Model-Predictive Control of Powershifts of Heavy-Duty Trucks with Dual-Clutch Transmissions

Tomas Szabo\textsuperscript{1}, Michael Buchholz\textsuperscript{1} and Klaus Dietmayer\textsuperscript{1}

Abstract—This paper presents a cascaded control system with a discrete-time model-predictive control (MPC) in the outer loop and a flatness-based 2-DOF controller in the inner loop. Hereby, the outer loop controller provides the trajectories for the inner loop controller. To significantly lower the computational effort of the MPC, Laguerre functions are used to approximate the control input of the MPC. Since the flatness-based inner loop control also needs the time-derivatives of the provided trajectories, it will be shown how those can be directly obtained from the MPC results. Finally, this control system is applied to a test rig with a dual-clutch transmission for heavy-duty trucks to perform powershifts.

I. INTRODUCTION

Modern powertrain development of heavy-duty trucks tries to meet the rising demands for fuel efficiency and environmental requirements while further increasing comfort and safety. Dual-clutch transmissions (DCT) belong to the group of automated manual transmissions (AMT) and can be basically viewed as two separate transmissions, where each one is connected to the engine with its own clutch. The benefit of DCTs is that while one part of the transmission is transferring the current gear, the next gear can be preselected in the other part. The gearshift itself can then be performed by changing the engagement from one clutch to the other. This can be done in a way that there is a continuous torque transfer from the engine to the drive.

For the automation of powershifts, a powertrain controller is needed, combined with separate controllers for the clutch positions. In control theory, this can be described as a cascaded control structure, which appears in a wide variety of control applications. Common examples are flow rate (or pressure) control in heat exchangers and boilers or motor current control of DC motors. In those applications, the inner loop dynamics are usually much faster than the outer loop dynamics, resulting in a notable simplification of the control problem.

The cascaded control structure discussed in this paper consists of a linear, discrete-time MPC for the powertrain (outer loop) and a flatness-based 2-DOF clutch position control (inner loop). Hereby, the MPC is used to compute the reference trajectories for the inner loop control. To enhance computational efficiency, Laguerre functions (based on \cite{8}) are used to approximate the input of the MPC. It will be shown how this approach can be cleverly used to obtain the time derivatives needed for the flatness-based feedforward control inside the inner loop directly as part of the MPC solution.

Flatness-based feedforward control can also be beneficial for systems with input time-delays (see e.g. \cite{4}, \cite{10}). Because pneumatic clutches exhibit input time-delay, it is briefly shown how the given cascaded control structure can be extended to compensate those time-delays.

This paper is structured as follows: Section II introduces Laguerre functions, which are used to enhance the solution of the discrete-time MPC. Section III discusses the overall control problem with the extension for systems with input time-delay in the inner loop. Section IV presents the powertrain model and the pneumatic clutch models, which are then used an example system for the given control structure. The measurement results of this system are presented in Section V. The paper closes with a conclusion in Section VI.

II. LAGUERRE FUNCTIONS

This section gives a brief introduction of Laguerre functions, which will be used in Section III to enhance the solution of a classic, discrete-time MPC (see e.g. \cite{1}, \cite{9}). Laguerre functions have an asymptotically decaying characteristic (see Fig. 1) and form an orthogonal function set. The continuous-time Laguerre functions are defined as

\[ L_n(t) = \sqrt{2a} e^{-at} \frac{d^n}{dt^n} \left(t^n e^{-2at}\right) \]

and were transformed in \cite{7} into the \(\mathcal{Z}\)-domain using a bilinear transformation, resulting in

\[ \Gamma_n(z) = \frac{\sqrt{1 - a^2}}{1 - az^{-1}} \left(1 - az\right)^{n-1}. \]

In \cite{8}, \cite{9}, the corresponding sequence \(l_1(k), \ldots, l_{N_i}(k)\) calculated from the inverse \(\mathcal{Z}\)-transform of \(\Gamma_n(z)\) was used as basis for the approximation of the control input \(u_k\) of a discrete-time MPC. For the \(i\)-th input, the sequence \(\tilde{l}_{i,k} = [l_1(k)\ l_2(k)\ \ldots\ l_{N_i}(k)]^T\) can be recursively computed by

\[ \tilde{l}_{i,k+1} = A_{i,k} \tilde{l}_{i,k}, \]

where

\[ (A_{i,j})_{jk} = \begin{cases} (-1)^{j-k+1}a^{j-k-1}\beta, & j > k \\ a, & j = k, j, k = 1, \ldots, N_i \\ 0, & j < k \end{cases} \]

with

\[ \tilde{L}_{i,0} = \sqrt{\beta} \left[1 - a\ a^2\ \ldots\ (-1)^{N_i-1}a^{N_i-1}\right]^T \]
and $\beta = 1 - a^2$, with the dimensions $A_i,l \in \mathbb{R}^{N_i \times N_i}$ and $\tilde{L}_{i,k} \in \mathbb{R}^{N_i}$. $0 \leq a \leq 1$ is called the pole of the Laguerre function set and is a design parameter of the network. $N_i$ is the number of Laguerre functions used for the approximation and can be freely chosen depending on the desired accuracy. Fig. 1 shows the first four Laguerre functions for the design parameters $a = 0.85$ and $a = 0.9$ for $k = 20$, respectively a time frame of $t = 0.1s$. As can be seen, the smaller $a$, the faster the functions decay. Depending on the length of the control horizon, the design parameter $a$ would be chosen rather large to cover a long horizon and rather small for a short horizon. For $a = 0$, the sequence $\tilde{L}_{i,k}$ converges to a mere impulse sequence for $k \leq N_i$ and equals zero for $k > N_i$. As consequence, one would have to set $N_i = N_c$ to approximate the control input over the complete horizon.

With (3), the control sequence $u_{i,k}, u_{i,k+1}, \ldots, u_{i,k+N_i-1}$ for the $i$-th input can be approximated by the set of Laguerre functions $l_1(k), l_2(k), \ldots, l_{N_i}(k)$ such that

$$u_{i,k} = \sum_{j=1}^{N_i} c_j l_j(k) = \tilde{L}_{i,k}^T \tilde{\eta}_i,$$

(5)

where $\tilde{\eta}_i = [c_1 \ c_2 \ \ldots \ c_{N_i}]^T$ is a vector which holds the $N_i$ Laguerre weighting coefficients. For the MIMO case, (5) has to be extended to

$$u_k = \begin{bmatrix} \tilde{L}_{1,k}^T & 0 & \cdots & 0 \\ 0 & \tilde{L}_{2,k}^T & \cdots & 0 \\ 0 & 0 & \cdots & \tilde{L}_{p_1,k}^T \end{bmatrix} \begin{bmatrix} \tilde{\eta}_1 \\ \tilde{\eta}_2 \\ \vdots \\ \tilde{\eta}_{p_1} \end{bmatrix} = L_k^T \eta.$$

(6)

With $i = 1, \ldots, p_1$ and $N = \sum_{i=1}^{p_1} N_i$, it follows that $\eta \in \mathbb{R}^N$ and $L_k^T \in \mathbb{R}^{p_1 \times N}$.

III. CONTROL PROBLEM

The structure of the control problem consists of two systems. System $\Sigma_1$ is linear and time-invariant, given by the state-space representation

$$\dot{x}_1 = A^* x_1 + B^* u_1$$

(7a)

$$y_1 = C^* x_1$$

(7b)

with $A^* \in \mathbb{R}^{n_1 \times n_1}$, $B^* \in \mathbb{R}^{n_1 \times p_1}$, $C^* \in \mathbb{R}^{k_1 \times n_1}$. System $\Sigma_2$ is a nonlinear SISO system given by

$$\dot{x}_2 = f(x_2, u_2)$$

(8a)

$$y_2 = h(x_2)$$

(8b)

with $x \in \mathbb{R}^{n_2}$ and the flat output $z = y_2 \in \mathbb{R}$. The flat output $y_2$ of system $\Sigma_2$ is the $i$-th input $u_{1,i}$ of system $\Sigma_1$. Assuming full relative degree of $\Sigma_2$, the following known properties (see e.g. [2]) for flat SISO systems are

$$z = \Phi(x_2),$$

(9)

$$x_2 = \Psi_x(z, \dot{z}, \ldots, z^{(n_2-1)}),$$

(10)

$$y_2 = \Psi_{y_2}(z, \dot{z}, \ldots, z^{(n_2)}),$$

(11)

$$u_2 = \Psi_u(z, \dot{z}, \ldots, z^{(n_2)}),$$

(12)

where $z = [z_1 \ z_2 \ \ldots \ z_{n_2+1}] = [z \ \dot{z} \ \ldots \ z^{(n_2)}]$ holds for system $\Sigma_2$.

In the problem discussed in this paper, system $\Sigma_1$ is used within the MPC to compute a desired trajectory for $u_{1,i}$, such that $y_2 = u_{1,i}$ holds. With that, the input $u_2$ of system $\Sigma_2$ can be calculated using (12):

$$u_2 = \Psi_u(z = u_{1,i}, \dot{z} = u_{1,i}, \ldots, z^{(n_2)} = u_{1,i}).$$

(13)

The resulting control loops can be viewed as an outer control loop for system $\Sigma_1$ and an inner control loop for system $\Sigma_2$. Fig. 2 shows a block diagram of this control structure. The other elements $u_{1,j}, j = 1, \ldots, p_1$, and $j \neq i$, can be used for further inner control loops, or can be directly applied on actuators. The example in Section IV uses a combination of both.

For the inner control loop, a 2-DOF controller with a flatness-based feedforward control $\Sigma_{ff}$ and a stabilizing feedback control $\Sigma_{fb}$ is used. $\Sigma_{ff}$ provides the feedforward control action to follow the desired trajectory, whereas $\Sigma_{fb}$ stabilizes the output around the trajectory, compensating disturbances and model errors.

For the outer control loop, a discrete-time MPC based on Laguerre functions is used because of two reasons. Firstly, the application of Laguerre functions with $a > 0$ reduces the dimension of the optimization problem and thus substantially decreases the computational effort to compute the MPC. Secondly, by approximating the control input $u_i$ of $\Sigma_1$ with Laguerre functions, the derivatives $\dot{u}_{1,i}, \ldots, u_{1,i}^{(n_2)}$, which are needed to compute (12), are directly part of the MPC solution.

A. Discrete-time model predictive control

To implement a discrete-time MPC, the continuous-time system $\Sigma_1$ has to be discretized resulting in

$$x_{k+1} = Ax_k + Bu_k,$$

(14a)

$$y_k = Cx_k,$$

(14b)

with $A \in \mathbb{R}^{n_1 \times n_1}$, $B \in \mathbb{R}^{n_1 \times p_1}$, $C \in \mathbb{R}^{k_1 \times n_1}$, and $n_1 = n_1^1 + p_1^1$, $p_1 = p_1^1$, where the former state vector $x_1$ is augmented by the former input $u_1$ to $x_k = [x_1 \ u_1]^T$. The new input $u_k$ is $u_k = u_{1,k} - u_{1,k-1}$. Through subsequent
The Laguerre functions introduced in Section II can be used to approximate the control input $u_k$ at the time instant $k$ by inserting (6) into (15), leading to

$$x_{k+m|k} = A^m x_k + \sum_{j=0}^{m-1} A^{m-j-1} B u_{k+j}. \quad (15)$$

The cost functional can then be written as

$$J(\eta) = \sum_{m=1}^{N_p} (x_{k+m|k} - A^m x_{ref})^T Q (x_{k+m|k} - A^m x_{ref}) + \eta^T R_L \eta$$

with $R_L \in \mathbb{R}^{N \times N}$. The prediction horizon $N_p \in \mathbb{N}$ specifies how far into the future the model behavior will be predicted. Computing $\frac{\partial J(\eta)}{\partial \eta} = 0$ leads to the optimal coefficient vector for minimal costs

$$\eta^* = \left[ \sum_{m=1}^{N_p} \Phi(m) Q \Phi^T(m) + R_L \right]^{-1} \left[ \sum_{m=1}^{N_p} \Phi(m) Q A^m (x_k - x_{ref}) \right]$$

$$= K (x_k - x_{ref}), \quad (18)$$

where the matrix $K \in \mathbb{R}^{N \times n_1}$ can be computed offline. This finally leads to the optimal control law

$$u_k = - \begin{bmatrix} \tilde{L}_1^T & 0 & \ldots & 0 \\ 0 & \tilde{L}_2^T & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \tilde{L}_{n_1}^T \end{bmatrix} \eta^*$$

$$= - L_0^* K (x_k - x_{ref}), \quad (19)$$

The dimension of (19) does not directly depend on the prediction horizon, but one has to choose $N$ and $\alpha$ such that the set of Laguerre functions can approximate the optimal solution as good as possible. This will usually result in a suboptimal solution, but with the benefit of a considerably lower computational effort, facilitating a real-time solution even on less powerful hardware.

Recalling the overall control structure, the coupling of (14) and (7) has to be explained more precisely. Returning briefly to the continuous-time domain, the flat output of $\Sigma_2$ is $z = y_2$. Since we augmented the state vector $x_1$ with $u_1$ and used $du_1/dt$ as new input, it follows that $\dot{z} = x_1 n_1^T + \dot{z} = \dot{u}_1$. At this point, the time derivatives $\dot{z}, \ldots, \dot{z}^{(n_2)}$ have still to be computed, since they are not part of the solution yet.

As already mentioned in Section II, the discrete Laguerre functions $\Gamma_n(z)$ are computed by a bilinear transformation from $L_n(t)$, thus (2) are not the correct continuous-time representations. If the exact continuous-time representations of $\Gamma_n(z)$ and their time derivatives were known, the result from the discrete-time MPC would automatically provide those time-derivatives. Therefore, in the appendix, the computation of the corresponding time domain representations $\gamma_n(t), \gamma_n(t), \ldots$ by applying the inverse $z$-transformation on $\Gamma_n(z)$ is shown. Fig. 1 also shows the first four continuous-time Laguerre functions resulting from the transformation, which are exactly matching the discrete counterparts at the sample times.

With known $\gamma_n(t)$, the control input (5) can be rewritten to

$$u_k(t) = [\gamma_1(t) \gamma_2(t) \ldots \gamma_{N_i}(t)] \eta_k. \quad (20)$$

The subsequent time-derivatives can be computed based on (38). Fig. 3 shows the block diagram of the control structure, where

$$L_n^0 = \begin{bmatrix} \gamma_1(0) & \gamma_2(0) & \ldots & \gamma_{N_i}(0) \end{bmatrix} \quad (21)$$

and $L_0^0$ can be similarly calculated as (6) (note that $L_{i,0}^0 = \tilde{L}_{i,0}$).

B. Extension to nonlinear systems with input time-delay in the inner loop

This subsection discusses how the control structure can be used to additionally compensate for a constant time-delay $T_d$ in the input of the inner loop system. In this case, system
The idea behind this approach is to provide the future trajectory shifted by \(T_d\) for the feedforward controller of the inner control loop. Then, the feedforward controller computes the input for \(\Sigma_2\) \(T_d\) steps in advance, thus compensating the time-delay in system (22). Using the MPC in the outer loop, where the predicted control trajectory is already part of the solution, the control law (19) can be modified to incorporate the input time-delay within the inner loop.

The sample period \(T_s\) of system (14) has to be chosen such that \(T_s/T_d \in \mathbb{N}_0^+\) to facilitate a compensation of the time-delay. Due to the assumed coupling between \(\Sigma_1\) and \(\Sigma_2\), the time-delay acts likewise on \(u_{i1,i}\) in (14), which has to be accounted for in the control design. Assuming equal time-delay on all inputs of system \(\Sigma_1\) and including it in (15) results in

\[
x_{k+m|k} = A^m x_k + \sum_{i=0}^{m-1} A^{m-i-1} B u_{k+i-T_d},
\]

which means that the past \(T_d\) inputs are needed to compute the forced action \(H_{k+m|k} = \sum_{i=0}^{m-1} A^{m-i-1} B u_{k+i-T_d}\).

Since at a current time instant \(k\) the MPC can influence the system only beginning from \(k+T_d\) in the future, the MPC is computed on the predicted state \(x_{k+T_d|k}\). Fig. 4 shows the block diagram of the resulting control structure, where the buffer holds the past \(T_d\) values of \(u_k\) and is initialized with zeros.

C. Discussion

In system dynamics, cascaded control structures resulting from a decomposition of a system into a slow and a fast subsystem are known under singular perturbation models (see e.g. [3] and [6]). Singular perturbation models allow the inner system with fast dynamics to be treated as static for the slow system, resulting in a simplification of the control design. To guarantee stability of the cascade, the inner system usually has to be much faster than the outer system. The control structure in this paper, as will be also presented in the example system, does not necessarily assume to have different dynamics for both systems. Nevertheless, to obtain a stable control loop, the inner loop control has to be able to follow the reference trajectories computed by the outer loop control. Since the dynamics of the control trajectory is determined by the solution of the MPC, the MPC has to be adequately tuned to deliver suitable trajectories. This can be done by means of the weighting matrices of the cost functional. The given MPC can also be used to incorporate state and input constraints (see [9]). Using input constraints, the control input could thus be forced to dynamics which can be followed by the inner loop control. Applying constraints comes of course with a much higher computation requirement to solve the underlying QP-problem.

IV. APPLICATION: POWERSHIFTS WITH A DUAL-CLUTCH TRANSMISSION FOR HEAVY DUTY TRUCKS

As mentioned in the introduction, the presented control structure will be applied to perform powershifts with a dual-clutch transmission for heavy-duty trucks on a test rig. For this purpose, the models of the powertrain and the pneumatic clutches are introduced in this section.

A. Powertrain

Fig. 5 shows the scheme of a powertrain test rig with dual-clutch transmission. The motor \(m1\) is connected to the dual-clutch with the clutch disks \(cd1\) and \(cd2\). The motor torque is transmitted through the torsional dampers \(td1\) and \(td2\) and the drive shafts \(ds1\) and \(ds2\) into the gearbox. The transmission ratios of the two transmissions are given by \(i_1\) and \(i_2\), respectively. The efficiency coefficient \(\eta\) accounts for the motor power loss in the gearbox due to friction and churning. On the test rig, the motor \(m2\) is simulating the drive load. The inputs of the system are the motor torque \(M_{m1}\) and the two clutch torques \(M_{c1}\) and \(M_{c2}\). The load torque \(M_{m2}\) is regarded as a disturbance, since in a vehicle, it would consist of the external drag forces (i.e. grade resistance, roll and air friction), which are not present on the test rig. \(J\) denotes the inertias, \(c\) and \(d\) the stiffnesses and inner dampings of the shafts, \(\varphi\) the angles, \(d_{\text{MotDyn}}\) viscous friction coefficients (e.g. bearings) and \(M_{\text{fric}}\) static friction torque losses (e.g. sealings).

The mathematical equations result in three linear, time-invariant models in the form of (7). If either clutch 1 or 2 is closed, the model has an order of 11. If both clutches are opened or slipping, the model is of 13th order. An appropriate switching between the three models has to be incorporated in the control and observer implementation. The exact equations can be found in [5].
B. Pneumatic clutches

The dual-clutch module consists of two dry clutches actuated by pneumatics. Fig. 6 shows a simplified model of a single pneumatic dry clutch. The on/off solenoid valves (a) connect the pressure tank (b) to the chamber of the concentric pneumatic actuator (c). The piston (d) of the actuator pushes against the diaphragm spring (e), which causes the pressure plate (f) to press the clutch disk (g) against the fly wheel (h). The only moving part of the system is the piston of the clutch actuator. Its position \( x_1 \) is measured.

The given system can be divided into a mechanical and a pneumatic part, both acting on the piston. The mechanical part is a mass-spring second order system. The dynamics of the pneumatical part can be modeled using the ideal gas law. Applying the equilibrium of force on the piston yields the system model

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -\frac{b}{M} x_2 - \frac{1}{M} \Psi(x_1) + \frac{A_p}{M} (x_3 - p_{atm}) \\
\dot{x}_3 &= -\frac{A_p}{A_p x_1 + V_0} x_2 x_3 + \frac{RT}{A_p x_1 + V_0} u(t - T_d)
\end{align*}
\]

\( y = x_1 \),

where \( x_1 \) is the piston position, \( x_2 \) is the piston velocity and \( x_3 \) is the pressure within the clutch actuator. Except from the nonlinear spring characteristic \( \Psi(x_1) \), all other parameters are constant. The input \( u \) of the system is the mass flow into the actuator, the output \( y \) the piston position \( x_1 \), which can be quite easily verified to be flat regarding the model (24). Transport time-delays in the pipes and switching time-delays of the solenoid valves add up to a bulk time-delay \( T_d \) of the input \( u \).

C. Powershifts

Fig. 7 shows the overall control structure. When receiving a shift request, the setpoint \( x_{ref} \) for the MPC control law (18) (outer loop) of the powertrain after the powershift has to be computed. The inner control loops consist of the position controls of the two clutches and the engine control. On the test rig the engine control is part of the converter driving the motors. This control structure matches the structure discussed Section III.

V. Results

In this section, some results from measurements of a powerupshift on the test rig will be presented. A powershift can be divided into three phases:

1) Positioning of the clutches to initial values.
2) The powershift phase, where the engaging clutch takes over the torque of the disengaging clutch.
3) Adjusting the engine speed to the speed of the new gear.

The control structure described in Section III will be applied during the powershift phase. During the powerupshift, the disengaging clutch should remain in stiction during the whole powershift phase, while the engaging clutch takes over. At the end of the powershift, the engaging clutch should transmit the complete engine torque and the disengaging clutch should be completely disengaged.

Fig. 8 shows measurements of the angular speeds as well as the torques and their time derivatives during the described powerupshift. From the angular speeds, one can see that the controller is able to hold the disengaging clutch in stiction during the whole powershift phase, because the tracking of the computed torque trajectories of the MPC by the clutch position control in the inner loop is very good.

An important point to draw attention to is the fact that the torque transfer of dry clutches is due to dry friction, which is a highly nonlinear phenomenon and only roughly known by means of an adapted characteristic. It has to be expected that this characteristic is erroneous to some extent. Another source of error is coming from the clutch position, which usually will not exactly match the reference. Fig. 8 was obtained using a quite exact clutch characteristic. To investigate the controller performance for erroneous clutch characteristics, the clutch characteristic for the disengaging clutch has been exemplarily altered such that approximately the doubled torque output within the range of interest is used within the controller model. Results taken with this considerably falsified characteristic are shown in Fig. 9. Due to the incorrect value of the characteristic, the disengaging clutch does not remain in stiction and the engine speed increases, because the sum of both clutch torques is smaller than expected by the controller. At this point, also a switching
between the 11th and 13th order models occurs in the controller.

For a satisfactory powerupshift, the controller should limit the speed increase as much as possible, otherwise in a real vehicle, a driver might get the feeling of the engine running off. From the speeds in Fig. 9, one can see that the controller is able to hold the engine speed within a reasonable range. This is done mainly by decreasing the engine torque as can be seen on the time-derivatives of the control input of the engine torque.

VI. Conclusions

This paper presented a cascaded control structure with an MPC in the outer loop and flatness-based 2-DOF inner control loops. The MPC is used to compute the trajectories and their time-derivatives for the inner loop controls. Therefore, an MPC with Laguerre functions based on [8] was extended by calculating the exact continuous-time representations of the used Laguerre functions such that the needed time-derivatives can be directly obtained from the solution of the MPC. The control structure is applied to perform powershifts with a dual-clutch heavy-duty truck transmission and has shown very good results on a test rig. Even with highly falsified clutch characteristics, the controller is capable to yield satisfactory shifting performance on the test rig.

REFERENCES

APPENDIX

Subsequently, the continuous-time description and the time derivatives of the discrete Laguerre functions (2) will be computed. Starting with the partial fraction decomposition of the fraction

\[
\frac{(1 - az)^{n-1}}{(z - a)^n}
\]

by applying the binomial theorem

\[
(\alpha - \beta)^n = \sum_{k=0}^{n} \binom{n}{k} \alpha^{n-k} \beta^k
\]
on the align

\[
(1 - az)^{n-1} = (-a)^{n-1} \cdot \left[ (z - a) - \left( \frac{1}{a} - a \right) \right]^{n-1},
\]

(25) can be expressed by

\[
\frac{(1 - az)^{n-1}}{(z - a)^n} = \frac{(-a)^{n-1}}{(z - a)^n} \sum_{k=0}^{n-1} \frac{(n-1)!}{(n-1-k)! \cdot k!} \cdot (-1)^k (z - a)^{n-1-k} \left( \frac{1}{a} - a \right)^k
\]

\[
= a^{n-1} \sum_{k=0}^{n-1} \frac{(n-1)!}{(n-1-k)! \cdot k!} \cdot (-1)^k (z - a)^{n-1-k} \left( \frac{1}{a} - a \right)^k
\]

Applying the inverse \(Z\)-transformation on (2)

\[
Z^{-1} \left[ \Gamma_n(z) \right] = \sqrt{1 - a^2} \cdot Z^{-1} \left[ \frac{(1 - az)^{n-1}}{(z - a)^n} \right]
\]

and substituting (28) leads to

\[
Z^{-1} \left[ \Gamma_n(z) \right] = \sqrt{1 - a^2} \cdot a^{n-1} \sum_{k=0}^{n-1} \left( \frac{1}{a} \right)^k \cdot \frac{1}{(n-1-k)! \cdot k!} \cdot (-1)^{n-1+k} \left( \frac{z}{a} \right)^{k+1} \left( \frac{z - a}{a^2} \right)^k (n-1)!
\]

with the correspondence

\[
Z^{-1} \left[ \frac{z}{(z - \beta)^n} \right] = \frac{\beta^{n+1} n}{(\beta - n + 1)! (n-1)!} \frac{z^n}{n!} \quad (31)
\]

and the abbreviation

\[
c_{nk} := \sqrt{1 - a^2} a^{n-1} \left( -1 \right)^{n-1-k} \left( \frac{1 + \frac{1}{T} \cdot \beta}{a} \right)^k \left( \frac{1}{n-1} \right)! \left( \frac{T}{k} \right)^2 T^k,
\]

the continuous-time description results in

\[
\gamma_n(t) = \sum_{k=0}^{n-1} c_{nk} \cdot T^k \cdot \frac{1}{(T/k)!} \cdot \frac{T}{k}.
\]

With the correlation

\[
\frac{\frac{T}{k}!}{(T/k)!} = \frac{1}{T} \cdot \prod_{m=0}^{k-1} (t-mT),
\]

for \(k \neq 0\) and \(a T = e^{ln a \cdot t}, (33)\) can be written as

\[
\gamma_n(t) = \left\{ \begin{array}{ll}
\gamma_0(t) & n = 1 \\
\gamma_0(t) + \sum_{k=1}^{n-1} c_{nk} \cdot \prod_{m=0}^{k-1} (t-mT) \cdot e^{ln a \cdot t}, & n > 1
\end{array} \right.
\]

In the next step, the time derivatives of \(I_n(t)\) will be computed. Introducing

\[
\alpha(t) := \prod_{m=0}^{k-1} (t-mT) \cdot e^{ln a \cdot t}
\]

and

\[
\beta(t) := \frac{T}{T-lT},
\]

with \(\alpha'(t) = \beta(t) \alpha(t)\) results in

\[
\gamma_n(t) = \left\{ \begin{array}{ll}
\gamma_0(t) & n = 1 \\
\gamma_0(t) + \sum_{k=1}^{n-1} c_{nk} \cdot \beta(t) \cdot \alpha(t), & n > 1
\end{array} \right.
\]

All subsequent derivatives \(\gamma_n^{(n)}\) can be computed by applying the product rule.

4561