

# データベース駆動型制御を用いた 巻取機におけるフィルムの張力制御コントローラ開発

## Development of Web Tension Controller for Winding Machine Using Database - Driven Control



浜本 裕貴\*  
Hiroki Hamamoto



平川 友大\*  
Tomohiro Hirakawa



日原 啓太\*  
Keita Hihara



落岩 崇\*  
Takashi Ochiwa



博士(工学) 富山 秀樹\*  
Dr. Hideki Tomiyama



博士(工学) 木下 拓矢\*\*  
Dr. Takuya Kinoshita



博士(工学) 山本 透\*\*  
Dr. Toru Yamamoto

### 要 旨

巻取機は、製造された紙やプラスチックフィルムなどのウェブを搬送して、製品不良のない状態でロール状に巻き取る装置である。複数の駆動ロールで構成されることによるロール同士の相互干渉や巻き太りによるシステム特性の変化、温度などの環境によるウェブの特性の変化によって、システム特性を把握することが困難であった。

本稿では、巻取機のモデルを作成し、張力制御コントローラにデータベース駆動型制御法を適用したシミュレーション検証を実施した。データベース駆動型制御法には、類似度に基づいた制御パラメータの調整法を適用することで計算コストの削減を可能とした。

### — Synopsis —

A winding machine is a device that transports a web such as manufactured paper or plastic film and winds it in a roll shape without product defects. In conventional system, it has been difficult to grasp the system characteristics due to mutual interference between the rolls due to the composition of a plurality of drive rolls, changes in system characteristics due to winding fatness, and changes in web characteristics due to the environment such as temperature.

In this paper, we created a model of a winder and conducted a simulation verification by using database-driven method for the tension controller. For the database-driven control method, it is possible to reduce the calculation cost by applying a method of adjusting control parameters based on similarity.

### 1. 緒 言

モデルベース開発(Model Based Development:MBD)とは、自動車業界などで多く用いられる製品開発手法である。一般に製品開発はV字プロセスと呼ばれるプロセスに従って開発が行われるが、試作後に検証して要求性能を満たしていない場合は、要求性能を満たす製品が

完成するまでV字プロセスを何度も繰り返すこととなる。MBD手法では、コンピュータ上で作成したモデルを活用し、シミュレーション上で検証を行うことで製品企画の段階からシステムの妥当性を検証することができる(図1)。そのため、開発における手戻りを低減させることができるので、複雑なシステムにおいても開発期間の短縮が可能である。また、シミュレーション技術を活

\*: イノベーションマネジメント本部 先端技術研究所  
Advanced Technology Laboratory,  
Innovation Management Headquarters

\*\* : 広島大学大学院先進理工系科学研究科  
Graduate School of Advanced Science and Engineering, Hiroshima University

用することで、複雑なシステムにおいても装置が故障することなく、システムの妥当性を検証することができるため、製品の性能向上が可能となる手法である。当社は、MBD手法を活用した制御システムの開発に取り組んでいる。

一般に、現在の産業界では、制御パラメータの調整が容易であるなどの理由からPID制御が広く用いられている。しかしながら、制御対象となるシステムの多くは非線形性を有しているため、固定の制御パラメータによるPID制御器では常に所望の応答を得られるとは限らない。紙やプラスチックフィルムなどのウェブを搬送してロール状に巻き取る装置(以降、巻取機)においても、しわや巻きズレなどを抑制するためにウェブの張力制御を行っているが、駆動ロール間での相互干渉や、巻き太りによるシステム変動などによる非線形性を有しており、制御パラメータを決定することが困難である。このような背景から、非線形システムの制御対象に対して制御パラメータであるPIDゲインを逐次更新する手法の1つとしてデータベース駆動型(Data Driven:DD)PID制御<sup>(1),(2)</sup>が提案されている。データベース駆動型PID制御では、まず取得可能な入出力データとそのときのPIDゲインを1セットとしてデータベースを作成する。そして、現在のデータとデータベース内のデータの比較を行い、現在のデータと類似度の高いデータを参照してPIDゲインを逐次決定・更新する。本取り組みでは、巻取機を対象として、ウェブのしわや巻きズレなどの製品不良を未然に防止する制御システム開発を行う。その最初のステップとして、それらの製品不良の原因として考えられているウェブの張力に着目し、広島大学大学院先進理工系科学研究科システム制御論研究室の山本教授と共同でウェブの張力制御にデータベース駆動型PID制御を適用した張力制御システムの開発に取り組んでいる。本稿では、高精度な巻取機制御システムの開発について述べる。

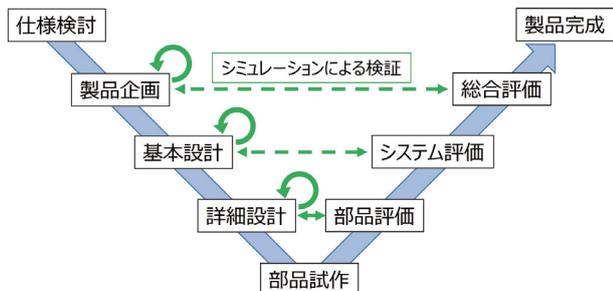


図1 MBD手法によるV字プロセス

## 2. 巻取機概要

### 2.1 巻取時の製品不良

二軸延伸フィルム製造ラインを図2に示す。巻取機は、上流にて製造されたフィルムを搬送し、ロール状に巻き取る装置である。上流で製造されたフィルムの品質を変えることなく、製品不良がないように巻き取る必要がある。

ウェブを巻き取る上での重要な要素の一つに巻固さがある。巻固さは材料そのものの硬さ、巻層間に含む空気量の量、巻上げ後の残留張力による締め付け圧力、フィルムの摩擦係数などによって決まる要素である。図3に巻固さが影響する代表的な製品不良を模式的に示す。スターディフェクトは、ロール内部でのウェブの座屈現象が原因で生じる製品不良であり、巻固さが過剰な状態で発生しやすい。テレスコープは、ロール内のウェブ層間のスリップが原因で生じる製品不良であり、巻固さが小さい状態で発生しやすい。ゲージバンドは、ウェブの幅方向の厚さムラがある場合に生じる製品不良であり、こちらは巻固さが過剰な状態で発生しやすい。これらの製品不良には、トレードオフの関係があるため、製品不良を防ぐためには最適な条件で巻き取る必要がある<sup>(3)</sup>。ここでは、これらの製品不良に大きく関係するウェブの張力に着目し、目標張力に追従するような張力制御コントローラについて考える。

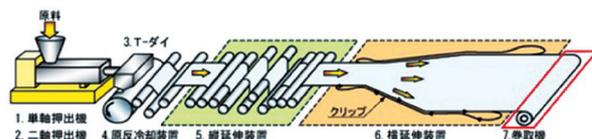


図2 二軸延伸フィルム製造ライン

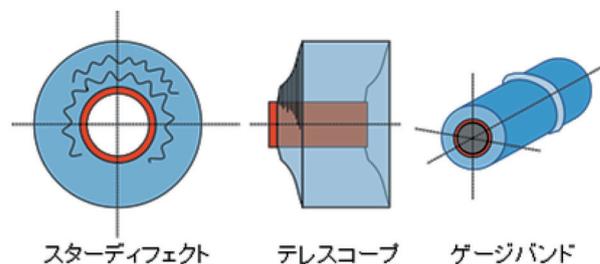


図3 代表的な製品不良

### 2.2 ウェブの張力制御

本開発の取り組みを進めるにあたり、広島中央サイエンスパークに実機検証用の巻取試験機を設置した(図4)。巻取試験機の構成概要図を図5に示す。本装置は、駆動ロールが巻出軸、フィードロール1、フィードロール2、巻取軸の計4本で構成されている。これらの駆動ロール

のトルクもしくは速度を制御することで各駆動ロール間のウェブの張力制御を行っている。具体的には、①巻出軸 - フィードロール1間、②フィードロール1 - フィードロール2間、③フィードロール2 - 巻取軸間でそれぞれ張力制御を行っており、①と③は巻出軸や巻取軸の巻径を利用したトルク制御、②はフィードロール1とフィードロール2の速度差を利用した速度制御による張力制御を行っている。

本開発では、③フィードロール2 - 巻取軸間におけるトルク制御による張力制御に着目する。ここで、トルク制御による張力制御は、ウェブの張力をフィードバックする張力フィードバック制御とウェブの張力をフィードバックしないオープンループ制御がある。張力フィードバック制御を用いる場合、オープンループ制御よりも高い制御性能を得られるが、一方で張力検出器が不可欠であり、オープンループ制御よりもコストが高くなるため、制御性能とコストはトレードオフの関係となっている。そのため、制御方式の特徴と巻取対象となるウェブの特性を考慮して制御方式を決定する必要がある。本稿では、張力フィードバック制御の制御性能向上を目的としてデータベース駆動型PID制御を適用した取り組みについて述べる。

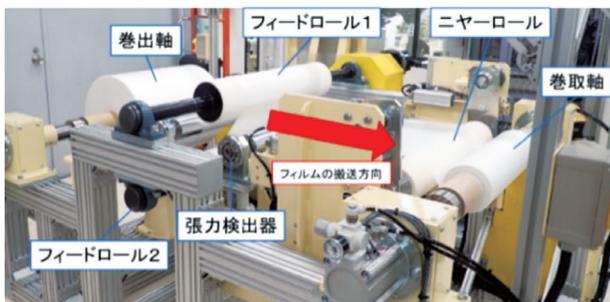


図4 巻取試験機

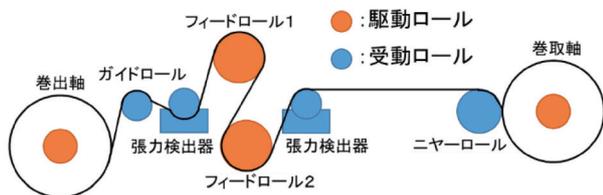


図5 巻取試験機の構成概要図

### 3. データベース駆動型PID制御

#### 3.1 データベース駆動型制御

データベース駆動型制御とは、制御対象が非線形性を有する場合でも、データベースに格納されたデータセットに基づいて適応的に制御パラメータを調整する手法で

ある(図6)。制御パラメータの修正(学習)を行う手法については、作業中に学習を行うオンライン学習と事前に学習を行うオフライン学習がある。

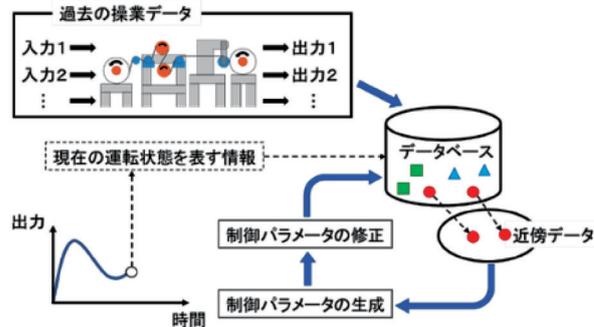


図6 データベース駆動型制御系概要図

#### 3.2 オフライン学習

ここでは、事前に取得した操業データを用いて、作業前にデータベース内の学習(オフライン学習)を行う手法による、データベース駆動型制御の詳細なアルゴリズムを示す(図7)。

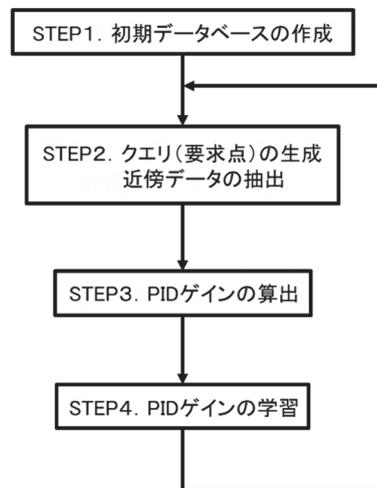


図7 データベース駆動型制御フローチャート

#### 【STEP1. 初期データベースの作成】

データベース駆動型制御では、原理上データベース内にデータセットが存在しない場合、パラメータ調整を行うことができない。そのため、固定のPID制御器などを用いて以下の操業データ  $\bar{\phi}(t_j)$  を取得する。

$$\bar{\phi}(t_j) = [r(t_j), y(t_j), u(t_j)]$$

ここで、 $r(t_j)$  は目標値、 $y(t_j)$  はシステム出力、 $u(t_j)$  は制御入力、 $t$  は時間、 $j$  はデータベースの  $j$  番目のデータセットであることを示している。そして得られた操業データ  $\bar{\phi}(t_j)$  と PID ゲイン  $\theta(t_j)$  から各時刻に

におけるデータセットを生成してデータベースに格納することで初期データベースを作成する。ここで、 $\theta(t_j)$  は以下である。

$$\theta(t_j) = [K_P(t_j), K_I(t_j), K_D(t_j)]$$

$K_P(t_j)$  は比例ゲイン、 $K_I(t_j)$  は積分ゲイン、 $K_D(t_j)$  は微分ゲインを表す。STEP1 の時点では、固定の初期 PID ゲインによって操業されているため、 $\theta(t_1) = \dots = \theta(t_{N(0)})$  である。ただし、 $N(0)$  は初期データベースにおけるデータセットの数を表している。

#### 【STEP2. クエリの生成・近傍データの抽出】

時刻  $t$  における操業データ  $\bar{\phi}_0(t)$  をクエリ(要求点)と呼ぶ。操業データを用いて時刻  $t$  におけるクエリ  $\bar{\phi}_0(t)$  を以下のように生成する。

$$\bar{\phi}_0(t) = [r_0(t+1), r_0(t), y_0(t), \dots, y_0(t-n_y + 1), u_0(t-1), \dots, u_0(t-n_u + 1)]$$

ここで、 $r_0, y_0, u_0$  は、初期の操業データの目標値、システム出力、制御入力を表している。

次にクエリ  $\bar{\phi}_0(t)$  とデータベース内の情報ベクトル  $\bar{\phi}(t_j)$  の距離  $d_j$  を計算する。距離  $d_j$  の算出には式(1)で示す重み付き  $\mathcal{L}_1$  ノルムを使用する。

$$d_j(\bar{\phi}_0(t), \bar{\phi}(t_j)) = \sum_{l=1}^{n_y+n_u+1} \left| \frac{\bar{\phi}_{0,l}(t) - \bar{\phi}_l(t_j)}{\max_m(\bar{\phi}_l(m)) - \min_m(\bar{\phi}_l(m))} \right| \quad \dots (1)$$

$(j = 1, \dots, N(0))$

ここで、 $n_y$  はシステム出力の次数、 $n_u$  は制御入力の次数、 $l$  はクエリの要素番号、 $N(0)$  は初期データベース内に蓄えられているデータセットの数を表している。また、 $\max_m(\bar{\phi}_l(m))$  は、データベースにあるすべての情報ベクトル  $\bar{\phi}(t_j)$  ( $j = 1, 2, \dots, N(t)$ ) の  $l$  番目の要素の中で最も大きな要素、 $\min_m(\bar{\phi}_l(m))$  は同様に、最も小さな要素を示している。式(1)により求められた距離  $d_j$  が小さいものから上位  $n$  個の操業データをクエリの近傍データとして抽出する。

#### 【STEP3. PID ゲインの算出】

STEP2 において選択された近傍データに対して式(2)に示す重み付き線形平均法を用いることで PID ゲイン  $\theta^{\text{old}}(t)$  を算出する。

$$\theta^{\text{old}}(t) = \sum_{i=1}^n \omega_i \theta(t_i) \quad \dots (2)$$

(10)

ここで、 $\theta^{\text{old}}(t)$  は以下である。

$$\theta^{\text{old}}(t) = [K_P^{\text{old}}(t), K_I^{\text{old}}(t), K_D^{\text{old}}(t)]$$

ただし、 $\omega_i$  は選択された第  $i$  番目の情報ベクトルに含まれる  $\theta(t_i)$  に対する重みであり、 $\sum_{i=1}^n \omega_i = 1$  を満足するものとする。

#### 【STEP4. PID ゲインの学習】

STEP3 で得られた PID ゲインを Fictitious Reference Iterative Tuning (FRIT) 法に基づき修正し、近傍データとして抽出されたデータセットの PID ゲインを学習結果に基づき更新する。FRIT 法と最急降下法に基づくオフライン学習則を式(3)に示す。

$$\theta^{\text{new}}(t) = \theta^{\text{old}}(t) - \eta \frac{\partial J(t+1)}{\partial \theta^{\text{old}}(t)} \quad \dots (3)$$

$$J(t) = \left\{ \frac{1}{2} (y_0(t) - y_r(t))^2 \right\}$$

ここで、 $\theta^{\text{new}}(t)$  は以下である。

$$\theta^{\text{new}}(t) = [K_P^{\text{new}}(t), K_I^{\text{new}}(t), K_D^{\text{new}}(t)]$$

$$\eta = [\eta_P, \eta_I, \eta_D]$$

ただし、 $\eta$  は学習係数ベクトル、 $y_r(t)$  は疑似参照入力に対する参照モデル出力を表している。オフライン学習では、クエリが初期の操業データのみで生成されているため、繰り返し学習を行うと、同じクエリを持つデータセットが増え続けてしまう。そこで、学習された PID ゲインの結果に基づき近傍データがもつ PID ゲインに上書きすることにより、初期データベースのサイズは変えずに、PID ゲインの修正を行う。

STEP2 ~ STEP4 を繰り返すことでオフライン学習を行う。

## 4. 数値例

### 4.1 簡易モデルを用いた検証

ここでは、MathWorks 社の MATLAB/Simulink を用いてシミュレーションを実施した。まず初めに、図8に示すような巻出軸 - フィードロール - 巻取軸で構成される装置を制御対象とした簡略モデルの作成を行った<sup>(4),(5)</sup>。このときの簡略モデルブロック線図を図9に示す。

巻出軸は受動ロールのため、コントローラが存在せず、制御対象モデルのみとなっている。ここではスパン2の張力制御コントローラ(図9赤枠部)に対してデータベース駆動型制御を適用することでスパン2における張力  $T_2$  の制御を行った。ただし、スパン2の張力制御コ

ントローラには比例先行型PI制御器を適用した。

まず、初期データベース作成のために張力制御コントローラのPIゲインを固定PIゲインとしてシミュレーションを行った。このときのシミュレーション結果を図10に示す。ただし、巻出軸の半径  $r_1$  と巻取軸の半径  $r_3$  について、巻出軸の半径  $r_1$  は時間経過と共に減少させ、巻取軸の半径  $r_3$  は時間経過と共に増加させた。

図10より立ち上がりはオーバーシュートして振動しているが、発散することなく制御できていることが分かる。図10のシミュレーション結果を初期データベースのデータセットとして、データベース駆動型制御器の適用を行った。

データベース駆動型制御を適用したときのシミュレーション結果を図11、各制御手法の目標値に対して誤差が10%以内となるまでの整定時間を表1に示す。図11より、データベース駆動型制御を適用することでオーバーシュートが低減され、振動が抑制されていることが分かる。また表1より、データベース駆動型制御を適用した場合の方が固定PI制御より整定時間が約63.5%早くなっていることが分かる。これは、運転状況に応じた最適なPIゲインを用いて制御が行えるようになったためであると考えられる。以上より、簡易モデルに対するデータベース駆動型制御の有効性を確認できた。

#### 4.2 巻取試験機モデルを用いた検証

ここでは巻取試験機モデルを用いた検証を行った。まず、実機検証で使用する巻取試験機と同等のモデルを作成した。巻取試験機モデルの構成図を図12、ブロック線図を図13に示す。図12、図13において青色の破線で囲まれた部分が簡易モデルからの変更箇所を表している。巻取試験機モデルは、フィードロールが2本あり、巻出軸、フィードロール1、2、巻出軸の全ての軸が駆動ロールとなっている。簡易モデルは駆動ロールが2軸であったのに対して、巻取試験機モデルは駆動ロールが4軸のため、図13からも分かるようにシステムが複雑になっていることが確認できる。ここではスパン1の張力制御コントローラ(図13赤枠部)に対してデータベース駆動型制御を適用することでスパン1における張力の制御を行った。

巻取試験機モデルにデータベース駆動型制御を適用したときのシミュレーション結果を図14に示す。赤色の実線がデータベース駆動型制御を適用したときの張力制御結果、緑色の破線は初期データベース作成に用いた固定PIゲインを用いたときの張力制御結果を示している。また、表2に各制御手法の平均立ち上がり時間を示す。図14と表2より、固定PI制御の結果よりもデータ

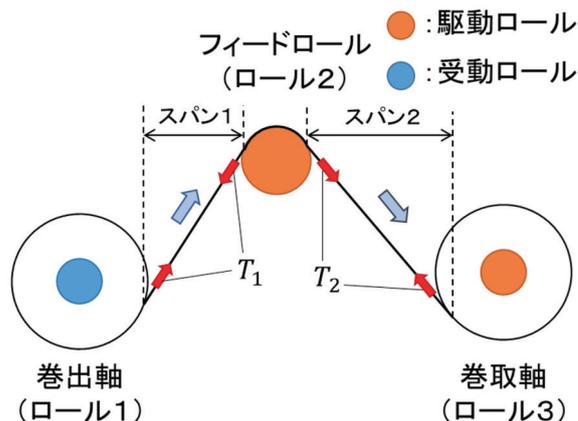


図8 簡略モデル構成図

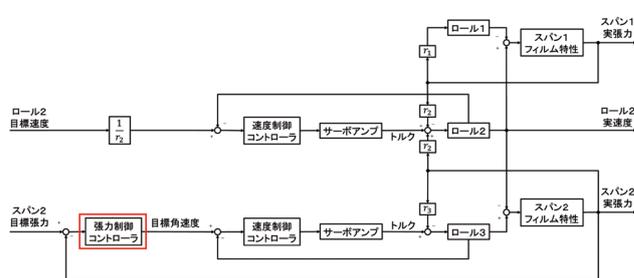


図9 簡略モデルブロック線図

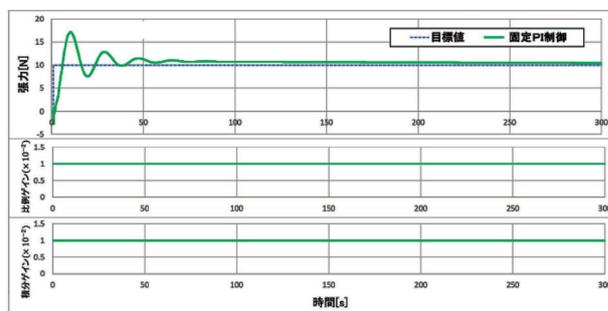


図10 固定PIコントローラでの張力制御結果

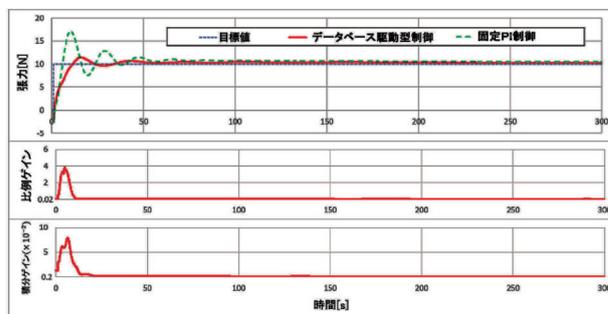


図11 データベース駆動型制御適用結果

表1 各制御手法の整定時間

制御手法	整定時間(s)
固定PI制御	51.68
データベース駆動型制御	18.87

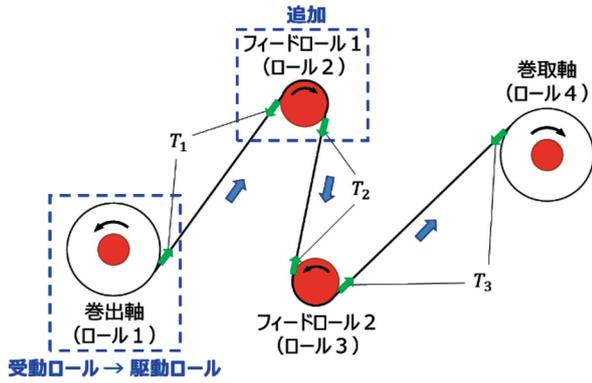


図 12 巻取試験機モデルの構成図

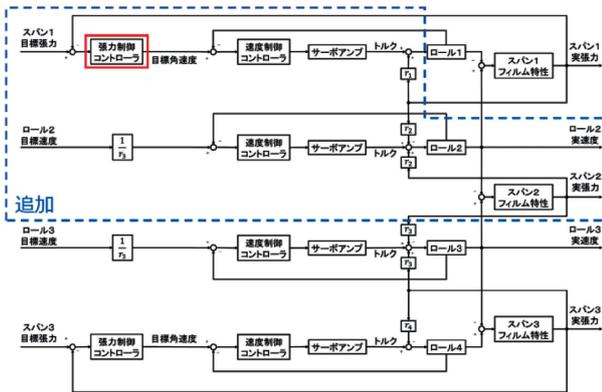


図 13 巻取試験機モデルのブロック線図

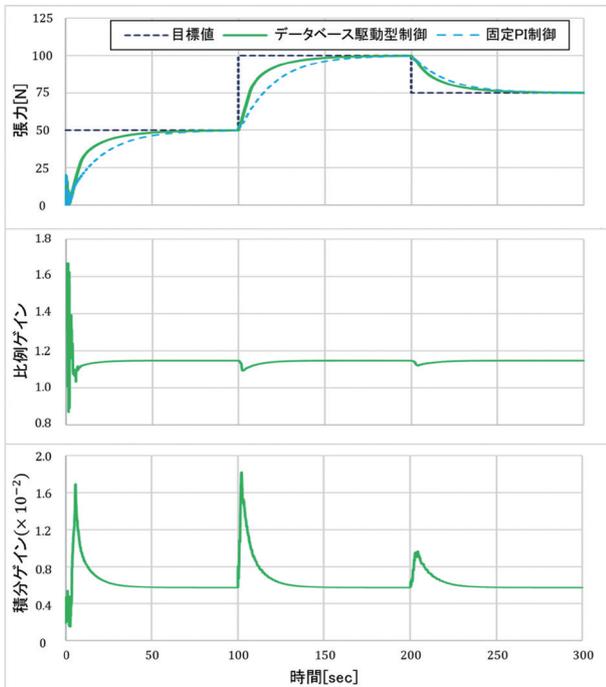


図 14 データベース駆動型制御による張力制御結果

表 2 各制御手法の平均立ち上がり時間

制御手法	平均立ち上がり時間(s)
固定PI制御	41.70
データベース駆動型制御	29.38

ベース駆動型制御の結果の方が、応答が早くなっており、立ち上がり時間が約 29.6 % 改善されていることが分かる。これは、データベース駆動型制御を適用したことにより、システム特性に応じた PI ゲインを用いた制御が行えたためであると考えられる。一方で 200 秒以降では、データベース駆動型制御の制御結果と固定 PI 制御結果にあまり差が見られないが、これは目標値の変化が小さかったため、PI ゲインの調整幅も小さくなったことが要因であると考えられる。こちらは学習回数を増やすことで制御性能の向上が期待できる。以上より、巻取試験機モデルに対してデータベース駆動型制御の有効性を確認できた。

### 4.3 類似度に基づくデータベース駆動型制御手法

4.2 節では、巻取試験機モデルにデータベース駆動型制御を適用した場合の有効性を確認できた。しかしながら、データベース駆動型制御を実際の装置に適用することを考えたとき、計算コストが課題として挙げられる。オフライン学習によるデータベース駆動型制御では、データベース内の全データを参照して PI ゲインの調整を行うため、データベース内のデータ数が多いと計算コストも増加する。そのため、計算コストが大きくなると装置の制御周期内に計算が終わらず、制御ができない可能性がある。こうした問題を解決するため、カーネル密度推定を用いた類似度に基づくデータベース駆動型制御法が提案されている<sup>(6)</sup>。本手法では、データセットに新規データを追加する際とデータベース駆動型制御適用時に近傍データを抽出する際、それぞれにおいてデータベース内の全データとの類似度を算出する。データセットに新規データを追加する際においては、算出した類似度に応じて新規データをデータベース内に格納するかを判断できる。そのため、類似度が高いデータを格納しないことでデータベース内のデータ数を削減できるので計算コストの低減が可能となる。近傍データを抽出する際には、類似度の高いデータのみを近傍データとして抽出できるため制御性能の向上が可能となる。以下に本手法の詳細を示す。

ある新規データ  $\bar{\phi}(t)$  を既存のデータセットに追加する場合を考える。新規データを追加する際は、データベース内のデータセット  $\bar{\phi}_A(k_j)$  から式(4)を用いてカーネル密度関数に基づく類似度  $S$  を算出する。

$$\begin{aligned}
 S(\bar{\phi}(t), \bar{\phi}_A(k_j)) &= \prod_{i=1}^{n_y+n_u+1} \frac{1}{\sqrt{2\pi}h_i} \exp\left(-\frac{(\bar{\phi}(t,i) - \bar{\phi}_A(k_j,i))^2}{2h_i^2}\right) \\
 &\dots \quad (4)
 \end{aligned}$$

ただし、式(4)における  $\bar{\phi}_A(k_j, i)$  とは、データセット  $\bar{\phi}_A$  における  $j$  番目の情報ベクトル、 $i$  番目の要素を指している。また、 $h_i$  はバンド幅を表しており、プラグ・イン法を用いている。式(4)において、類似度が最も高くなるのは、情報ベクトルが全く同じ場合であるため、類似度の最大値は以下となる。

$$S(\bar{\phi}(t), \bar{\phi}_A(k_j)) = \prod_{i=1}^{n_y+n_u+1} \frac{1}{\sqrt{2\pi h_i}} \quad \dots (5)$$

一方で、類似度が低い場合は  $S \rightarrow 0$  となる。ここで、新規データをデータベース内に格納するかを判断するための閾値  $\alpha_{th} (0 < \alpha_{th} \leq 1)$  を導入する。新規データを既存のデータベース内の全データについて式(6)を満たす場合のみ追加する。

$$S(\bar{\phi}(t), \bar{\phi}_A(k_j)) < \alpha_{th} \cdot \prod_{i=1}^{n_y+n_u+1} \frac{1}{\sqrt{2\pi h_i}} \quad \dots (6)$$

ある時刻  $t$  におけるクエリ  $\bar{\phi}(t)$  に対する近傍データを抽出する際においても、上記と同様にして、閾値  $\beta_{th} (0 < \beta_{th} \leq 1)$  によって決まる式(7)を満たすものを近傍データとして抽出する。

$$S(\bar{\phi}(t), \bar{\phi}_A(k_j)) \geq \beta_{th} \cdot \prod_{i=1}^{n_y+n_u+1} \frac{1}{\sqrt{2\pi h_i}} \quad \dots (7)$$

以上を踏まえて、スパン1の張力制御コントローラ(図13赤枠部)に対して類似度に基づくデータベース駆動型制御を適用したシミュレーションを実施した。結果を図15に示す。赤色の実線は類似度に基づくデータベース駆動型制御を適用したときの張力制御結果、緑色の一点鎖線は従来のデータベース駆動型制御を適用したときの張力制御結果、水色の破線は固定PIゲインを用いたときの張力制御結果を示している。また、表3に各制御手法の平均立ち上がり時間を示す。図15より、従来のデータベース駆動型制御の結果と比べて類似度に基づくデータベース駆動型制御の結果の方が、応答が早くなっており、立ち上がり時間が約48.3%改善されていることが分かる。これは、従来のデータベース駆動型制御では近傍データに類似度の低いデータが使用されていたが、類似度に基づくデータベース駆動型制御では類似度の高いデータのみを近傍データとして抽出したため、より最適なPIゲインが算出できたためであると考えられる。また、2つの手法の平均計算時間とデータベース内のデータ数を表4に示す。表4より、類似度に基づくデータベース駆動型制御は従来のデータベース駆動型制御に比べて

平均計算時間を85%程度低減することができたことが分かる。これは、類似度の高いデータをデータベース内に格納しないことにより、データベース内のデータ数が減少したためであると考えられる。したがって、類似度に基づくデータベース駆動型制御の有効性を確認することができた。

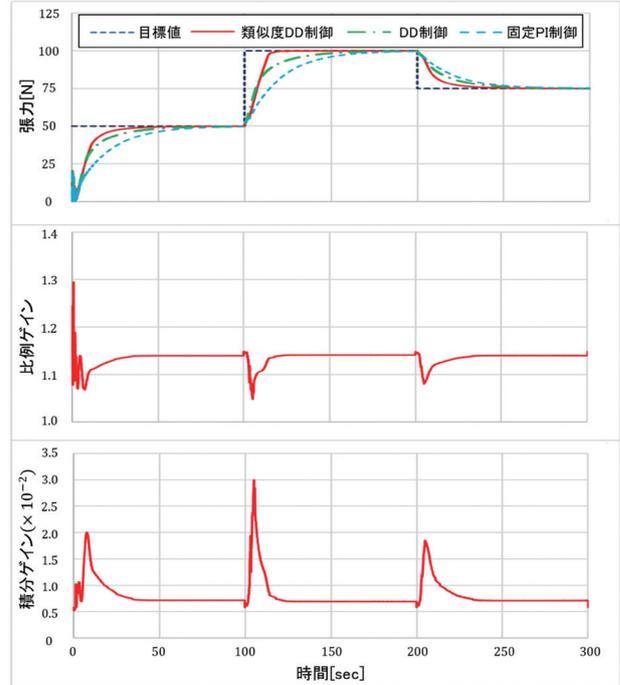


図15 提案法による張力制御結果

表3 各制御手法の平均立ち上がり時間

制御手法	平均立ち上がり時間(s)
固定PI制御	41.70
DD制御	29.38
類似度DD制御	15.20

表4 各手法の平均計算時間とデータ数

制御手法	平均計算時間(ms)	データベース内のデータ数
DD制御	1.53	2997
類似度DD制御	0.21	234

## 5. 結 言

本稿では、駆動軸が2軸の簡易モデルと駆動軸が4軸の巻取試験機モデルを作成し、巻取機の張力制御コントローラに対してデータベース駆動型制御を適用したシミュレーションを行った。その結果、簡易モデル、巻取試験機モデル共に固定PI制御よりも制御性能が向上することが確認できた。その後、オフライン学習によるデータベース駆動型制御の計算コストの課題を解決するために類似度に基づくデータベース駆動型制御を適用してシミュレーションを実施した。その結果、従来のデータベース駆動型制御と比べて計算コストが低減されるだけでなく、制御性能が向上することも確認できた。今後は巻取試験機を用いた検証を行い、有効性を確認する所存である。

## 参 考 文 献

- (1) 山本透編著：データ指向型PID制御，森北出版（2020）
- (2) 木下拓矢：“データベース駆動型ソフトセンサを用いたフィードバック制御系の一設計”，2021 IEE JAPAN
- (3) 橋本巨：入門ウェブハンドリング，加工技術研究会（2010）
- (4) 坂本哲三：“ウェブの張力制御系の解析と制御”，電気学会論文誌D（産業応用部門誌）Vol. 117（1997）No. 3, pp. 274-280
- (5) 福島幸平，坂本哲三：“評価関数を線形化したFRITとOPSOによるウェブ張力制御系のセルフチューニング”，電気学会論文誌D（産業応用部門誌），Vol. 138（2018）No. 2, pp. 128-134
- (6) 木下拓矢，山本透：“類似度に基づくデータベース駆動型制御系の一設計”，電気学会論文誌C（電子・情報・システム部門誌），Vol. 140（2020）No. 3, pp. 312-319