(19) Deutsches Patent- und Markenamt

(10) **DE 10 2013 220 432 A1** 2015.04.16

(12) # Offenlegungsschrift

(21) Aktenzeichen: **10 2013 220 432.9**
(22) Anmeldetag: **10.10.2013**
(43) Offenlegungstag: **16.04.2015**

(51) Int Cl.: ***G06F 17/10*** (2006.01)
F02D 41/26 (2006.01)
G05B 13/04 (2006.01)
F02D 41/14 (2006.01)

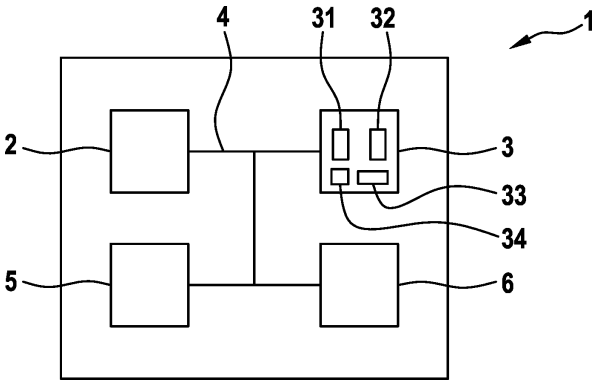
(71) Anmelder:
Robert Bosch GmbH, 70469 Stuttgart, DE

(72) Erfinder:
Vietinghoff, Anne von, 76137 Karlsruhe, DE; Markert, Heiner, 70178 Stuttgart, DE; Guntoro, Andre, 75177 Pforzheim, DE; Hanselmann, Michael, 70825 Korntal-Münchingen, DE

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Modellberechnungseinheit für einen integrierten Steuerbaustein zur Berechnung von LOLIMOT**

(57) Zusammenfassung: Die Erfindung betrifft eine Modellberechnungseinheit (3) für einen integrierten Steuerbaustein (1), der durch rein hardwarebasierte Implementierung mit einer Exponentialfunktion, Summierfunktionen und Multiplikationsfunktionen in mindestens einer inneren und einer äußeren Schleife versehen ist, um ein datenbasiertes Funktionsmodell, insbesondere ein Gauß-Prozess-Modell, zu berechnen, wobei die Modellberechnungseinheit (3) ferner dazu ausgebildet ist, ein LOLIMOT-Modell, das mehrere lokale Teilmodelle beinhaltet, zu berechnen.



Beschreibung

Technisches Gebiet

[0001] Die Erfindung betrifft integrierte Steuerbausteine mit einer Modellberechnungseinheit, in der datenbasierte Funktionsmodelle hardwaremäßig berechnet werden können.

Stand der Technik

[0002] Aus dem Stand der Technik sind Steuergeräte mit integrierten Steuerbausteinen mit einer Hauptrecheneinheit und einer separaten Modellberechnungseinheit zur Berechnung von datenbasierten Funktionsmodellen bekannt.

[0003] So zeigt beispielsweise die Druckschrift DE 10 2010 028 266 A1 einen integrierten Steuerbaustein mit einer Hauptrecheneinheit, einer Speichereinheit und einer zusätzlichen Logikschaltung als Modellberechnungseinheit, die zur rein hardwarebasierten Berechnung einer Abfolge von Exponentialfunktionen sowie Additions- und Multiplikationsoperationen ausgebildet ist. Dies ermöglicht es, die Berechnung von Bayes-Regressionsverfahren, die insbesondere für die Berechnung von Gauß-Prozess-Modellen benötigt werden, durch eine Hardware-Einheit in dem integrierten Steuerbaustein zu unterstützen.

[0004] Die Modellberechnungseinheit ist insgesamt zur Durchführung mathematischer Prozesse zur Berechnung des datenbasierten Funktionsmodells basierend auf bereitgestellten Hyperparametern und Stützstellen bzw. Trainingsdaten ausgelegt. Insbesondere sind die Funktionen der Modellberechnungseinheit zur effizienten Berechnung von Exponentialfunktionen und Summenfunktionen ausschließlich in Hardware realisiert, so dass es ermöglicht wird, Gauß-Prozess-Modelle mit einer deutlich höheren Rechengeschwindigkeit zu rechnen, als dies in der softwaregesteuerten Hauptrecheneinheit erfolgen könnte.

[0005] In integrierten Steuerbausteinen zum Einsatz in Steuergeräten für Kraftfahrzeuge ist neben datenbasierten Funktionsmodellen auch die Berechnung weiterer rechenintensiver Funktionen relevant. So kann für einige Aufgabenstellungen auch die Evaluierung von LOLIMOT(LOcal Linear MOdel Tree)-Modellen von Interesse sein. LOLIMOT-Modelle verwenden Gaußsche Gewichtungsfunktionen, die ähnlich den zur Auswertung von datenbasierten Funktionsmodellen, insbesondere Gauß-Prozess-Modellen, verwendeten Funktionen sind. LOLIMOT-Modelle werden beispielsweise in der Druckschrift O. Nelles, S. Sinsel, R. Isermann, UKACC International Conference an Control, 1996, beschrieben. Da herkömmliche, in Hardware implementierte Funktionen zur Berechnung von datenbasierten Funktionsmodellen nicht vollständig zur Berechnung von LOLIMOT-Modellen geeignet sind, ist eine Anpassung der integrierten Steuerbausteine, insbesondere der dort vorgesehenen Modellberechnungseinheiten, wünschenswert.

Offenbarung der Erfindung

[0006] Erfindungsgemäß sind eine Modellberechnungseinheit für einen integrierten Steuerbaustein gemäß Anspruch 1 sowie ein integrierter Steuerbaustein gemäß dem nebengeordneten Anspruch vorgesehen.

[0007] Weitere Ausgestaltungen sind in den abhängigen Ansprüchen angegeben.

[0008] Gemäß einem ersten Aspekt ist eine Modellberechnungseinheit für einen integrierten Steuerbaustein vorgesehen, der durch rein hardwaremäßige Implementierung mit einer Exponentialfunktion, Summierfunktionen und Multiplikationsfunktionen in mindestens einer inneren und einer äußeren Schleife versehen ist, um ein datenbasiertes Funktionsmodell, insbesondere ein Gauß-Prozess-Modell, zu berechnen, wobei die Modellberechnungseinheit ferner dazu ausgebildet ist, ein LOLIMOT-Modell, das mehrere lokale Teilmodelle beinhaltet, zu berechnen.

[0009] Durch die Möglichkeit der hardwarebasierten Berechnung von LOLIMOT-Modellen in einer Modellberechnungseinheit in einem integrierten Speicherbaustein, die auch für die Berechnung von datenbasierten Funktionsmodellen, insbesondere Gauß-Prozess-Modellen, ausgelegt ist, wird ermöglicht, die Berechnung von sowohl datenbasierten Funktionsmodellen als auch von LOLIMOT-Modellen hardwarebasiert in einer gemeinsamen Modellberechnungseinheit durchzuführen. Dadurch können mit nur einer Hardware-Einheit sowohl datenbasierte Funktionsmodelle als auch LOLIMOT-Modelle berechnet werden, was im Vergleich zu einer Auswertung mithilfe eines Software-Algorithmus deutlich schneller durchzuführen ist.

[0010] Weiterhin kann die Modellberechnungseinheit ausgebildet sein, um jeweils zur Berechnung eines datenbasierten Funktionsmodells eine global gültige vorgegebene Längenskala (Lengthscale) und zur Berechnung eines LOLIMOT-Modells für jede Dimension eine eigene Längenskala zu verwenden.

[0011] Gemäß einer Ausführungsform kann jeweils das datenbasierte Funktionsmodell mit vorgegebenen Stützstellenpunkten und das LOLIMOT-Modell mit Zentren für lokale Funktionen berechenbar sein, wobei die Modellberechnungseinheit ausgebildet ist, um die Stützstellenpunkte und die Zentren für eine Berechnung in einer der inneren Schleifen in gleicher Weise zu verwenden.

[0012] Es kann vorgesehen sein, dass die Modellberechnungseinheit ausgebildet ist, um im Falle einer Berechnung eines LOLIMOT-Modells das Ergebnis der in der inneren Schleife berechneten Exponentialfunktion zu normieren.

[0013] Weiterhin kann die Modellberechnungseinheit ausgebildet sein, um im Falle einer Berechnung eines LOLIMOT-Modells das Ergebnis der Exponentialfunktion mit einem für jeden Abfragepunkt jedes lokalen Teilmodells ermittelbaren Gewicht zu multiplizieren.

[0014] Ferner kann die Modellberechnungseinheit ausgebildet sein, um im Falle einer Berechnung eines datenbasierten Funktionsmodells das Ergebnis der Exponentialfunktion mit einem vorgegebenen Gewicht pro Stützstellenpunkt zu multiplizieren.

[0015] Insbesondere kann die Modellberechnungseinheit ausgebildet sein, um im Falle einer Berechnung eines LOLIMOT-Modells Eingangsdimensionen nur linear, nur nichtlinear oder linear und nichtlinear zu berücksichtigen Gemäß einem weiteren Aspekt ist ein integrierter Steuerbaustein mit einer softwaregesteuerten Hauptrecheneinheit und der obigen Modellberechnungseinheit vorgesehen.

[0016] Die Modellberechnungseinheit kann ausgebildet sein, um im Falle einer Berechnung eines LOLIMOT-Modells Zwischenergebnisse der Anwendung der Exponentialfunktion zu addieren und einen Rückgabewert durch die Summe der Zwischenergebnisse zu dividieren.

[0017] Gemäß einem weiteren Aspekt ist ein integrierter Steuerbaustein mit einer softwaregesteuerten Hauptrecheneinheit und der obigen Modellberechnungseinheit vorgesehen, wobei die Hauptrecheneinheit die Modellberechnungseinheit so betreibt, dass die Division durch die Summe der Zwischenergebnisse entweder durch eine Näherungsfunktion in der Modellberechnungseinheit oder in der Hauptrecheneinheit durchgeführt wird.

Kurzbeschreibung der Zeichnungen

[0018] Ausführungsformen werden nachfolgend anhand der beigefügten Zeichnungen näher erläutert. Es zeigen:

[0019] Fig. 1 eine schematische Darstellung eines integrierten Speicherbausteins mit einer Modellberechnungseinheit,

Beschreibung von Ausführungsformen

[0020] Fig. 1 zeigt eine schematische Darstellung einer Hardwarearchitektur für einen integrierten Steuerbaustein **1**, z. B. in Form eines Mikrocontrollers, in dem in integrierter Weise eine softwaregesteuerte Hauptrecheneinheit **2** und eine Modellberechnungseinheit **3** zur rein hardwarebasierten Berechnung eines datenbasierten Funktionsmodells und eines LOLIMOT-Modells vorgesehen sind. Die Hauptrecheneinheit **2** und die Modellberechnungseinheit **3** stehen über eine interne Kommunikationsverbindung **4**, wie z. B. einen Systembus, miteinander in Kommunikationsverbindung.

[0021] Grundsätzlich ist die Modellberechnungseinheit **3** im Wesentlichen hartverdrahtet und dementsprechend nicht wie die Hauptrecheneinheit **2** dazu ausgebildet, einen Softwarecode auszuführen. Alternativ ist eine Lösung möglich, in der die Modellberechnungseinheit **3** zur Berechnung des datenbasierten Funktionsmodells oder des LOLIMOT-Modells einen eingeschränkten, hochspezialisierten Befehlssatz zur Verfügung stellt. In der Modellberechnungseinheit **3** ist kein Prozessor vorgesehen. Dies ermöglicht eine ressourcenoptimierte Realisierung einer solchen Modellberechnungseinheit **3** bzw. einen flächenoptimierten Aufbau in integrierter Bauweise.

[0022] Die Modellberechnungseinheit **3** weist einen Rechenkern **31** auf, der eine Berechnung eines vorgegebenen Algorithmus rein in Hardware implementiert. Der Rechenkern **31** steht mit einer Abbrucheinheit **32** in Verbindung, die einen Abbruch der Berechnung des Algorithmus signalisiert, wenn eine Abbruchbedingung vorliegt. Die Modellberechnungseinheit **3** kann des Weiteren einen lokalen SRAM **33** für die Speicherung der Konfigurationsdaten umfassen. Die Modellberechnungseinheit **3** kann ebenfalls eine lokale DMA-Einheit **34** (DMA = Direct Memory Access) umfassen. Mittels der DMA-Einheit **34** ist es möglich, auf die integrierten Ressourcen des Steuerbausteins **1**, insbesondere auf einen internen Speicher **5**, zuzugreifen.

[0023] Der Steuerbaustein **1** kann einen internen Speicher **5** und eine weitere DMA-Einheit **6** (DMA = Direct Memory Access) umfassen. Der interne Speicher **5** und die weitere DMA-Einheit **6** stehen in geeigneter Weise, z. B. über die interne Kommunikationsverbindung **4**, miteinander in Verbindung. Der interne Speicher **5** kann einen (für die Hauptrecheneinheit **2**, die Modellberechnungseinheit **3** und ggf. weitere Einheiten) gemeinsamen SRAM-Speicher und einen Flash-Speicher für die Konfigurationsdaten (Parameter und Stützstellendaten) umfassen.

[0024] Die Verwendung von nicht parametrischen, datenbasierten Funktionsmodellen basiert auf einem Bayes-Regressionsverfahren. Die Grundlagen der Bayes-Regression sind beispielsweise in C. E. Rasmussen et al., „Gaussian Processes for Machine Learning", MIT Press 2006, beschrieben. Bei der Bayes-Regression handelt es sich um ein datenbasiertes Verfahren, das auf einem Modell basiert. Zur Erstellung des Modells sind Messpunkte von Trainingsdaten sowie zugehörige Ausgangsdaten einer zu modellierenden Ausgangsgröße erforderlich. Die Erstellung des Modells erfolgt anhand der Verwendung von Stützstellendaten, die den Trainingsdaten ganz oder teilweise entsprechen oder aus diesen generiert werden. Weiterhin werden abstrakte Hyperparameter bestimmt, die den Raum der Modellfunktionen parametrisieren und effektiv den Einfluss der einzelnen Messpunkte der Trainingsdaten auf die spätere Modellvorhersage gewichten.

[0025] Die abstrakten Hyperparameter werden durch ein Optimierungsverfahren bestimmt. Eine Möglichkeit für ein solches Optimierungsverfahren besteht in einer Optimierung einer Marginal Likelihood p(Y|H,X). Die Marginal Likelihood p(Y|H,X) beschreibt die Plausibilität der gemessenen y-Werte der Trainingsdaten, dargestellt als Vektor Y, gegeben die Modellparameter H und die x-Werte der Trainingsdaten. Im Modelltraining wird p(Y|H,X) maximiert, indem geeignete Hyperparameter gesucht werden, die zu einem Verlauf der durch die Hyperparameter und die Trainingsdaten bestimmten Modellfunktion führen und die Trainingsdaten möglichst genau abbilden. Zur Vereinfachung der Berechnung wird der Logarithmus von p(Y|H,X) maximiert, da der Logarithmus die Stetigkeit der Plausibilitätsfunktion nicht verändert.

[0026] Für die Berechnung des Gauß-Prozess-Modells werden die Eingangswerte $\tilde{u}_d$ für einen Testpunkt u (Eingangsgrößenvektor) zunächst normiert, und zwar gemäß der folgenden Formel:

$$u_d = \frac{\tilde{u_d} - (m_x)_d}{(s_x)_d}.$$

[0027] Dabei entsprechen m_x der Mittelwertfunktion bezüglich eines Mittelwerts der Eingangswerte der Stützstellendaten, s_y der Varianz der Eingangswerte der Stützstellendaten und d dem Index für die Dimension D des Testpunkts u.

[0028] Als Ergebnis der Erstellung des nicht parametrischen, datenbasierten Funktionsmodells erhält man:

$$v = \sum_{i=1}^{N} (Q_y)_i \sigma_f \exp\left(-\frac{1}{2}\sum_{d=1}^{D} \frac{((x_i)_d - u_d)^2}{l_d}\right).$$

[0029] Der so ermittelte Modellwert v wird mithilfe einer Ausgangsnormierung normiert, und zwar gemäß der Formel:

$\tilde{v} = us_y + m_y$.

[0030] Dabei entsprechen v einem normierten Modellwert (Ausgangswert) an einem normierten Testpunkt u (Eingangsgrößenvektor der Dimension D), $\tilde{v}$ einem (nicht normierten) Modellwert (Ausgangswert) an einem (nicht normierten) Testpunkt $\tilde{u}$ (Eingangsgrößenvektor der Dimension D), x_i einer Stützstelle der Stützstellendaten, N der Anzahl der Stützstellen der Stützstellendaten, D der Dimension des Eingangsdaten-/Trainings-

daten-/Stützstellendatenraums, sowie l_d und σ_f den Hyperparametern aus dem Modelltraining. Der Vektor Q_y ist eine aus den Hyperparametern und den Trainingsdaten berechnete Größe. Weiterhin entsprechen m_y der Mittelwertfunktion bezüglich eines Mittelwerts der Ausgangswerte der Stützstellendaten und s_y der Varianz der Ausgangswerte der Stützstellendaten.

[0031] LOLIMOT-Modelle (LOLIMOT: LOcal LInear MOdel Tree) gehören zur Familie der Neuro-Fuzzy-Modelle. Ein LOLIMOT-Modell entspricht einem neuronalen Netz mit einer verdeckten Schicht, das mehrere lineare Modelle mithilfe von normierten Gaußfunktionen überlagert. Die einzelnen lokalen linearen Teilmodelle werden dann mit Gaußschen Glockenfunktionen gewichtet und normiert überlagert. So entstehen weiche Übergänge zwischen den einzelnen linearen Teilmodellen.

[0032] Ein Standardfall bei der LOLIMOT-Modellierung betrifft eine normierte Gaußsche Gewichtungsfunktion und die Verwendung linearer lokaler Modelle. Seien I_{Lin}, $I_{nl} \subseteq \{1, ..., D\}$ Indexmengen der Eingangsdimensionen, die linear bzw. nichtlinear in die Vorhersage eingehen. Die Vorhersageformel für einen Vorhersagewert y an einem Testpunkt u lautet dann

$$y = \sum_{n=1}^{N} \Theta_n(u, c_n, \sigma_n) \left(w_{n0} + \sum_{d \in I_{Lin}} w_{nd} u_d \right)$$

mit

$$\Theta_n(u, c_n, \sigma_n) = \frac{z_n}{\sum_{j=1}^{N} z_j}$$

und

$$z_j = \exp\left(- \sum_{d \in I_{nl}} \frac{(c_{jd} - u_d)^2}{2\sigma_{jd}} \right) = \exp\left(- \sum_{d \in I_{nl}} l_{jd} (c_{jd} - u_d)^2 \right)$$

für $l_{jd} := \frac{1}{2\sigma_{jd}}$.

Wir definieren $Q_n := w_{n0}$ und erhalten

$$y = \frac{1}{\sum_{j=1}^{N} z_j} \sum_{n=1}^{N} z_n \left(Q_n + \sum_{d \in I_{Lin}} w_{nd} u_d \right).$$

[0033] Hierbei sind C_{jd} die Zentren der j = 1, ..., N lokalen Funktionen der Dimension $|I_{nl}|$, σ_{jd} die Varianz der Gaußschen Gewichtungsfunktion von Modell j = 1, ..., N bzgl. der Dimensionen I_{nl} und w_{nd} die Parameter des n-ten linearen Regressionsmodells mit n = 1, ..., N und $d \in I_{Lin}$. N bezeichnet somit die Anzahl der lokalen linearen Modelle. Die D Eingangsgrößen werden unterschieden nach $|I_{nl}| \leq D$ Eingangsgrößen, die nichtlinear eingehen, und $|I_{Lin}| \leq D$ Eingangsgrößen, die linear eingehen. Die maximale Anzahl an benötigten Parametern ergibt sich somit, wenn alle Eingangsgrößen sowohl linear als auch nichtlinear eingehen, d. h.

$|I_{nl}| = |I_{Lin}| = D$.

[0034] Diese Berechnungsformel weist eine starke Ähnlichkeit mit der obigen Formel für die Berechnung der Gaußprozesse auf. Es gibt jedoch vier wesentliche Unterschiede:

1. Manche Eingangsdimensionen werden an manchen Stellen der Berechnung ignoriert. Im Speziellen ist es möglich, dass einige Eingangsdimensionen nur linear, nur nichtlinear oder linear und nichtlinear eingehen.
2. Jede lokale Funktion hat pro Dimension eine eigene Längenskala (Lengthscale) statt einer global gültigen Längenskala pro Dimension.

3. Der Ausgang der Exponentialfunktion wird normiert.
4. Anstatt den Ausgang der Exponentialfunktion mit einem festen Gewicht pro Stützstellenpunkt zu multiplizieren, wird das Gewicht für einen Abfragepunkt aus dem linearen Regressionsmodell jedes lokalen Modells berechnet.

[0035] Um die Modellberechnungseinheit **3** so auszubilden, dass abhängig von der Bedatung sowohl Gauß-Prozess-Modelle als auch LOLIMOT-Modelle berechnet werden können, bestehen grundsätzlich zwei Möglichkeiten. In beiden Fällen werden die Zentren c_{jd} der LOLIMOT-Teilmodelle wie die Stützstellenpunkte eines Gauß-Prozess-Modells behandelt. Die Berechnung wird in allen Fällen durch Übergabe der Parameter an die Modellberechnungseinheit **3** gestartet.

Variante 1: Explizites Speichern der Indexmenge

[0036] Die beiden Indexmengen I_{Lin}, I_{nl} werden direkt als Parameter gespeichert. Dies hat den Vorteil, dass Rechenzeit und Speicherbedarf minimal gehalten werden, wenn viele Zentren nicht gleichzeitig linear und nichtlinear in die Modellvorhersage eingehen.

[0037] Pro lokaler Funktion/Stützstellenpunkt und Dimension $d \in I_{nl}$ werden jeweils eine eigene Längenskala und pro lokaler Funktion/Stützstellenpunkt j je $|I_{Lin}| + 1$ Gewichte w_{jd} gespeichert. Zusätzlich werden die Indexmengen gespeichert.

[0038] Verglichen mit einer Hardwareimplementierung für die Modellberechnungseinheit **3** zur Berechnung von datenbasierten Funktionsmodellen ergeben sich die in fett markierten folgenden Änderungen im Pseudo-C-Code:

```
/* Initialisierung von y zur Verwendeung eines Berechnungsstopps
und einer Wiederaufnahme der Berechnung */
y = p9;
```

```
/* Stufe 1: Eingangsnormierung */
for (k = 0; k < p7; k++) {
  ut[k] = u[k]*p1[k] + p2[k];
}


/* Alternativ kann e_sum aus dem Parameter e_init initialisiert
werden, um eine Unterbrechung und Wiederaufnahme der Berechnung
zu ermöglichen*/
e_sum = 0.0;
/* Stufe 2: y Akkumulator - Berechnung der äußeren Schleife */
for (j = p8; j < p6; j++) {
  i = j * p7;

  /* Stufe 2a1: t Akkumulator- Berechnung der inneren Schleife:
Berechne nichtlinearen Term */
  t = 0.0;

  for (k = 0; k < p7; k++) {
    d = ut[p11[k]] - V[i+k];
    d = d * d;
    if (loliflag)
      m = i+k;
    else
      m = k;
    t += L[m] * d;
  }

  /* Stufe 2a2: w Akkumulator - Berechnung der inneren Schleife:
Berechne linearen Term */
  /* wenn loliflag gesetzt ist, sollte p12 0 sein, so dass der
nachfolgende "if"-Ausdruck weggelassen werden kann*/
  w = p3[j];
  if (loliflag)
    for (k=0; k<p12; k++) {
      w += p10[i+k] * ut[p13[k]];
```

```
  }

  /* Stufe 2b: exp() */
  e = exp(-t);
  if (loliflag)
    e_sum += e;

  /* Stufe 2c: y Akkumulator */
  y += w * e;
}

/* Stufe 3: Ausgangsnormierung*/
if (cfg_setz)
  z = p5;
else
  z = zold;
z += y*p4

/* Für die LOLIMOT Division kann z/e_sum in Software berechnet
werden! */
return {z, e_sum};
```

[0039] Der Pseudo-C-Code beinhaltet teilweise auch die Steuerung und den Zustandsautomaten, die nicht in der Formel auftreten. Folgende Zuordnungen zu den Variablen des obigen Gauß-Prozess-Modells und des LOLIMOT-Modells wurden für die Darstellung des Algorithmus im Pseudo-C-Code verwendet. Die nur für das LOLIMOT-Modell verwendeten Parameter sind durch Fettdruck markiert.

Formel	Pseudocode	Dimension	Beschreibung
$\tilde{u}_d$	u	Dx1	Abfragepunkt
$(s'_x)_d$	p1	Dx1	Inverse Varianzen bzgl. der x-Werte in D Dimensionen
$(m'_x)_d$	p2	Dx1	Negierte punktweise Division der Mittelwerte und Varianzen bzgl. der x-Werte in D Dimensionen
$(Q'_y)_n$	p3	Nx1	Gewichte für die Stützstellenpunkte,

			beinhalten y-Werte der Stützstellenpunkte
l'_d bzw. l_{jd}	L	**1xp7 (loliflag=0); Nxp7 (loliflag=1)**	Inverse Längenskalen
$(x_n)_d$	v	Nxp7	Normalisierte Stützstellenpunkte (x-Werte)
v	z	1x1	Rückgabewert
s_y	p4	1x1	Varianz bzgl. y
m_y	p5	1x1	Mittelwert bzgl. y
N	p6	1x1	Anzahl Stützstellenpunkte
$\lvert I_{nl} \rvert$	**p7**	**1x1**	**Anzahl nichtlinear eingehender Größen/Dimensionen, Mächtigkeit von I_{nl}**
nStart	p8	1x1	Startwert der äußeren Schleife
vInit	p9	1x1	Startwert für Akkumulator
cfg_setz	cfg_setz	1x1	Flag, ob auf vorherige Modellausgabe aufakkumuliert wird oder nicht
W	**p10**	**Nxp12**	**Parameter der N lokalen linearen Regressionsmodelle Wird bei loliflag = 0 nicht benötigt**
I_{nl}	**p11**	**p7x1**	**Indexmenge nichtlinear eingehender Größen, aufsteigend sortiert**
$\lvert I_{Lin} \rvert$	**p12**	**1x1**	**loliflag = 1: Anzahl linear eingehender Größen/Dimensionen, Mächtigkeit von I_{Lin} loliflag = 0: Nullsetzen**
I_{Lin}	**p13**	**p12x1**	**Indexmenge linear eingehender Größen, aufsteigend sortiert**
loliflag	**loliflag**	**1x1**	**lolimot = 1: LOLIMOT-Berechnung lolimot = 0: Gauß-Prozess-Modell**

[0040] Die Bedatung eines Gauß-Prozess-Modells und eines LOLIMOT-Modells erfolgt direkt durch Einsetzen der jeweiligen Parameter. Insbesondere gelten für die Parameter zur Berechnung eines LOLIMOT-Modells:

- loliflag = 1
- Die Zeilen von v enthalten die Zentren der lokalen Funktionen c_j.

[0041] Für die Berechnung eines Gauß-Prozess-Modells müssen folgende Parameterwerte gesetzt werden:

- loliflag = 0
- p12 = 0
- p7 = D
- p11 = 1, ..., D
- L mit der Größe 1 × D

Variante 2: Implizites Speichern der Indexmenge in den Gewichtungsvektoren

[0042] Die Nichtbeachtung einzelner Eingangsdimensionen wird durch geeignetes Nullsetzen der Modellparameter erreicht. Diese Variante erfordert nur geringe Hardwareänderungen, benötigt jedoch etwas mehr Speicher als bei der ersten Variante sowie gegebenenfalls geringfügig mehr Rechenzeit.

```
/* Initialisierung von y zur Verwendung eines Berechnungsstopps
und einer Wiederaufnahme der Berechnung */
y = p9;

/* Stufe 1: Eingangsnormierung */
for (k = 0; k < p7; k++) {
  ut[k] = u[k]*p1[k] + p2[k];
}

/* Alternativ kann e_sum aus dem Parameter e_init initialisiert
werden, um eine Unterbrechung und Wiederaufnahme der Berechnung
zu ermöglichen*/
e_sum = 0.0;
/* Stufe 2: y Akkumulator – Berechnung der äußeren Schleife */
for (j = p8; j < p6; j++) {
  i = j * p7;
```

```
  /* Stufe 2a: t und w Akkumulator- Berechnung der inneren
Schleife */
  t = 0.0;
  w = p3[j];
  for (k = 0; k < p7; k++) {
    d = ut[k] - V[i+k];
    d = d * d;
    if (loliflag)
      m = i+k;
    else
      m = k;
    t += L[m] * d;
    if (loliflag)
      w += p10[m] * ut[k];
  }

  /* Stufe 2b: exp() */
  e = exp(-t);
  if (loliflag)
    e_sum += e;

  /* Stufe 2c: y Akkumulator */
  y += w * e;
}

/* Stufe 3: Ausgangsnormierung*/
if (cfg_setz)
  z = p5;
else
  z = zold;
z += y*p4

/* Für die LOLIMOT Division kann z/e_sum in Software berechnet
werden! */

return {z, e_sum};
```

[0043] Der Pseudo-C-Code beinhaltet teilweise auch die Steuerung und den Zustandsautomaten, die nicht in der Formel auftreten. Folgende Mappings wurden für die Darstellung des Algorithmus im Pseudo-C-Code verwendet:

Formel	Pseudocode	Dimension	Beschreibung
$\tilde{u}_d$	u	Dx1	Abfragepunkt
$(s'_x)_d$	p1	Dx1	Inverse Varianzen bzgl. der x-Werte in D Dimensionen
$(m'_x)_d$	p2	Dx1	Negierte punktweise Division der Mittelwerte und Varianzen bzgl. der x-Werte in D Dimensionen
$(Q'_y)_n$	p3	Nx1	Gewichte für die Stützstellenpunkte, beinhalten y-Werte der Stützstellenpunkte
l'_d **bzw.** l_{jd}	**L**	**1xD (loliflag=0) NxD (loliflag=1)**	**Inverse Längenskalen**
$(x_n)_d$	V	NxD	Normalisierte Stützstellenpunkte (x-Werte)
v	z	1x1	Rückgabewert
s_y	p4	1x1	Varianz bzgl. y
m_y	p5	1x1	Mittelwert bzgl. y
N	p6	1x1	Anzahl Stützstellenpunkte
D	p7	1x1	Anzahl Dimensionen
nStart	p8	1x1	Startwert der äußeren Schleife
vInit	p9	1x1	Startwert für Akkumulator
cfg_setz	cfg_setz	1x1	Flag, ob auf vorherige Modellausgabe aufakkumuliert wird oder nicht
W	**p10**	**NxD**	**Parameter der N lokalen linearen Regressionsmodelle**
loliflag	**loliflag**	**1x1**	**Flag für Aktivierung LOLIMOT-Modus, Default ist 0, also GP und kein LOLIMOT; Alternativ: Verwendung von zwei separaten Flags für Normalisierung und Berechnung der linearen lokalen Funktion**

[0044] Verglichen mit der Gauß-Prozess-Modell-Parametrierung verdreifacht sich hierdurch der Speicherplatzbedarf in etwa, falls man davon ausgeht, dass für das Gauß-Prozess-Modell jeweils die gleiche Anzahl an lokalen LOLIMOT Funktionen und Stützstellenpunkten verwendet wird. Insbesondere benötigt ein Gauß-Prozess-Modell insgesamt N*D + N + D Parameter, während ein LOLIMOT-Modell entsprechend der oben beschriebenen zweiten Variante 3*N*D + N + 2*D Parameter benötigt. In der Praxis ist jedoch zu erwarten, dass die Anzahl der lokalen Funktionen des LOLIMOT-Modells wesentlich kleiner als die Anzahl der Stützstellenpunkte des Gauß-Prozess-Modells ist.

[0045] Unter der Annahme, dass pro Teilmodell mindestens D + 1 Punkte verfügbar sein müssen, um das lineare Modell zu fitten, ergibt sich, dass typischerweise bei N Stützstellenpunkten nicht mehr als N/(D + 1) <

N/D Teilmodelle verwendet werden. Daraus ergibt sich bei einer Dimension von D ≥ 3, dass das LOLIMOT-Modell weniger Speicherplatz benötigt als ein entsprechendes Gauß-Prozess-Modell.

[0046] Die Bedatung eines LOLIMOT-Modells in der zweiten Variante erfolgt durch Setzen der folgenden Parameter:

- loliflag = 1
- p10[:,{1, ..., D}\I_{Lin}] = 0, d. h. Nullsetzen aller Spalten, die zu Dimensionen gehören, die nicht linear eingehen
- Setzen der übrigen Spalten von p10 entsprechend den linearen Gewichten
- L[:,{1, ..., D}\I_{nl}] = 0, d. h. Nullsetzen aller Spalten, die zu Dimensionen gehören, die nicht nichtlinear eingehen
- Setzen der übrigen Spalten von L auf die entsprechenden halbierten inversen Varianzen $1/(2\sigma)$
- Die Zeilen von V enthalten die Zentren der lokalen Funktionen c_j.
- V[:,{1, ..., D}\I_{nl}] = 0, d. h. Nullsetzen der nicht benötigten Spalten von V

[0047] Für die Berechnung eines Gauß-Prozess-Modells müssen folgende Werte gesetzt werden:

- loliflag = 0
- Die Zeilen von V enthalten die Stützstellenpunkte des Gauß-Prozess-Modells
- L mit der Dimension 1 × D

[0048] Beiden Varianten gemein ist, dass mit ihnen auch RBF-Netze auf der AMU berechenbar sind, die für jedes Neuron eine eigene Parametrierung der RBF-Kernel-Funktion aufweisen, d. h. für jedes Neuron sind eigene Längenskalen pro Dimension möglich.

Berechnung der Division:

[0049] Die in der letzten Zeile des Pseudo-C-Codes angegebene Division kann beispielsweise auf die folgende Weise berechnet werden:

– Division in Software. Hierzu muss neben dem nicht normierten Ergebnis der Normierungsfaktor an die Hauptrecheneinheit **2** zurückgegeben werden.
– Division in Hardware durch Nutzung einer in der Modellberechnungseinheit **3** dedizierten Einheit.
– Approximation der Division

[0050] Im Folgenden wird eine Möglichkeit zur Approximation diskutiert:
Das nicht normierte Ergebnis muss durch eine Summe von Exponentialfunktionen dividiert werden, d. h. es muss der Faktor

$$\frac{1}{\sum_{j=1}^{N} e^{(-\sum\ldots)}} = \frac{1}{\sum_{j=1}^{N} z_j}$$

berechnet werden. Aufgrund des eingeschränkten Wertebereichs für jede einzelne exp-Funktion gilt $0 \le z_j \le 1$ und damit, dass der Summand s im Nenner zwischen 0 und N liegt. Er kann, wie jede Floatingpoint-Zahl, dargestellt werden als $s = 2^q p$ mit $0.5 \le p < 1$. Die Division durch ein Vielfaches von 2 ist sehr einfach und effizient durch Bitshifts realisierbar. Probleme bereitet die Division durch eine beliebige Zahl p. Wir schlagen vor, den Term 1/p durch eine geeignet skalierte exp-Funktion g(p) mit

$$g(p) = c \exp(-(ap + b)) + d$$

zu approximieren. Um einen minimalen (durchschnittlichen oder absoluten) Approximationsfehler zu erhalten, müssen die Parameter a bis d optimiert werden. Eine mögliche (aber u. U. nicht optimale) Lösung ist die folgende:

$a = 2\ln 2$, $b = -2\ln 2$, $c = 1$, $d = 0$.

[0051] Sie sorgt dafür, dass bei Eingabe der Grenzen, also p = 0,5 bzw. p = 1, die korrekten Ausgaben g(0,5) = 2 = 1/0,5 bzw. g(1) = 1 = 1/1 erreicht werden. Für p = 0,75 erhält man g(0,75) = sqrt(2) und damit eine Abweichung von etwa 0,08 von der korrekten Lösung.

ZITATE ENTHALTEN IN DER BESCHREIBUNG

Diese Liste der vom Anmelder aufgeführten Dokumente wurde automatisiert erzeugt und ist ausschließlich zur besseren Information des Lesers aufgenommen. Die Liste ist nicht Bestandteil der deutschen Patent- bzw. Gebrauchsmusteranmeldung. Das DPMA übernimmt keinerlei Haftung für etwaige Fehler oder Auslassungen.

Zitierte Patentliteratur

- DE 102010028266 A1 [0003]

Zitierte Nicht-Patentliteratur

- O. Nelles, S. Sinsel, R. Isermann, UKACC International Conference an Control, 1996 [0005]
- C. E. Rasmussen et al., „Gaussian Processes for Machine Learning", MIT Press 2006 [0024]

Patentansprüche

1. Modellberechnungseinheit (**3**) für einen integrierten Steuerbaustein (**1**), der durch rein hardwarebasierte Implementierung mit einer Exponentialfunktion, Summierfunktionen und Multiplikationsfunktionen in mindestens einer inneren und einer äußeren Schleife versehen ist, um ein datenbasiertes Funktionsmodell, insbesondere ein Gauß-Prozess-Modell, zu berechnen, wobei die Modellberechnungseinheit (**3**) ferner dazu ausgebildet ist, ein LOLIMOT-Modell, das mehrere lokale Teilmodelle beinhaltet, zu berechnen.

2. Modellberechnungseinheit (**3**) nach Anspruch 1, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um jeweils zur Berechnung eines datenbasierten Funktionsmodells eine global gültige vorgegebene Längenskala und zur Berechnung eines LOLIMOT-Modells für jede Dimension eine eigene Längenskala zu verwenden.

3. Modellberechnungseinheit (**3**) nach Anspruch 1 oder 2, wobei jeweils das datenbasierte Funktionsmodell mit vorgegebenen Stützstellenpunkten und das LOLIMOT-Modell mit Zentren für lokale Funktionen berechenbar ist, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um die Stützstellenpunkte und die Zentren für eine Berechnung in einer der inneren Schleifen in gleicher Weise zu verwenden.

4. Modellberechnungseinheit (**3**) nach Anspruch 3, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um im Falle einer Berechnung eines LOLIMOT-Modells das Ergebnis der in der inneren Schleife berechneten Exponentialfunktion zu normieren.

5. Modellberechnungseinheit (**3**) nach Anspruch 4, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um im Falle einer Berechnung eines LOLIMOT-Modells das Ergebnis der Exponentialfunktion mit einem für jeden Abfragepunkt jedes lokalen Teilmodells ermittelbaren Gewicht zu multiplizieren.

6. Modellberechnungseinheit (**3**) nach einem der Ansprüche 1 bis 5, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um im Falle einer Berechnung eines datenbasierten Funktionsmodells das Ergebnis der Exponentialfunktion mit einem vorgegebenen Gewicht pro Stützstellenpunkt zu multiplizieren.

7. Modellberechnungseinheit (**3**) nach einem der Ansprüche 1 bis 6, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um im Falle einer Berechnung eines LOLIMOT-Modells Zwischenergebnisse der Anwendung der Exponentialfunktion zu addieren und einen Rückgabewert durch die Summe der Zwischenergebnisse zu dividieren.

8. Modellberechnungseinheit (**3**) nach einem der Ansprüche 1 bis 7, wobei die Modellberechnungseinheit (**3**) ausgebildet ist, um im Falle einer Berechnung eines LOLIMOT-Modells Eingangsdimensionen nur linear, nur nichtlinear oder linear und nichtlinear zu berücksichtigen.

9. Integrierter Steuerbaustein (**1**) mit einer softwaregesteuerten Hauptrecheneinheit (**2**) und der Modellberechnungseinheit (**3**) nach einem der Ansprüche 1 bis 8.

10. Integrierter Steuerbaustein (**1**) mit einer softwaregesteuerten Hauptrecheneinheit (**2**) und der Modellberechnungseinheit (**3**) nach Anspruch 7, wobei die Hauptrecheneinheit (**2**) die Modellberechnungseinheit (**3**) so betreibt, dass die Division durch die Summe der Zwischenergebnisse entweder durch eine Näherungsfunktion in der Modellberechnungseinheit (**3**) oder in der Hauptrecheneinheit (**2**) durchgeführt wird.

Es folgt eine Seite Zeichnungen

Anhängende Zeichnungen

Fig. 1