocity bound will not be violated: equation 25 establishes steps 8 and 9 of the algorithm.

Step 10 introduces the velocity bound again; the maximal velocity is obtained at t_3 (see 22):

$$\hat{v} = v(t_3) = \bar{\jmath} t_{\bar{\jmath}}^2 + \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}} \tag{26}$$

If $\hat{v} > \bar{v}$ the bound is violated and we can recalculate $t_{\bar{a}}$ as follows:

$$t_{\bar{a}} = \frac{\bar{v} - \bar{\jmath} t_{\bar{\jmath}}^2}{\bar{\jmath} t_{\bar{\jmath}}} = \frac{\bar{v}}{\bar{\jmath} t_{\bar{\jmath}}} - t_{\bar{\jmath}} = \frac{\bar{v}}{\bar{a}} - t_{\bar{\jmath}} \tag{27}$$

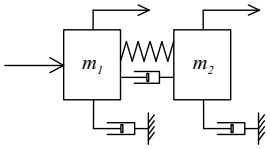
Now we have determined $t_{\bar{\jmath}}$ and $t_{\bar{a}}$ to be the time-optimal solution under the restriction of the given bounds (step 11).

As the next lower derivative is already v, we can skip step 12 and continue with step 13: the determination of the constant velocity time interval $t_{\bar{v}}$ such that the required total displacement $\bar{x}$ is obtained. From the previous calculations follows that $t_{\bar{v}} = 0$ if $t_{\bar{a}}$ is according to equation 25. But if $t_{\bar{a}}$ is reduced according to equation 27, we will have $x(t_7) < \bar{x}$ and we must add a constant velocity phase to the trajectory such that $x(t_4) - x(t_3) = \bar{x} - 2x(t_3)$. With equation 22 this implies that we can calculate $t_{\bar{v}}$ as:

$$t_{\bar{v}} = \frac{\bar{x} - 2\bar{\jmath} t_{\bar{\jmath}}^3 - 3\bar{\jmath} t_{\bar{\jmath}}^2 t_{\bar{a}} - \bar{\jmath} t_{\bar{\jmath}} t_{\bar{a}}^2}{\bar{v}} \tag{28}$$

This completes the calculation of the characteristics of the time-optimal third order profile for a given distance $\bar{x}$ and given bounds $\bar{v}$, $\bar{a}$ and $\bar{\jmath}$. The total displacement $\bar{x}$ can be expressed as a function of $\bar{\jmath}$ and the times $t_{\bar{\jmath}}$, $t_{\bar{a}}$ and $t_{\bar{v}}$:

$$\begin{aligned} \bar{x} &= \bar{\jmath}\left(2t_{\bar{\jmath}}^3 + 3t_{\bar{\jmath}}^2 t_{\bar{a}} + t_{\bar{\jmath}} t_{\bar{a}}^2\right) + v(t_3) t_{\bar{v}} \\ &= \bar{\jmath}\left(2t_{\bar{\jmath}}^3 + 3t_{\bar{\jmath}}^2 t_{\bar{a}} + t_{\bar{\jmath}} t_{\bar{a}}^2 + t_{\bar{\jmath}}^2 t_{\bar{v}} + t_{\bar{\jmath}} t_{\bar{a}} t_{\bar{v}}\right) \end{aligned} \tag{29}$$



6 Fourth order trajectory planning

This chapter will provide the derivations and calculations necessary to 'fill in' the algorithm developed in chapter 4 for fourth order trajectory planning. First the main algorithm will be derived, analogous to the third order algorithm in the previous chapter. Section 6.2 will then give the derivation of the solution of the third order polynomial equation that will appear in one of the steps of this main algorithm.

6.1 Main algorithm

Again we will assume that a symmetrical trajectory must be planned for a point to point move over a distance $\bar{x}$. We have bounds on velocity ($\bar{v}$), acceleration ($\bar{a}$), jerk ($\bar{j}$) and derivative of jerk ($\bar{d}$). The trajectory planning algorithm will now be based on the construction of a derivative of jerk profile that can be integrated four times to obtain the fourth order position trajectory. A symmetrical trajectory is completely determined by four time intervals: the constant derivative of jerk interval $t_{\bar{d}}$, the constant jerk interval $t_{\bar{j}}$, the constant acceleration interval $t_{\bar{a}}$ and the constant velocity interval $t_{\bar{v}}$. The resulting profiles are given in figure 7.

Including the starting time of the trajectory at t_0, there are now sixteen time instances at which the derivative of jerk changes.

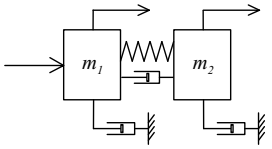
For step 1 of the algorithm, we will discard the bounds on jerk, acceleration and velocity, and only consider the bound on the derivative of jerk. This implies $t_{\bar{v}} = 0$, $t_{\bar{a}} = 0$ and $t_{\bar{j}} = 0$ and it is clear that this will provide us with a lower bound for the trajectory execution time $t_{\bar{x}} = 8 \times t_{\bar{d}}$.

To obtain $t_{\bar{d}}$ (step 2), we need the relation between $t_{\bar{d}}$ and $\bar{x}$ with given $\bar{d}$. For this we make use of the constant value of derivative of jerk of $+\bar{d}$ or $-\bar{d}$ during each interval. If we set the time instance of derivative of jerk change to 0, it is easily verified that for any time t during the constant derivative of jerk interval the fourth order profiles for acceleration, velocity and position can be expressed as follows:

$$\begin{aligned} j(t) &= d_0 t + j_0 \\ a(t) &= \tfrac{1}{2} d_0 t^2 + j_0 t + a_0 \\ v(t) &= \tfrac{1}{6} d_0 t^3 + \tfrac{1}{2} j_0 t^2 + a_0 t + v_0 \\ x(t) &= \tfrac{1}{24} d_0 t^4 + \tfrac{1}{6} j_0 t^3 + a_0 t^2 + v_0 t + x_0 \end{aligned} \tag{30}$$

Because we assume that the bounds on jerk, acceleration and velocity are not violated, we have $t_{\bar{j}} = 0$, $t_{\bar{a}} = 0$ and $t_{\bar{v}} = 0$, and consequently from figure 7: $t_2 = t_1 = t_{\bar{d}}$, $t_4 = t_3 = 2t_{\bar{d}}$, $t_6 = t_5 = 3t_{\bar{d}}$ and $t_8 = t_7 = 4t_{\bar{d}}$. Hence:

$$\begin{aligned} j(t_1) &= \bar{d} t_{\bar{d}} \\ a(t_1) &= \tfrac{1}{2} \bar{d} t_{\bar{d}}^2 \\ v(t_1) &= \tfrac{1}{6} \bar{d} t_{\bar{d}}^3 \\ x(t_1) &= \tfrac{1}{24} \bar{d} t_{\bar{d}}^4 \end{aligned} \tag{31}$$



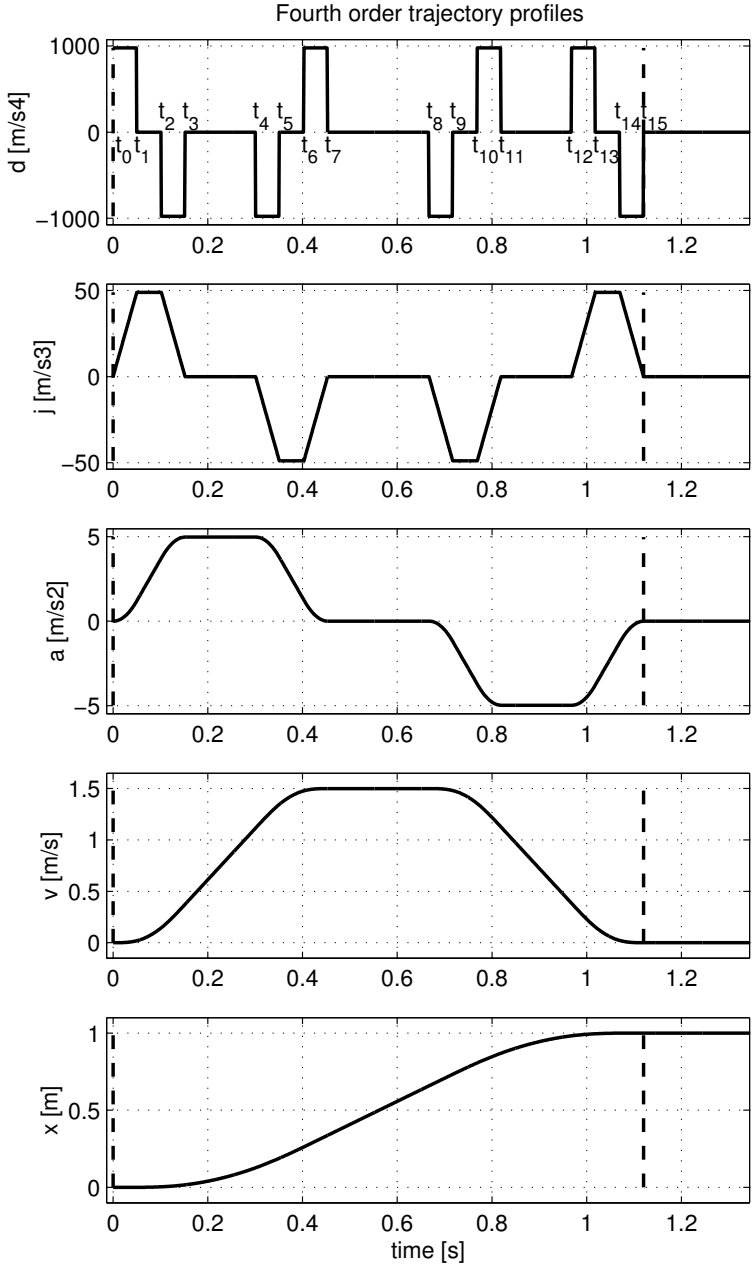
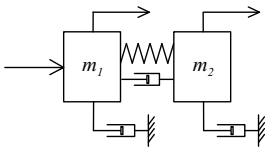


Figure 7: Fourth order trajectory planning.



and for the next period in which $d = -\bar{d}$:

$$\begin{aligned} \jmath(t_3) &= -\bar{d}t_{\bar{d}} + \jmath(t_2) \\ a(t_3) &= -\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 + \jmath(t_2)t_{\bar{d}} + a(t_2) \\ v(t_3) &= -\tfrac{1}{6}\bar{d}t_{\bar{d}}^3 + \tfrac{1}{2}\jmath(t_2)t_{\bar{d}}^2 + a(t_2)t_{\bar{d}} + v(t_2) \\ x(t_3) &= -\tfrac{1}{24}\bar{d}t_{\bar{d}}^4 + \tfrac{1}{6}\jmath(t_2)t_{\bar{d}}^3 + \tfrac{1}{2}a(t_2)t_{\bar{d}}^2 + v(t_2)t_{\bar{d}} + x(t_2) \end{aligned} \tag{32}$$

Substitution gives:

$$\begin{aligned} \jmath(t_3) &= 0 \\ a(t_3) &= \bar{d}t_{\bar{d}}^2 \\ v(t_3) &= \bar{d}t_{\bar{d}}^3 \\ x(t_3) &= \tfrac{7}{12}\bar{d}t_{\bar{d}}^4 \end{aligned} \tag{33}$$

Again using equation 30, we can calculate the results of the subsequent periods as:

$$\begin{aligned} \jmath(t_5) &= -\bar{d}t_{\bar{d}} \\ a(t_5) &= \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 \\ v(t_5) &= 1\tfrac{5}{6}\bar{d}t_{\bar{d}}^3 \\ x(t_5) &= 2\tfrac{1}{24}\bar{d}t_{\bar{d}}^4 \end{aligned} \tag{34}$$

and:

$$\begin{aligned} \jmath(t_7) &= 0 \\ a(t_7) &= 0 \\ v(t_7) &= 2\bar{d}t_{\bar{d}}^3 \\ x(t_7) &= 4\bar{d}t_{\bar{d}}^4 \end{aligned} \tag{35}$$

and due to symmetry of the profile: $x(t_{15}) = 2x(t_7) = 8\bar{d}t_{\bar{d}}^4$. From this we can calculate $t_{\bar{d}}$ as required by steps 1 and 2 of the trajectory planning algorithm as:

$$t_{\bar{d}} = \sqrt[4]{\frac{\bar{x}}{8\bar{d}}} \tag{36}$$

To continue with step 3 we have that the maximal velocity is obtained at time instance t_7. Hence, from equation 35 follows:

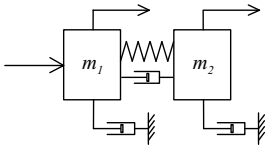
$$\hat{v} = v(t_7) = 2\bar{d}t_{\bar{d}}^3 \tag{37}$$

If $\hat{v} > \bar{v}$ we must recalculate $t_{\bar{d}}$ as:

$$t_{\bar{d}} = \sqrt[3]{\frac{\bar{v}}{2\bar{d}}} \tag{38}$$

which establishes step 4. Next, step 5 follows from the calculation of maximal acceleration obtained at time instance t_3. From equation 33 we have:

$$\hat{a} = a(t_3) = \bar{d}t_{\bar{d}}^2 \tag{39}$$



and if $\hat{a} > \bar{a}$ we must recalculate $t_{\bar{d}}$ as:

$$t_{\bar{d}} = \sqrt{\frac{\bar{a}}{\bar{d}}} \tag{40}$$

which establishes step 6. In this case we have one further test (step 7) to do on the maximal jerk obtained at t_1:

$$\hat{\jmath} = \jmath(t_1) = \bar{d}t_{\bar{d}} \tag{41}$$

and if $\hat{\jmath} > \bar{\jmath}$ we must recalculate $t_{\bar{d}}$ as:

$$t_{\bar{d}} = \frac{\bar{\jmath}}{\bar{d}} \tag{42}$$

Hence, we now have a value for $t_{\bar{d}}$ that complies with all of the bounds $\bar{\jmath}$, $\bar{a}$ and $\bar{v}$.

The next thing to do is to calculate $t_{\bar{\jmath}}$ under the assumption that bounds $\bar{a}$ and $\bar{v}$ are not violated (steps 8 and 9). To do the necessary calculations, assume that $t_{\bar{\jmath}} > 0$, such that we now have $t_2 > t_1$ and $t_6 > t_5$ (from figure 7). Equation 30 and the fact that $d = 0$ during the period from t_1 to t_2 now enables us to calculate:

$$\begin{aligned}
\jmath(t_2) &= \bar{d}t_{\bar{d}} \\
a(t_2) &= \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}} \\
v(t_2) &= \tfrac{1}{6}\bar{d}t_{\bar{d}}^3 + \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 \\
x(t_2) &= \tfrac{1}{24}\bar{d}t_{\bar{d}}^4 + \tfrac{1}{6}\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + \tfrac{1}{4}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \tfrac{1}{6}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3
\end{aligned} \tag{43}$$

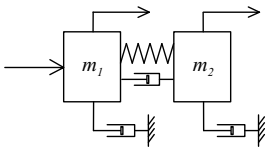
Again using equation 30 we can then calculate:

$$\begin{aligned}
\jmath(t_3) &= 0 \\
a(t_3) &= \bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}} \\
v(t_3) &= \bar{d}t_{\bar{d}}^3 + 1\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 \\
x(t_3) &= \tfrac{7}{12}\bar{d}t_{\bar{d}}^4 + 1\tfrac{1}{6}\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + \tfrac{3}{4}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \tfrac{1}{6}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3
\end{aligned} \tag{44}$$

$$\begin{aligned}
\jmath(t_5) &= -\bar{d}t_{\bar{d}} \\
a(t_5) &= \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}} \\
v(t_5) &= 1\tfrac{5}{6}\bar{d}t_{\bar{d}}^3 + 2\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 \\
x(t_5) &= 2\tfrac{1}{24}\bar{d}t_{\bar{d}}^4 + 3\tfrac{1}{6}\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 1\tfrac{1}{4}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \tfrac{1}{6}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3
\end{aligned} \tag{45}$$

$$\begin{aligned}
\jmath(t_6) &= -\bar{d}t_{\bar{d}} \\
a(t_6) &= \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 \\
v(t_6) &= 1\tfrac{5}{6}\bar{d}t_{\bar{d}}^3 + 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 \\
x(t_6) &= 2\tfrac{1}{24}\bar{d}t_{\bar{d}}^4 + 5\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 4\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3
\end{aligned} \tag{46}$$

$$\begin{aligned}
\jmath(t_7) &= 0 \\
a(t_7) &= 0 \\
v(t_7) &= 2\bar{d}t_{\bar{d}}^3 + 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 \\
x(t_7) &= 4\bar{d}t_{\bar{d}}^4 + 8\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 5\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3
\end{aligned} \tag{47}$$



Now with $t_8 = t_7$ and due to symmetry of the profile we have:

$$x(t_{15}) = 2x(t_7) = 8\bar{d}t_{\bar{d}}^4 + 16\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 10\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + 2\bar{d}t_{\bar{d}} t_{\bar{\jmath}}^3 \tag{48}$$

By setting $\bar{x} = x(t_{15})$ we can then solve $t_{\bar{\jmath}}$ from the following:

$$\begin{aligned} &(2\bar{d}t_{\bar{d}})t_{\bar{\jmath}}^3 + (10\bar{d}t_{\bar{d}}^2)t_{\bar{\jmath}}^2 + (16\bar{d}t_{\bar{d}}^3)t_{\bar{\jmath}} + (8\bar{d}t_{\bar{d}}^4 - \bar{x}) = 0 \\ &\Rightarrow t_{\bar{\jmath}}^3 + (5t_{\bar{d}})t_{\bar{\jmath}}^2 + (8t_{\bar{d}}^2)t_{\bar{\jmath}} + (4t_{\bar{d}}^3 - \tfrac{\bar{x}}{2\bar{d}t_{\bar{d}}}) = 0 \end{aligned} \tag{49}$$

The derivation of the solution of this third order polynomial equation is given in section 6.2. There it will be proven that there always exists a positive (or zero) real solution:

$$\begin{aligned} p &:= -\tfrac{1}{9}t_{\bar{d}}^2 \\ q &:= -\tfrac{1}{27}t_{\bar{d}}^3 - \tfrac{\bar{x}}{4\bar{d}t_{\bar{d}}} \\ D &:= p^3 + q^2 \\ r &:= \sqrt[3]{-q + \sqrt{D}} \\ t_{\bar{\jmath}} &= r - \tfrac{p}{r} - \tfrac{5}{3}t_{\bar{d}} \end{aligned} \tag{50}$$

After this, step 10 introduces the velocity and acceleration bounds again; the maximal velocity is obtained at t_7 (see 47):

$$\hat{v} = v(t_7) = 2\bar{d}t_{\bar{d}}^3 + 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \bar{d}t_{\bar{d}} t_{\bar{\jmath}}^2 \tag{51}$$

If $\hat{v} > \bar{v}$ the bound is violated and we can recalculate $t_{\bar{\jmath}}$ from the second order polynomial:

$$t_{\bar{\jmath}}^2 + 3t_{\bar{d}}t_{\bar{\jmath}} + 2t_{\bar{d}}^2 - \frac{\bar{v}}{\bar{d}t_{\bar{d}}} = 0 \tag{52}$$

The positive real solution for this is:

$$t_{\bar{\jmath}} = -1\frac{1}{2}t_{\bar{d}} + \sqrt{\frac{t_{\bar{d}}^2}{4} + \frac{\bar{v}}{\bar{d}t_{\bar{d}}}} \tag{53}$$

Next, the maximal acceleration is obtained at t_3 (see 44):

$$\hat{a} = a(t_3) = \bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}} \tag{54}$$

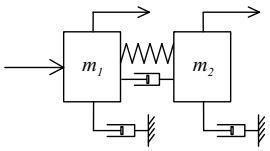
If $\hat{a} > \bar{a}$ we can recalculate $t_{\bar{\jmath}}$ as:

$$t_{\bar{\jmath}} = \frac{\bar{a}}{\bar{\jmath}} - t_{\bar{d}} \tag{55}$$

and we arrive at step 11 of the planning algorithm.

In accordance with step 12 there is one further extension of the trajectory required for the calculation of $t_{\bar{a}}$. To do the necessary calculations, assume that $t_{\bar{a}} > 0$, such that we now have $t_4 > t_3$ (from figure 7). Equation 30 and the fact that $d = 0$ and $\jmath = 0$ during the period from t_3 to t_4 now enables us to calculate:

$$\begin{aligned} \jmath(t_4) = &\ 0 \\ a(t_4) = &\ \bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}} \\ v(t_4) = &\ \bar{d}t_{\bar{d}}^3 + 1\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}^2 t_{\bar{a}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}} \\ x(t_4) = &\ \tfrac{7}{12}\bar{d}t_{\bar{d}}^4 + 1\tfrac{1}{6}\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + \tfrac{3}{4}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \tfrac{1}{6}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3 + \\ &\ \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{a}}^2 + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}}^2 + \bar{d}t_{\bar{d}}^3 t_{\bar{a}} + 1\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}t_{\bar{a}} + \\ &\ \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 t_{\bar{a}} \end{aligned} \tag{56}$$



$$
\begin{aligned}
\jmath(t_5) = &\quad -\bar{d}t_{\bar{d}} \\
a(t_5) = &\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}} \\
v(t_5) = &\quad 1\tfrac{5}{6}\bar{d}t_{\bar{d}}^3 + 2\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}^2 t_{\bar{a}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}} \\
x(t_5) = &\quad 2\tfrac{1}{24}\bar{d}t_{\bar{d}}^4 + 3\tfrac{1}{6}\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 1\tfrac{1}{4}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \tfrac{1}{6}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3 + \\
&\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{a}}^2 + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}}^2 + 2\bar{d}t_{\bar{d}}^3 t_{\bar{a}} + 2\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}t_{\bar{a}} + \\
&\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 t_{\bar{a}}
\end{aligned}
\tag{57}
$$

$$
\begin{aligned}
\jmath(t_6) = &\quad -\bar{d}t_{\bar{d}} \\
a(t_6) = &\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 \\
v(t_6) = &\quad 1\tfrac{5}{6}\bar{d}t_{\bar{d}}^3 + 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}^2 t_{\bar{a}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}} \\
x(t_6) = &\quad 2\tfrac{1}{24}\bar{d}t_{\bar{d}}^4 + 5\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 4\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3 + \\
&\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{a}}^2 + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}}^2 + 2\bar{d}t_{\bar{d}}^3 t_{\bar{a}} + 3\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}t_{\bar{a}} + \\
&\quad 1\tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 t_{\bar{a}}
\end{aligned}
\tag{58}
$$

$$
\begin{aligned}
\jmath(t_7) = &\quad 0 \\
a(t_7) = &\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 \\
v(t_7) = &\quad 2\bar{d}t_{\bar{d}}^3 + 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}^2 t_{\bar{a}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}} \\
x(t_7) = &\quad 4\bar{d}t_{\bar{d}}^4 + 8\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 5\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3 + \\
&\quad \tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{a}}^2 + \tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}}^2 + 3\bar{d}t_{\bar{d}}^3 t_{\bar{a}} + 4\tfrac{1}{2}\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}t_{\bar{a}} + \\
&\quad 1\tfrac{1}{2}\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 t_{\bar{a}}
\end{aligned}
\tag{59}
$$

Now, given that $t_{\bar{d}}$ and $t_{\bar{\jmath}}$ are already determined and the planned trajectory is symmetrical, we can solve $t_{\bar{a}}$ such that $x(t_{15}) = 2x(t_7) = \bar{x}$ from the following polynomial equation:

$$
\begin{aligned}
&\{\bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}\}t_{\bar{a}}^2 + \{6\bar{d}t_{\bar{d}}^3 + 9\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + 3\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2\}t_{\bar{a}} + \\
&\{8\bar{d}t_{\bar{d}}^4 + 16\bar{d}t_{\bar{d}}^3 t_{\bar{\jmath}} + 10\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}}^2 + 2\bar{d}t_{\bar{d}}t_{\bar{\jmath}}^3 - \bar{x}\} = 0
\end{aligned}
\tag{60}
$$

Clearly $t_{\bar{a}}$ must be the positive root of this second order polynomial (compare with equation 25).

Now we can introduce the velocity bound again; the maximal velocity is obtained at t_7 (see 59):

$$
\hat{v} = 2\bar{d}t_{\bar{d}}^3 + 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2 + \bar{d}t_{\bar{d}}^2 t_{\bar{a}} + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}t_{\bar{a}} \tag{61}
$$

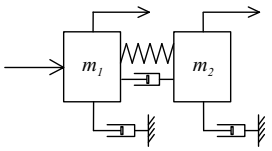
If $\hat{v} > \bar{v}$ the bound is violated and we can recalculate $t_{\bar{a}}$ as:

$$
t_{\bar{a}} = \frac{\bar{v} - 2\bar{d}t_{\bar{d}}^3 - 3\bar{d}t_{\bar{d}}^2 t_{\bar{\jmath}} - \bar{d}t_{\bar{d}}t_{\bar{\jmath}}^2}{\bar{d}t_{\bar{d}}^2 + \bar{d}t_{\bar{d}}t_{\bar{\jmath}}} \tag{62}
$$

Now we have determined $t_{\bar{d}}$, $t_{\bar{\jmath}}$ and $t_{\bar{a}}$ under the restriction of the given bounds and, as the next derivative is v, this establishes step 12 of the algorithm.

Finally, step 13 will determine the constant velocity time interval $t_{\bar{v}}$ such that the required total displacement $\bar{x}$ is obtained. From the previous calculations follows that $t_{\bar{v}} = 0$ if $t_{\bar{a}}$ is according to equation 60. But if $t_{\bar{a}}$ is reduced according to equation 62, we will have $x(t_{15}) < \bar{x}$ and we must add a constant velocity phase to the trajectory such that $x(t_4) - x(t_3) = \bar{x} - 2x(t_7)$. With equation 59 this implies that we can calculate $t_{\bar{v}}$ as:

$$
t_{\bar{v}} = \frac{\bar{x} - 2x(t_7)}{\bar{v}} \tag{63}
$$



This completes the calculation of the characteristics of the symmetrical fourth order profile for a given distance $\bar{x}$ and given bounds $\bar{v}$, $\bar{a}$, $\bar{j}$ and $\bar{d}$. We can further state that if the trajectory contains a constant velocity phase (i.e. $t_{\bar{v}} > 0$), the trajectory is time-optimal. The total displacement $\bar{x}$ can be expressed as a function of $\bar{d}$ and the times $t_{\bar{d}}$ $t_{\bar{j}}$, $t_{\bar{a}}$ and $t_{\bar{v}}$:

$$\begin{aligned}\bar{x} = \bar{d}\ (\ & t_{\bar{d}}^2 t_{\bar{a}}^2 + t_{\bar{d}} t_{\bar{j}} t_{\bar{a}}^2 + 6 t_{\bar{d}}^3 t_{\bar{a}} + 9 t_{\bar{d}}^2 t_{\bar{j}} t_{\bar{a}} + 3 t_{\bar{d}} t_{\bar{j}}^2 t_{\bar{a}} + \\ & 8 t_{\bar{d}}^4 + 16 t_{\bar{d}}^3 t_{\bar{j}} + 10 t_{\bar{d}}^2 t_{\bar{j}}^2 + 2 t_{\bar{d}} t_{\bar{j}}^3 + 2 t_{\bar{d}}^3 t_{\bar{v}} + \\ & 3 t_{\bar{d}}^2 t_{\bar{j}} t_{\bar{v}} + t_{\bar{d}} t_{\bar{j}}^2 t_{\bar{v}} + t_{\bar{d}}^2 t_{\bar{a}} t_{\bar{v}} + t_{\bar{d}} t_{\bar{j}} t_{\bar{a}} t_{\bar{v}}\ )\end{aligned} \tag{64}$$

6.2 Solution of equation 49

A solution of the third order polynomial equation 49 is a valid solution $t_{\bar{j}}$ if and only if the solution is real and larger than or equal to zero. This section will consider the existence of such a solution and derive its calculation. The results of this section are based on Cardan's formula, as can for instance be found in [2].

Given a third order polynomial equation (also known as a cubic equation) in variable x of the form:

$$x^3 + ax^2 + bx + c = 0 \tag{65}$$

with constant real parameters a, b and c. This can be rewritten as:

$$y^3 + 3py + 2q = 0 \tag{66}$$

with new variable y defined as:

$$y = x + \tfrac{1}{3}a \tag{67}$$

and new parameters p and q defined as:

$$\begin{aligned} p &= \tfrac{1}{3}b - \tfrac{1}{9}a^2 \\ q &= \tfrac{1}{27}a^3 - \tfrac{1}{6}ab + \tfrac{1}{2}c \end{aligned} \tag{68}$$

Applying this to equation 49 gives:

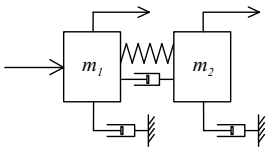
$$\begin{aligned} p &= \tfrac{8}{3}t_{\bar{d}}^2 - \tfrac{25}{9}t_{\bar{d}}^2 &&= -\tfrac{1}{9}t_{\bar{d}}^2 \\ q &= \tfrac{125}{27}t_{\bar{d}}^3 - \tfrac{40}{6}t_{\bar{d}}^3 + 2t_{\bar{d}}^3 - \tfrac{\bar{x}}{4\bar{d}t_{\bar{d}}} &&= -\tfrac{1}{27}t_{\bar{d}}^3 - \tfrac{\bar{x}}{4\bar{d}t_{\bar{d}}} \end{aligned} \tag{69}$$

From [2] we have that the characteristics of the solution are determined by the sign of the discriminant D:

$$\begin{aligned} D &= p^3 + q^2 = (-\tfrac{1}{9}t_{\bar{d}}^2)^3 + (-\tfrac{1}{27}t_{\bar{d}}^3 - \tfrac{\bar{x}}{4\bar{d}t_{\bar{d}}})^2 \\ &= \underbrace{-\frac{1}{729}t_{\bar{d}}^6 + \frac{1}{729}t_{\bar{d}}^6}_{=0} + \underbrace{\frac{\bar{x}t_{\bar{d}}^2}{54\bar{d}} + \frac{\bar{x}^2}{16\bar{d}^2 t_{\bar{d}}^2}}_{>0} > 0 \end{aligned} \tag{70}$$

Now $D > 0$ implies that there exists one real and two complex conjugated solutions, and only the real solution is a valid candidate for $t_{\bar{j}}$. Based on Cardan's formula for equation 67 the real solution can be found from:

$$y = \sqrt[3]{-q + \sqrt{D}} + \sqrt[3]{-q - \sqrt{D}} \tag{71}$$



such that:

$$x = y - \tfrac{1}{3}a \;\Rightarrow\; t_{\bar{j}} = y - \tfrac{5}{3}t_{\bar{d}} \tag{72}$$

Next, to prove that this real solution must also be larger than or equal to zero, consider the signs of the coefficients of equation 49:

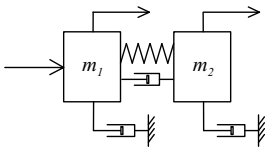
$$\begin{aligned} a &= 5t_{\bar{d}} > 0 \\ b &= 8t_{\bar{d}}^2 > 0 \\ c &= 4t_{\bar{d}}^3 - \tfrac{\bar{x}}{2\bar{d}t_{\bar{d}}} \leq 0 \end{aligned} \tag{73}$$

Now consider the function $f(x) = x^3 + ax^2 + bx + c$. Obviously, if $c = 0$ the solution is $x = 0$ (i.e. $t_{\bar{j}} = 0$). If $c < 0$, we have $f(0) = c < 0$ and $f(x)$ is monotone increasing for $x \geq 0$ and $\lim_{x\to\infty} f(x) = \infty$. Hence, the function $f(x)$ must cross the positive real axis, thus establishing a positive real solution. This solution must be given by equations 71 and 72, as this is the only real solution.

For easier implementation of this solution, equations 71 and 72 are simplified as follows:

$$\begin{aligned} t_{\bar{j}} &= \sqrt[3]{-q+\sqrt{D}} + \sqrt[3]{-q-\sqrt{D}} - \tfrac{5}{3}t_{\bar{d}} \\ &= \sqrt[3]{-q+\sqrt{D}} + \frac{\sqrt[3]{-q+\sqrt{D}}\sqrt[3]{-q-\sqrt{D}}}{\sqrt[3]{-q+\sqrt{D}}} - \tfrac{5}{3}t_{\bar{d}} \\ &= \sqrt[3]{-q+\sqrt{D}} + \frac{\sqrt[3]{-q^2-D}}{\sqrt[3]{-q+\sqrt{D}}} - \tfrac{5}{3}t_{\bar{d}} \\ &= \sqrt[3]{-q+\sqrt{D}} - \frac{p}{\sqrt[3]{-q+\sqrt{D}}} - \tfrac{5}{3}t_{\bar{d}} \end{aligned} \tag{74}$$

With this result it is not necessary to calculate the second term: $\sqrt[3]{-q-\sqrt{D}}$, which saves time and also has a positive effect on accuracy of finite precision calculations.



7 Implementation aspects

7.1 Switching times

From the previous chapters it is clear that the derivations for especially fourth order trajectory planning are quite elaborate. However, the calculations actually required for implementation of this planner are relatively straightforward. The algorithm of chapter 4 consists of a combination of polynomial calculations with simple if-then-else tests for which most state-of-the-art motion control hardware has standard algorithms.

When considering implementation of the planned trajectory into a feedforward control scheme, it is important to consider the effect of discretization. This implies that the integrators and the feedforward filter as given in the diagram of figure 5 will all have to be implemented in discrete time at some given sampling time interval T_s. This also implies that the switching time instances of the planned trajectory must be synchronized with the sampling time instances: i.e. the time intervals $t_{\bar{v}}$, $t_{\bar{a}}$, etc. must be multiples of T_s.

To obtain this, it must be accepted that time-optimality as resulting from the continuous time calculations in the previous chapters is lost. The calculated time intervals must be rounded-off towards a multiple of T_s. This must be done such that the given bounds are not violated, but at the same time approximated as closely as possible. The approach proposed here is to extend the algorithm of chapter 4. After each calculation or re-calculation of a time interval, the result is rounded-off upward to the next multiple of T_s. Next the maximal value of the highest derivative is calculated accordingly.

As an example consider the first calculation of $t_{\bar{d}}$ for a fourth order trajectory (equation 36). The rounded-off value for $t_{\bar{d}}$ can be calculated as:

$$t'_{\bar{d}} = \text{ ceil}\left(\frac{t_{\bar{d}}}{T_s}\right) \times T_s \tag{75}$$

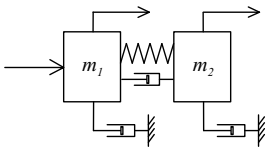
with ceil(.) denoting the rounding off towards the next higher integer. From equation 36 we can then calculate a new value for $\bar{d}$:

$$\bar{d}' = \frac{\bar{x}}{8t'^{4}_{\bar{d}}} \tag{76}$$

Note that with $t'_{\bar{d}} \geq t_{\bar{d}}$ we must have $\bar{d}' \leq \bar{d}$.

It can be verified that this same approach is valid for the calculation or re-calculation of all time intervals. It is important to note that with each new calculation of $\bar{d}'$ its value must reduce. This guarantees that none of the bounds that were checked in earlier steps of the algorithm will be violated.

State-of-the-art motion controllers are equipped with a high resolution floating point calculation unit, such that quantization effects are usually negligible. To check this, we can make use of equation 64 to calculate the obtained position in case a quantized $\bar{d}'$ is used. The difference between desired position and calculated position can be accounted for by means of a correction signal on the position trajectory. Typically in a digital system, the position signal is also quantized (usually in encoder increments). The correction signal can then be implemented by adding some increments to the position reference at each sampling instance during the trajectory. In relevant cases only one or two correction increments at each sampling instance should be sufficient (to prevent deterioration of the feedforward control). In case larger corrections are necessary, the accuracy



and/or sampling rate of the motion control hardware should be increased.

A complete algorithm for fourth order trajectory planning is given in the appendix. This algorithm is given as a Matlab .M function, which calculates the times $t_{\bar{d}}$ $t_{\bar{j}}$, $t_{\bar{a}}$ and $t_{\bar{v}}$. It also includes the possibility to specify a sampling time interval. It then uses equations 75 and 76 to synchronize the switching time instances with the sampling time instances and to calculate $\bar{d}'$. Finally, it allows compensation for the effect of quantization by specifying a number of significant decimals for $\bar{d}'$ and calculating the required amount of correction increments for the position reference signal.

7.2 Synchronization of profiles

Consider the continuous time implementation of the feedforward signal calculation as given in figure 5. Now, the generation of the derivative of jerk profile and the four integrators to obtain jerk, acceleration, velocity and position must be implemented in digital hardware. A straightforward approach is to replace the continuous time integrators by forward Euler discrete time integrators as indicated in figure 8. Clearly, all required profiles are thus calculated. However, due to the zero

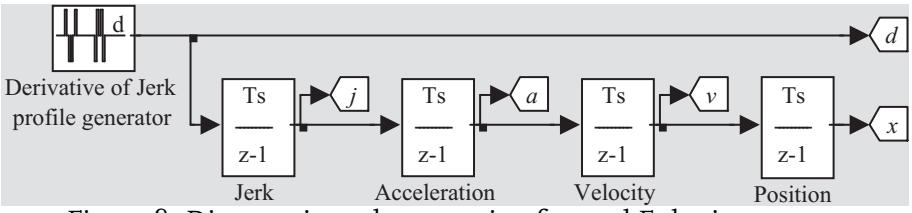
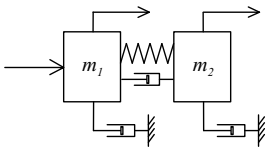


Figure 8: Discrete time planner using forward Euler integrators.

order hold effect, each of the four integrators introduces a specific delay time. This can be seen in figure 9, in which the discrete time profiles are compared with the corresponding continuous time profiles. Note that the chosen sampling time is 0.05 seconds, which is large in relation with the required trajectory to show the discretization effect more clearly. To fix this effect, the higher order profiles can be delayed individually such that the symmetry of the complete set of profiles is restored. Figure 10 shows the effect of this in comparison with the continuous time profiles. The latter are delayed with 0.1 seconds (or 2 times the sample time interval), to show that the results are now perfectly synchronized. Note that the derivative of jerk profile must be delayed with $2 \times T_s$, the jerk profile with $1.5 \times T_s$, the acceleration profile with $1 \times T_s$ and finally the velocity profile with $0.5 \times T_s$. To obtain a delay of $0.5 \times T_s$ when sampling with T_s the average value is taken from the current and previous amplitude of the considered profile. This operation appears to work very well, although the associated smoothing effect is undesirable.



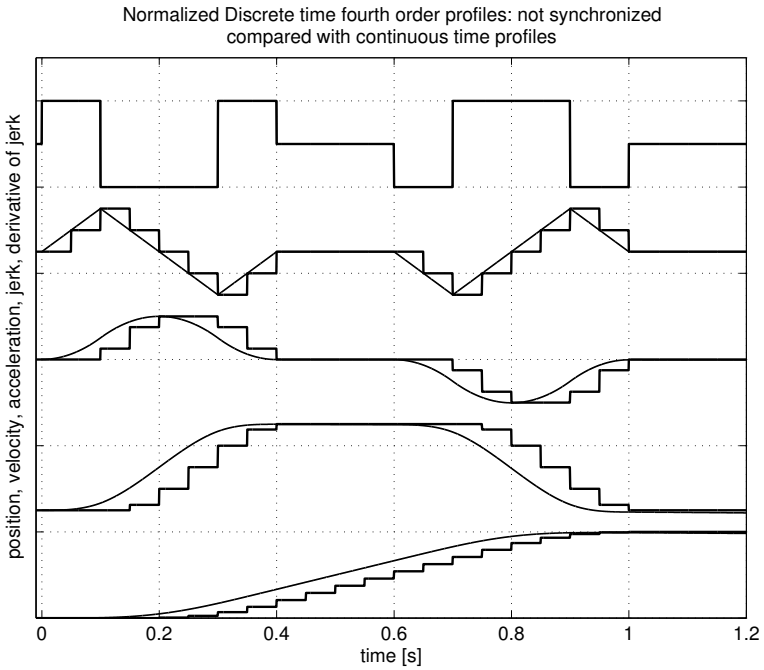


Figure 9: Discrete time profiles using forward Euler integrators, compared with continuous time profiles.

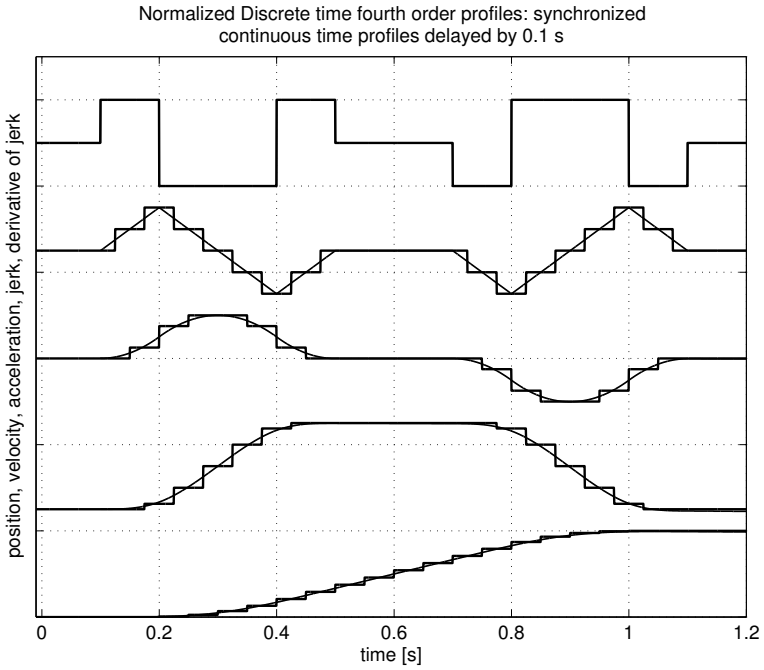
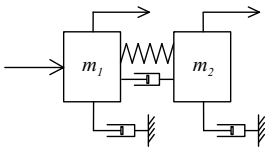


Figure 10: Synchronized discrete time profiles using forward Euler integrators, compared with continuous time profiles delayed with 0.1 seconds.



7.3 Implementation of first order filter

All required profiles for calculation of the feedforward signal are now available. The multiplication with factors q_1 to q_4 followed by summation as indicated in figure 5 is straightforward. The first order filtering is less trivial, as it must also be transferred to the discrete time domain. A possible implementation that prevents problems with unwanted time delays and gives good results is to make use of the trapezoidal integration method. Figure 11 shows the equivalence of the first order filter in figure 5 with several possible implementations using the trapezoidal integration method. Note that the inherent disadvantage of using the trapezoidal integration method in

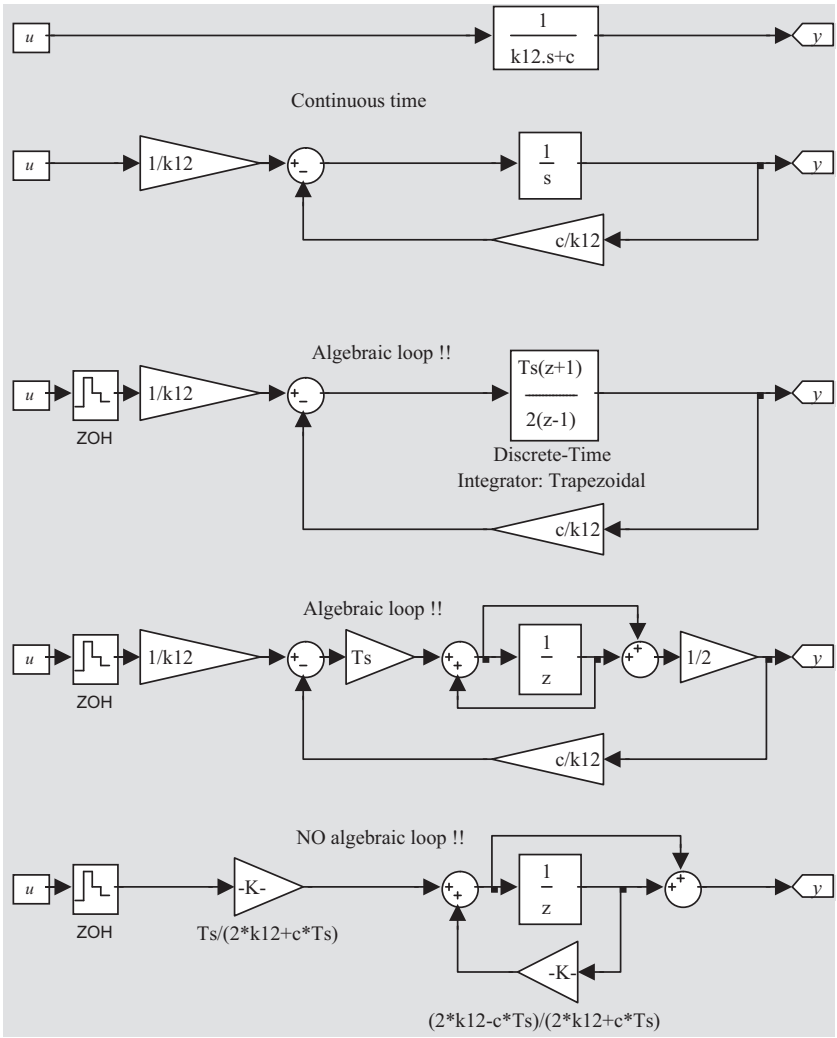
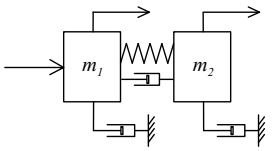


Figure 11: Discrete implementation of first order filter using the trapezoidal integration method.

a first order filter—the occurrence of an algebraic loop—can be prevented by re-calculating the loop. This can be checked by verifying that the discrete implementations all have the following transfer function:

$$y = \frac{\frac{T_s}{2k_{12}+cT_s}(z+1)}{z - \frac{2k_{12}-cT_s}{2k_{12}+cT_s}}u \tag{77}$$

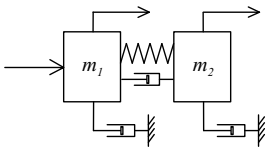


7.4 Calculation of reference trajectory

A final point on synchronization must be made with respect to the calculation of the reference trajectory that is used for feedback control. When applying the feedforward signal as calculated above, based on the synchronized profiles as illustrated in figure 10, the actual plant's response will be close to the ideal continuous time response with a delay of $2 \times T_s$. However, in order to compare this signal with the reference trajectory it must be sampled with the same sampling frequency as is used for generation of the reference trajectory. The sample and hold device used for this will then introduce an average delay of exactly $0.5 \times T_s$.

Hence, to compensate for this further delay, it is necessary to also delay the reference trajectory with this same value. This additional delay can be implemented in the same manner as mentioned before: by taking the average between the current and previous reference trajectory value. The result of this will be that the control error will not be affected by sampling. The controller will only act on the effects of disturbances and on discrepancies between the actual plant and the modelled fourth order behavior.

Obviously, this is only true if the sampling frequency is sufficiently high: otherwise the momentary control error may deviate significantly from the average value. If this is the case, an increase in sampling frequency must be considered. Usually however, the sampling frequency is more significantly determined by the demands on stability and performance of the (digital) feedback controller.



8 Simulation results

A theoretical motivation for the application of fourth order feedforward as an extension of rigid body feedforward was already given in chapter 3. Next, subsequent chapters have shown that fourth order feedforward is feasible in the sense of trajectory planning and (digital) feedforward implementation. This chapter will give further motivation for application of fourth order feedforward by considering some simulation results.

The main concern when considering model based feedforward control is the occurrence of discrepancies between the behavior of the actual motion system and the used model. This often motivates the use of rigid body feedforward, as the total mass of the motion system is usually known within tight boundaries. Furthermore, a well designed motion system will have limited friction and damping: especially dry friction must be minimized as it leads to classical problems in both feedforward and feedback. Linear viscous damping can more easily be dealt with, but often shows time dependent behavior: if damping has a reasonably constant value, it can be compensated as shown in chapter 2. Now, fourth order feedforward introduces several additional physical parameters that may, or may not be constant: a mass ratio (division of mass in m_1 and m_2), a damping ratio (division of damping in k_1 and k_2), a spring stiffness c and an internal damping k_{12} (see figures 1 and 4). The simulation results in this chapter will show that fourth order feedforward will give a significant improvement of performance (in the sense of servo errors during or after trajectory execution) in comparison with rigid body feedback, even if these additional parameters have relatively large uncertainties.

All simulations will be performed using the configuration of figure 5. The motion system parameters and their variations used in the simulations are given in table 1 The trajectory pa-

Parameter	Value	Unit	Variation
m_1	20	Kg	$m_1 \in \{15 \cdots 25\}$,
m_2	10	Kg	$m_2 = 30 - m_1$
k_1	10	Ns/m	$k_1 \in \{5 \cdots 15\}$,
k_2	10	Ns/m	$k_2 = 20 - k_1$
c	$6 \cdot 10^5$	N/m	$\pm 33\%$
k_{12}	500	Ns/m	$\pm 100\%$

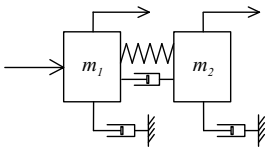
Table 1: Simulation parameters

rameters are defined in table 2. When comparing with rigid body feedforward, the fourth order

Parameter	Symbol	Value	Unit
derivative of jerk	$\bar{d}$	1000	m/s^4
jerk	$\bar{j}$	50	m/s^3
acceleration	$\bar{a}$	5	m/s^2
velocity	$\bar{v}$	1	m/s
displacement	$\bar{x}$	1	m

Table 2: Trajectory definition parameters

trajectory will be used, such that the smoothing effect of using a high order trajectory is not accountable for the difference. In that case, it can easily be verified that optimal rigid body feedforward is obtained by setting $m_1 = 30$, $m_2 = 0$, $k_1 = 20$, $k_2 = 0$ and $k_{12} = 0$. From



equation 9 we then have $q_1 = q_2 = 0$, $q_3 = m_1 c$ and $q_4 = k_1 c$. Furthermore, the first order filter reverts to a constant gain of $\frac{1}{c}$ (see also equation 77) and equation 10 reverts to equation 1. Note that the value of c becomes unimportant as it drops out of the calculations. To remove the effect of feedback control from the results, most simulations are performed in open loop.

Figure 12 shows the performance of the continuous time fourth order feedforward control for a fourth order motion system model with perturbations according to table 1. For comparison, the response of the nominal motion system model with the optimal rigid body feedforward controller is given. Note that in spite of the significant plant variations, the fourth order feedforward controller performs at least twice as good.

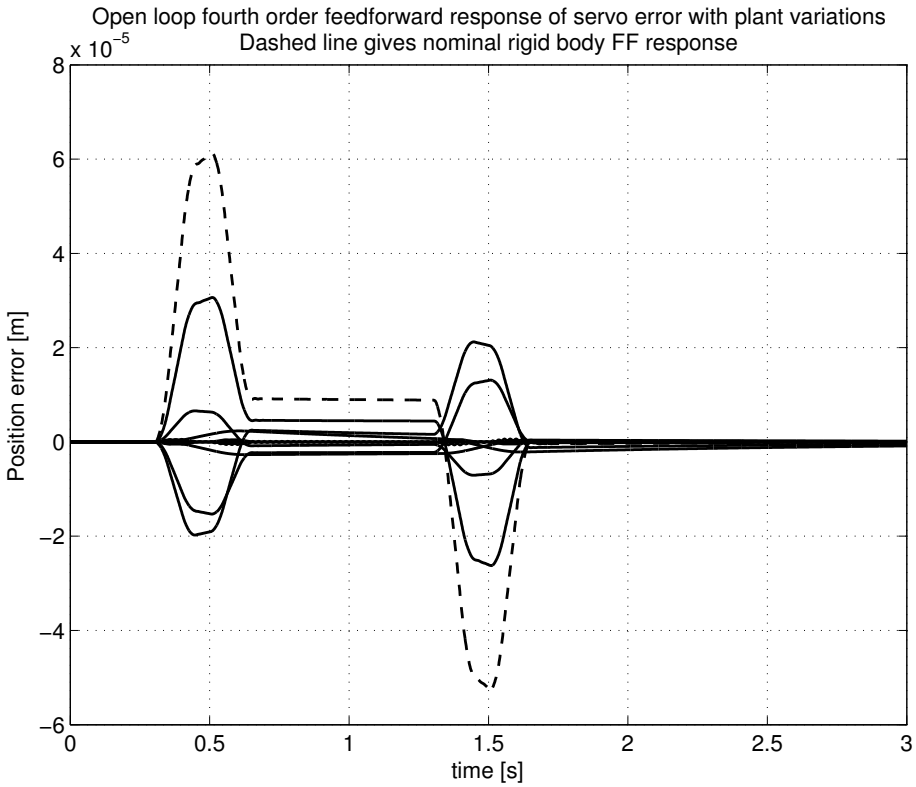
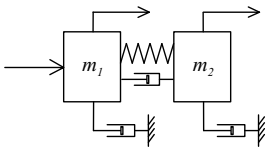


Figure 12: Open loop simulation results of fourth order feedforward controller with plant variations, in comparison with optimally tuned rigid body feedforward.

The approach of choosing the parameters such that fourth order feedforward reverts to rigid body feedforward suggests the use of third order feedforward as an intermediate form. This can be done by setting $m_1 = 30$ and $m_2 = 0$, while leaving the viscous damping parameters unchanged: now only q_1 reverts to zero. This would allow to implement the simpler third order trajectory planner, with jerk being the highest bounded derivative. However, figure 13 shows that the addition of a jerk feedforward term (i.e. q_2) does not have a significant effect: it is the derivative of jerk term q_1 that makes the difference. The fact that in practice it appears that third order trajectory planning may lead to strong improvement of performance is therefore almost exclusively due to smoothing.

Figure 14 shows that the servo error responses will not significantly deteriorate if fourth order



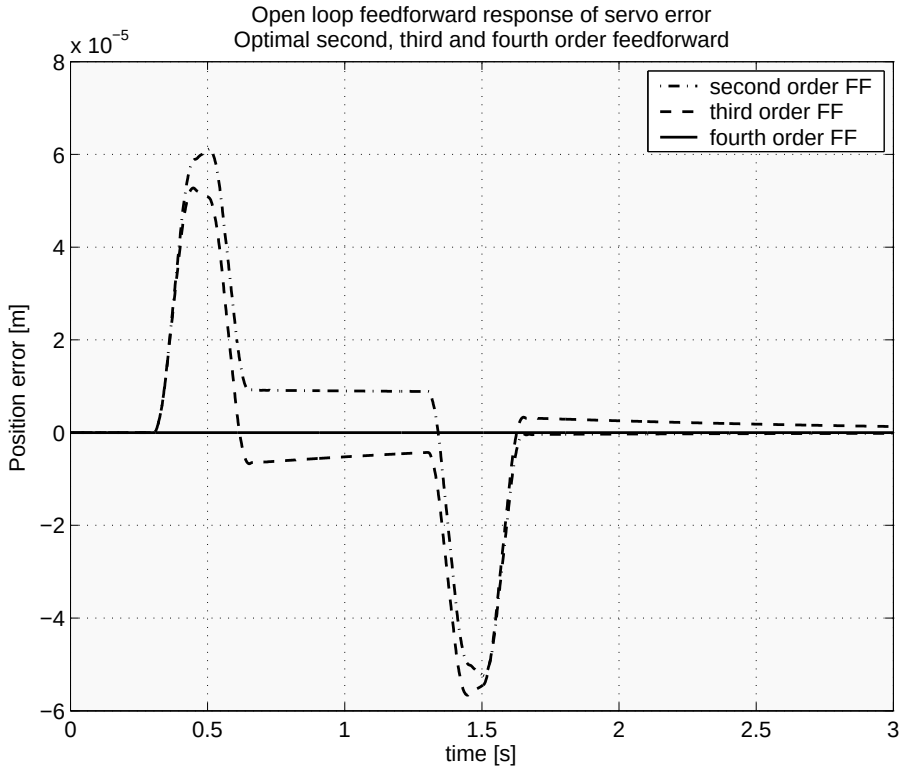
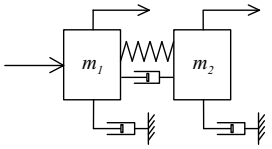


Figure 13: Open loop simulation results of optimal second, third and fourth order feedforward controller with nominal plant

feedforward is implemented in discrete time. As an example, the motion system with minimal spring-stiffness is considered. Both open loop and closed loop results are given: the feedback controller is tuned for a bandwidth of about 10 Hz, whereas the motion system's first resonance mode is at 50 Hz. The digital feedforward controller is combined with a discrete time feedback controller, both with a sampling rate of 200 Hz. Note that this is a low sampling rate for a high performance servo system: this is chosen to demonstrate the discretization effects more clearly.



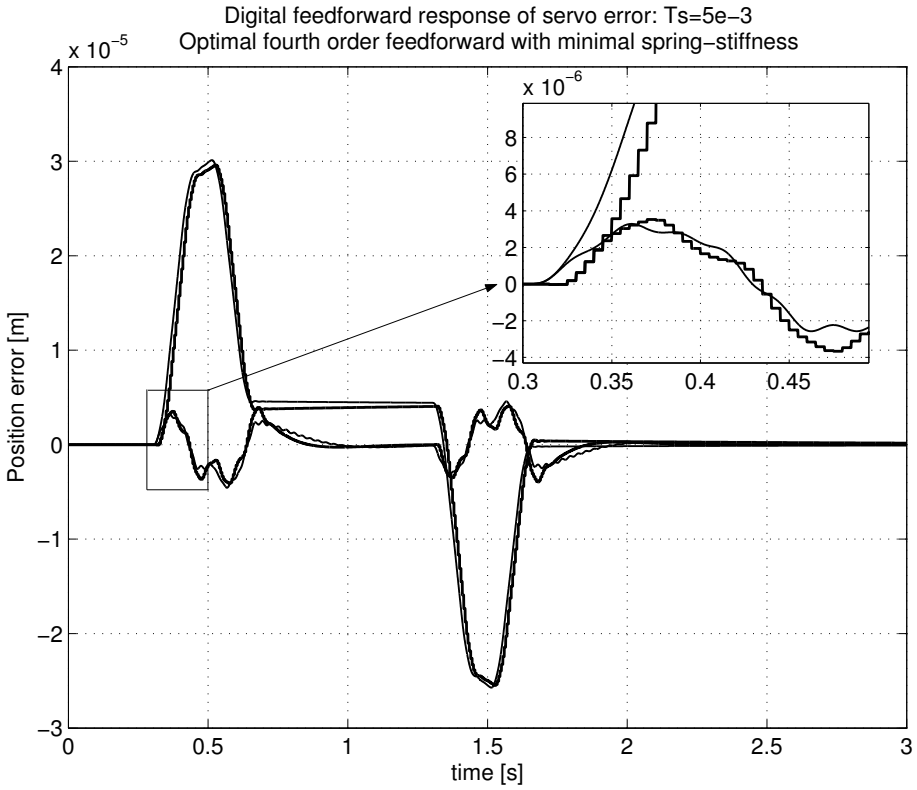
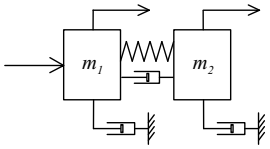


Figure 14: Simulation results of open loop and closed loop discrete time fourth order feedforward controller with minimal stiffness plant. Thick lines: discrete, thin lines: continuous time.



9 Conclusions

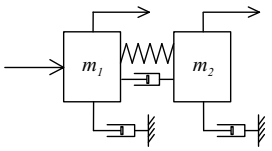
For high performance motion control, especially for electromechanical motion systems, the usefulness of feedforward control is well known and often implemented. This report shows that the popular simple feedforward scheme known as 'mass-feedforward', 'rigid-body feedforward' or 'second-order feedforward' can be extended to higher order feedforward while still maintaining important practical properties like time-optimality, actuator effort limitation, reliability, implementability and accuracy. Furthermore, it is argumented that the increase in complexity is manageable in state-of-the-art motion control hardware.

Third order feedforward, which is increasingly applied in practice, appears to be mainly effective due to 'smoothing' of the trajectory, i.e. reduction of the high frequency content of the trajectory, such that feedback control can be more effective. The use of fourth order feedforward for high performance electromechanical motion systems is motivated by considering an appropriate model and supported by simulation results. Apart from the mentioned smoothing effect, the feedforward of the derivative of jerk profile appears to result in a significant performance improvement. This is even more remarkable when considering that, for relevant cases, the calculated feedforward force is hardly affected.

A high level algorithm is given to calculate higher order trajectories for point-to-point moves; other motion commands, like speed change operations, can be derived from this. For third and fourth order trajectory planning, the details of the algorithm are worked out, resulting in practical, reliable and accurate algorithms.

Further implementation issues, like discrete time calculations, quantization effects and synchronization, are explicitly addressed. The trajectory planning algorithms can be implemented such that switching times are exactly synchronized with sampling instances. A digital implementation is suggested that takes care of the synchronization of the various profiles with each other and the position trajectory, and also with the measured position. It is shown that deterioration of the continuous time results due to sampling are small when applying a sufficient sampling rate. Experience shows that a sampling rate that is required for stable feedback control is also sufficient for feedforward control.

Simulation results show that the improvement obtained with fourth order feedforward is not overly sensitive to variations in the parameters that are additional to the 'classic' parameters of rigid-body feedforward (i.e. mass and viscous damping). Obviously, the feedforward performance will improve if the dynamical behavior of the actual motion system is closer to that of the fourth order model and said parameters are known within tighter bounds. An important advantage of the suggested implementation is the possibility to manually fine-tune the feedforward amplification factor for each profile (the q factors). This can be seen as a simple, direct extension of the well known practice of fine-tuning rigid-body feedforward.



References

[1] M. Boerlage, M. Steinbuch, P. Lambrechts, M. van de Wal, 'Model based feedforward for motion systems'. *Submitted*, 2003.

[2] I.N. Bronshtein, K.A. Semendyayev, 'A 'Guide book to mathematics'. *Verlag Harri Deutsch, Frankfurt*, ISBN 3-87144-095-7, 1973.

[3] S. Devasia, 'Robust inversion-based feedforward controllers for output tracking under plant uncertainty'. *Proc. of the American Control Conference*, 2000, pp.497-502.

[4] B.G. Dijkstra, N.J. Rambaratsingh, C. Scherer, O.H. Bosgra, M. Steinbuch, S. Kerssemakers, 'Input design for optimal discrete-time point-to-point motion of an industrial XY positioning table', *in Proc. 39th IEEE Conference on Decision and Control*, 2000, pp.901-906.

[5] L. Hunt, G. Meyer, 'Noncausal inverses for linear systems'. *IEEE Trans. on Automatic Control*, 1996, Vol. 41(4), pp.608-611.

[6] P.H. Meckl, 'Discussion on: comparison of filtering methods for reducing residual vibration'. *European Journal of Control*, 1999, Vol. 5, pp.219-221.

[7] P.H. Meckl, P.B. Arestides, M.C. Woods, 'Optimized S-curve motion profiles for minimum residual vibration'. *Proc. of the American Control Conference*, 1998, pp.2627-2631.

[8] B.R. Murphy, I. Watanaabe, 'Digital shaping filters for reducing machine vibration'. *IEEE Trans. on Robotics and Automation*, 1992, Vol. 8(2), pp.285-289.

[9] F. Paganini, A. Giusto, 'Robust synthesis of dynamic prefilters', *Proc. of the American Control Conference*, 1997, pp.1314-1318.

[10] H. S. Park, P.H. Chang, D.Y. Lee, 'Concurrent design of continuous zero phase error tracking controller and sinusoidal trajectory for improved tracking control'. *J. Dynamic Systems, Measurement, and Control*, 2001, Vol. 5, pp.3554-3558.

[11] D. Roover, 'Motion control for a wafer stage', *Delft University Press*, The Netherlands, 1997.

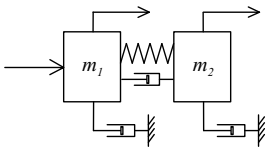
[12] D. Roover, F. Sperling, 'Point-to-point control of a high accuracy positioning mechanism', *Proc. of the American Control Conference*, 1997, pp.1350-1354.

[13] N. Singer, W. Singhose, W. Seering, 'Comparison of filtering methods for reducing residual vibration'. *European Journal of Control*, 1999, Vol.5, pp.208-218.

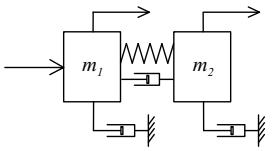
[14] M. Steinbuch, M.L. Norg, 'Advanced motion control: an industrial perspective', *European Journal of Control*, 1998, pp.278-293.

[15] M. Tomizuka, 'Zero phase error tracking algorithm for digital control'. *J. Dynamic Systems, Measurement, and Control*, 1987, Vol.109, pp.65-68.

[16] D.E. Torfs, J. Swevers, J. De Schutter, 'Quasi-perfect tracking control of non-minimal phase systems'. *Proc. of the 30th Conference on Decision and Control*, 1991, pp.241-244.

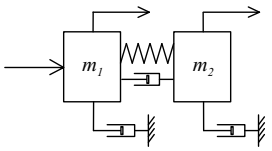


[17] D.E. Torfs, R. Vuerinckx, J. Swevers, J. Schoukens, 'Comparison of two feedforward design methods aiming at accurate trajectory tracking of the end point of a flexible robot arm', *IEEE Trans. on Control Systems Technology*, 1998, Vol.6(1), pp.1-14.



A List of symbols

$\bar{x}$	bound on $\lvert x\rvert$
$\hat{x}$	maximum value obtained by $\lvert x\rvert$ if bound is not considered
x_0	initial value
$t_{\bar{x}}$	time interval during which $\lvert x\rvert$ obtains its bound
$t_i,\ i \in N$	switching time instances
D	discriminant of cubic equation
F	actuator force, feedforward force [N]
T_s	sampling time [s]
a	acceleration (profile) [m/s^2], parameter of cubic equation
b	parameter of cubic equation
c	spring stiffness [N/m], parameter of cubic equation
d	derivative of jerk (profile) [m/s^4]
j	jerk (profile) [m/s^3]
k	viscous damping coefficient [Ns/m]
m	mass [Kg]
p	parameter of cubic equation
q	parameter of cubic equation
$q_{1\cdots4}$	feedforward parameters
r	auxiliary parameter
s	Laplace transform variable
t	time [s]
u	input signal
v	velocity (profile) [m/s]
x	position, displacement (profile) [m]
y	variable, output signal
z	shift operator



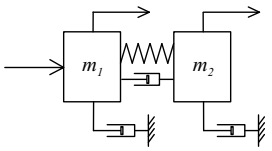
B The Matlab function MAKE4.m

```
function [t,dd]=make4(varargin)

% [t,dd] = make4(p,v,a,j,d,Ts,r,s)
%
% Calculate timing for symmetrical 4th order profiles.
%
% inputs:
%      p    = desired path (specify positive)              [m]
%      v    = velocity bound (specify positive)            [m/s]
%      a    = acceleration bound (specify positive)        [m/s2]
%      j    = jerk bound (specify positive)                [m/s3]
%      d    = derivative of jerk bound (specify positive)  [m/s4]
%      Ts   = sampling time            [s]        (optional, if not specified or 0: continuous time)
%      r    = position resolution      [m]        (optional, if not specified: 10*eps)
%      s    = number of decimals for digitized
%             derivative of jerk bound             (optional, if not specified: 15)
%
% outputs:
%      t(1) = constant djerk phase duration
%      t(2) = constant jerk phase duration
%      t(3) = constant acceleration phase duration
%      t(4) = constant velocity phase duration
%
%       t1              t1              t1     t1
%       .-.             .-.             .-.    .-.
%       | |             | |             | |    | |
%       | |t2    t3    t2| |   t4       t2| | t3 | |t2
%       '-'--.-.----.-.--' '---------.-.--'-'----'-'--.-.--
%            | |    | |             | |             | |
%            | |    | |             | |             | |
%            '-'    '-'             '-'             '-'
%            t1     t1              t1              t1
%
% In case of discrete time, derivative of jerk bound d is reduced to dd and
% quantized to ddq using position resolution r and number of significant decimals s
% Two position correction terms are calculated to 'repair' the position error
% resulting from using ddq instead of dd:
%   cor1  gives the number of position increments that can equally be divided
%         over the entire trajectory duration
%   cor2  gives the remaining number of position increments
% The result is given as:
%                          dd = [ ddq  cor1  cor2  dd ]
%

% Paul Lambrechts, TUE fac. WTB, last modified: April 1, 2003.
%

% Checking validity of inputs
if nargin < 5 | nargin > 8
   help make4
   return
else
   p=abs(varargin{1});
   v=abs(varargin{2});
   a=abs(varargin{3});
   j=abs(varargin{4});
   d=abs(varargin{5});
   if nargin == 5
      Ts=0; r=eps; s=15;
   elseif nargin == 6
      Ts=abs(varargin{6});
      r=eps; s=15;
```



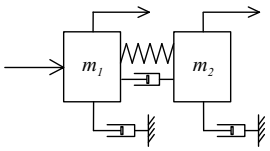

```
  elseif nargin == 7
      Ts=abs(varargin{6});
      r=abs(varargin{7});
      s=15;
  elseif nargin == 8
      Ts=abs(varargin{6});
      r=abs(varargin{7});
      s=abs(varargin{8});
   end
end

if length(p)==0 | length(v)==0 | length(a)==0 | length(j)==0 | length(d)==0 | ...
   length(Ts)==0 | length(r)==0 | length(s)==0
   disp('ERROR: insufficient input for trajectory calculation')
   return
end

tol = 3*eps;   % tolerance required for continuous time calculations
dd = d;      % required for discrete time calculations

% Calculation constant djerk phase duration: t1
t1  = (p/(8*d))^(1/4) ;  % largest t1 with bound on derivative of jerk
if Ts>0
   t1 = ceil(t1/Ts)*Ts;
   dd  = p/(8*(t1^4));
end
% velocity test
if v < 2*dd*t1^3           % v bound violated ?
   t1 = (v/(2*d))^(1/3) ;  % t1 with bound on velocity not violated
   if Ts>0
      t1 = ceil(t1/Ts)*Ts;
      dd  = v/(2*(t1^3));
   end
end
% acceleration test
if a < dd*t1^2         % a bound violated ?
   t1 = (a/d)^(1/2) ;  % t1 with bound on acceleration not violated
   if Ts>0
      t1 = ceil(t1/Ts)*Ts;
      dd  = a/(t1^2);
   end
end
% jerk test
if j < dd*t1    % j bound violated ?
   t1 = j/d ;    % t1 with bound on jerk not violated
   if Ts>0
      t1 = ceil(t1/Ts)*Ts;
      dd  = j/t1;
   end
end
d = dd;  % as t1 is now fixed, dd is the new bound on derivative of jerk

% Calculation constant jerk phase duration: t2
% largest t2 with bound on jerk
P  = -1/9*t1^2;
Q  = -1/27*t1^3-p/(4*d*t1);
D  = sqrt(P^3+Q^2);
% D  = sqrt( p/d * (t1^2/54 + p/t1^2/(16*d)));  % alternative calculation
R  = (-Q+D)^(1/3);
t2 = R - P/R - 5/3*t1;
% is solution of: t2^3+5*t1*t2^2+8*t1^2*t2+4*t1^3-p/(2*d*t1) = 0
if Ts>0
   t2 = ceil(t2/Ts)*Ts;
   dd  = p/( 2*t1*(4*t1^3+t2*(8*t1^2+t2*(5*t1+t2))) );
       % = p/( 8*t1^4 + 16*t1^3*t2 + 10*t1^2*t2^2 + 2*t1*t2^3 );
```




```
end
if abs(t2)<tol t2=0; end % for continuous time case
% velocity test
if v < (t1*dd*(2*t1^2 + t2*(3*t1 + t2)));          % = (2*dd*t1^3 + 3*dd*t1^2*t2 + dd*t1*t2^2)
   t2 = ( (t1^2)/4 + v/(d*t1) )^(1/2) - 1.5*t1 ;   % t2 with bound on velocity not violated
   if Ts>0
      t2 = ceil(t2/Ts)*Ts;
      dd = v/( t1*(2*t1^2 + t2*(3*t1 + t2)) );     % = v/( 2*t1^3 + 3*t1^2*t2 + t1*t2^2 )
   end
end
if abs(t2)<tol t2=0; end % for continuous time case
% acceleration test
if a < ( dd*t1*(t1 + t2) )   % a bound violated ?
   t2 = a/(d*t1) - t1 ;      % t2 with bound on acceleration not violated
   if Ts>0
      t2 = ceil(t2/Ts)*Ts;
      dd  = a/( t1*(t1 + t2) );
   end
end
if abs(t2)<tol t2=0; end % for continuous time case
d = dd;  % as t2 is now fixed, dd is the new bound on derivative of jerk

% Calculation constant acceleration phase duration: t3
c1 = t1*(t1+t2) ;                                    % = t1^2+t1*t2
c2 = 3*t1*( 2*t1^2 + t2*(3*t1 + t2) ) ;              % = 6*t1^3 + 9*t1^2*t2 + 3*t1*t2^2
c3 = 2*t1*( t1*( 4*t1^2 + t2*(8*t1 + 5*t2) ) + t2^3 ) ; % = 8*t1^4 + 16*t1^3*t2 + 10*t1^2*t2^2
                                                     %                          + 2*t1*t2^3
t3 = (-c2 + sqrt(c2^2-4*c1*(c3-p/d)))/(2*c1) ;       % largest t3 with bound on acceleration
if Ts>0
   t3 = ceil(t3/Ts)*Ts;
   dd = p/( t3*( c1*t3 + c2) + c3 );   % = p/( c1*t3^2 + c2*t3 + c3 )
end
if abs(t3)<tol t3=0; end % for continuous time case
% velocity test
if v < dd*t1*( t1*( 2*t1 + 3*t2 + t3 ) + t2*(t2+t3) )
       % = 2*dd*t1^3 + 3*dd*t1^2*t2 + dd*t1*t2^2 + dd*t1^2*t3 + dd*t1*t2*t3
   % t3, bound on velocity not violated
   t3 = ( t1*( t1*( 2*t1 - 3*t2) - t2^2 ) + v/d ) / (t1*(t1+t2));
      % = -(2*t1^3 + 3*t1^2*t2 + t1*t2^2 - v/d)/(t1^2 + t1*t2)
   if Ts>0
      t3 = ceil(t3/Ts)*Ts;
      dd = v/( t1*( t1*( 2*t1 + 3*t2 + t3 ) + t2*(t2+t3) ) );
         % = v/( 2*t1^3 + 3*t1^2*t2 + t1*t2^2 + t1^2*t3 + t1*t2*t3 )
  end
end
if abs(t3)<tol t3=0; end % for continuous time case
d = dd;  % as t3 is now fixed, dd is the new bound on derivative of jerk

% Calculation constant velocity phase duration: t4
% t4 with bound on velocity
t4 = ( p - d*( t3*(c1*t3 + c2) + c3) )/v ;  % = ( p - d*(c1*t3^2 + c2*t3 + c3) )/v
if Ts>0
   t4 = ceil(t4/Ts)*Ts;
   dd = p/( t3*( c1*t3 + c2) + c3 + t4*( t1*( t1*( 2*t1 + 3*t2 + t3 ) + t2*(t2+t3) ) ) ) ;
      % = p/( c1*t3^2 + c2*t3 + c3 + t4*(2*t1^3 + 3*t1^2*t2 + t1*t2^2 + t1^2*t3 + t1*t2*t3) )
end
if abs(t4)<tol t4=0; end % for continuous time case

% All time intervals are now calculated
t=[t1 t2 t3 t4] ;

% This error should never occur !!
if min(t)<0
    disp('ERROR: negative values found')
end
```

```
% Quantization of dd and calculation of required position correction (decimal scaling)
if Ts>0
   x=ceil(log10(dd));          % determine exponent of dd
   ddq=dd/10^x;                % scale to 0-1
   ddq=round(ddq*10^s)/10^s;   % round to s  decimals
   ddq=ddq*10^x;
   % actual displacement obtained with quantized dd
   pp = ddq * p/dd;
      %= ddq*( c1*t3^2 + c2*t3 + c3 + t4*(2*t1^3 + 3*t1^2*t2 + t1*t2^2 + t1^2*t3 + t1*t2*t3) )
   dif=p-pp;          % position error due to quantization of dd
   cnt=round(dif/r);  % divided by resolution gives 'number of increments'
                      % of required position correction
   % smooth correction obtained by dividing over entire trajectory duration
   tt = 8*t(1)+4*t(2)+2*t(3)+t(4);
   ti = tt/Ts;        % should be integer number of samples
   cor1=sign(cnt)*floor(abs(cnt/ti))*ti;   % we need cor1/ti increments correction at each
                                           % ... sample during trajectory
   cor2=cnt-cor1;                          % remaining correction: 1 increment per sample
                                           % ... during first part of trajectory
   dd=[ddq cor1 cor2 dd];
else
   dd=[dd 0 0 dd]; % continuous time result in same format
end

% Finished.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```