Schiele, Martin; Augsburg, Klaus:

# Datenbasierte Simulationsumgebung für das Training autonomer, maschineller Regelungssysteme

**DOI:** 10.22032/dbt.40838

**URN:** urn:nbn:de:gbv:ilm1-2020200271

*Original published in:* 14. Magdeburger Maschinenbau-Tage 2019 - Magdeburger Ingenieurtage / Kasper, Roland. - Magdeburg : Otto von Guericke Universität Magdeburg, Fakultät Maschinenbau, Institut für Mobile Systeme - Lehrstuhl Mechatronik, 2019. - (2019), p. 300-309.

*Original published:* 2019-09

*ISBN:* 978-3-944722-81-8

[*Visited:* 2020-02-25]

# Datenbasierte Simulationsumgebung für das Training autonomer, maschineller Regelungssysteme

M. Sc. Martin Schiele [1)], Univ.-Prof. Dr.-Ing. Klaus Augsburg[2)]

[1)] Fahrzeugtechnik, TU Ilmenau, Deutschland, Martin.Schiele@tu-ilmenau.de, +49 3677 69 49 50

[2)] Fahrzeugtechnik, TU Ilmenau, Deutschland, Klaus.Augsburg@tu-ilmenau.de, +49 3677 69 38 42

## Zusammenfassung

Nach der Auslegung und Produktion komplexer Systeme, folgen oft umfangreiche Tests zum Nachweis der Produktfähigkeit über die Grenzen der vorausgelegten Randbedingungen hinweg. Solche Tests erfolgen häufig automatisiert unter bestmöglicher Reproduktion der späteren Einsatzbedingungen. Die dabei gemessenen Daten spiegeln das Systemverhalten in Form von Mess-, Berechnungs- und Stellgrößen wieder. Mithilfe dieser Daten soll in der vorliegenden Arbeit ein Vorgehen beschrieben werden, dass sie nutzt und durch den Einsatz neuronaler Netze in eine Black-Box Simulationsumgebung überführt. Diese Simulationsumgebung wird dann dazu verwendet, dass Systemverhalten vorherzusagen, Abweichungen zu erkennen und vor allem autonome Lernalgorithmen auf das System anzuwenden, denn selbst wenn das System fertig entwickelt ist, so bedarf es oft immer noch einer manuellen Parametrisierung der Systemsteuerung.

Schlüsselwörter: neuronale Netze, Black Box, maschinelles Lernen, big data, digitaler zwilling

## 1. Einleitung

Komplexe Systeme, wie Verbrennungsmotoren und Kraftwerke, bestehen oft selbst aus einer ganzen Reihe von Subsystemen. Bei der Auslegung des Gesamtsystems werden diese Subsysteme für gewöhnlich als Baugruppen zusammengeführt und übernehmen gemeinsam bestimmte Aufgaben. In einem Verbrennungsmotor sind das Systeme wie der Turbolader zur Erhöhung der Zylinderfüllung, die Kühlwasserpumpe zur aktiven Temperaturregelung und das Einspritzsystem zur Kraftstoffversorgung des Brennraumes, um nur einige wenige zu nennen.
All diese Subsysteme werden von unterschiedlichen Herstellern geliefert und müssen nach Vorgaben eines Lastenheftes gewisse Anforderungen erfüllen. Zum Nachweis der Maschinenfähigkeit, werden dazu umfangreiche Tests auf dafür eigens aufgebauten Komponentenprüfständen durchgeführt und überwacht. Die Prüfstände sind hierfür mit umfangreicher Sensorik ausgestattet und zeichnen das Systemverhalten im relevanten Bereich (durch Lastenheft vorgegeben) auf. Bei der Kreation neuer Subsysteme werden oft Simulationsmodelle mit modellbasiertem Ansatz generiert. Diese dienen dann als Auslegungswerkzeug, müssen jedoch im Anschluss mit den am Prüfstand gewonnenen Daten validiert werden. Erst wenn Modell-Simulation und reale Messwerte übereinstimmen, gilt das Modell als validiert und kann innerhalb des validierten Bereiches zur weiteren Modellauslegung verwendet werden. Der in dieser Arbeit beschriebene Ansatz umgeht die Modellerstellung und nutzt die gewonnenen Daten direkt als Grundlage zur Generierung einer Black Box Simulationsumgebung, oder auch „digitaler Zwilling“, was allerdings erst nach der Fertigstellung des Produktes (Subsystem) möglich ist, da erst dann Messdaten vorliegen.
Dieser digitale Zwilling soll in der weiteren Verwendung vor allem dazu dienen eine nahezu exakte Kopie der realen Umgebung / des Simulationsobjektes darzustellen, um maschinelle Lernalgorithmen darauf anwenden zu können. So ist es möglich unter Vorgabe gewisser Zielfunktionen (Belohnungsfunktion) zu einer Steuerungslösung zu gelangen ohne am realen System arbeiten zu müssen. Dieses Vorgehen verspricht Havarie Sicherheit, hohen Parallelisierungsgrad und enorme Kostenersparnis durch große „sample efficiency“.

## 2. Maschinelles Lernen

Maschinelle Lernalgorithmen und die dazugehörigen Wissenschaftsfelder existieren seit weit über 50 Jahren und ihr Beginn wird mit der Dartmouth Conference 1955 in New Hampshire benannt [1]. Im Laufe der Jahrzehnte

wurden Modelle und Vorgehensweisen entwickelt, um Algorithmen zu ermöglichen, bestimmte Aufgaben zu erfüllen. Seit Anfang des 21 Jahrhunderts kam es durch verbesserte Rechentechnologien, vor allem seitens *NVIDIA* und deren *CUDA* Plattform, zu zunehmendem Interesse der Wissenschaft am Lösen parallelisierter Rechenaufgaben.

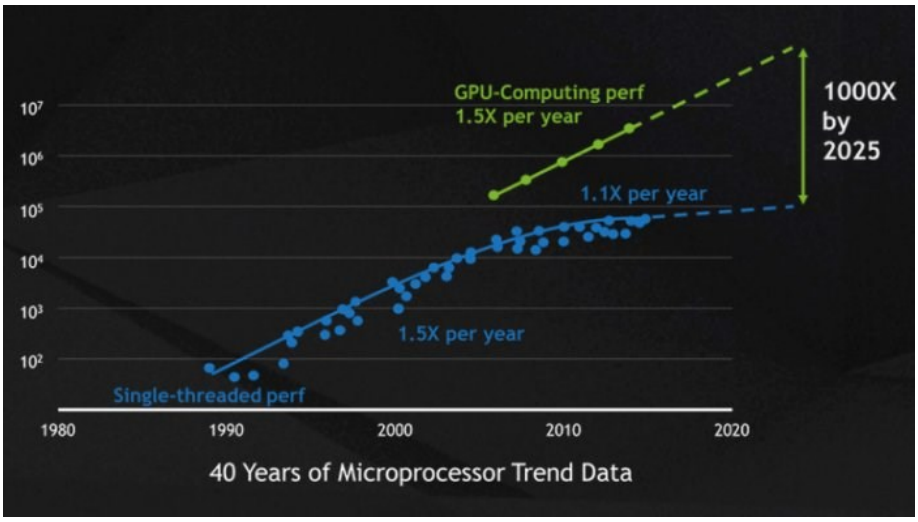


**Abbildung 1. Darstellung von *Nvidia* zur Rechenleistungssteigerung der letzten 40 Jahre [2]**

Das maschinelle Lernen setzt sich dabei aus drei Bereichen zusammen. Dem unbeaufsichtigten Lernen, dem beaufsichtigten Lernen und dem bestärkten Lernen. Unbeaufsichtiges Lernen ist kein Baustein dieser Arbeit und wird daher nicht thematisiert, es beschäftigt sich allerdings mit dem Aufbereiten von Daten, aus denen keine klare Struktur bekannt ist. Das beaufsichtigte Lernen dient der Erstellung von Modellen mit klar bezeichneten Input und Output Parametern, sogenannten Features. Vor allem Klassifikations- und Regressionsaufgaben lassen sich mit dieser Form des maschinellen Lernens lösen und sie ist ein wichtiger Teil der in dieser Arbeit beschriebenen Vorgehensweise. Bestärkendes Lernen ist ebenfalls ein in dieser Arbeit genutztes, wichtiges Teilgebiet. Es liefert Methoden, die dazu dienen einen Algorithmus oder eine Funktion so anzupassen, dass diese nach bestimmten Vorgaben Entscheidungen trifft.

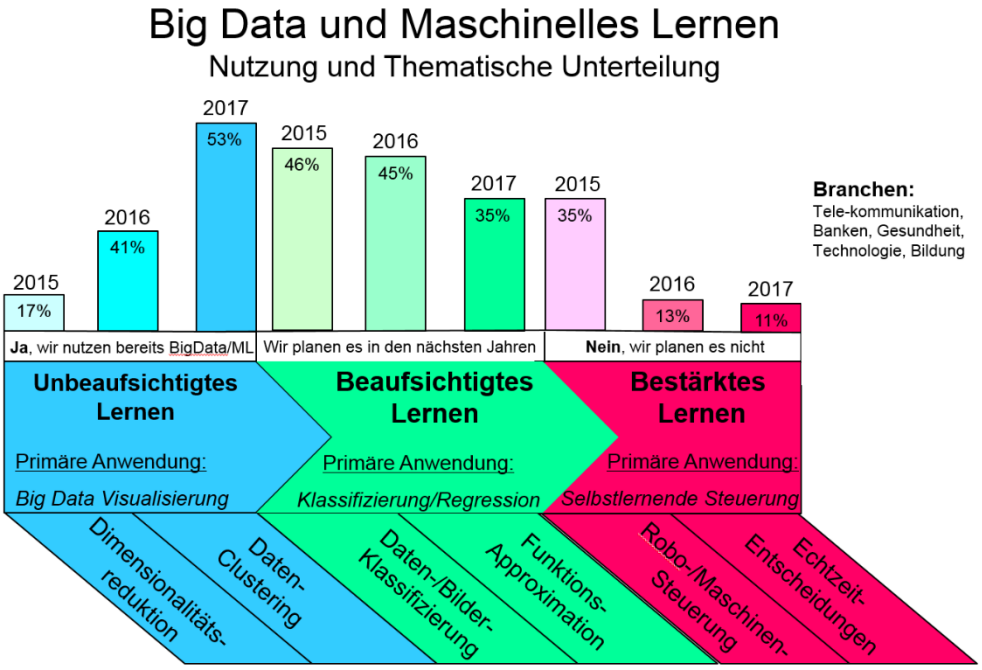


**Abbildung 2. Unterteilung maschinellen Lernens und diverse Brancheninteressen am Thema [3]**

Wie in Abbildung 2 ersichtlich, steigt seit Jahren das Interesse von Unternehmen sich dem Thema zu widmen. Zum einen, weil es ein Trend ist und zum anderen, weil sich daraus tatsächlich neue Möglichkeiten ergeben. Vor allem für Unternehmen, die viele Daten „produzieren".
Eine solche Institution ist für gewöhnlich auch eine Forschungseinrichtung. An Fachgebiet Kraftfahrzeugtechnik der TU Ilmenau werden ständig Versuchsaufbauten an Prüfständen mit umfangreicher Messelektronik ausgestattet und getestet. Immer liegt diesem Vorgehen ein bestimmter Zweck zugrunde, wobei jedoch viele gemessene Daten nicht zwangsläufig weiterverwendet werden. Die Daten repräsentieren das Systemverhalten in verschlüsselter Form. Ihnen liegen zwar physikalische, chemische und biologische Gesetzmäßigkeiten zugrunde, diese sind aber nicht direkt sichtbar. Maschinelles Lernen bietet jedoch die Möglichkeit, auch ohne Modellansatz, zu korrekten Korrelationen zu finden.

In den vergangenen Jahren wurde das bestärkende Lernen zunehmend populärer [4–6]. Vor allem in Simulationsumgebungen, wie Videospielen und zum Erlernen einer Steuerung für Roboter [7]. Es zeigten sich vor allem in Spielumgebungen Leistungen, die denen von Menschen rasch überlegen waren [4, 8]. Das führt zu der Frage, in wie weit sich solche Algorithmen auf Steuerungsaufgaben übertragen lassen, die bisher von Menschen übernommen werden und was könnten das konkret für Aufgaben sein.

Als Beispiel wird der Verbrennungsmotor betrachtet. Dieser wird im Auslieferungszustand über ein fertig parametriertes Motorsteuergerät (ECU – Electronic Control Unit, oder Engine Control Unit) kontrolliert [9]. Dieses Steuergerät ist jedoch ein Produkt, dass nach Fertigstellung des Motors erst von Menschenhand parametrisiert werden muss [10]. Das bedeutet ein Produkt, dass von Menschen entworfen, wieder und wieder detailverändert und nach klaren Vorgaben funktioniert, am Ende der Produktionskette erst einmal lauffähig programmiert werden muss. Und auch wenn die Funktionsweisen klar sind, ist dies ein Prozess der enorm viel Aufwand bedeutet. Professionelle Anbieter wie die IAV (Ingenieurgesellschaft Auto und Verkehr), bieten diese Leistung für Automobilkonzerne an. Mit dieser Arbeit soll versucht werden eine Vorgehensweise aufzuzeigen, die es ermöglicht nahezu ohne menschliche Zuhilfenahme, jedwede Form komplexer Systeme zu steuern.

## 3. Bottom Up – Maschinelles Lernen am virtuellen Baugruppensystem

Ein System wie der Verbrennungsmotor lässt sich in einzelne Subsysteme unterteilen. Diese Subsysteme beschreiben in Summe das Gesamtsystem, Motor. Ausgehend von dieser Annahme, ist es möglich eine Simulationsumgebung aufzubauen, welche aus Blackbox Modellen der Subsysteme besteht.

Die folgende Illustration 3, zeigt den beispielhaften Zusammenhang anhand des Verbrennungsmotors.

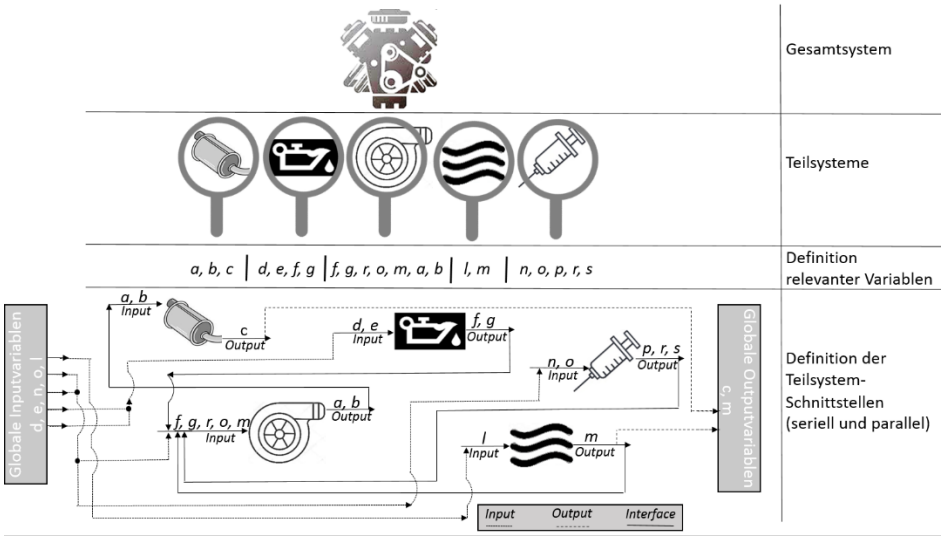


**Abbildung 3. Gesamtsystem Verbrennungsmotor unterteilt in Teilsysteme**

Jedes der Subsysteme weist ein eigenes Systemverhalten auf. Wird ein Verbrennungsmotor zusammengestellt, gibt ein Lastenheft die Anforderungen an jedes der einzelnen Subsysteme vor. Lieferanten dieser Subsysteme

testen ihre Produkte für gewöhnlich sehr umfangreich an dafür vorgesehenen Komponentenprüfständen, um die Maschinenfähigkeit für den Kunden nachzuweisen.

Bei diesen, oft sehr umfangreichen Tests, entstehen große Datensätze, da die Prüflinge mit Messinstrumenten ausgestattet werden, welche im Serienbetrieb oft nicht mehr angebracht sind. Wird also beispielsweise ein Turbolader am Heißgasprüfstand auf Lebensdauer, Maximaldrehzahl und sein Kennfeld getestet, so liefern diese Tests Daten, welche zur Validierung einer modellbasierten Simulationsumgebung dienen könnten, oder gleich als Blackbox Simulationsumgebung, ohne Umweg über ein entsprechendes Werkzeug, genutzt werden können.

Dazu sind, wie in Abbildung 3 gezeigt wird, die Schnittstellen zwischen den Subsystemen zu definieren und alle relevanten Variablen zu messen. Das Verfahren kann als „Bottom – Up“ bezeichnet werden, da jedes Teilsystem in ein eigenes Blackbox Modell überführt werden muss, um anschließend zum Gesamtsystem zusammengeschaltet zu werden. Dafür wird überwachtes Lernen eingesetzt und neuronale Netze so trainiert, dass Sie die Systemzusammenhänge akkurat approximieren können (Black Box Simulation).

## 4. Simulationswerkzeuge – neuronale Netze

Zur Simulation von verschiedenen physikalischen, chemischen und biologischen Systemen existieren einige kommerzielle Werkzeuge. Einfach betrachtet ist das Simulieren eines Systems nicht viel mehr als das Aufstellen einer oft nichtlinearen Funktion. Unter bestimmten Eingangsumständen $X$ ergeben sich Ausgangsvariablen $Y$. Demnach liegt hier ein Regressionsproblem vor. In der realen Welt ist oft mit kontinuierlichen Aktions- und Zustandsräumen umzugehen. Aufgrund ihrer Kontinuität existieren praktisch unendlich viele Kombinationen, welche nie durch Messungen an Komponentenprüfständen vollumfänglich abgebildet werden können. Das zu nutzende Werkzeug, muss also zumindest die Fähigkeit besitzen, unter Eingabe diskreter Messpunkte die dazwischenliegenden Zusammenhänge zu interpolieren.

Als Blackbox Simulationswerkzeug wird in dieser Arbeit auf neuronale Netzwerke zurückgegriffen. Ihre Fähigkeiten komplexe Zusammenhänge abzubilden und zu interpolieren sind weitreichend bekannt [11–13]. Dabei wird wie folgt vorgegangen. Eine Reihe von zusammengehörigen Datenpunkten wird in „Inputs“ und „Outputs“ unterteilt. Über das Gradientenabstiegsverfahren werden die zufällig initialisierten Gewichtungen der einzelnen Knotenpunkte soweit angepasst, dass die benutzten Trainingsdatensätze innerhalb einer Bestimmten Fehlerquote die „Inputs“ optimal zu den „Outputs“ umrechnen. Dieser Zusammenhang soll an einem einfachen Multi Layer Perceptron (MLP) in Abbildung 4 gezeigt werden.

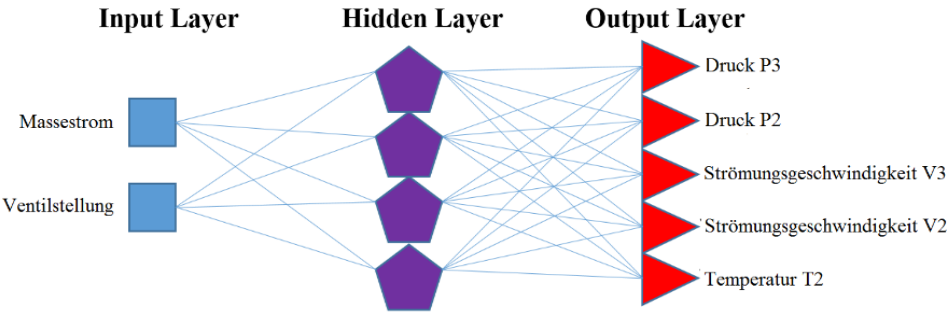


**Abbildung 4. Voll verknüpftes MLP Netzwerkt zur Regression/Simulation von Drücken, Strömungsgeschwindigkeiten und einer Temperatur [14]**

Das genannte Beispiel wird im weiteren Verlauf noch ausführlich beschrieben. Einfach illustriert sind hier zwei Inputvariablen „Massestrom“ und „Ventilstellung“ welche am Ende in 5 Outputvariablen umgewandelt werden. Das neuronale Netzwerk stellt damit über diese Variablen ein Systemverhalten dar.

Übertragen auf das Beispiel des Verbrennungsmotors, ließen sich alle Subsysteme virtuell durch neuronale Netzwerke „substituieren“. Ihr spezifisches Systemverhalten, welches den gemessenen Daten der Prüfstandtests inhärent ist, kann somit simuliert werden. Abbildung 5 zeigt eine Abwandlung von Abbildung 3 derart, dass die einzelnen Teilsysteme nun durch neuronale Netze ersetzt wurden. Typische Netztopologien können MLPs, LSTMs (long short term memory) oder GRUs (gated recurrent unit) sein [15, 16]. Dargestellt sind allerdings in jeder Illustration ausschließlich MLPs, um den Detailgrad nicht unnötig, unübersichtlich zu steigern.

In der vorliegenden Quelle [17] ist ein Überblick verschiedener, bereits unternommener Versuche zu finden neuronale Netze (NN) in den Betrieb von Verbrennungsmotoren mit einzubinden. Vor allem das Ersetzen von LookUp-Tables durch NN scheint hierbei oft hilfreich, da sie kontinuierlich Werte zu interpolieren vermögen.

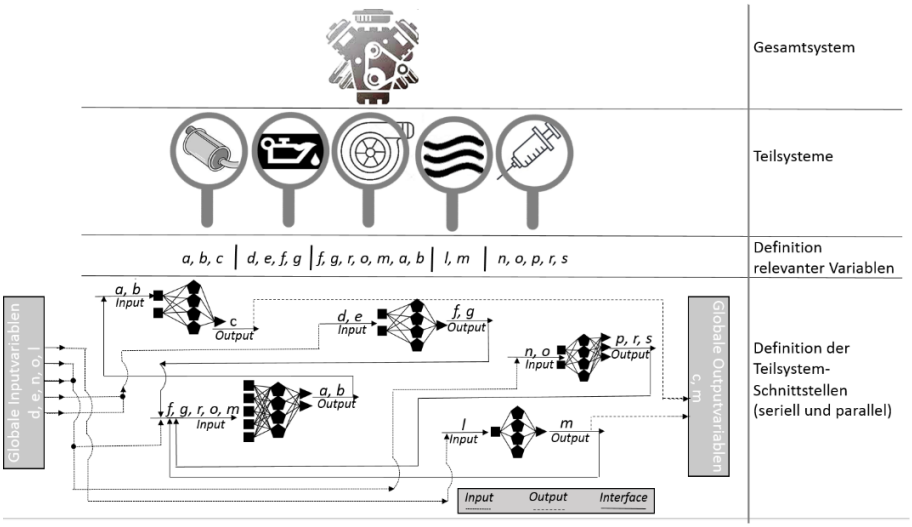


**Abbildung 5. Gesamtsystem Verbrennungsmotor in Teilsysteme unterteilt und Teilsysteme durch neuronale Netzwerke approximiert / simuliert (Blackbox Modelle)**

## 5. Der digitale Zwilling als Werkzeug

Das Nachbilden eines Gesamtsystems aus Blackbox Simulationsmodellen seiner Teilsysteme hat den Zweck die für gewöhnlich anfallenden Messdaten sinnvoll zu Nutzen und schon vor Fertigstellung des realen Gesamtsystems eine auf empirischen Daten beruhende Simulationsumgebung zu schaffen. Abbildung 6 fasst die „Transformation“ noch einmal zusammen (die Illustration ist idealisiert, inwiefern Systemschnittstellen seriell oder parallel verlaufen ist systemabhängig).

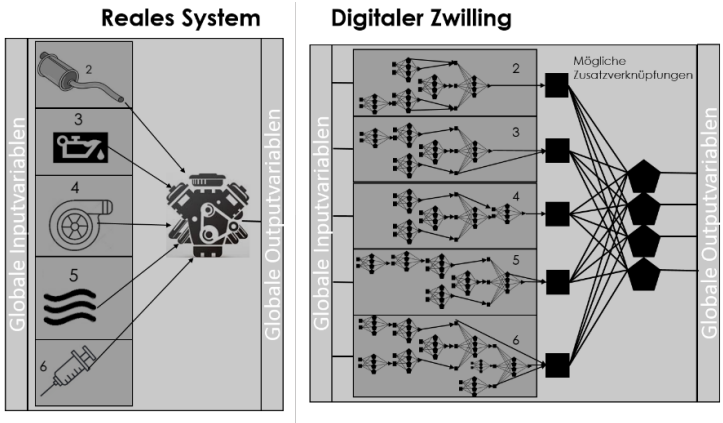


**Abbildung 6. Gesamtsystem mit Subsystemen, real und digital**

Wie schon in der Einleitung beschrieben, gibt es mehrere Anwendungsmöglichkeiten für einen solchen digitalen Zwilling.

1. Als Simulationsmodell welches es ermöglicht „offline“, also nicht am realen System, sondern am Computer Steuerungsaufgaben über einen „bestärkenden Lernalgorithmus“ anzutrainieren.

2. Als Vorhersagemodell im Realbetrieb. Das bedeutet man kann etwaige Aktionen schon vor realer Durchführung am digitalen Zwilling ausprobieren und schauen in wie fern Randbereiche des Gesamtsystems gefährdet sind.

3. Zur Verschleiß und Toleranzkontrolle. Wird beispielsweise ein Master Turbolader zur Messung eines Kennfeldes benutzt um dessen Daten zur Generierung eines Subsystemzwillings verwendet können, treten beim Einsetzen eines Serienturboladers im Realsystem fertigungstoleranzbedingt Unterschiede auf. Diese ließen sich Messen und gegebenenfalls über ganze Chargen überwachen. Auch Sensoralterung ließe sich vorhersagen. Sind im Realsystem Sensoren verbaut, welche auch beim Erstellen des digitalen Zwillings vorhanden waren und messen diese Trotz gleicher Randbedingungen zu große Unterschiede, so kann dies sowohl an einem fehlerhaften System, als auch an einem Sensorfehler, oder an Sensoralterung liegen. In jedem Fall wird ein auffälliger Unterschied zwischen Vorhersagewert und Realwert zu erhöhter Aufmerksamkeit am entsprechenden Subsystem führen und eine mögliche Havarie rechtzeitig vorhersagen oder andeuten.

4. Virtuelle Sensoren stehen ebenfalls zur Verfügung. Hat man wie beim Verbrennungsmotor im Serienbetrieb nur wenige verfügbare Messwerte, so können nicht verfügbare Werte aus anderen korreliert werden.

Im Folgenden soll Punkt 1 genauer an einem Beispiel vorgestellt werden. Die detaillierte Modellerstellung wird nicht genauer erläutert um den Fokus auf die Methode und die erzielten Ergebnisse zu legen. Sie ist genauer in Quelle [14] erläutert.

## 6. Testsystem Heißgasprüfstand

Um die vorgestellte Methode für den Anwendungsfall einer reglerfreien Steuerung, basierend auf bestärkendem lernen zu validieren, wird folgend ein reales, wenn auch einfaches Problem vorgestellt (Abbildung 7).

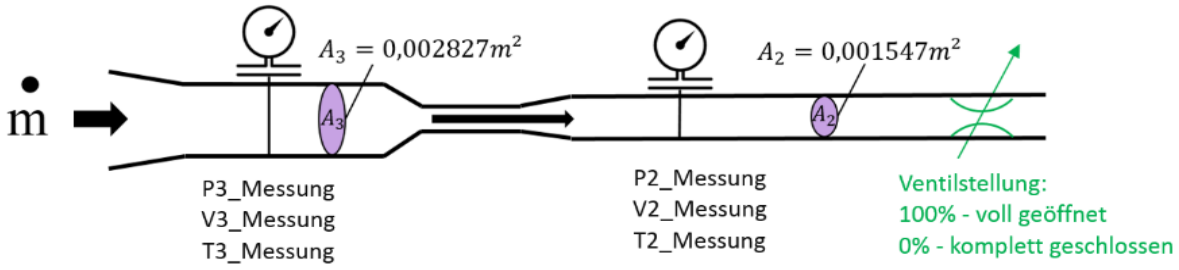


**Abbildung 7. Testsetup Heißgasprüfstand, Massestrom und Ventilsteuerung [14]**

Am Heißgasprüfstand des Fachgebietes Fahrzeugtechnik wurde eine simple Druckluftverrohrung installiert. Es gibt zwei Input Variablen (Steuergrößen), nämlich Massestrom und Ventilstellung, welche in Kombination bestimmte Drücke, Strömungsgeschwindigkeiten und Temperaturen an zwei verschiedenen Messstellen kontrollieren. Stellt man also einen festen Massestrom ein und verringert über das Ventil den Strömungsquerschnitt steigt der Systemdruck. Drücke, Strömungsgeschwindigkeiten und Temperaturen sind Output Variablen und der Variablenzusammenhang ist bereits in Abbildung 4 gezeigt worden. Dieses Setup und seine Digitalisierung wurde schon ausführlich in [14] beschrieben. Der Zusammenhang ist teilweise stark nichtlinear, besonders zwischen den Outputvariablen P3 und P2. Da hier ein Blackbox Modell verwendet wird, soll nicht explizit auf den physikalisch korrekten Zusammenhang über Gleichungen eingegangen. Dieser Zusammenhang wird aufgegriffen, wenn es den Ergebniserklärungen dienlich erscheint. Reale physikalische Zusammenhänge sind für Blackbox Simulationen und insbesondere neuronale Netze belanglos, sie approximieren ausschließlich Muster. Ähnlich einem Menschen, der unter der Dusche steht und über einen Drehregler den optimalen Komfortbereich via Massestrom und Temperatur zum Duschen einstellen möchte. Dabei ist es egal wie hoch der Wärmeübergangskoeffizient der Haut oder die Wärmekapazität des Wassers ist, die Erinnerung spiegelt ein Muster der Drehreglerstellung im gewollten Komfortbereich wieder.

Wenn man nun in der Lage ist über Eingabe zweier Variablen (Massestrom und Ventilstellung) den Zustand des Systems (P3, T3, V3, P2, T2, V2) vorherzusagen, ist die Simulationsumgebung geschaffen. Abbildung 8 zeigt die Zusammenhänge in Form zweier Diagramme.

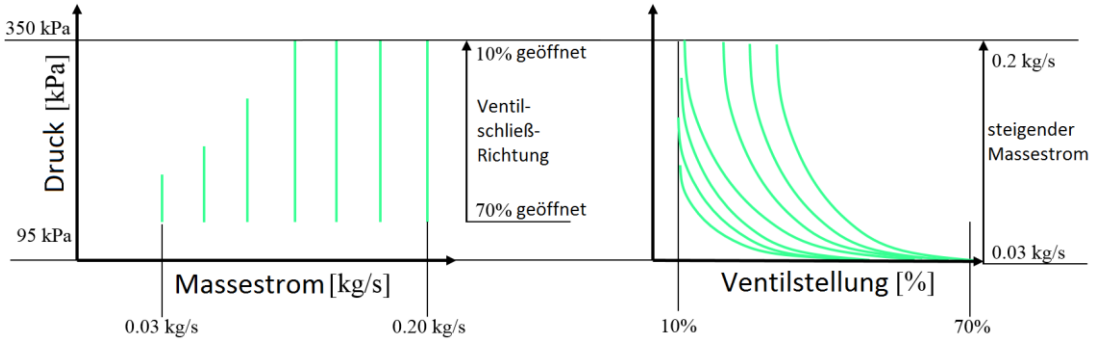


**Abbildung 8. Qualitative Kennfelddarstellung für Strömungsgeschwindigkeit (links) und Druck (rechts)**

Die erreichte Abweichung zwischen Modellvorhersage aus trainierten Trainingsdatensätzen und einem Validierungsdatensatz beträgt etwa 0,44% Wurzel des mittleren quadratischen Fehlers (RMSE – root mean squared error) und 0,0022% des mittleren quadratischen Fehlers (MSE – mean squared error).

Mithilfe dieser Simulationsumgebung ist es nun möglich einen maschinellen Lernalgorithmus zu Trainieren. Im Folgenden wird ein Actor-Critic (AC) verfahren verwendet, das den Namen „D4PG – Distributed Distributional Deep Deterministic Policy Gradient“ [18] trägt. Der Algorithmus ist in der Lage kontinuierliche Kontrollaufgaben nach Stand der Technik zu übernehmen. Zwei neuronale Netze, der „Actor“ und der „Critic“ werden dazu trainiert und zyklisch kopiert. Das Lernen geschieht recht langsam, sorgt aber dafür, dass die Lernkurve in der Regel konvergiert. Für das vorliegende System wurden rund 3 Millionen Zustandsänderungen durchlaufen.

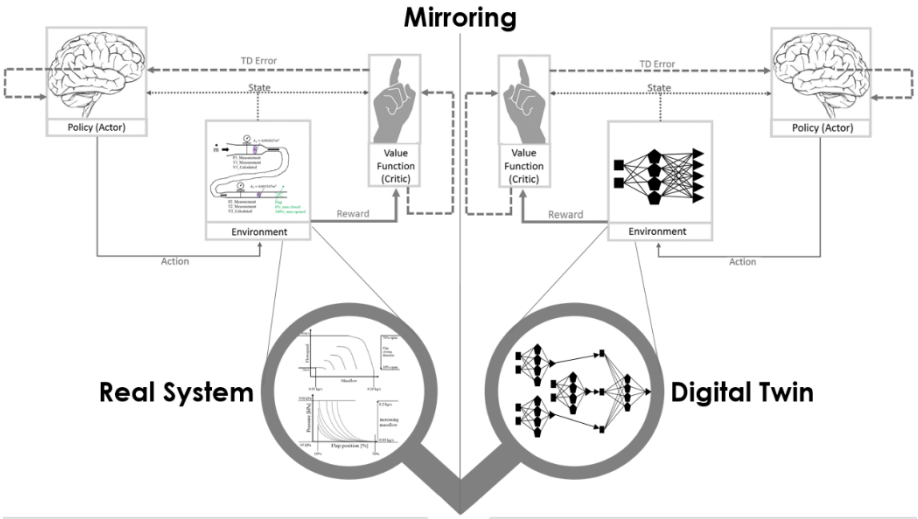


**Abbildung 9. Gespiegeltes System Heißgasprüfstand, eingebunden in eine Actor-Critic Lernumgebung**

In Abbildung 9 ist die Lernumgebung illustriert. Der Actor ist vergleichbar mit einem Kind, dass auf dem Spielplatz spielt, wobei der Critic die Mutter darstellt, welche die einzelnen Aktionen des Kindes bewertet und so seine Prägung auf bestimmte Verhaltensmuster bildet. Der AC-Algorithmus trainiert jetzt unter Vorgabe einer Belohnungsfunktion, die Steuerung des Prüfstandes. Als Zielvorgaben werden ein zu erreichender Zieldruck und eine zu erreichende Zielströmungsgeschwindigkeit nach Gleichung 5.1 vorgegeben. In der hier behandelten Beispielumgebung wurden viele verschiedene Zielfunktionen getestet, jedoch hier nur diese eine genauer beschrieben.

$$Reward = -abs(P_{Ziel} - P_{Ist}) - abs(V_{Ziel} - V_{Ist}) \tag{5.1}$$

Gleichung 5.1 wird nur dann zu Null, wenn der gewünschte Druck und die gewünschte Strömungsgeschwindigkeit erreicht werden. Der Algorithmus erhält über die „Reward-Funktion“ eine Belohnung für Zustands-Aktions Tupel. So lernt er durch optimieren des Summenrewards, also den über mehrere Episoden aufsummierten Reward seinen Aktionsspielraum (Gleichung 5.2) so zu nutzen, die Belohnung möglichst groß (gegen 0 strebend) ausfallen zu lassen. Der Aktionsspielraum Sieht wie folgt aus (2 Werte):

$$Aktionsraum = [\pm \Delta Ventil, \pm \Delta \dot{m}] = [\pm 0{,}8\ \%, \pm 0{,}005\ \tfrac{kg}{s}] \tag{5.2}$$

Der übergebene Zustandsraum lässt sich via Gleichung 5.3 folgendermaßen darstellen (8 Werte):

$$Zustandsraum = [Aktion_{Ventil}, Aktion_{\dot{m}}, Ventil_{ist}, \dot{m}_{ist}, P_{ist}, V_{ist}, P_{ist} - P_{soll}, V_{ist} - V_{soll}] \quad (5.3)$$

Nach dem Trainingsprozess wird das Actor-Netzwerk extrahiert und am Prüfstand eingesetzt. Nun erhält man ein neuronales Netzwerk, dass den Umgebungszustand (Gleichung 5.3) als Input zugeführt bekommt und nach Zielvorgaben (Gleichung 5.1) selbstständig die nötigen Steuerelemente bedient (Gleichung 5.2) um sie zu erreichen. Im PKW könnte später die Zielvorgabe sein, die vom Fahrer gewünschte Fahrzeugleistung via Pedalstellung zu erreichen, unter der Prämisse, wenig Schadstoffe auszustoßen. Die am Prüfstandbeispiel möglichen Aktionen sind dabei das Verstellen des Massestroms im Bereich von 0,03kg/s bis 0,2kg/s und das Verstellen des Ventils im Bereich von 10 % bis 70% Öffnungsweite.

Wie beschrieben wurde das Actor-Netzwerk am Prüfstand getestet. Abbildung 10 zeigt einen solchen Versuch. Nach Vorgabe einer gewünschten Strömungsgeschwindigkeit und eines gewünschten Zieldruckes, werden innerhalb des Actor-Netzwerkes die antrainierten Zielfunktionen in Form von Aktionen versucht zu erfüllen. Selbst wenn ein Wert aufgrund unrealer Werte nicht erreichbar wäre, würde das System dennoch versuchen nächstmöglich dran zu kommen.

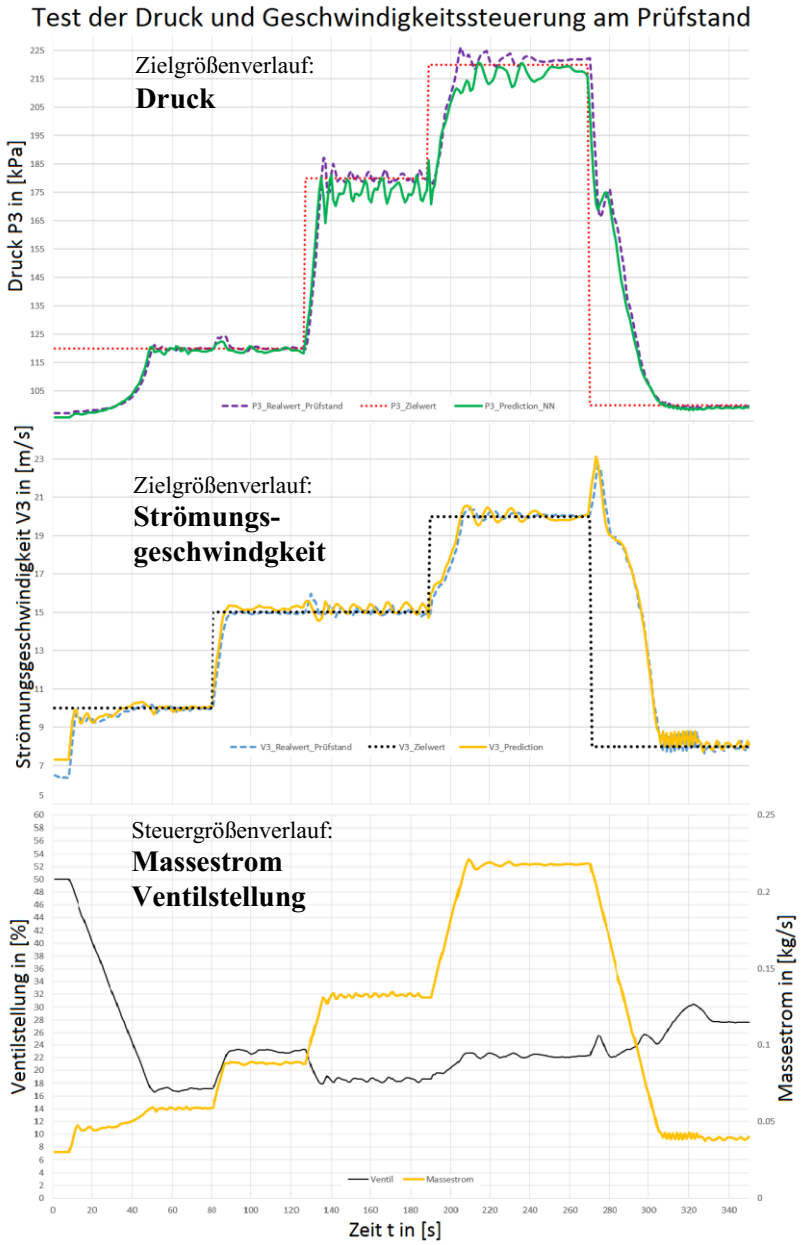


**Abbildung 10. Ergebnisse des realen Prüfstandversuches, eines D4PG trainierten Actor-Netzwerkes**

Je nach Trainingsdauer und Anwendung des Steueralgorithmus kommt es zu Einschwingprozessen. Wird das Ventil zugestellt, verringert sich die Strömungsgeschwindigkeit und der Druck steigt. Erhöht sich der Massestrom, steigen Druck und Strömungsgeschwindigkeit gemeinsam. Sowohl die Zielgeschwindigkeit, als auch der Zieldruck werden immer sehr gut erreicht. Wichtig ist dabei noch einmal anzumerken, dass der handelnde Agent die reale Umgebung nie gesehen hat. Er lernte das Umgebungsverhalten ausschließlich aus dem durch Messwerte erstellten, digitalen Zwilling der Umgebung. Damit kann gezeigt werden, dass reale Probleme, anhand von digitalen, neuronalen Netzwerk-Zwillingen gelöst werden können, insofern eine kleine Datenbasis vorliegt.

Des Weiteren hat sich gezeigt, dass der digitale Zwilling keine perfekte Kopie der Umgebung sein muss. Selbst wenn Vorhersagewerte und Realwerte nicht ansatzweise übereinstimmen, so regelt der Agent die Zielvorgaben trotzdem genauso gut ein, als wäre beides gleich, insofern der digitale Zwilling die relativen Variablenzusammenhänge richtig wiedergibt. Das liegt darin begründet, dass der Agent lernt, welchen Einfluss jede mögliche Aktion bezüglich der Zielwerte hat. Nämlich den Zusammenhang zwischen Verstellen von Massestrom und Ventilstellung als Einfluss auf Strömungsgeschwindigkeit und Druck. Er orientiert sich also an Relativwerten, an Gradienten und niemals an Absolutwerten (sofern diese im State – Zustand mit übergeben werden).

## 7. Fazit und Ausblick

Es konnte gezeigt werden, dass es möglich ist ein reales System mit Mess- Berechnungs- und Stellgrößen zu erfassen und anschließend als Black Box Simulationsmodell in Form neuronaler Netzwerke zu überführen. Mithilfe dieser digitalen Kopie eines Realweltproblems, lassen sich aus der Informatik bekannte, maschinelle Lernalgorithmen auf Steuer- und Optimierungsprobleme anwenden. Diese Algorithmen brauchen viele Trainingsdatensätze und Hyperparameteroptimierung, wodurch ein Lernen am echten System ineffizient wäre. Außerdem müsste das Explorationsverhalten stark eingeschränkt werden, um Havarien vorzubeugen. Das Nutzen einer digitalen Kopie gelang im vorliegenden Fall für die Druck- und Strömungsgeschwindigkeitsregelung eines Rohrsystems via D4PG Actor-Critic Verfahren.

Da es das Ziel sein soll zu zeigen, dass ein komplexes System wie der Verbrennungsmotor auf diese Weise automatisch steuerbar wird, ist der nächste Schritt ein Subsystem, den Abgasturbolader, digital zu kopieren. Ist es möglich, den Abgasturbolader in Verbindung mit dem Motor anhand realer Messwerte zu simulieren, kann ein maschineller Lernalgorithmus versuchen die Ladedrucksteuerung vollautomatisch, anhand übergeordneter Optimierungsfunktionen zu übernehmen. Das digitale Kopieren ist jüngst gelungen[19], Abbildung 11.

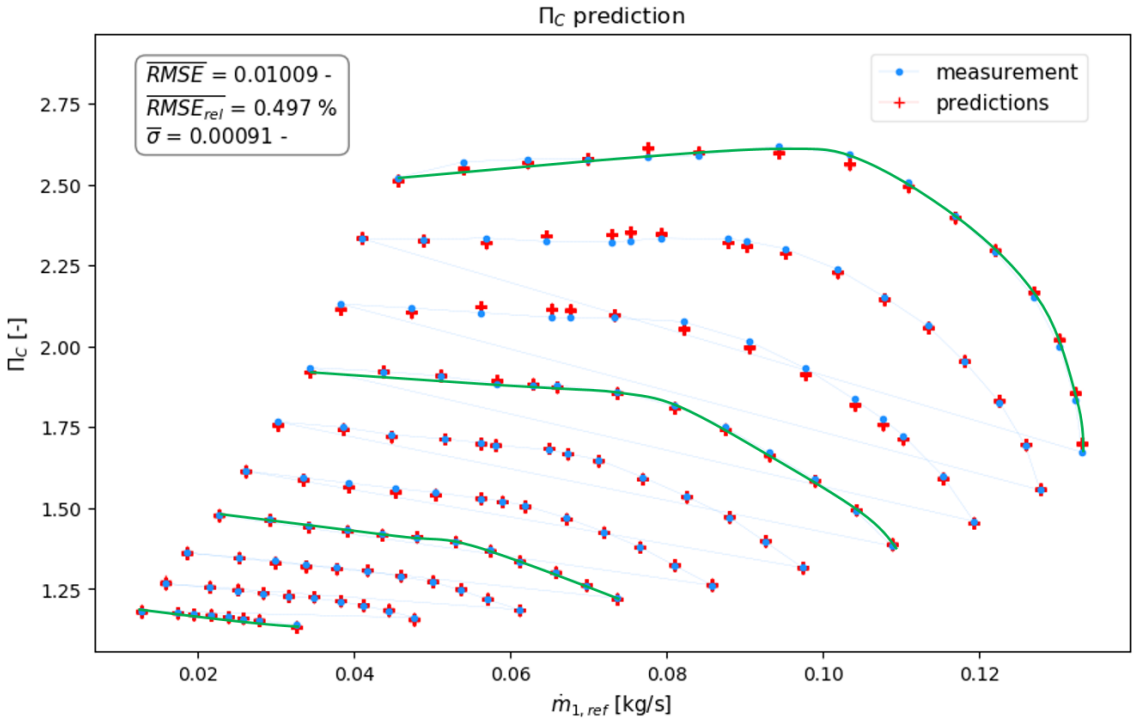


**Abbildung 11. Black Box Abgasturboladersimulation, Verdichterkennfeld. Trainierte Linien in Grün**

Sämtliche Zusammenhänge innerhalb des Turboladers haben sich mit Fehlerquoten weit unter 1% RMSE wiedergeben lassen. Dabei wurden 10 Kennlinien gemessen (blaue Punkte), jedoch nur 4 antrainiert. Abbildung 11 zeigt, wie gut das neuronale Netzwerk darin war die unbekannten Messpunkte vorherzusagen (rote Kreuze). Auch Verdichterwirkungsgrad und Turbinenkennfelder haben sich sehr gut vorhersagen lassen. Die Wiedergabegenauigkeit steigt zunehmend mit der Anzahl an Trainingsdaten, wobei das Nutzen von 40% der Messwerte bereits zu mehr als zufriedenstellenden Ergebnissen führte. Doch was bedeutet das für weitere Untersuchungen?

Es eröffnet zukünftig nun die Möglichkeit, anhand eines „digitalen Turboladers“ eine Ladedrucksteuerung zu trainieren, die dann am Motor implementiert werden kann. So wäre es möglich die Motorsteuerung Subsystem, für Subsystem zu optimieren, oder bei Neuaufbau initial zu installieren.

## Literaturverzeichnis


[1] M. Khosrow-Pour, Hg., *Advanced methodologies and technologies in artificial intelligence, computer simulation, and human-computer interaction*. Hershey, PA: Engineering Science Reference, an imprint of IGI Global, 2019.

[2] Martin Heller, *What is CUDA? Parallel programming for GPUs: You can accelerate deep learning and other compute-intensive apps by taking advantage of CUDA and the parallel processing power of GPUs.* Verfügbar unter: https://www.infoworld.com/article/3299703/what-is-cuda-parallel-programming-for-gpus.html (14.06.2019).

[3] Louis Columbus, *53% Of Companies Are Adopting Big Data Analytics.* Verfügbar unter: https://www.forbes.com/sites/louiscolumbus/2017/12/24/53-of-companies-are-adopting-big-data-analytics/ (14.06.2019).

[4] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, *Playing Atari with Deep Reinforcement Learning.*

[5] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever und I. Mordatch, “Emergent Complexity via Multi-Agent Competition,” Okt. 2017.

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang und W. Zaremba, “OpenAI Gym,” Jun. 2016.

[7] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng und W. Zaremba, “Learning Dexterous In-Hand Manipulation,” Aug. 2018.

[8] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar und D. Silver, “Rainbow: Combining Improvements in Deep Reinforcement Learning,” Okt. 2017.

[9] V. D. Bhise, *Automotive product development: A systems engineering implementation*. Boca Raton, London, New York: CRC PressTaylor & Francis Group, 2017.

[10] T. Cuatto, C. Passerone, C. Sansoè, F. Gregoretti, A. Jurecska und A. Sangiovanni-Vincentelli, “A Case Study in Embedded Systems Design: An Engine Control Unit,” *Design Automation for Embedded Systems*, 6. Jg., Nr. 1, S. 71–88, 2000.

[11] D. F. Specht, “A general regression neural network,” (eng), *IEEE transactions on neural networks*, 2. Jg., Nr. 6, S. 568–576, 1991.

[12] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong und W.-c. Woo, “Deep Learning for Precipitation Nowcasting: A Benchmark and A New Model,” Jun. 2017.

[13] I. E. Lagaris, A. Likas und D. I. Fotiadis, “Artificial neural networks for solving ordinary and partial differential equations,” (eng), *IEEE transactions on neural networks*, 9. Jg., Nr. 5, S. 987–1000, 1998.

[14] M. Schiele und K. Augsburg, “Using a Multidimensional Input/Output Neural Network-Regression for Experienced Replay Suitability on Real World Test Bench Data,” in *2019 IEEE International Conference on Mechatronics (ICM)*: IEEE, Mrz. 2019 - Mrz. 2019, S. 433–439.

[15] S. Hochreiter und J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, 9. Jg., Nr. 8, S. 1735–1780, 1997.

[16] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk und Y. Bengio, “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation,” Jun. 2014.

[17] R. F. Turkson, F. Yan und M. K. A. Ali, *Artificial Neural Network Applications in the Calibration of Engines,* 1. Aufl. Saarbrücken: LAP LAMBERT Academic Publishing, 2016.

[18] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. TB, A. Muldal, N. Heess und T. Lillicrap, “Distributed Distributional Deterministic Policy Gradients,” Apr. 2018.

[19] Sebastian Berger, “Interpolation nichtlinearer Zusammenhänge am Beispiel des Abgasturboladers,” Masterarbeit, Kraftfahrzeugtechnik, 2019.