Methoden

Christoph Rösmann, Frank Hoffmann* und Torsten Bertram

Prädiktive Regelung mit Timed-Elastic-Bands

Timed-Elastic-Bands for Predictive Control

Zusammenfassung: Der Beitrag beschreibt eine Erweiterung zur zeitoptimalen nichtlinearen modellprädiktiven Regelung. Die Trajektorie im Zustandsraum und die Steuerfolge werden durch ein Timed-Elastic-Band (TEB) repräsentiert. Diese Beschreibung in Form eines TEBs ermöglicht die Überführung der Regelungsaufgabe in ein nichtlineares, gewichtetes Ausgleichsproblem mit weichen Nebenbedingungen, für welches effiziente numerische Optimierungsverfahren zur Verfügung stehen. In Analogie zu modellprädiktiven Reglern wird die Planung der optimalen Trajektorie mit einer Zustandsrückführung im Regelkreis integriert. Die vergleichende Analyse der durch den TEB-Ansatz optimierten Steuerfolge mit der analytischen, optimalen Lösung belegt die Praxistauglichkeit der Methode.

Schlüsselwörter: (Modell-) Prädiktive Regelung, Trajektorienplanung, Zeitoptimale Regelung.

Abstract: This contribution presents an extension to time-optimal nonlinear model predictive control. The state space trajectory and the control inputs are represented by a Timed-Elastic-Band (TEB). This representation in terms of a TEB allows the transformation of the control task into a nonlinear, weighted regulation problem with soft constraints, for which efficient numerical solvers are available. In analogy to model predictive control planning of the optimal trajectory is integrated with state feedback in the control loop. The comparative analysis of the control sequence obtained from the TEB with the analytical, optimal solution confirms the feasibility of the method.

Keywords: (model-) predictive control, trajectory planning, time-optimal control.

DOI 10.1515/auto-2014-1120
Eingang 13. Mai 2014; angenommen 3. August 2014

***Korrespondenzautor: Frank Hoffmann,** Technische Universität Dortmund, E-Mail: frank.hoffmann@tu-dortmund.de
Christoph Rösmann, Torsten Bertram: Technische Universität Dortmund

1 Einleitung

In der Prozessautomatisierung hat sich die modellprädiktive Regelung (MPC) seit längerem etabliert [1]. Modellprädiktive Regelungsverfahren basieren auf der wiederholten Lösung eines Optimalsteuerungsproblems über einem endlichen Horizont und berücksichtigen Restriktionen der Stell- und Zustandsgrößen sowie Mehrgrößensysteme [2]. Innerhalb der letzten Dekade ist das Interesse an effizienten MPC-Implementierungen gestiegen mit dem Ziel, diese Entwurfsmethode für mechatronische Systeme mit niedrigen Zeitkonstanten zu erschließen. [3] wendet das Mehrfachschießverfahren zur Beschleunigung der Berechnungszeit an. Hierbei wird der Zeithorizont in einzelne Teilintervalle unterteilt, für welche die optimale Teiltrajektorie als zunächst isoliertes Anfangswertproblem parallel gelöst wird. Der Ansatz wird in [4] erweitert, indem eine zunächst grobe Approximation der optimalen Steuerfolge während der Laufzeit iterativ verfeinert wird. Neuere Ansätze [5–7] formulieren das Optimierungsproblem neu, um es dadurch einer effizienten Lösung mit Inneren-Punkte-Verfahren zugänglich zu machen. Durch die simultane Umstrukturierung der Zustands- und Steuerfolge resultiert eine insgesamt zwar größere, dafür jedoch dünnbesetzte Hesse-Matrix, die es erlaubt, das Optimierungsproblem durch spezialisierte Methoden effizient zu lösen. In [8] wird ein effizienter Lösungsansatz basierend auf projizierte Gradienten vorgestellt. Die Implementierung erfolgt innerhalb eines echtzeitfähigen MPC-Schemas. Einen MPC-Ansatz mit garantierter Stabilität und Realisierbarkeit für harte Echtzeitbedingungen stellt [9] für lineare Systeme vor. Die explizite modellprädiktive Regelung löst das allgemein parametrisierte Optimierungsproblem vollständig offline und hinterlegt die Lösungen für den nächsten Schritt in einer *Lookup*-Tabelle [10]. Der Regler entnimmt die für den aktuellen Zustand optimale Stellgröße der Tabelle. Ein Nachteil dieses Ansatzes ist, dass er eine vollständige Parametrierung als a-priori bekannt voraussetzt. Zudem wächst der Speicherbedarf der *Lookup*-Tabelle exponentiell mit der Anzahl der freien Parameter.

Dieser Beitrag setzt es sich zum Ziel, die modellprädiktive Regelung unter Zeitoptimalität für Systeme mit hohen

Abtastraten im Bereich unterhalb von 50 ms zu erschließen. Die Implementierung einer effizienten MPC beruht auf drei aufeinander abgestimmten Bausteine:

1. Repräsentation der Steuerfolge und zugehörigen Zustandstrajektorie.
2. Formulierung des Optimalsteuerungsproblems in Form einer (approximativen) Gütefunktion.
3. Numerisches Optimierungsverfahren zur Bestimmung der optimalen Steuerfolge

Den Kern des hier vorgestellten Ansatzes bilden *Timed-Elastic-Bands* (TEB) zur Repräsentation der Zustandstrajektorie und Steuerfolge. In Verbindung mit der Formulierung des Optimalsteuerungsproblems als Ausgleichproblem ermöglicht dieses die Verwendung einer dedizierten, effizienten Levenberg-Marquardt Optimierung.

Der ursprüngliche TEB-Ansatz wurde für die Planung und Optimierung der Trajektorien mobiler Systeme konzipiert. Grundlage ist der Ansatz von [11], welcher einen grob geplanten, rein geometrischen Pfad unter der Einwirkung von internen und externen Kräften deformiert. Interne Kräfte kontrahieren den Pfad im Sinne des kürzesten Weges, externe Kräften stoßen den Pfad von Hindernissen ab. Das *Elastic-Band* wird in [12] auf die Deformation von Trajektorien erweitert. [13] führt das zeitbehaftete TEB ein und formuliert die Trajektorienplanung als Ausgleichsproblem mit dünnbesetzter Struktur. Verletzungen der kinematischen und (kino-)dynamischen Beschränkungen einer Roboterbewegung werden in der Kostenfunktion bestraft. In [14] wird das TEB erfolgreich für die Planung eines Notausweichmanövers im Automotive-Bereich angewendet.

Die Verwendung des TEBs für die modellprädiktive Regelung erfordert eine Erweiterung ihrer bisherigen Repräsentation als reine Trajektorie zu einer über die Systemdynamik untereinander verknüpften Zustands- und Steuerfolge. Dazu werden die Gleichheitsbedingungen der Zustandsgleichung unter Erhaltung der dünnbesetzten Struktur des Optimierungsproblems in die Kostenfunktion integriert. In dieser Formulierung eignet sich der Ansatz zunächst für die in diesem Beitrag behandelten zeitoptimalen Regelungsprobleme. Die Übertragung des TEB-Ansatzes auf verbrauchs- und verlaufsoptimale Gütemaße wird an dieser Stelle nicht betrachtet.

Konventionelle MPC-Ansätze operieren mit einer konstanten Abtastzeit welche bei fester Länge der Steuerfolge zu einem fixierten Zeithorizont führen. Ein MPC-Ansatz, der sich auf das Folgen eines Referenzpfades in minimaler Zeit spezialisiert, wird in [15] vorgestellt. Zeitoptimalität lässt sich hierbei näherungsweise bei hinreichend groß gewähltem Zeithorizont erreichen. Die zeitoptimale Punkt-zu-Punkt-Regelung durch MPC erfordert bisher eine aufwendige, zweistufige Lösung des Optimalsteuerungsproblems [16]. Für einen durch die feste Länge der Steuerfolge vorgegebenen Zeithorizont wird der Abstand des Endzustands zum Zielzustand minimiert. In der äußeren Schleife wird nun die Länge der Steuerfolge solange reduziert, bis der Endzustand die Zielumgebung verfehlt. Im ungünstigsten Fall muss die Länge der Steuerfolge in jedem Abtastschritt neu angepasst werden.

Dagegen behandelt der TEB-Ansatz die Diskretisierungsschrittweite explizit als kontinuierliche Optimierungsgröße und erlaubt dadurch eine die für eine (quasi-)zeitoptimale Regelung notwendige Kontraktion oder Expansion der Steuerfolge in der Zeit.

Abschnitt 2 stellt das für die modellprädiktive Regelung erweiterte TEB vor. Anwendungsbeispiele und Simulationen, sowie die vergleichende Analyse der TEB-Lösung mit der bekannten optimalen Steuerfolge erfolgen in Abschnitt 3. Abschließend erfolgt in Abschnitt 4 eine Zusammenfassung und ein Ausblick.

2 Timed-Elastic-Band (TEB)

2.1 Formulierung des Optimierungsproblems

Ein nichtlineares, autonomes Deskriptor-System mit p Zustandsgrößen und q Eingangsgrößen wird mathematisch wie folgt beschreiben:

$$\mathbf{f}(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t)) = 0\,, \quad \mathbf{x}(t=0) = \mathbf{x}_s \tag{1}$$

Dabei bezeichnet $\mathbf{x} \in \mathbb{R}^p$ den Zustandsvektor und $\mathbf{u} \in \mathbb{R}^q$ den Eingangsvektor. Die diskrete Approximation von (1) mit Hilfe finiter Differenzen $\dot{\mathbf{x}}(t) \approx (\Delta T)^{-1}(\mathbf{x}_{k+1} - \mathbf{x}_k)$ führt zu der Differenzengleichung:

$$\tilde{\mathbf{f}}(\mathbf{x}_{k+1}, \mathbf{x}_k, \mathbf{u}_k, \Delta T) = 0 \quad k = 1, 2, \ldots, n-1 \tag{2}$$

Diese Approximation erlaubt eine diskrete Parametrierung der Zustandstrajektorie und Steuerfolge durch n Paare von Zustands- und Eingangsvektoren mit einer zeitlichen Diskretisierung ΔT. Diese beiden Vektoren bilden zusammen mit der variablen Schrittweite ΔT das TEB $\mathcal{B} \subseteq \mathbb{R}^d$ der Dimension $d = np + (n-1)q + 1$:

$$\mathcal{B} := \{\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2, \mathbf{u}_2, \ldots, \mathbf{x}_{n-1}, \mathbf{u}_{n-1}, \mathbf{x}_n, \Delta T\} \tag{3}$$

Im Gegensatz zu der ursprünglichen Formulierung in [13], bei der die Steuerfolge $\mathbf{u}_k$ durch Invertierung von (2) implizit aus den benachbarten Zuständen $\mathbf{x}_k, \mathbf{x}_{k+1}$ rekonstruiert

wird, wird sie für dynamische Systeme nach (1) als explizite freie Optimierungsvariable hinzugefügt. Zur Verringerung der Anzahl an Optimierungsvariablen wird in (2) und (3) eine konstante Diskretisierungsschrittweite ΔT entlang der gesamten Trajektorie gewählt. Beim TEB unterliegt die Diskretisierungsschrittweite ΔT selber der Optimierung. Zudem wird in jedem neuen Abtastintervall der Regelung die Anzahl der Diskretisierungszeitpunkte n an die Entfernung des aktuellen Zustands zum Zielzustand angepasst. Diese Flexibilität hinsichtlich der zeitlichen Diskretisierung bietet gegenüber herkömmlichen MPC-Ansätzen auf Basis rein diskreter oder kontinuierlicher Modelle, insbesondere bei zeitoptimaler Regelung und Aufgaben mit variablem Zeithorizont einen Vorteil.

[13] formuliert das Optimierungsproblem des TEBs als nichtlineares, gewichtetes Ausgleichsproblem, für welches effiziente Lösungsalgorithmen zur Verfügung stehen. Das Optimierungsziel setzt sich aus $i = 1, 2, \ldots, z$ quadratischen Kostenfunktionen $C_i : \mathcal{B} \times \mathbb{R}^{v \times v} \rightarrow \mathbb{R}^v$ der Form

$$C_i(\mathcal{B}, \boldsymbol{\Sigma}_i) = \mathbf{c}_i(\mathcal{B})^T \boldsymbol{\Sigma}_i \mathbf{c}_i(\mathcal{B}) := \|\mathbf{c}_i(\mathcal{B})\|^2_{\Sigma,i} \tag{4}$$

zusammen. $\mathbf{c}_i : \mathcal{B} \rightarrow \mathbb{R}^v$ ist eine im Allgemeinen vektorielle nichtlineare Funktion, $\boldsymbol{\Sigma}_i \in \mathbb{R}^{v \times v}$ eine positive definite Gewichtungsmatrix.

Die Zustände des TEBs unterliegen der Systemdynamik in (2), welche dem Ausgleichsproblem als Gleichheitsnebenbedingungen hinzugefügt werden:

$$\min_{\mathcal{B}} \quad \sum_{i=1}^{z} \underbrace{\|\mathbf{c}_i(\mathcal{B})\|^2_{\Sigma,i}}_{C_i(\mathcal{B},\boldsymbol{\Sigma}_i)} \tag{5}$$

$$\begin{aligned} \text{N.B.:} \quad & \mathbf{x}_1 = \mathbf{x}_s \\ & \mathbf{x}_n = \mathbf{x}_g \\ & \tilde{\mathbf{f}}_k(\mathbf{x}_{k+1}, \mathbf{x}_k, \mathbf{u}_k, \Delta T) = \mathbf{0} \quad (k = 1, 2, \ldots, n-1) \\ & \mathbf{g}_k(\mathbf{x}_k, \mathbf{u}_k) \geq \mathbf{0} \qquad\quad (k = 1, 2, \ldots, n-1) \end{aligned}$$

Die konstanten Komponenten $\mathbf{x}_1, \mathbf{x}_n$ des TEBs werden durch den Anfangs- $\mathbf{x}_s$ und Endzustand $\mathbf{x}_g$ vorgegeben. Die vektorielle Ungleichheitsnebenbedingung $\mathbf{g}_k : \mathbb{R}^p \times \mathbb{R}^q \rightarrow \mathbb{R}^r$ berücksichtigt r beliebige Zustands- und Stellgrößenbeschränkungen.

Die Minimierung der benötigten Übergangszeit T vom Anfangszustand $\mathbf{x}_0$ in einen gewünschten Zielzustand $\mathbf{x}_g$ lässt sich nach der diskreten Approximation (2) mit $T \approx (n-1)\Delta T$ beschreiben. Für den Kostenterm $\mathbf{c}_1(\mathcal{B})$ gilt:

$$\mathbf{c}_1(\mathcal{B}) = \mathbf{c}_1(\Delta T) = (n-1)\Delta T \tag{6}$$

Gleichung (4) besitzt mit (6) sein Minimum in $\Delta T = 0$, steht jedoch im Konflikt mit der Einhaltung der Systemdynamik und dem Diskretisierungsfehler. Weitere Optimierungsziele C_i, wie beispielsweise verbrauchs- und verlaufsoptimale Ziele, können optional hinzugefügt werden. Durch die Wahl der Gewichtungsmatrizen $\boldsymbol{\Sigma}_i$ lässt sich ein Kompromiss zwischen den Kostentermen einstellen.

Zur effizienteren Lösung wird (5) durch ein Optimierungsproblem mit weichen Nebenbedingungen ersetzt. Weiche Nebenbedingungen sind Bedingungen, deren Verletzung erlaubt ist, aber bestraft wird. Die Gleichheitsnebenbedingung der Systemdynamik (2) wird mit Hilfe der weichen Bestrafungskostenfunktion $\phi(\tilde{\mathbf{f}}_k, \sigma_1)$ angenähert.

$$\phi(\tilde{\mathbf{f}}_k, \sigma_1) = \sigma_1 \tilde{\mathbf{f}}_k^T \mathbf{I} \tilde{\mathbf{f}}_k = \sigma_1 \|\tilde{\mathbf{f}}_k\|_2^2 \tag{7}$$

mit σ_1 als Gewichtungsfaktor. Im Allgemeinen kann anstelle von $\sigma_1 \mathbf{I}$ eine beliebige Gewichtungsmatrix angesetzt werden. Da $\tilde{\mathbf{f}}_k \geq 0$ möglichst für alle Komponenten einzuhalten ist, wird eine gleichmäßige Gewichtung gewählt. Die Ungleichheitsnebenbedingungen $\mathbf{g}_k$ werden durch quadratische Bestrafungsfunktionen ersetzt:

$$\chi(\mathbf{g}_k, \sigma_2) = \sigma_2 \| \min\{[\mathbf{0}, \mathbf{g}_k(\mathbf{x}_k, \mathbf{u}_k)]\} \|_2^2 := \sigma_2 \|\tilde{\mathbf{g}}_k\|_2^2 \tag{8}$$

Der min-Operator wird dazu zeilenweise auf die Matrix $[\mathbf{0}, \mathbf{g}_k(\mathbf{x}_k, \mathbf{u}_k)]$ angewendet. Es gilt $\tilde{\mathbf{g}}_k : \mathbb{R}^p \times \mathbb{R}^q \rightarrow \mathbb{R}^r$. Zur Lösung des Optimierungsproblems wird die stetige Differenzierbarkeit von $\chi(\mathbf{g}_k, \sigma_2)$ für alle verschwindenden Komponenten von $\mathbf{g}_k$ vorausgesetzt. Die Nebenbedingungen $\mathbf{x}_1 = \mathbf{x}_s$ und $\mathbf{x}_n = \mathbf{x}_g$ können durch direktes Einsetzen eliminiert werden. $\mathbf{x}_1$ und $\mathbf{x}_n$ sind daher während der Optimierung konstant.

Für das approximierte, nicht länger restringierte Optimierungsproblem gilt:

$$\mathcal{B}^* = \underset{\mathcal{B}\setminus\{\mathbf{x}_1,\mathbf{x}_n\}}{\arg\min} \underbrace{\sum_{i=1}^{z} C_i(\mathcal{B}, \boldsymbol{\Sigma}_i) + \sum_{k=0}^{n-1} \left[\phi(\tilde{\mathbf{f}}_k, \sigma_1) + \chi(\mathbf{g}_k, \sigma_2)\right]}_{\Phi(\mathcal{B},\boldsymbol{\Sigma}_i,\sigma_1,\sigma_2)} \tag{9}$$

Das optimale TEB $\mathcal{B}^*$ resultiert aus der Minimierung der Kostenfunktion $\Phi(\mathcal{B}, \boldsymbol{\Sigma}_i, \sigma_1, \sigma_2)$ bezüglich der Komponenten des TEBs $\mathcal{B}$. Für $\sigma_{1,2} \rightarrow \infty$ konvergiert die Lösung $\mathcal{B}^*$ gegen ein lokales Minimum des ursprünglichen restringierten Optimierungsproblems (5). Ein Ansatz besteht darin σ_1 und σ_2 mit jeder Iteration sukzessive zu erhöhen, um sich so von außen, im Sinne einer Verletzung der Beschränkungen, der optimale Lösung anzunähern. Für die Online-Optimierung des TEBs ist es in der Regel nicht erforderlich, σ_1 und σ_2 beliebig anwachsen zu lassen, um eine hinreichend gute Approximation der optimalen Lösung zu finden. Der geschlossene Regelkreis unterdrückt nicht nur die von außen auf das System einwirkenden Störungen, sondern regelt ebenso die durch den

Diskretisierungs- und Approximierungsfehler verursachten Abweichungen zwischen dem vom TEB geplanten und tatsächlichem Zustand aus. Für konkrete dynamische Systeme liefert eine Sensitivitätsanalyse Aufschluss über die durch die Approximation in der Optimierung induzierten Zustandsabweichungen. Die Robustheit der Lösung kann durch einen restriktiveren Offset $\boldsymbol{\epsilon} \in \mathbb{R}^r$ in der Bestrafungskostenfunktion $\min\{[\mathbf{0}, \mathbf{g}_k(\mathbf{x}_k, \mathbf{u}_k) - \boldsymbol{\epsilon}]\}$ (8) verbessert werden. Die robustere Lösung ist dann jedoch suboptimal bezüglich des gewählten Gütekriteriums.

2.2 Lösung des Optimierungsproblems

Das nicht länger approximierte Optimierungsproblem (9) setzt sich aus der gewichteten Superposition einzelner quadratischer Kostenterme zusammen. Damit handelt es sich um ein gewichtetes Ausgleichsproblem, welches zur nachfolgenden Beschreibung des Lösungswegs in eine übliche Form gebracht wird.

Zunächst werden die veränderlichen Variablen des TEBs $\mathcal{B}$ in einem Optimierungsvektor $\mathbf{b} \in \mathcal{B} \setminus \{\mathbf{x}_1, \mathbf{x}_n\} \subseteq \mathbb{R}^l$ mit $l = p(n-2)q(n-1) + 1$ zusammengefasst:

$$\mathbf{b} := [\mathbf{u}_1, \mathbf{x}_2, \mathbf{u}_2, \ldots, \mathbf{x}_{n-1}, \mathbf{u}_{n-1}, \Delta T] \tag{10}$$

Anfangs- $\mathbf{x}_1$ und Endzustand $\mathbf{x}_n$ sind gegeben und bleiben innerhalb der Optimierung konstant.

Bestandteile der einzelnen quadratischen Kostenfunktionen aus (9) sind vektorielle nichtlineare Funktionen, die in einer globalen Funktion $\mathbf{e}(\mathbf{c}(\mathcal{B}), \tilde{\mathbf{f}}(\mathcal{B}), \tilde{\mathbf{g}}(\mathcal{B})) : \mathcal{B} \rightarrow \mathbb{R}^r$ mit $r = z + 2(n-1)$ zusammengefasst werden:

$$\mathbf{e}(\mathcal{B}) = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_z, \tilde{\mathbf{f}}_1, \tilde{\mathbf{f}}_2, \ldots, \tilde{\mathbf{f}}_{n-1}, \tilde{\mathbf{g}}_1, \tilde{\mathbf{g}}_2, \ldots, \tilde{\mathbf{g}}_{n-1}]^T \tag{11}$$

Aus Gründen der Übersichtlichkeit wird auf die Angabe einzelner Argumente verzichtet. Die Abhängigkeit der Funktion $\mathbf{e}(\mathcal{B})$ vom vollständigen TEB $\mathcal{B}$ darf nicht auf den Optimierungsvektor $\mathbf{b}$ reduziert werden, da Anfangs- $\mathbf{x}_1$ und Endzustand $\mathbf{x}_n$ ebenfalls in der globalen Kostenfunktionen zu berücksichtigen sind, beispielsweise in (9) mit $k = 1$ und $k = n - 1$. Um diese Abhängigkeit beizubehalten wird $\mathbf{e}(\mathbf{b}) := \mathbf{e}(\mathbf{b}, \mathbf{x}_1, \mathbf{x}_n) := \mathbf{e}(\mathcal{B})$ definiert.

Die Gewichte und Gewichtungsmatrizen der jeweiligen Kostenfunktionen werden in einer Diagonalmatrix $\boldsymbol{\Omega} \in \mathbb{R}^{r \times r}$ zusammengefasst:

$$\Omega = \operatorname{diag}(\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \ldots, \boldsymbol{\Sigma}_z, \sigma_1 \mathbf{I}, \ldots, \sigma_1 \mathbf{I}, \sigma_2 \mathbf{I}, \ldots, \sigma_2 \mathbf{I}) \tag{12}$$

Damit ergibt sich das gewichtete Ausgleichsproblem nach (9) zu:

$$\mathbf{b}^* = \arg\min_{\mathbf{b}} \Phi(\mathbf{b}, \boldsymbol{\Omega}) = \mathbf{e}(\mathbf{b})^T \boldsymbol{\Omega} \mathbf{e}(\mathbf{b}) \tag{13}$$

$\mathbf{b}^*$ stellt die optimale Lösung der veränderlichen TEB-Variablen dar. Ausgleichsprobleme dieser Art werden mit einem nichtlinearen Quasi-Gauß-Newton- beziehungsweise Levenberg-Marquardt-Ansatz gelöst. Durch sukzessive Linearisierungen der Funktion $\mathbf{e}(\mathbf{b})$ wird das Ausgleichsproblem iterativ durch wiederholte Betrachtung linearer Gleichungssysteme gelöst.

Die Funktion $\mathbf{e}(\mathbf{b})$ wird in der Umgebung einer initialen Schätzung $\breve{\mathbf{b}}$ mit Hilfe einer Taylor-Reihe erster Ordnung linearisiert:

$$\mathbf{e}(\breve{\mathbf{b}} + \Delta\mathbf{b}) \approx \mathbf{e}(\breve{\mathbf{b}}) + \mathbf{J}\Delta\mathbf{b} \tag{14}$$

$\mathbf{J} \in \mathbb{R}^{r \times l}$ ist die Jacobi-Matrix von $\mathbf{e}(\mathbf{b})$ berechnet in $\breve{\mathbf{b}}$. Unter Berücksichtigung der Linearisierung (14) in (13) folgt:

$$\begin{aligned}
&\Phi(\breve{\mathbf{b}} + \Delta\mathbf{b}, \boldsymbol{\Omega}) && (15)\\
&= \mathbf{e}(\breve{\mathbf{b}} + \Delta\mathbf{b})^T \boldsymbol{\Omega}\, \mathbf{e}(\breve{\mathbf{b}} + \Delta\mathbf{b}) && (16)\\
&= (\mathbf{e}(\breve{\mathbf{b}}) + \mathbf{J}\Delta\mathbf{b})^T \boldsymbol{\Omega}\, (\mathbf{e}(\breve{\mathbf{b}}) + \mathbf{J}\Delta\mathbf{b}) && (17)\\
&= \mathbf{e}(\breve{\mathbf{b}})^T \boldsymbol{\Omega} \mathbf{e}(\breve{\mathbf{b}}) + 2\mathbf{e}(\breve{\mathbf{b}})^T \boldsymbol{\Omega}\mathbf{J}\Delta\mathbf{b} + \Delta\mathbf{b}^T \mathbf{J}^T \boldsymbol{\Omega}\mathbf{J}\Delta\mathbf{b} && (18)\\
&= d + 2\mathbf{p}^T \Delta\mathbf{b} + \Delta\mathbf{b}^T \mathbf{H}\Delta\mathbf{b} && (19)
\end{aligned}$$

$\mathbf{H} = \mathbf{J}^T \boldsymbol{\Omega}\mathbf{J} \in \mathbb{R}^{l \times l}$ kann als Hesse-Matrix des Ausgleichsproblems interpretiert werden. Die Minimierung von (19) bezüglich $\Delta\mathbf{b}$ erfordert die Lösung des linearen Gleichungssystems:

$$\mathbf{H}\Delta\mathbf{b}^* = -\mathbf{p} \tag{20}$$

In der aktuellen Iteration folgt die Lösung $\mathbf{b}^*$ des Ausgleichsproblems aus der Addition $\mathbf{b}^* = \breve{\mathbf{b}} + \Delta\mathbf{b}^*$.

Eine robuste Erweiterung von Quasi-Gauß-Newton-Verfahren ist die Levenberg-Marquardt-Methode, welche die zu invertierende Hesse-Matrix um einen Dämpfungsfaktor λ erweitert:

$$(\mathbf{H} + \lambda\mathbf{I})\Delta\mathbf{b}^* = -\mathbf{p} \tag{21}$$

Für kleine λ verhält sich der Lösungsalgorithmus wie der Quasi-Gauß-Newton-Ansatz und konvergiert in einer Umgebung der optimalen Lösung lokal quadratisch, für große λ geht (21) in einen Gradientenabstieg über, der zwar langsamer, dafür jedoch garantiert konvergiert. Über die Anpassung von λ wird nach jeder Iteration eine Reduktion der Kosten erzwungen. Zusätzlich können über den Dämpfungsfaktor singuläre Konfigurationen in Form einer Regularisierung behandelt werden, bei denen $\mathbf{H}$ für sich alleine keinen vollen Rank besitzt. In der Literatur finden sich unterschiedliche Vorgehensweisen, sodass im Folgenden ein im Zusammenhang mit dem TEB funktionierender Algorithmus vorgestellt wird. Dieser basiert auf einer effizienten Implementierung in [17]. Für vertiefende

Details zum Levenberg-Marquardt-Algorithmus sei auf [18] verwiesen.

Prozedur LevenbergMarquardt($\mathbf{b}, \boldsymbol{\Omega}$)
  Berechne $\mathbf{J}$, $\mathbf{H}$ und $\mathbf{p}$
  $\lambda \leftarrow 10^{-5} \max(\operatorname{diag} \mathbf{H})$
  $\rho \leftarrow 0$
  $\nu \leftarrow 2$
  Für alle Iterationen 1 bis I_{LM} :
    Wiederhole
      Löse $(\mathbf{H} + \lambda\mathbf{I})\Delta\mathbf{b}^* = -\mathbf{p}$
      $\hat{\rho} \leftarrow (\Phi(\mathbf{b}) - \Phi(\mathbf{b} + \Delta\mathbf{b}^*))$
      $\rho \leftarrow \hat{\rho}(\Delta\mathbf{b}^{*T}(\lambda\Delta\mathbf{b}^* + \mathbf{p}))^{-1}$
      Wenn $\rho > 0$ **dann**
        $\mathbf{b}^* \leftarrow \mathbf{b} + \Delta\mathbf{b}^*$
        Berechne $\mathbf{J}$, $\mathbf{H}$ und $\mathbf{p}$
        $\alpha \leftarrow \max(\min(2/3, 1 - (2\rho - 1)^3), 1/3)$
        $\lambda \leftarrow \lambda\alpha$
        $\nu \leftarrow 2$
      sonst
        $\lambda \leftarrow \lambda\nu$
        $\nu \leftarrow 2\nu$
    bis $\rho > 0$
  Rückgabe (Sub-)Optimales TEB $\mathbf{b}^*$

Die Jacobi-Matrix $\mathbf{J}$ wird entweder analytisch vorgegeben oder numerisch berechnet. In den nachfolgenden Beispielen wird sie numerisch über zentrale Differenzen $\partial\mathbf{e}(\breve{\mathbf{b}})/\partial b_i \approx 0{,}5h^{-1}[\mathbf{e}(\breve{\mathbf{b}} + \mathbf{h}_i) - \mathbf{e}(\breve{\mathbf{b}} - \mathbf{h}_i)], b_i \in \breve{\mathbf{b}}$, bestimmt. $\mathbf{h}_i = [0, \ldots, 0, h, 0, \ldots, 0]^T$ beinhaltet an i. Stelle die Schrittweite $h = 10^{-9}$.

In jeder Iteration des Levenberg-Marquardt-Algorithmus wird zunächst das lineare Gleichungssystem (21) gelöst. Anschließend wird überprüft, ob sich der Gesamtfehler $\Phi(\mathbf{b} + \Delta\mathbf{b}^*)$ für die neue Lösung reduziert. In diesem Fall ist die aktuelle Iteration beendet und der Dämpfungsfaktor wird für zukünftige Iterationen verringert. Ist dies nicht der Fall, wird (21) mit doppeltem λ wiederholt gelöst. Nach unseren Erfahrungen führen bereits wenige, in der Regel circa 5-10 Iterationen I_{LM} zu einer hinreichend guten Näherung der optimalen Steuerfolge.

Innerhalb des Levenberg-Marquardt-Algorithmus wird wiederholt das lineare Gleichungssystem (21) gelöst. Daher hat der eingesetzte Lösungsalgorithmus einen wesentlichen Einfluss auf die benötigte Optimierungszeit. Die zu invertierende Matrix $(\mathbf{H} + \lambda\mathbf{I})$ ist symmetrisch positiv semidefinit, sodass sich die Cholesky-Zerlegung zur effizienten und robusten Lösung heranziehen lässt [19]. Diese zerlegt die reelle Matrix $(\mathbf{H} + \lambda\mathbf{I}) = \mathbf{L}\mathbf{L}^T$ in das Produkt einer unteren Dreiecksmatrix $\mathbf{L}$ mit der zugehörigen Transponierten $\mathbf{L}^T$. Die Lösung $\Delta\mathbf{b}^*$ in (21) wird anschließend mit dem Hilfsvektor $\bar{\mathbf{b}}$ durch Vorwärts- und Rückwärtseinsetzen gelöst:

$$\mathbf{L}\bar{\mathbf{b}} = -\mathbf{p} \tag{22}$$

$$\mathbf{L}^T\Delta\mathbf{b}^* = \bar{\mathbf{b}} \tag{23}$$

Die Struktur der Hesse-Matrix $\mathbf{H}$ und damit auch der Matrix $(\mathbf{H} + \lambda\mathbf{I})$ ist bei geeigneter Wahl der Kostenfunktionen $C_i(\mathbf{b})$ dünnbesetzt, sodass sich zur Lösung dedizierte Verfahren einsetzen lassen. Die Funktionen $\tilde{\mathbf{f}}_k$ und $\tilde{\mathbf{g}}_k$ berücksichtigen nach (5) mit Ausnahme von ΔT ausschließlich benachbarte Zustände, sodass die Jacobi-Matrix $\mathbf{J}$ nur wenige von Null verschiedene Elemente aufweist. Da $\mathbf{H} = \mathbf{J}^T\boldsymbol{\Omega}\mathbf{J}$ gilt und $\boldsymbol{\Omega}$ eine Diagonal-/Bandmatrix darstellt, ist die Hesse-Matrix ebenfalls dünnbesetzt. In Folge der verschachtelten Anordnung der Zustands- und Eingangsvektoren in $\mathbf{b}$ ist $\mathbf{H}$ zusätzlich näherungsweise band-diagonal. Abbildung 5 zeigt eine beispielhafte Besetzung der Hesse-Matrix für das Szenario in Abschnitt 3.1. Das Beispiel zeigt, dass als einzige die Komponente ΔT des TEBs über die Systemdynamik (2) mit allen übrigen Zuständen $\mathbf{x}_k$ verknüpft ist. Diese Kopplung führt zu einer einzelnen dichtbesetzten Spalte und Reihe in der Hesse-Matrix. Durch eine Erhöhung der Optimierungsvariablen und damit der Dimension d des TEBs in Form von verschiedenen $\Delta T_k, k = 1, 2, \ldots, n-1$, anstelle einer zeitlich äquidistanten Diskretisierung ΔT, wird eine vollständige Bandstruktur erzielt. Diese Erweiterung führt jedoch bei den hier untersuchten Beispielen zu keiner signifikanten Verbesserung der Lösung und wird daher an dieser Stelle nicht weiter betrachtet.

Dedizierte Algorithmen für Cholesky-Zerlegungen dünnbesetzter Matrizen werden in [19] zusammengefasst. Die prinzipielle Vorgehensweise besteht in einer anfänglichen graphenbasierten Umsortierung der Matrix, um die Anzahl der von Null verschiedenen Elemente in der späteren Dreiecksmatrix $\mathbf{L}$ a-priori zu reduzieren. Anschließend wird über eine symbolische Cholesky-Zerlegung das vollständige Lösungsmuster bestimmt. Da an dieser Stelle die numerischen Werte selber nicht von Interesse sind, beschränkt sich der Algorithmus ausschließlich auf symbolische Manipulationen, die neue Einträge in $\mathbf{L}$ erzeugen. Die eigentliche rechenintensive numerische Cholesky-Zerlegung erfolgt abschließend nach dem zuvor symbolisch ermittelten Lösungsmuster. Überflüssige Operationen mit Nullelementen werden dadurch von vornherein ignoriert.

Das g2o-Framework [17] ist ein frei verfügbares generisches C++ Optimierungsframework mit aktuellen Algorithmen zum Lösen dünnbesetzter, nichtlinearer Aus-

gleichsprobleme und eignet sich daher besonders für die Optimierung des TEBs. Ausgleichsprobleme nach (9) werden dabei als Hypergraphen formuliert. Ein Hypergraph besitzt im Gegensatz zu konventionellen Graphen sogenannte Hyperkanten welche nicht nur zwei, sondern eine beliebige Anzahl an Knoten miteinander verbinden. Dabei repräsentieren Knoten die Optimierungsvariablen (Zustände und Eingangsgrößen), welche über die Kanten, in diesem Fall lokale Kostenfunktionen verknüpft werden. Diese lokalen Kostenfunktionen entsprechen den einzelnen quadratischen Termen in (9), die im Idealfall nur auf wenigen, im TEB zudem benachbarten, Optimierungsvariablen beruhen. Diese graphenbasierte Implementierung des Ausgleichsproblems erlaubt neben einer modularen Handhabung von Kostenfunktionen auch einen Geschwindigkeitsvorteil bei großen Optimalsteuerungsproblemen. So gehen in die Berechnung der Jacobi-Matrix einer Hyperkante ausschließlich die angebundenen Knoten ein. Unnötige Nullelemente werden damit von vornherein übersprungen.

Abbildung 1 zeigt die Berechnungszeit für zehn Levenberg-Marquardt-Iterationen bei verschiedenen Lösungsansätzen des linearen Gleichungssystems (21) über die Anzahl der Elemente im Optimierungsvektor $\mathbf{b}$ (siehe Abschnitt 3 für den verwendeten Rechner). Die Implementierung des TEBs im g2o-Framework basiert auf *CSparse* zur dünnbesetzten Cholesky-Zerlegung [19]. Für die gewöhnliche und dünnbesetzte Cholesky-Zerlegung ohne Graphenrepräsentation werden als Referenz effiziente Cholesky-Implementierungen der *Eigen* C++ Bibliothek [20] verwendet. Der Vergleich in Abbildung 1 belegt, das dass g2o-Framework bereits ab circa 200 Optimierungsvariablen die identische Lösung effizienter als Implementierungen mit Matrixrepräsentationen berechnet.

Neben dem hier vorgestellten Levenberg-Marquardt-Algorithmus wurde eine alternative TEB-Implementierung mit dem Levenberg-Marquardt-Algorithmus der *Eigen* Bibliothek durchgeführt. Gewichte werden hier nicht explizit berücksichtigt, sondern sind in den Kostentermen enthalten. Die Linearisierung erfolgt in $\Phi(\breve{\mathbf{b}} + \Delta\mathbf{b})$ anstatt in $\mathbf{e}(\breve{\mathbf{b}} + \Delta\mathbf{b})$. Es zeigte sich eine langsamere Konvergenz bei gleichzeitig langsameren Optimierungszeiten. Eine weitere Implementierung verwendet den Gauß-Newton-Algorithmus. Bei einer ungünstigen und im Zustandsraum weit entfernten initialen Lösung führte dieser Ansatz nicht immer zur Konvergenz.

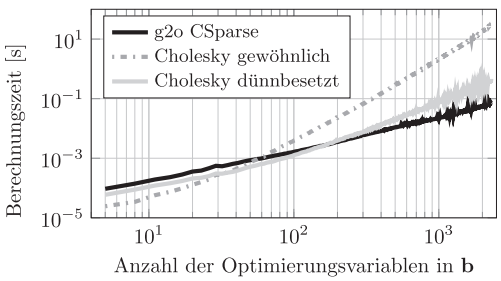

Abbildung 1: Berechnungszeit von zehn Levenberg-Marquardt-Iterationen verschiedener Ansätze zur Cholesky-Zerlegung.

2.3 Timed-Elastic-Band-Algorithmus

Im Folgenden wird der Einsatz des TEBs im geschlossenen Regelkreis beschrieben.

Abbildung 2 zeigt das zugehörige Blockschaltbild. In Abhängigkeit des aktuellen Zustandes der Regelstrecke $\mathbf{x}_k$ ermittelt der TEB-Algorithmus die optimale Trajektorie im Sinne des Optimierungsproblems (9) zur Überführung in den gewünschten Zielzustand $\mathbf{x}_g$. Für nicht vollständig messbare Zustandsvektoren $\mathbf{x}_k$, wird der Zustand, unter Voraussetzung der vollständigen Beobachtbarkeit, durch einen Zustandsbeobachter geschätzt. Die optimierte Trajektorie $\mathbf{b}^*$ dient gleichzeitig als Startlösung $\mathbf{b}$ für die Optimierung im darauffolgenden Abtastintervall. Der TEB-Algorithmus lässt sich folgendermaßen zusammenfassen:

1: **Prozedur** TimedElasticBand($\mathbf{b}, \mathbf{x}_s, \mathbf{x}_g$)
2: Initialisiere bzw. aktualisiere TEB
3: $\sigma \leftarrow \sigma^{(0)}$ ▷ Hier: $\sigma = \sigma_1 = \sigma_2$
4: **Für alle** Iterationen 1 bis I_{teb} :
5: Anpassung der Dimension d des TEBs
6: $\mathbf{b}^* \leftarrow$ LevenbergMarquardt($\mathbf{b}, \sigma$)
7: $\sigma \leftarrow \kappa \cdot \sigma$
Rückgabe (Sub-)Optimales TEB $\mathbf{b}^*$

Vor dem ersten Abtastintervall muss die Startlösung der Optimierung $\mathbf{b}$ geeignet initialisiert werden. Erste Erfahrungen zeigen, dass für viele Probleme eine lineare Interpolation $\mathbf{x}_k = \mathbf{x}_s + kn^{-1}(\mathbf{x}_g - \mathbf{x}_s)$ zwischen Anfangs- $\mathbf{x}_s$ und Endzustand $\mathbf{x}_g$ bei zunächst verschwindender Steuerfolge $\mathbf{u}_k = 0$ zum Erfolg führt. Noch einfacher kann alternativ $\mathbf{x}_k, k \in 1, 2, \ldots, n-1$ durchgängig mit dem Anfangszustand $\mathbf{x}_s$ initialisiert werden. Diese Initialisierungsmöglichkeiten lassen sich jedoch nicht für beliebige dynamische Systeme verallgemeinern, da Newton-basierte Lö-

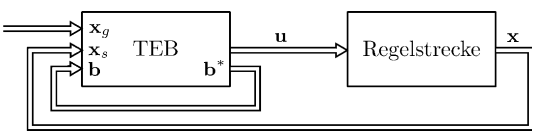

Abbildung 2: TEB im geschlossenen Regelkreis.

sungsansätze ausschließlich lokale Minima finden. In diesem Fall ist es erforderlich, die TEB-Optimierung durch eine globale Planung der Trajektorie zu initialisieren. Für alle weiteren Abtastintervalle dient das TEB $\mathbf{b}^*$ der vorherigen Optimierung zur Initialisierung der neuen Optimierung. Dabei wird das TEB aktualisiert, indem die erste und letzte Komponente des TEBs $\mathbf{x}_1$ und $\mathbf{x}_n$, mit dem aktuellen Zustand $\mathbf{x}_s$ und dem Endzustand $\mathbf{x}_g$ besetzt werden. An dieser Stelle können ggf. erweiterte *Warmstart*-Methoden eingesetzt werden, um die Initialisierung zu verbessern. Die Gewichte der Bestrafungsfunktionen (7) und (8) werden mit einem Startwert $\sigma_1 = \sigma_2 = \sigma^{(0)}$ initialisiert.

Im Anschluss an die Initialisierung erfolgt die Optimierung des TEBs. Hierzu wird über eine Schleife mit I_{teb} TEB-Iterationen iteriert. Der erste Schritt besteht in der Anpassung der Anzahl der Diskretisierungszeitpunkte n und damit der Dimension des TEBs. Dies hat den Vorteil, dass der Rechenaufwand zur Berechnung der Optimalsteuerfolge mit geringerem räumlichen und zeitlichen Abstand zum Zielzustand sinkt. Somit steigt bei gleicher Rechendauer die Anzahl der Iterationen und mit ihr die Güte der Optimierung. Störungen oder Änderungen des Zielzustandes führen umgekehrt zu einer Erhöhung der Dimension des TEBs, um bei einer länger werden Trajektorie dennoch den Diskretisierungsfehler zwischen den Zuständen konstant zu halten. Ein weiterer Vorteil der Adaption der Dimension und Schrittweite ist die Vermeidung numerischer Instabilitäten und daraus resulierender Probleme. Bei einer zu hohen Auflösung der Trajektorie enstehen andernfalls Probleme durch zu große und kleine numerische Werte, besonders bei der Division durch ein zu kleines Zeitintervall ΔT. Die Anpassung erfolgt auf Basis der Bewertung des aktuellen Zeitintervalls ΔT:

- Falls $\Delta T > \Delta T_{soll} + \Delta T_{hyst} \wedge n > n_{min}$ wird der Zustand $\mathbf{x}_{n-1}$ entfernt.
- Falls $\Delta T < \Delta T_{soll} - \Delta T_{hyst}$ wird ein neuer Zustand an $(n-1)$. Stelle eingefügt. Dabei wird dieser durch Interpolation zwischen den benachbarten Zuständen initialisiert.

Nach der Dimensionsanpassung werden I_{LM} Iterationen des Levenberg-Marquardt-Algorithmus zur Optimierung des TEBs durchgeführt. Nach der Betrachtung der Berechnungszeiten in Abschnitt 2.2 erfolgt die Optimierung für die gezeigten Beispiele mit dem g2o-Framework [17]. In jedem Durchgang der Schleife werden die Bestrafungsgewichte σ um den Faktor κ erhöht.

Der TEB-Algorithmus führt mit den vorherigen Überlegungen zu einem *elastischen Band* im Zustandsraum, dessen Konvergenzverhalten sich durch die Wahl von I_{teb}, I_{LM} und κ einstellen lässt. Damit sich das TEB bei der vorgestellten Implementierung robust zusammenzieht, bedarf es mindestens eines Optimierungszieles C_k welches eine Kontraktion der TEB-Zustände in Raum oder Zeit begünstigt. Als naheliegende Kostenfunktion dienen entweder der euklidische Abstand der Zustände (kürzeste Trajektorie) oder die insgesamt benötigte Zeit (schnellste Trajektorie, siehe (6)).

Die Stabilität des TEB-Algorithmus hängt von der Konvergenz der unterlagerten Optimierung ab. Unter der Annahme, dass das Minimum des nicht restringierten Problems mit dem Minimum des restringierten Problems übereinstimmt und die Optimierung in dieses globale Optimum konvergiert ist, besitzt die Kostenfunktion die Eigenschaften einer Lyapunovfunktion. Eine Konvergenz der lokalen nichtlinearen Optimierung in das globale Optimum hängt wegen der nichtkonvexen Kostenfunktion (9) von der Anfangslösung ab. Eine Aussage über den Einzugsbereich der globalen Lösung für allgemeine Systeme (1) ist nicht möglich. Für die in diesem Beitrag betrachteten Systeme ist das wie oben beschrieben initialisierte TEB stets in das globale Optimum konvergiert. Der Nachweis der Stabilität unter den obigen Annahmen wird in dieser Arbeit nicht geführt. Für ereignisbasierte MPCs bei denen eine variable Abtastzeit aus asynchronen Mess- und Stellsignalen erwächst, hat [21] die Stabilität des geschlossenen Regelkreises nachgewiesen. Die Übertragung der dortigen Beweisführung auf den TEB-Algorithmus bleibt zukünftigen Untersuchungen vorbehalten.

3 Beispiele und Simulationen

Die Leistungsfähigkeit des vorgestellten TEB-Algorithmus wird in diesem Abschnitt anhand von Beispielen untersucht. Dabei werden sowohl Simulationen für die optimierte Steuerfolge im offenen Regelkreis, als auch für das TEB im geschlossenen Regelkreis nach Abbildung 2 durchgeführt. Die Simulation der Regelstrecken erfolgt in *Matlab Simulink* über die kontinuierliche Zustandsraumdarstellung (1) und dem *ode45*-Algorithmus. Der auf dem diskretisierten Systemmodell basierende TEB-Algorithmus zur Regelung erfolgt in *C++* (3,4 Ghz Intel i7 CPU).

In den vorgestellten Beispielen wird jeweils im Sinne einer zeitoptimalen Regelung die quadratische Kostenfunktion nach (4) mit (6) gewählt. Für die Parameter des TEB-Algorithmus werden $\Delta T_{soll} = 0{,}05\,\text{s}$, $\Delta T_{hyst} = 0{,}03\,\text{s}$, $\sigma^{(0)} = 1$, $\kappa = 2$ und $n_{min} = 8$ gewählt. Falls nicht anders angegeben, gilt für die Anzahl der Iterationen $I_{teb} = 3$ und $I_{LM} = 10$. Als Initialisierung wird $\mathbf{x}_k = \mathbf{x}_s$, $\mathbf{u}_k = \mathbf{0}$ $(k = 1, 2, \ldots, n-1)$, $\Delta T = \Delta T_{soll}$ und $\mathbf{x}_n = \mathbf{x}_g$ vorgegeben.

3.1 Dreifach-Integrator

Dieses Beispiel betrachtet das lineare, dynamische System $0{,}2\ddot{x}(t) = u(t)$. Mit dem Zustandsvektor $\mathbf{x} = [x(t), \dot{x}(t), \ddot{x}(t)]^T$ lässt sich die Zustandsraumdarstellung angeben:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} \tag{24}$$

Die Zeitargumente werden im Folgenden aus Gründen der Übersichtlichkeit weggelassen. Für das Optimierungsproblem des TEB-Algorithmus wird (24) diskretisiert:

$$\tilde{\mathbf{f}}(\mathbf{x}_{k+1}, \mathbf{x}_k, u_k, \Delta T) = (\Delta T)^{-1}(\mathbf{x}_k - \mathbf{x}_{k+1}) + \mathbf{A}\mathbf{x}_k + \mathbf{b}u_k \tag{25}$$

Unter Berücksichtigung einer Stellgrößenbeschränkung wird

$$|u| \leq 1 \rightarrow g_k(u) = -|u| + 1 \tag{26}$$

gefordert.

Die Analyse vergleicht die über das TEB optimierte Zustands- und Steuerfolge mit der des analytisch berechneten, zeitoptimalen Zweipunktreglers.

Die Regelungsaufgabe besteht aus der Überführung des Systems von $\mathbf{x}_s = [4, 2, -1]^T$ nach $\mathbf{x}_g = \mathbf{0}$. Abbildung 3 stellt den zeitlichen Verlauf der Zustandsgrößen für den geschlossenen Regelkreis (GRK), den offenen Regelkreis (ORK) und den analytischen Zweipunktregler dar. Die zugehörigen Stellgrößen am Eingang der Strecke sind in Abbildung 4 dargestellt. Die Abbildungen zeigen ein *Rattern* des Zweipunktreglers nach Erreichen des gewünschten Endzustands. Bei der Regelung mit dem TEB werden die Zeit-Kosten (6) durch eine Skalierung mit der Anzahl an Diskretisierungszeitpunkten angepasst. In der Nähe des Endzustandes wird die untere Schranke erreicht. Dies führt durch ein fallendes ΔT in Relation zu den Bestrafungstermen zu veringerten Kosten und damit implizit einem glatteren Verlauf der Stellgröße, auch wenn eine

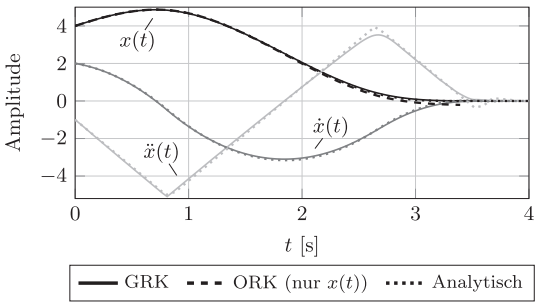

Abbildung 3: Zustandsgrößenverlauf des Dreifach-Integrators.

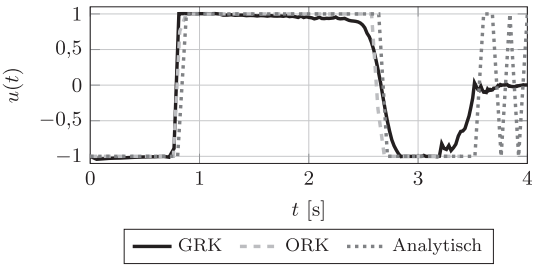

Abbildung 4: Eingangsgrößenverlauf am Dreifach-Integrator.

(quasi-)zeitoptimale Regelung zur Stabilisierung um einen festen Arbeitspunkt nur bedingt passend ist.

Die Trajektorie $x(t)$ im offenen Regelkreis (siehe Abbildung 3) verfehlt aufgrund der nicht kompensierten Diskretisierungs- und Approximationsfehlern den Endzustand mit einer Abweichung von 0,2. Das TEB im geschlossenen Regelkreis korrigiert diese Abweichung und überführt das System in den gewünschten Endzustand. Zur Bewertung der Regelung im Vergleich zur analytischen Referenz wird das R^2-Bestimmtheitsmaß berechnet. Für die Zustandstrajektorien des geschlossenen Regelkreises ergeben sich: $R^2(x) = 0{,}99$, $R^2(\dot{x}) = 0{,}97$ und $R^2(\ddot{x}) = 0{,}93$. Die Qualität der Differentiationen lässt durch die angewendete Diskretisierung leicht nach.

Abbildung 5 zeigt das Besetzungsmuster der Hesse-Matrix des Optimierungsproblems. Die Anordnung der Optimierungsvariablen in den Zeilen und Spalten der Matrix folgt der Reihenfolge in $\mathbf{b}$. Die Matrix ist dünnbesetzt mit circa 6 % von Null verschiedenen Elementen.

3.2 Van-der-Pol Oszillator

Der Van-der-Pol Oszillator stellt ein schwingungsfähiges System zweiter Ordnung mit nichtlinearer Dämpfung dar. Das Systemverhalten wird durch die Differentialgleichung $\ddot{x}(t) + (x(t)^2 - 1)\dot{x}(t) + x(t) = u(t)$ beschrieben. Die Sprungantwort des Oszillators ist in Abbildung 6 darge-

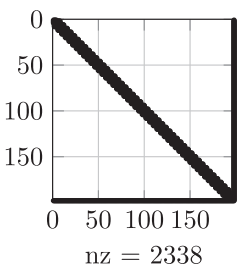

Abbildung 5: Besetzung der Hesse-Matrix.

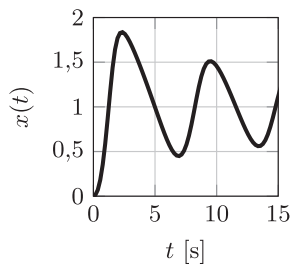

Abbildung 6: Sprungantwort des Van-der-Pol-Systems.

stellt. Das System lässt sich mit dem Zustandsvektor $\mathbf{x} = [x(t), \dot{x}(t)]^T$ unter Vernachlässigung der Zeitargumente in die Form

$$\dot{\mathbf{x}} = \boldsymbol{\xi}(\mathbf{x}, u) = \begin{bmatrix} \dot{x} \\ -(x^2 - 1)\dot{x} - x - u \end{bmatrix} \tag{27}$$

überführen. Für die diskrete Beschreibung folgt:

$$\tilde{\mathbf{f}}(\mathbf{x}_{k+1}, \mathbf{x}_k, u_k, \Delta T) = \Delta T^{-1}(\mathbf{x}_k - \mathbf{x}_{k+1}) + \boldsymbol{\xi}(\mathbf{x}_k, u_k) \tag{28}$$

Die Stellgröße wird durch $g_k(u) = -|u| + 1$ auf $|u| \leq 1$ beschränkt.

Als Referenz wird das Optimierungsproblem in ein kontinuierliches Randwertproblem (RWP) überführt und eine zeitoptimale Steuerfolge generiert. Für die Umsetzung ist eine Dimensionserweiterung und Glättung der sign-Funktion im Eingangsraum notwendig. Die Zeitoptimalität als Optimierungsziel wird durch eine Zeittransformation berechenbar. Für vertiefende Details sei auf [22] verwiesen. Das RWP wird in *Matlab* unter Verwendung des *bvp4c*-Algorithmus gelöst.

Zunächst wird das Übergangsverhalten des offenen und geschlossenen Regelkreises von $\mathbf{x}_s = \mathbf{0}$ nach $\mathbf{x}_g = [1, 0]^T$ unter Variation der TEB-Iterationen I_{teb} betrachtet. Abbildung 7 zeigt den Zustandsgrößenverlauf des offenen und geschlossenen Regelkreises, sowie die Lösung des RWP als Referenz. Die dazugehörigen Stellgrößen am Eingang des Systems sind in Abbildung 8 dargestellt. Die Anzahl der TEB-Iterationen I_{teb} wird für den ORK und GRK variiert: $I_{teb} \in \{2, 3, 5, 10, 20, 30\}$. Die Verläufe $u(t)$ und $\dot{x}(t)$ des ORK zeigen eine quasi-stetige Deformation in Richtung der Referenz bei steigendem I_{teb}. Im GRK liegen die Verläufe dagegen fast vollständig übereinander. Für alle I_{teb} ergibt eine Bewertung der Zustandstrajektorien im GRK mit der Referenz: $R^2(x) \approx 0{,}99$ und $R^2(\dot{x}) \approx 0{,}94$. Das TEB verwendet in jedem Abtastintervall die zuvor berechnete Lösung zur Initialisierung, sodass die Güte der Lösung mit der Anzahl der zurückliegenden Abtastintervalle steigt. Dieses Beispiel zeigt, dass bereits nach wenigen Iterationen eine suboptimale Lösung vorliegt, die Beschränkungen und Systemdynamiken berücksichtigt. Im geschlossenen Regelkreis lassen sich nur geringfügige Abweichungen zur offline, numerisch berechneten, optimalen Lösung feststellen. In der folgenden Simulation des geschlossenen Regelkreises wird eine Sprungfolge für den Zustand $x_{soll}(t)$ vorgeben. Es gilt $\dot{x}_{soll}(t) = 0$. Dem TEB ist ausschließlich der aktuelle Sprung, nicht jedoch die Folgesprünge bekannt. Erst beim Umschalten auf den nächsten Sprung wird $\mathbf{x}_g$ neu vorgegeben. Alle weiteren Zustände bleiben im TEB-Algorithmus unverändert. Die resultierenden Zustandstrajektorien sind in Abbildung 9 dargestellt. Abbildung 10 zeigt den Eingangsgrößenverlauf. An den Umschaltzeitpunkten wird der Endzustand verändert, was unmittelbar zu einer Verlängerung der benötigten Zeit und damit zu einer Vergrößerung der Dimension des TEBs führt. Abbildung 11 stellt diese Größenänderung beim vorliegenden Szenario über die Zeit dar. Die Größe des TEBs hat einen direkten Einfluss auf die Optimierungs-

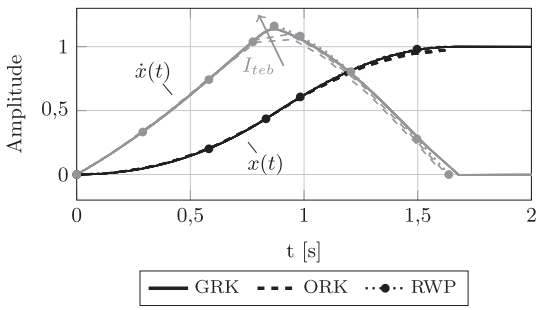

Abbildung 7: Zustandsgrößenverlauf des Van-der-Pol-Systems bei steigendem I_{teb}.

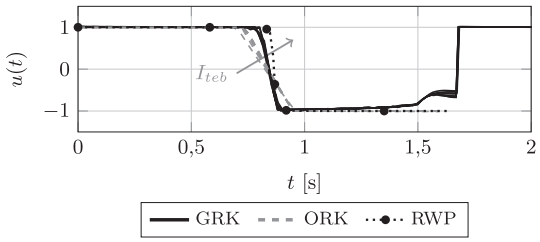

Abbildung 8: Eingangsgrößenverlauf beim Van-der-Pol-System bei steigendem I_{teb}.

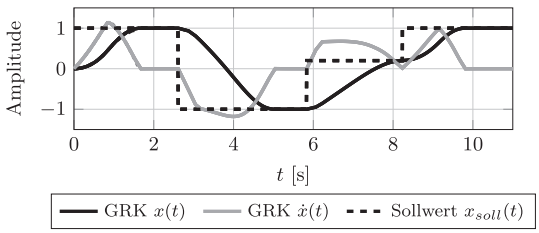

Abbildung 9: Systemantwort des Van-der-Pol-Systems auf eine Sprungfolge.

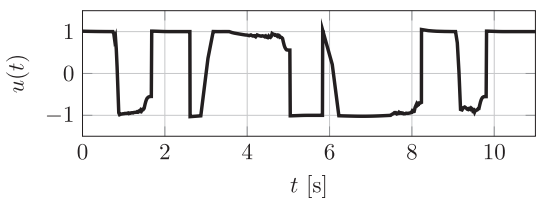

Abbildung 10: Stellgröße am Van-der-Pol-System bei der Sprungfolge.

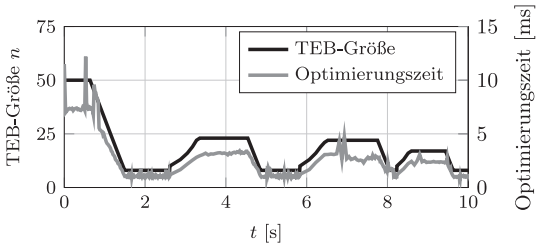

Abbildung 11: Laufzeit und TEB-Größe beim Van-der-Pol-System.

zeit pro Abtastintervall. In der Nähe des Endzustandes gilt $n = n_{min}$, sodass hier I_{teb} zur Verbesserung der Güte erhöht werden kann. In den vorliegenden Beispielen wird auf diese Anpassung verzichtet.

3.3 Mobiler Roboter

In diesem Beispiel dient das TEB zur Trajektorienplanung und Navigation eines mobilen Roboters in Gegenwart eines statischen Hindernisses. Bei dem simulierten Roboter handelt es sich um einen Pioneer 3DX, dessen Systembeschreibung und Parameter [23] entnommen sind. Die nicht-holonome Kinematik, sowie die Dynamik des Differentialantriebs werden durch folgende Gleichungen beschrieben:

$$\dot{x} = 0{,}05\,(\dot{\theta}_r + \dot{\theta}_l)\cos\beta$$
$$\dot{y} = 0{,}05\,(\dot{\theta}_r + \dot{\theta}_l)\sin\beta$$
$$\dot{\beta} = 0{,}33\,(\dot{\theta}_r - \dot{\theta}_l)$$
$$\begin{bmatrix}\ddot{\theta}_l\\ \ddot{\theta}_r\end{bmatrix} = 10^{-5}\begin{bmatrix}-4{,}1 & 1{,}6\\ 1{,}6 & -4{,}1\end{bmatrix}\begin{bmatrix}\dot{\theta}_l\\ \dot{\theta}_r\end{bmatrix} + \begin{bmatrix}11{,}7 & -4{,}6\\ -4{,}6 & 11{,}7\end{bmatrix}\begin{bmatrix}M_l\\ M_r\end{bmatrix} \tag{29}$$

Dabei stellen x, y und β Position und Orientierung in der Ebene dar. $\dot{\theta}_l$ und $\dot{\theta}_r$ sind die Winkelgeschwindigkeiten des linken und rechten Rades. Der Roboter lässt sich über die Radmomente M_l und M_r antreiben. Mit dem Zustandsvektor $\mathbf{x} = [x, y, \beta, \dot{\theta}_l, \dot{\theta}_r, M_l, M_r]^T$ lässt sich das System in die Darstellung $\dot{\mathbf{x}} = \boldsymbol{\xi}(\mathbf{x}, \mathbf{u})$ überführen und nach (28) diskretisieren. Stellgrößenbeschränkungen $|M_l| \leq 1$ und $|M_r| \leq 1$ werden komponentenweise wie in (26) behandelt. Die Hindernisvermeidung wird über eine Zustandsbeschränkung eingefügt, sodass ein Abstand von 0, 5 m eingehalten werden soll. In diesem Szenario wird ein punktförmiges Hindernis bei $x_h = 2,5$ m und $y_h = -0,01$ m berücksichtigt. Zur Glättung wird ausschließlich die orthogonale Projektion des Abstandes auf das TEB betrachtet. Mit diesen Betrachtungen sei

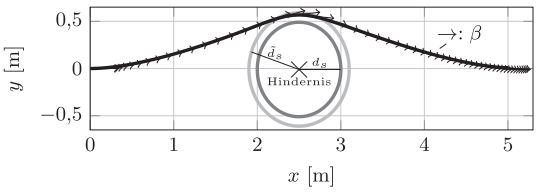

Abbildung 12: Zurückgelegte Trajektorie des mobilen Roboters.

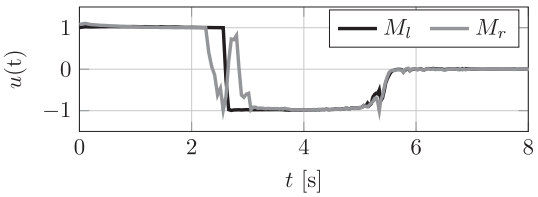

Abbildung 13: Gestellte Radmomente am mobilen Roboter.

$$\mathbf{g}_k(\mathbf{x}_k, \mathbf{u}_k) = \begin{bmatrix} ||\mathbf{d}_k||\,|\sin(\operatorname{atan2}(\Delta y_k, \Delta x_k) - \beta_k)| - d_s \\ -|M_l| + 1 \\ -|M_r| + 1 \end{bmatrix} \tag{30}$$

Dabei ist $\mathbf{d}_k = [\Delta x_k, \Delta y_k]^T = [x_k - x_h, y_k - y_h]^T$ der Abstandsvektor vom Hindernis zur Postion in $\mathbf{x}_k$. d_s stellt den minimal einzuhaltenden Abstand dar. Durch das Hinzufügen vieler Nebenbedingungen kann in Folge der *weichen* Approximation nicht immer eine vollständige Einhaltung garantiert werden. Der Kompromiss zwischen den Nebenbedingungen wird durch die Wahl der Gewichte σ eingestellt. Bei einem einheitlichem σ fallen die Kostenanteile für die Abstandsbestrafung geringer aus, da eine geringe Anzahl an Zuständen betroffen ist. Um einer geringen Verletzung entgegenzuwirken, wird der minimale Abstand $\tilde{d}_s = d_s + \epsilon$ mit $\epsilon = 0{,}1$ m vorgegeben. Abbildung 13 zeigt die zurückgelegte Trajektorie des Roboters im geschlossenen Regelkreis. Die Orientierung β_k wird durch Pfeile verdeutlicht. Die Initialisierung des TEBs durch eine kürzeste Gerade vom Start zum Ziel ignoriert zunächst das Hindernis. Durch das Hinzufügen des Hindernisses bei $x_h = -0{,}01$ m wird somit in jedem Schritt eine Homotopie des zuvor berechneten TEBs erzeugt. Damit liegt die resultierende Trajektorie in der Homotopieklasse der Initialisierung und damit oberhalb des Hindernisses. Die gestellten Momente an den Rädern sind in Abbildung 13 dargestellt.

3.4 Integratoren verschiedener Ordnungen

Dieser Abschnitt gibt einen Überblick der benötigten Optimierungszeit in Abhängigkeit der Systemordnung. In diesem Beispiel wird ein Integrator p. Ordnung, beschrieben

Tabelle 1: Optimierungszeit bei Integratoren p. Ordnung.

p	$\bar{\tau} \pm \sigma_\tau$	τ_{max}	n_{max}	l_{max}
1	1,3 ms ± 0,9 ms	3,7 ms	30	58
2	2,8 ms ± 1,7 ms	5,4 ms	30	86
3	6,6 ms ± 3,3 ms	11,0 ms	36	138
4	8,5 ms ± 5,8 ms	22,2 ms	45	217
5	23,5 ms ± 9,9 ms	34,4 ms	55	320
6	38,5 ms ± 11,8 ms	57,1 ms	63	429
7	68,5 ms ± 10,3 ms	80,0 ms	75	586

durch die Differentialgleichung $x^{(p)}(t) = u(t)$, betrachtet. Die Regelungsaufgabe besteht aus der Überführung des Systems von $\mathbf{x}_s = \mathbf{0}$ nach $\mathbf{x}_g = [1, 0, \ldots, 0]^T \in \mathbb{R}^p$ in minimaler Zeit.

Der Regelkreis wird nach Abbildung 2 geschlossen. In jedem Abtastintervall wird die benötigte Optimierungszeit τ des TEB-Algorithmus gemessen. Neben der Größe p des Zustandsvektors beeinflusst das Hinzufügen/Entfernen von Diskretisierungszeitpunkten zur Einhaltung von $\Delta T_{soll} = 0{,}05$ s die Anzahl l der Optimierungsvariablen in $\mathbf{b}$.

Tabelle 1 zeigt die Ergebnisse für Integratoren bis zur siebten Ordnung. n_{max} und l_{max} kennzeichnen die maximale Anzahl an Diskretisierungszeitpunkten und Optimierungsvariablen. Die Optimierungszeit τ ist als Mittelwert $\bar{\tau}$ mit Standardabweichung σ_{tau} und der oberen Schranke τ_{max} angegeben. Selbst bei einer maximalen Länge des Optimierungsvektors von mehr als 500 Parametern liegt die Optimierungszeit immer noch unter 100 ms. In Folge der p-fachen Anwendung des Differenzenquotienten und der gleichzeitig konstant gewählten Abtastzeit steigt der Diskretisierungsfehler mit erhöhter Ordnung. Um das simulierte kontinuierliche System dennoch wie gewünscht in den Arbeitspunkt zu überführen, wird das Gewicht σ_2 beziehungsweise der Faktor κ erhöht. Bei größeren Ordnungen ist ΔT_{soll} auf Kosten einer erhöhten Berechnungszeit zu verringern.

4 Zusammenfassung und Ausblick

Die Erweiterung des TEBs im Kontext der modellprädiktiven Regelung eignet sich insbesondere für zeitoptimale Probleme bei nichtlinearen Systemen unter Beschränkungen der Zustands- und Stellgrößen. Eine vergleichende Analyse der durch den TEB-Ansatz optimierten Steuerfolge mit der analytischen optimalen Lösung belegt die Praxistauglichkeit der Methode. Die Betrachtung der notwendigen Optimierungszeit bestätigt das Potential der optimalen Regelung mit TEBs auch für nichtlineare hochdynamische Systeme.

Zukünftige Arbeiten beschäftigen sich mit der Anpassung der Gewichtungsfaktoren $\sigma_{1,2}$ für die Verletzung der Nebenbedingungen während der Regelung, um die konfliktären Ziele der Minimierung der eigentlichen Kostenfunktion und der Erfüllung der Zustandsgleichung in Einklang zu bringen. Der vorgestellte Ansatz soll weiterhin für multikriterielle Optimierungsziele, wie beispielsweise eine Kombination aus zeit- und verbrauchsoptimalen Kriterien, erweitert und analysiert werden. Es soll darüber hinaus untersucht werden, inwieweit sich der TEB-Ansatz auf Systembeschreibungen in Eingangs-Ausgangsform und datenbasierte Modelle der Zustandsdynamik übertragen lässt.

Literatur

1. Piechottka, U.; Hagenmeyer, V.: A discussion of the actual status of process control in theory and practice: a personal view from German process industry. *at – Automatisierungstechnik* 62 (2014) 2.
2. Morari, M.; H. Lee, J.: Model predictive control: past, present and future. *Computers & Chemical Engineering* 23 (1999) 4-5, S. 667–682.
3. Diehl, M.; Bock, H. G.; Schlöder, J. P.; Findeisen, R.; Nagy, Z.; Allgöwer, F.: Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control* 12 (2002) 4, S. 577–585.
4. Diehl, M.; Bock, H. G.; Schlöder, J. P.: A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control. *SIAM Journal on Control and Optimization* 43 (2005) 5, S. 1714–1736.
5. Wang, Y.; Boyd, S. P.: Fast Model Predictive Control Using Online Optimization. In: *Proceedings of the 17th IFAC World Congress* (Dong-il, C.; Hyung, C.; Myung, C.; Wang, Y., Hg.), Bd. 17, S. 6974–6979. 2008.
6. Richards, A.: Fast model predictive control with soft constraints. In: *European Control Conference*, S. 1–6. 2013.
7. Pakazad, K. S.; Ohlsson, H.; Ljung, L.: Sparse Control Using Sum-of-norms Regularized Model Predictive Control. In: *IEEE Conference on Decision and Control*. 2013.
8. Graichen, K.; Kpernick, B.: A Real-Time Gradient Method for Nonlinear Model Predictive Control. In: *Frontiers of Model Predictive Control* (Zheng, T., Hg.). InTech. 2012.
9. Zeilinger, M. N.; Raimondo, D. M.; Domahidi, A.; Morari, M.; Jones, C. N.: On real-time robust model predictive control. *Automatica* 50 (2014) 3, S. 683–694.
10. Kouramas, K. I.; Faísca, N. P.; Panos, C.; Pistikopoulos, E. N.: Explicit/multi-parametric model predictive control (MPC) of linear discrete-time systems by dynamic and multi-parametric programming. *Automatica* 47 (2011) 8, S. 1638–1645.
11. Quinlan, S.; Khatib, O.: Elastic bands: connecting path planning and control. In: *IEEE International Conference on Robotics and Automation*, S. 802–807. 1993.

12. Delsart, V.; Fraichard, T.: Navigating dynamic environments using trajectory deformation. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, S. 226–233. 2008.
13. Rösmann, C.; Feiten, W.; Wösch, T.; Hoffmann, F.; Bertram, T.: Efficient trajectory optimization using a sparse model. In: *European Conference on Mobile Robots*, S. 138–143. 2013.
14. Keller, M.; Hoffmann, F.; Bertram, T.; Hass, C.; Seewald, A.: Planning of Optimal Collision Avoidance Trajectories with Timed Elastic Bands. In: *19th World Congress of the International Federation of Automatic Control*. 2014.
15. Lam, D.: *A model predictive approach to optimal path-following and contouring control*. Dissertation, The University of Melbourne. 2012.
16. Van den Broeck, L.; Diehl, M.; Swevers, J.: A model predictive control approach for time optimal point-to-point motion control. *Mechatronics* 21 (2011) 7, S. 1203–1212.
17. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W.: G2o: A general framework for graph optimization. In: *IEEE International Conference on Robotics and Automation*, S. 3607–3613. 2011.
18. Moré, J. J.: The Levenberg-Marquardt algorithm: Implementation and theory. In: *Numerical Analysis* (Watson, G. A., Hg.), Bd. 630 von *Lecture Notes in Mathematics*, S. 105–116. Springer Berlin Heidelberg. 1978.
19. Davis, T. A.: *Direct methods for sparse linear systems*. Fundamentals of algorithms. Philadelphia: Society for Industrial and Applied Mathematics. 2006.
20. Guennebaud, G.; Jacob, B.: Eigen v3. URL http://eigen.tuxfamily.org. 2010.
21. Faulwasser, T.; Kern, B.; Varutti, P.; Findeisen, R.: Prädiktive Regelung nichtlinearer Systeme unter asynchronen Mess- und StellsignalenNonlinear Predictive Control based on Asynchronous Measurement and Control Signals. *at - Automatisierungstechnik* 57 (2009) 6.
22. Avvakumov, S. N.; Kiselev, Yu. N.: Boundary Value Problem for Ordinary Differential Equations with Applications to Optimal Control. In: *World Multi-Conference on Systemics, Cybernetics and Informatics*. USA. 2004.
23. Ivanjko, E.; Petrinić, T.; Petrović, I.: Modelling of Mobile Robot Dynamics. In: *EUROSIM Congrees on Modelling and Simulation* (Čepek, M., Hg.). 2010.

M. Sc. Christoph Rösmann
Technische Universität Dortmund,
Lehrstuhl RST, 44221 Dortmund, Tel: +49 (0)231 755 3592
christoph.roesmann@tu-dortmund.de

M. Sc. Christoph Rösmann ist wissenschaftlicher Mitarbeiter am Lehrstuhl für Regelungssystemtechnik in der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität Dortmund. Arbeitsgebiete: Trajektorienplanung und -Regelung mobiler Plattformen und proxemische Robotik.

Dr. rer. nat. Frank Hoffmann
siehe oben
frank.hoffmann@tu-dortmund.de

Dr. rer. nat. Frank Hoffmann ist Akademischer Oberrat am Lehrstuhl für Regelungssystemtechnik in der Fakultät für Elektrotechnik und Informationstechnik an der Technischen Universität Dortmund. Seine Forschungsgebiete liegen in der Robotik, Bildverarbeitung und Computational Intelligence.

Prof. Dr.-Ing Prof. h.c. Dr. h.c. Torsten Bertram
siehe oben
torsten.bertram@tu-dortmund.de

Prof. Dr.-Ing Prof. h.c. Dr. h.c. Torsten Bertram ist Inhaber des Lehrstuhls für Regelungssystemtechnik in der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität Dortmund. Seine Hauptforschungsinteressen sind die Modellbildung, Regelung und Simulation mechatronischer Systeme für Anwendungen in der Fahrzeugsystemtechnik und Servicerobotik.