Rechenzeitoptimierte Modellierung nicht-linearer physikalischer Prozesse mit Surrogate Modellen zur Anwendung in echtzeitfähigen Optimierungsverfahren

Marian Göllner[1]*, Or Aviv Yarom[1], Jannis Fritz[1], Xiaobo Liu-Henke[1]

[1] Institut für Mechatronik, Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel; *mar.goellner@ostfalia.de

Abstract. Auf Grundlage eines nicht-linearen Modells der Regelstrecke eines Intralogistik- Transportsystems „S-Mobile“ wird in diesem Beitrag ein optimierender, nicht-linearen Polvorgaberegler auf Basis der Jacobi-Matrizen entwickelt. Bedingt durch die nicht-linearitäten, vor allem der trigonometrischen Funktionen innerhalb des nicht-linearen Streckenmodells und damit auch in den Jacobi-Matrizen, wird der Einfluss auf die Echtzeitfähigkeit untersucht. Mit der Landau Notation der Algorithmen-Komplexität der zugrundeliegenden Berechnungsalgorithmen der Grundfunktionen, wird eine Methode zur Prädiktion der zu erwartenden Berechnungszeit vorgestellt. Ein Algorithmus, welcher ein heuristisches Modell als Surrogate-Modell der nicht-Linearitäten so trainiert, dass sowohl die Abweichung zu der Referenz klein als auch die Algorithmen-Komplexität insgesamt geringer ist, wird entwickelt.

Einleitung und Motivation

Zur Verbesserung der dynamischen Eigenschaften von Gütertransportsystemen und deren Manövrierfähigkeit wurde an der Ostfalia Hochschule für angewandte Wissenschaften ein neuartiges Intralogistik Transportsystem „S-Mobile“, als ein hochdynamisches und unteraktuiertes System mit sphärischem Elektroantrieb, mechatronisch entwickelt. Da dieses unteraktuierte Mehrkörpersysteme nicht-minimalphasig ist und daher bei Arbeitspunktwechseln nicht-linearitäten aufweist und sogar instabil werden kann, ist die Regelung eine technische Herausforderung die nicht mehr mit konventioneller Regelungstechnik gelöst werden kann. Das in [1] sukzessiv entwickelte und in [2] vorgestellte nicht-lineare, hierarchische Regelungssystem basiert auf der Berechnung der Systemzustände mittels kontinuierlicher Nachführung eines linearen Ersatzmodells auf Grundlage der Jacobi-Ableitungsmatrizen der nicht-linearen Systemdynamik. Die Approximationsgüte ist für die Funktion des Regelsystems maßgebend, erfordert allerdings eine dementsprechend exakte Nachbildung die zu Konflikten mit der Echtzeitfähigkeit des Reglers führen. Damit das Regelsystem unter Echtzeitbedingen eine optimale Zustandsregelung durchführen kann, soll in dieser Arbeit eine allgemeine Methode zum finden eines Surrogate Modell hergeleitet werden.

1 Modellbildungsprozess

Im Rahmen dieser Arbeit wird unter anderem die Modellbildung des nicht-linearen Streckenverhaltens besprochen. Nachfolgend sollen alle grundlegenden Schritte für die spätere Implementierung des nicht-linearen Verhaltens in den Regler dargestellt werden.

1.1 Methode

Der dazu notwendige Modellbildungsprozess (vgl. Abbildung 1) basiert auf dem realen System, welches gemäß der Aufgabenstellung reduziert bzw. vereinfacht wird, sodass sich ein physikalisches Modell ergibt. Dieses wird mithilfe physikalischer Gesetzmäßigkeiten in ein mathematisches Modell überführt, welches wiederum bspw. in Form von Signalflussplänen im Rechner abgebildet und mithilfe von CAE-Werkzeugen und entsprechender Numerik simuliert werden kann.

Der Modellbildungsprozess umfasst zudem Messungen am realen System, um zum einen die Parameter des mathematischen Modells zu identifizieren und

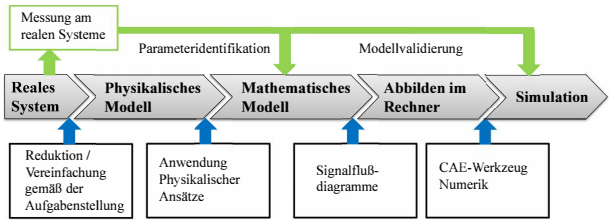

Abbildung 1: Modellbildungsprozess

zum anderen die Simulation zu validieren. Weitere Informationen sind für den planaren Fall in [3] und für die holistische driedimensionale Betrachtung in [1] gelistet.

1.2 nicht-lineares Zustandsraummodell

Ein Bereits in [2] für das S-Mobile auf Basis des Euler-Lagrange Ansatzes hergeleitetes Dynamikmodell wurde in die allgemeine Matrizenschreibweise überführt und dient als Grundlage der nachfolgenden Betrachtungen. Die nachfolgende Gleichung zeigt das verallgemeinerte Dynamikmodell mit Reibmodellierung aus [4] zur Berücksichtigung der Kraftübertragung.

$$\underline{\underline{M}}\left(\underline{q}\right)\cdot\ddot{\underline{q}}+\underline{\underline{C}}\left(\underline{q},\dot{\underline{q}}\right)\cdot\dot{\underline{q}}+\underline{G}\left(\underline{q}\right)=\underline{\underline{Q}}\left(\underline{q}\right)\cdot\underline{u} \qquad (1)$$

Die folgenden Abbildung 2 zeigt die allgemeine Struktur des Dynamikmodells der Strecke.

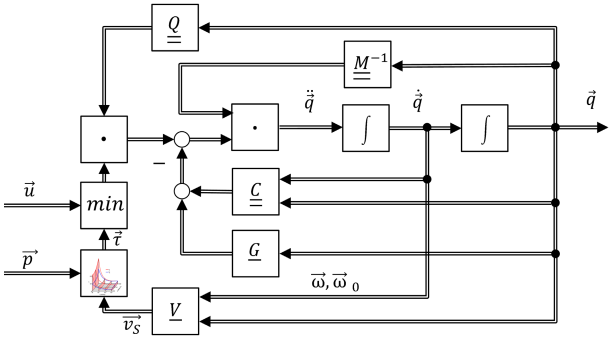

Abbildung 2: Modell der Strecke

Dieses Modell besitzt die Eingänge:

- $\underline{u} \in \mathbb{R}^n$ Krafteingangsvektor
- $\underline{p} \in \mathbb{R}^n$ Vektor der Pressungen

die Zustandsvektoren:

- $\underline{q} \in \mathbb{R}^m$ Zustandsvektor
- $\underline{v_s} \in \mathbb{R}^n$ Schlupfvektor
- $\underline{\tau} \in \mathbb{R}^n$ Kraftübertragungsvektor

und die charakterisierenden Matrizen und Vektoren:

- $\underline{\underline{M}} \in \mathbb{R}^{m\times m}$ Massenmatrix
- $\underline{\underline{C}} \in \mathbb{R}^{m\times m}$ Coriolis- oder Zentrifugalkraftmatrix
- $\underline{\underline{G}} \in \mathbb{R}^{m\times m}$ Gravitationsmatrix
- $\underline{\underline{V}} \in \mathbb{R}^{m\times m}$ Matrize der Geschwindigkeitsvektoren
- $\underline{\underline{Q}} \in \mathbb{R}^{m\times n}$ Eingangsmatrix der eingeprägten Kräfte

Der Eingangsvektor $\underline{u}$ wird über eine Minimalbedingung jedes einzelnen Vektorelements i definiert. Aus [4] wird ersichtlich, dass entweder direkt das Antriebsmoment der Aktuatoren $\underline{u}$ oder aber eine maximal übertragbare Kraft $\underline{\tau}$ in das Modell weitergeleitet wird.

$$\forall(u_i,\tau_i)\in(\underline{u}|\underline{\tau}):$$
$$\underline{u}=\min_{i=1\ldots n}[u_i,\tau_i]=\begin{cases}\underline{u}(i) & u_i\leq\tau_i\\ \underline{\tau}(i) & u_i>\tau_i\end{cases} \qquad (2)$$

Der Kraftübertragungsvektor $\underline{\tau}$ bildet sich aus den Anpresskräften $\underline{F_N}$ und dem Reibbeiwertsvektor $\underline{\mu}$. Es soll hierbei beachtet werden, dass durch mechanische Vorspannung der Räder die Anpresskräfte einstellbar und unabhängig von der Kinematik des Systems sein können.

$$\underline{\tau}=\underline{F_N}\cdot\underline{\mu} \qquad (3)$$

Dieses Modell ist offensichtlich nicht-linear und aus der Modellherleitung in [1] und [4] folgt das die Übertragungsmatrixen von Winkelfunktionen abhängen.

1.3 Linearisierung und Zustandsraum

Für die Spätere Reglersynthese ist das ableiten eines linearen Zustandsraummodells zu jedem Zeitschritt notwendig. Der Vorgang basiert auf einer Taylor-Reihenentwicklung unter Verwendung der Jacobi-Matrizen des nicht-linearen Modells. Die Taylor-Reihe führt auf die linearisierte Funktion $\Delta\underline{y}_{lin}$ an einem beliebigen Arbeitspunkt AP zu einem Zeitschritt i:

$$\Delta\underline{y}_{lin}=f\Big|_{AP_i}+\underline{\underline{J}}_{u}(i)\cdot\Delta\underline{u} \qquad (4)$$
$$+\underline{\underline{J}}_{q}(i)\cdot\Delta\underline{q}+\underline{\underline{J}}_{\dot{q}}(i)\cdot\Delta\dot{\underline{q}}+\underline{\underline{J}}_{\ddot{q}}(i)\cdot\Delta\ddot{\underline{q}}=0\in\mathbb{R}^5$$

Mit den partiellen Differentialen als Jacobi- Matrizen:

$$\begin{aligned} \underline{\underline{J}}_u(i) &= \left.\frac{\partial \underline{f}}{\partial \underline{u}}\right|_{AP_i}; \underline{\underline{J}}_q(i) = \left.\frac{\partial \underline{f}}{\partial \underline{q}}\right|_{AP_i}; \\ \underline{\underline{J}}_{\dot{q}}(i) &= \left.\frac{\partial \underline{f}}{\partial \underline{\dot{q}}}\right|_{AP_i}; \underline{\underline{J}}_{\ddot{q}}(i) = \left.\frac{\partial \underline{f}}{\partial \underline{\ddot{q}}}\right|_{AP_i} \end{aligned} \tag{5}$$

Das linearisierte Modell am aktuellen Arbeitspunkt ist entsprechend:

$$\begin{aligned} \Delta\underline{\ddot{q}} = & \underbrace{(-\underline{\underline{J}}_{\ddot{q}}(i)^{-1} \cdot \underline{\underline{J}}_q(i)) \cdot \Delta\underline{q} + (-\underline{\underline{J}}_{\ddot{q}}(i)^{-1} \cdot \underline{\underline{J}}_{\dot{q}}(i)) \cdot \Delta\underline{\dot{q}}}_{\text{für } \underline{\underline{A}}_i \cdot \underline{x}} \\ & + \underbrace{(-\underline{\underline{J}}_{\ddot{q}}(i)^{-1} \cdot \underline{\underline{J}}_u(i)) \cdot \Delta\underline{u}}_{\text{für } \underline{\underline{B}}_i \cdot \underline{u}} + \underbrace{(-\underline{\underline{J}}_{\ddot{q}}(i)^{-1}) \cdot f\Big|_{AP_i}}_{\text{für } \underline{\underline{E}}_i \cdot \underline{z}_i} \end{aligned} \tag{6}$$

Woraus sich das Zustandsraummodell allgemein ableitet:

$$\begin{aligned} \dot{\underline{x}} &= \underline{\underline{A}}_i \cdot \underline{x} + \underline{\underline{B}}_i \cdot \underline{u} + \underline{\underline{E}}_i \cdot \underline{z}_i \\ \underline{y} &= \underline{\underline{C}} \cdot \underline{x} + \underline{\underline{D}} \cdot \underline{u} \end{aligned} \tag{7}$$

Das Modell besitz die folgenden Zustandsvektoren:

- $\underline{x} \in \mathbb{R}^m$ Zustandsvektor
- $\underline{y} \in \mathbb{R}^p$ Ausgangsvektor
- $\underline{u} \in \mathbb{R}^n$ Eingangsvektor
- $\underline{z} \in \mathbb{R}^q$ Störvektor

und charakterisierenden Matrizen:

- $\underline{\underline{A}} \in \mathbb{R}^{m \times m}$ Dynamikmatrix
- $\underline{\underline{B}} \in \mathbb{R}^{m \times n}$ Eingangsmatrix
- $\underline{\underline{C}} \in \mathbb{R}^{p \times m}$ Ausgangsmatrix
- $\underline{\underline{D}} \in \mathbb{R}^{p \times n}$ Durchgriff
- $\underline{\underline{E}} \in \mathbb{R}^{m \times q}$ Störmatrix

Dieses Modell ist nun mit einer beschränkten Gültigkeit eng um den gewählten (oder gemessenen) Arbeitspunkt AP_i linear.

2 Reglersynthese

Der hier vorgestellte nicht-lineare Regler in Abbildung 3 basiert auf der Methode der Online-Linearisierung, optimiert aber zudem den Stellvektor anhand einer Polvorgabe. Diese wiederum nutzt optimale Pole eines geschlossenen linearen Modellregelkreises, welcher durch eine Form der Fehlerflächenreduktion gefunden wird. Wenn der Regler diese idealen Polstellen dem Gesamtsystem in jedem Zeitschritt und an jedem beliebigen Arbeitspunkt aufzwingen kann, so ist das geschlossene System nach außen hin linear und für hierarchisch übergeordnete Regler (wie in [2] beschrieben) somit auch einfach prädizier- und kontrollierbar.

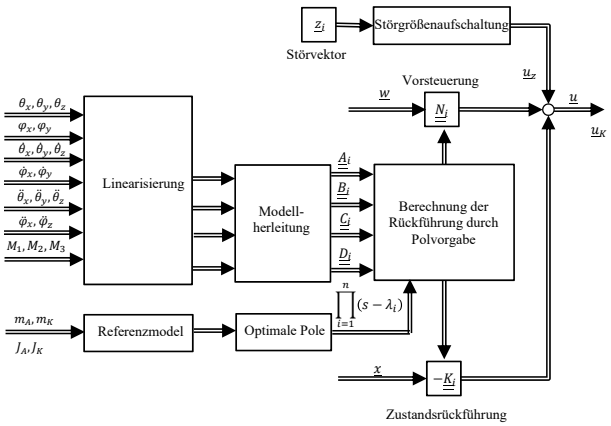

Abbildung 3: nicht-linearer Polvorgaberegler

2.1 Referenzmodell und optimale Pole

Auf Basis des in Kapitel 1.3 vorgestellten, linearen Zustandsraummodells der Regelstrecke sollen Polstellen abgeleitet werden, die für das ideale System am Referenzarbeitspunkt der quasi-stationären Ruhelage gelten. Dafür wird zunächst ein fiktiver idealer Regelkreis angenommen. da aber nur die Polstellen von Interesse sind, ist die Berechnung einer expliziten Stellgrößenrückführung nicht notwendig. Es wird eine Fehlerfunktion anhand der bereits in [1] vorgestellten Methode der Zustandsregelung angenommen:

$$\begin{aligned} \min_u(J_{LQ}) = & \frac{1}{2}\underline{x}^T(t_e) \cdot \underline{\underline{F}} \cdot \underline{x}(t_e) \\ & + \frac{1}{2}\int_{t_0}^{t_e} \underline{x}^T(t) \cdot \underline{\underline{Q}} \cdot \underline{x}(t)) + \underline{u}^T(t) \cdot \underline{\underline{R}} \cdot \underline{u}(t)\, dx) \end{aligned} \tag{8}$$

Mit den Gewichtungsfaktoren Q und R. Aus dieser wird zunächst die Hamilton-Funktion $\mathscr{H}(\underline{x},\underline{u},\underline{\lambda})$ abgeleitet:

$$\begin{aligned}\mathscr{H}(\underline{x},\underline{u},\underline{\lambda}) = \frac{1}{2}(\underline{x}^T(t)\cdot\underline{\underline{Q}}\cdot\underline{x}(t)) + \underline{u}^T(t)\cdot\underline{\underline{R}}\cdot\underline{u}(t)) \\ + \underline{\mathscr{L}}^T(t)(\underline{\underline{A}}\cdot\underline{x}+\underline{\underline{B}}\cdot\underline{u})\end{aligned} \tag{9}$$

Aus den Optimierungsbedingungen folgt:

$$\begin{aligned}\underline{\dot{\mathscr{L}}}^T(t) &= (\underline{\dot{P}}+\underline{\underline{P}}\cdot\underline{\underline{A}}-\underline{\underline{P}}\cdot\underline{\underline{B}}\cdot\underline{\underline{R}}^{-1}\cdot\underline{\underline{B}}^T\cdot\underline{\underline{P}})\cdot\underline{x}(t) \\ \underline{\dot{\mathscr{L}}}^T(t) &= -(\underline{\underline{A}}^T\cdot\underline{\underline{P}}+\underline{\underline{Q}})\cdot\underline{x}(t)\end{aligned} \tag{10}$$

Es ergibt sich die Ricatti-Gleichung mit Nebenbedingungen:

$$-\underline{\dot{P}} = \underline{\underline{A}}^T\cdot\underline{\underline{P}}-\underline{\underline{P}}\cdot\underline{\underline{B}}\cdot\underline{\underline{R}}^{-1}\cdot\underline{\underline{B}}^T\cdot\underline{\underline{P}}+\underline{\underline{Q}}+\underline{\underline{P}}\cdot\underline{\underline{A}} = 0 \tag{11}$$

Aus dieser lassen sich durch Lösen der Gleichung und Umformung die Polstellen leicht berechnen:

$$det(s\underline{\underline{I}}-\underline{\underline{A}}+\underline{\underline{B}}\cdot\underline{\underline{R}}^{-1}\cdot\underline{\underline{B}}^T\cdot\underline{\underline{P}}) = \prod_{i=1}^{n}(s-\lambda_i) \tag{12}$$

Diese Polstellen repräsentieren nun das ideale geschlossenen System, die weiteren Schritte dienen dazu dem realen, geschlossenen Regelkreis dieses dynamische Verhalten aufzuzwingen.

2.2 Linearisierung und Modellherleitung

Um die Regelung an jedem beliebigen Arbeitspunkt durchführen zu können, soll auf Basis der in Kapitel 1.3 hergeleiteten Jacobimatrizen der nicht-linearen Strecke ein zu dem jeweiligen *AP* gültiges Ersatzmodell in Zustandsraumdarstellung hergeleitet werden.

$$\begin{aligned}\underline{\dot{x}} &= (\underline{\underline{A}}_i-\underline{\underline{B}}_i\cdot\underline{\underline{K}}_i)\cdot\underline{x} \\ \underline{y} &= (\underline{\underline{C}}-\underline{\underline{D}}\cdot\underline{\underline{K}}_i)\cdot\underline{x}\end{aligned} \tag{13}$$

Dieses dient der nachgelagerten Polvorgabe als Grundlage zur Berechnung des Zustandsrückführungsvektors. Da dieser Prozess in jedem Zeitschritt auf Basis der nicht-linearen Jacobimatrizen durchgeführt wird, ist er für die Echtzeitfähig kritisch. Der vorliegende Beitrag konzentriert sich auf die Optimierung dieses Prozesses auf Basis einer heuristischen Näherung der Ausgabewerte $\underline{\underline{A}}_i, \underline{\underline{B}}_i, \underline{\underline{C}}_i, \underline{\underline{D}}_i$ bei jedem Zeitschritt i.

2.3 Zustandsrückführung und Polvorgabe

Auf Basis des online-linearisierten Zustandsraummodells zum Zeitschritt i am aktuellen Arbeitspunkt und den idealen Polen p wird nun durch Polvorgabe der Zustandsrückführungsvektor $\underline{\underline{K}}_i$ und der Vorfilter $\underline{\underline{N}}_i$ berechnet. Dazu werden die geregelten Pole des Systems

$$det(s\underline{\underline{I}}-\underline{\underline{A}}_i+\underline{\underline{B}}_i\cdot\underline{\underline{K}}_i) = \prod_{i=1}^{n}(s-\lambda_i) = const. \tag{14}$$

mit den optimalen Polen gleichgesetzt.

$$det(s\underline{\underline{I}}-\underline{\underline{A}}_i+\underline{\underline{B}}_i\cdot\underline{\underline{K}}_i) = det(s\underline{\underline{I}}-\underline{\underline{A}}+\underline{\underline{B}}\cdot\underline{\underline{R}}^{-1}\cdot\underline{\underline{B}}^T\cdot\underline{\underline{P}})$$

So kann durch Koeffizientenvergleich ein Zustandsrückführvektor $\underline{\underline{K}}_i$ berechnet werden. Ein Vorfilter soll das stationäre Verhalten verbessern. Die inverses der internen Systemdynamik ist allerdings aufgrund der Dimensionen nicht explizit zu berechnen und wird deshalb durch eine Pseudoinverse gebildet:

$$\underline{\underline{N}}_i = -(\underline{\underline{C}}(\underline{\underline{A}}_i+\underline{\underline{B}}_i\cdot\underline{\underline{K}}_i)^{-1})^+ \tag{15}$$

Es ergibt sich so das Stellgesetz des Reglers zu:

$$\underline{u} = -\underline{\underline{K}}_i\cdot\underline{x}+\underline{\underline{N}}_i\cdot\underline{w} \tag{16}$$

Für der Sollwerteingang $\underline{w}$ ist der geschlossener Regelkreis nun linear und hat stabile, konstante Pole.

3 Echtzeitoptimierung der Reglerfunkionen

Der in Kapitel 2 vorgestellte Regelalgorithmus arbeitet teilweise mit den Jacobi-Matrizen der nicht-linearen Strecke, welche Winkelfunktionen enthalten und deshalb die Echtzeitausführung des Codes auf einer Zielhardware besonders beeinflussen. Die Funktionen können teilweise durch eine heuristische Näherung ersetzt werden, welche Aufgrund der noch aufzuzeigenden niedrigeren Zeitkomplexität eine Chance zur Optimierung der Echtzeitanforderung hinsichtlich Genauigkeit, Rechenarchitektur und Berechnungsschrittweite bietet. In diesem Kapitel soll nun zunächst die Berechnungsdauer durch Klassifizierung der verwendeten Algorithmen nach der erstmals von Bachmann [5] verwendeten Landau Notation [6] in Abhängigkeit der Eingänge und Rechenoperationen prädiziert werden.

3.1 Herleitung der Komplexität des nicht-linearen Modells

Die Simulation anfangswertbehafteter, gewöhnlicher Differentialgleichungen führt unweigerlich auf die Verwendung numerischer Integrationsverfahren. Eine Untersuchung des optimalen Verfahrens unter Echtzeitanforderungen sollte mit einer Betrachtung des lokalen und globalen Fehlers der Integrationsmethode beginnen. Ausgehend von der Landau Notation [6] hat z.B. das einfachste Euler Verfahren 2. Ordnung einen lokalen Fehler von $\mathcal{O}(h^2)$ und einen globalen Fehler $\mathcal{O}(h)$ für die Schrittweite h. Das Heun- Trapezverfahren 3. Ordnung einen lokalen Fehler von $\mathcal{O}(h^3)$ und einen globalen Fehler $\mathcal{O}(h^2)$. Der maximale lokale Fehler eines Verfahrens der Ordnung p (für kleine Schrittweiten h) ist also eine Funktion proportional zu h^p oder mit Landau-Symbolen: $\mathcal{O}(h^p)$ [7]. Es entsteht dadurch ein Zielkonflikt, da ein höherwertiges Verfahren einen geringeren Fehler und somit eine größere Zeitschrittweite erlaubt. Es ist also nicht in allen Fällen davon auszugehen, dass ein einfaches Verfahren bei vorgegebener Fehlertoleranz grundsätzlich schneller rechnet, ein komplexeres kann diesen durch ein Aufweiten der Schrittweite kompensieren. Die Echtzeitanforderung muss also in Abhängigkeit des Fehlers und der Schrittweite optimiert werden.

Es ist weiterhin davon auszugehen, dass die dem nicht-linearen Modell zugrundeliegenden trigonometrischen Funktionen hinsichtlich ihrer Algorithmen-Komplexität die Echtzeitfähigkeit signifikant beeinflussen. Zur Berechnung der Winkelfunktionen wird von einer Darstellung über den CORDIC Algorithmus ausgegangen. Grundlage ist die Drehung eines Einheitsvektors, der in x-Achse liegt, um einen Winkel Θ.

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} = \cos\Theta \cdot \begin{bmatrix} 1 & -\tan\Theta \\ \tan\Theta & 1 \end{bmatrix}$$

Die Drehung wird aufgrund der Übergänge der Winkelfunktionen als Teildrehung $\sigma_i \in \{-1, 1\}$ um Teilwinkel α_i realisiert und entspricht:

$$\Theta = \sum_i \sigma_i \cdot \alpha_i \text{ mit } \sigma_i \quad = \begin{cases} -1 & \text{falls} \quad z_i \leq 0 \\ 1 & \text{sonst} \end{cases} \tag{17}$$

Über den Zusammenhang $\tan\alpha_i = 2^{-i}$ erhält man den folgenden Algorithmus:

$$\begin{aligned} \begin{bmatrix} x_n \\ y_n \end{bmatrix} &= \prod_{i=0}^{n-1} \cos\alpha_i \begin{bmatrix} 1 & -\sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \\ \begin{bmatrix} x_n \\ y_n \end{bmatrix} &= K \cdot \prod_{i=0}^{n-1} \begin{bmatrix} 1 & -\sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \end{aligned} \tag{18}$$

mit dem Skalierungsfaktor $K = \prod_{i=0}^{n-1} \cos\alpha_i \approx 0,60725...$ für $n \to \infty$, es entsteht die folgende, einfache Formelschar:

$$\begin{aligned} x_{i+1} &:= x_i - \sigma_i 2^{-i} y_i \\ y_{i+1} &:= y_i + \sigma_i 2^{-i} x_i \\ z_{i+1} &:= z_i - \sigma_i \arctan 2^{-i} \end{aligned} \tag{19}$$

Deren Algorithmen-Komplexität über die Multiplikation und die arctan-Funktion bestimmt sind. Der naive aus der Schule Bekannte „menschliche“ Algorithmus zur Multiplikation hat die Laufzeit $\mathcal{O}(n^2)$ [8], wenn als Effizienzmaß die Bitkomplexität auf mehrbändigen Turingmaschinen, also die maximale Laufzeit des Algorithmus gemessen als benötigte Bitoperationen in Abhängigkeit von der Bitlänge n der Eingabegrößen gewählt wird. Nach [9] hat der momentan effizienteste Multiplikationsalgorithmus eine Zeitkomplexität von:

$$\mathcal{O}(n \log n) \tag{20}$$

Die Zeitkomplexität in Abhängigkeit der Genauigkeit n (Anzahl der signifikanten Bits) der arctan-Funktion lässt sich über dessen Verbindung zum komplexen Logarithmus Naturalis herleiten [10]:

$$\arctan 2^{-i} = \frac{1}{2\mathrm{i}} \ln \frac{1 + \mathrm{i}2^{-i}}{1 - \mathrm{i}2^{-i}} \tag{21}$$

Die Algorithmen-Komplexität des komplexen Logarithmus ist über die Padé Approximation gegeben [11]:

$$\mathcal{O}(M(n) \log n) \tag{22}$$

Wobei $M(n)$ die bereits vorgestellte Komplexität des eingesetzten Multiplikationsalgorithmus ist.

3.2 Herleitung der Komplexität des Neuronalen Netzes

Das heuristisch genäherte Modelle hingegen wird als gerichteter Graph $G = f(U, C)$ aufgefasst, dessen Knoten $j \in U$ die Neuronen und dessen Kanten $c \in C$ die

Neuronenverbindungen sind. Jede Verbindung $(i,j) \in C$ innerhalb einer Neuronenschicht k besitzt eine Gewichtung w_{ij}. Die Netztopologie einer Neuronenschicht kann in Form einer $(d \times d)|d = |U|$ dimensionalen Adjazenzmatrix $\underline{\underline{A}} = (a_{ij}^k)$ mit den Elementen:

$$a_{ij}^k = \begin{cases} 1 & (i,j) \in C \\ 0 & (i,j) \notin C \end{cases} \tag{23}$$

beschrieben werden. Sie bestehen aus topologisch zusammenhängenden Neuronen welche immer auf der Basis eines allgemeinen Neurons ihre gewichteten Eingänge $x_1, \ldots, x_n$ mit den Gewichtungsfaktoren $w_{1j}, \ldots, w_{nj}$ zunächst mit der folgenden Übertragungsfunktion/Integrationsfunktion (Summation) in eine Netzausgabe wandeln:

$$net_j^k = \sum_{i=1}^{m^{k-1}} j_i^{k-1} = \sum_{i=1}^{m^{k-1}} w_{ij}^{k-1} \cdot out_i^{k-1} \tag{24}$$

Das Ergebnis durchläuft eine Aktivierungsfunktion act_j^k welche über einen erlernten oder vorgegebenen Schwellenwert θ_j die Ausgabefunktion z.B. die Sigmoide aktiviert:

$$out_j^k = f_j(act_j^k) \text{ mit } act_j^k = \begin{cases} net_j^k & net_j^k > \theta_j \\ 0 & net_j^k \leq \theta_j \end{cases} \tag{25}$$

Es ergibt sich so für eine Netzschicht k das folgende Gleichungssystem:

$$\begin{aligned} \underline{net}^k &= \underline{\underline{w}}^{k-1} \cdot \underline{out}^{k-1} \\ \underline{out}^k &= f(\underline{net}^k) \end{aligned} \tag{26}$$

mit den Dimensionen

$$\begin{aligned} \underline{net}^k &\in \mathbb{R}^{n^k \times 1} \\ \underline{\underline{w}}^{k-1} &\in \mathbb{R}^{n^k \times n^{k-1}} \\ \underline{out}^{k-1} &\in \mathbb{R}^{n^{k-1} \times 1} \end{aligned} \tag{27}$$

Es ist also ersichtlich das zur Lösung einer Netzschicht k mit einer einfachen Aktivierungsfunktion der Komplexität $\mathcal{O}_{trans}^{(k-1)\to k} = \mathcal{O}(n^k)$ grundsätzlich nur eine dreidimensionale Matrizenmultiplikation durchgeführt werden muss. diese hat eine Zeitkomplexität von $\mathcal{O}_{mult} = \mathcal{O}(a \cdot b \cdot c)$ mit den diskreten Dimensionsachsen a, b, c der Matrize. So lässt sich also ableiten:

$$\mathcal{O}(\underbrace{(n^k \cdot n^{k-1} \cdot 1)}_{\mathcal{O}_{mult}} + \underbrace{(n^k)}_{\mathcal{O}_{trans}}) \tag{28}$$

Für ein einfaches Neuronales Netz mit einer Eingangsschicht $k = 0$, einer Ausgangsschicht $k = 2$ und einem dazwischenliegenden sog. Hidden Layer $k = 1$ ergibt sich exemplarisch also folgende Zeitkomplexität:

$$\underline{out}^2 = \underline{out}^1 + \underline{out}^0 = \mathcal{O}((n^2 \cdot n^1) + (n^2) + (n^1 \cdot n^0) + (n^1))$$

Unter der Annahme das die Aktivierungsfunktion für alle Layer gleich ist, ist die Komplexität des einfachen Netzes:

$$\mathcal{O}((n^0 + n^2) \cdot 2n^1) \tag{29}$$

Allgemein für ein beliebig großes Netz folgt so:

$$\begin{aligned} \mathcal{O}_{trans} &= \sum_{i=1}^{k} n^i \\ \mathcal{O}_{mult} &= \sum_{i=1}^{k} (n^{i-1} \cdot n^i) \\ &= \underbrace{\sum_{j=1}^{i-1} (n^{j-1} \cdot n^j)}_{\text{const}} + \underbrace{(n^{i-1} \cdot n^{i+1}) n^i}_{\text{linear: } \mathcal{O}(n^i)} + \underbrace{\sum_{j=i+2}^{k} (n^{j-1} \cdot n^j)}_{\text{const}} \end{aligned} \tag{30}$$

Es ist zu erkennen das die Komplexität des Netzes von der Anzahl der in einem Layer enthaltenen Neuronen abhängt, da eine Erhöhung dieser zu einem linearen Anstieg der Berechnungskomplexität führt.

3.3 Algorithmus zur Auslegung der heuristischen Funktion

Auf Basis der gewonnenen Erkenntnisse lässt sich zunächst zusammenfassen, dass ein zu großes Netz eine höhere Zeitkomplexität als die Ursprungsfunktionen aufweist, ein zu kleines Netz oder aber zu einfache Aktivierungsfunktionen u.U. die geforderte Genauigkeit nicht erreichen. Es entsteht ein Zielkonflikt, den es über eine Optimierung zu lösen gilt. Dazu ist es zunächst notwendig die Struktur der zu ersetzenden nichtlinearitäten zu untersuchen. Dazu kann der Zusammenhang zwischen den linearisierten Teilmodellen in Zustandsraumdarstellung zu dem Referenzmodell betrachtet werden. Hierbei zeigen sich Zellen innerhalb der Matrizen $\underline{\underline{A}}_i, \underline{\underline{B}}_i$ die sich aufgrund linearer Zusammenhänge bei unterschiedlichen Arbeitspunkten nicht än-

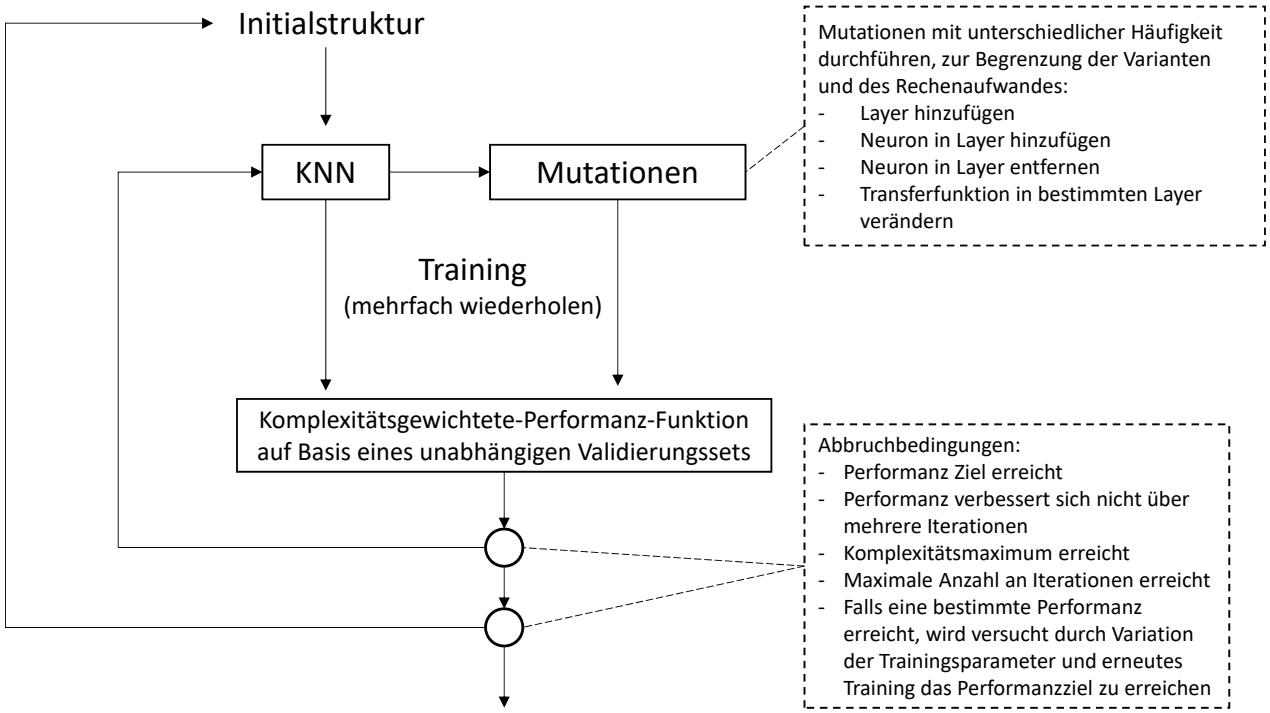

Abbildung 4: Optimierungsalgorithmus zum finden eines heuristischen Ersatzmodells

dern. Diese Zellen können als ideales lineares Übertragungsverhalten separiert werden. Die reduzierten Elemente sollen durch ein heuristisches Modell (hier: neuronales Netz) genähert werden. Dazu wird ein Trainingsalgorithmus entworfen der unter Beachtung der Algorithmen-Komplexität sowohl der Netzstruktur als auch der Aktivierungsfunktionen des Netzes eine optimale Lösung findet. Die folgende Abbildung 4 zeigt den Ansatz.

Zunächst müssen die Matrizenelemente so normiert werden, dass sie zu dem Wertebereich der Aktivierungsfunktion passen. es wird eine Normierung sowohl für die Netzeingänge als auch die Trainingsdaten auf einen Wertebereich von $[-1:1]$ durchgeführt. Nach dem Training wird die Normierung invers als Bias in die Ausgangsschicht zurückgespielt. Eine beliebige Initialstruktur wird mittels Mutation (siehe dazu auch [12]) hinsichtlich ihrer Neuronenanzahl, Schichttiefe, und Aktivierungsfunktion im jeweiligen Layer so lange trainiert und mit den vorherigen Netzen verglichen, bis sich ein Optimum einer Kostenfunktion ergibt. Die Kostenfunktion C setzt sich aus der Fehleranfälligkeit C_E und der Algorithmen-Komplexität C_O zusammen:

$$C = C_E \cdot C_O \tag{31}$$

Die Algorithmen-Komplexität C_O wurde bereits im vorherigen Kapitel ausführlich besprochen und setzt sich wie folgt zusammen:

$$C_O = \sum(\alpha_{g(x)} \cdot n_{g(x)}) + n_{mult} \tag{32}$$

Sie wird aus der Anzahl der Produktsummen n_{mult}, der Anzahl der Berechnungen, die für das Ausführen einer Aktivierungsfunktion nötig sind, $n_{g(x)}$ und des auf die Produktsumme normierten Gewichtungsfaktors der Aktivierungsfunktion $\alpha_{g(x)}$ berechnet. Dabei ist zu beachten das für Ausführen einer Aktivierungsfunktion hier eine von der Anzahl der Neuronen $n^(i)$ in einem Layer i abhängige Funktion angenommen wird, die über alle Layer k hinweg summiert wird:

$$n_{g(x)} = \sum_{i=1}^{k}(n^(i)) \tag{33}$$

Die Anzahl der Produktsummen n_{mult} muss bedingt durch die Struktur des Netzes in gleicher Weise aufsummiert werden:

$$n_{mult} = \sum_{i=1}^{k}(n^{(i-1)} \cdot n^i) \tag{34}$$

Die Fehleranfälligkeit C_E ist hingegen von den Ausgabewerten des Netzen $y_{i,net}$ in Relation zu den in den Trainingsdaten gespeicherten Ergebnissen $y_{i,train}$ abhängig und wird über die Anzahl der auswertbaren Daten d aufsummiert:

$$C_E = \frac{1}{d}\sum_{i=1}^{d}(y_{i,train} - y_{i,net})^2 \tag{35}$$

Der Fehler des mutierten Netzes muss sich somit überproportional verringern gegenüber der Komplexität, damit das Netz gegenüber dem Initialnetz bevorzugt wird. Somit wird das Netz zu einer besseren Echtzeitfähigkeit hin optimiert, ohne den Fehler zu verschlechtern.

4 Zusammenfassung

Im vorliegenden Beitrag wurde auf Grundlage eines nicht-linearen Modells der Regelstrecke eines Intralogistik- Transportsystems „S-Mobile“ zunächst ein optimierender, nicht-linearen Polvorgaberegler auf Basis der Jacobi-Matrizen entwickelt. Bedingt durch die nicht-linearitäten, vor allem der trigonometrischen Funktionen innerhalb des nicht-linearen Streckenmodells und damit auch in den Jacobi-Matrizen, wurde der Einfluss auf die Echtzeitfähigkeit untersucht. Mit der Landau Notation der Algorithmen-Komplexität der zugrundeliegenden Berechnungsalgorithmen der Grundfunktionen, wurde eine Methode zur Prädiktion der zu erwartenden Berechnungszeit vorgestellt. Auf dieser Grundlage wurde ein Algorithmus entwickelt, welcher ein heuristisches Modell als Surrogate-Modell der nicht-Linearitäten so trainiert, dass sowohl die Abweichung zu der Referenz klein als auch die Algorithmen-Komplexität insgesamt geringer ist.

Danksagung

Der vorliegende Beitrag wurde im Rahmen des Teilprojektes auto-EVM als Teil des Verbundprojekts autoMoVe unter dem Förderkennzeichen ZW6-85030889 gefördert. Die Verantwortung für den Inhalt liegt bei den Autoren. Für die Förderung bedanken sich diese herzlichst.

Referenzen

[1] Liu-Henke X, Göllner M, Tao H. An intelligent control structure for highly dynamic driving of a spherical electrical drive. In: *2017 Twelfth International Conference on Ecological Vehicles and Renewable Energies (EVER)*. Piscataway, NJ: IEEE. 2017; pp. 1–10.

[2] Göllner M, Zhang J, Liu-Henke X. Model predictive trajectory control for automated driving of a spherical electrical drive. In: *2018 IEEE International Systems Engineering Symposium (ISSE)*. IEEE. 10/1/2018 - 10/3/2018; pp. 1–6.

[3] Liu-Henke X, Göllner M, Tao H. Modellbasierte Reglerauslegung eines sphärischen Elektroantriebs. In: *Workshop der ASIM/GI-Fachgruppen "Simulation technischer Systeme", "Grundlagen und Methoden in Modellbildung und Simulation"*, edited by Wahmkow C, Roßmanek P, Wendorf R, ASIM-Mitteilungen, pp. 37–46. Stralsund: Fachhochschule Stralsund. 2015;.

[4] Göllner M, Liu-Henke X, Frerichs L. Analyse und Simulation des Kraftübertragungsverhaltens von Mecanum-Rädern. In: *Proceedings ASIM SST 2020*, edited by Deatcu C, Lückerath D, Ullrich O, Durak U, ASIM Mitteilung, pp. 89–98. Wien: ARGESIM. 2020;.

[5] Bachmann PGH. *Die analytische Zahlentheorie*, vol. 2 of *Zahlentheorie*. Leipzig: B.G. Teubner. 1894.

[6] Landau E. *Handbuch Der Lehre Von Der Verteilung Der Primzahlen*. Leipzig and Berlin: B.G. Teubner. 1909.

[7] Berg I. A Comparison of Numeric Integration Schemes: Runge Kutta vs. Euler: On the Influence of time step sizes on the accuracy of numerical simulations.

[8] Newman JR. *The World of Mathematics*, vol. 1. New York, NY: Simon and Schuster. 1956.

[9] Harvey D, van der Hoeven J. Integer multiplication in time O(n log n). *Princeton University, Department of Mathematics*. 2019;.

[10] Cotes R. Logometria. *Philosophical Transactions Royal Society*. 1714;.

[11] Padé H. Sur la généralisation des fractions continues algébriques. *Journal de Mathématiques Pures et Appliquées*. 1894;pp. 291–330.
URL `http://eudml.org/doc/235790`

[12] Yarom OA, Jacobitz S, Liu-Henke X. Design of Genetic Algorithms for the Simulation-Based Training of Artificial Neural Networks in the Context of Automated Vehicle Guidance. In: *19th International Conference on Mechatronics – Mechatronika*, edited by Maga D. 2020; pp. 254–261.