情報処理学会論文誌 Vol.56 No.8 1560–1567 (Aug. 2015)

プロセッサ故障にともなう過負荷状態の回避を考慮した最適出力フィードバック制御器の設計

吉本 達也[1,a)] 潮 俊光[1,b)] 安積 卓也[1,c)]

受付日 2014年11月19日, 採録日 2015年2月4日

概要：組込みシステムにおいて，プロセッサの故障にともなう過負荷状態を回避し，制御性能の劣化を最小化することはシステムの信頼性，安全性を保つうえで重要である．プロセッサが過負荷状態に陥ると，デッドラインを満たせないジョブが発生し，それらのジョブの実行に時間を要したために他のジョブのデッドラインミスにも影響する．そこで，過負荷状態において受理するジョブと破棄するジョブを選択し，受理されたジョブのデッドラインが必ず満たされる方法としてジョブスキッピングが考える．しかし，従来研究ではジョブスキッピングパターンを決定するものの，制御性能の劣化を抑える制御器の設計に関してはあまり検討されていない．本論文では．ジョブスキッピングに基づくスケジューリングを用いたリアルタイム制御システム上の最適出力フィードバック制御器の設計を行う．シミュレーションでは，ジョブスキッピングが発生したときに従来の最適出力フィードバック制御器に比べて提案する制御器によって制御性能の劣化が抑えられることを示す．

キーワード：過負荷状態回避，ジョブスキッピング，最適制御，オブザーバ．

Design of Optimal Output Feedback Controller under Overload Avoidance Caused by Processor Failure

Tatsuya Yoshimoto[1,a)] Toshimitsu Ushio[1,b)] Takuya Azumi[1,c)]

Received: November 19, 2014, Accepted: February 4, 2015

Abstract: In embedded control systems, it is important to avoid an overload situation and minimize a degradation of control performance caused by a failure of a processor for keeping reliability and safety. If the processor is under the overload situation, several jobs cannot meet their deadlines and their fruitless execution compress the other jobs' deadlines. Then, we consider a job skipping approach which is a policy to classify released jobs into accepted and skipped jobs to meet deadlines of all accepted jobs. However, conventional approaches had not addressed a design that suppresses an effect of job skipping. In this paper, we design an optimal output-feedback controller on a real-time system with a scheduler using the job skipping policy. By a numerical simulation, we show that the proposed optimal controller suppresses the degradation of control performance when jobs are skipped.

Keywords: Overload avoidance, job skipping, optimal control, observer.

1. はじめに

組込みシステムは物理的制限や製造コストの面から，利用可能な計算リソースが制限される場合が多い．限られた計算リソースを有効利用し，最大限の性能を実現することは組込みシステムを設計，開発するうえで重要な課題である [1]．組込みシステムにおいて複数の制御タスクが実装され，制御対象（プラント）を制御する．たとえば，組込みシステムとして代表的な自動車はエンジン制御，ブレーキ制御，クルーズ制御など機能ごとのサブシステムに分割され，車体に搭載された複数のプロセッサ上で分散制御される．制御タスクは処理単位であるジョブを連続的にリリー

[1] 大阪大学
Osaka University, Toyonaka, Osaka 560–8531, Japan
a) yoshimoto@hopf.sys.es.osaka-u.ac.jp
b) ushio@sys.es.osaka-u.ac.jp
c) takuya@sys.es.osaka-u.ac.jp

スし，プロセッサがそれぞれのデッドラインまでに実行完了することで所望の制御性能を達成する．すべてのジョブがデッドラインを満たすためにジョブの実行順序を決定することをスケジューリングといい，古典的なスケジューリングアルゴリズムには Rate Monotonic（RM）や Earliest Deadline First（EDF）がある [2]．

組込みシステムの設計者は与えられた計算リソース仕様に基づき，所望の制御性能を達成するように制御タスクやスケジューラを設計する．しかし，過負荷状態が発生した場合，すべての制御タスクのスケジュール可能性を達成することが不可能となり，本来の制御性能を達成できない [3]．例として，2 つのプロセッサ上で複数の制御タスクが分散的に実行されているとき，片方のプロセッサが故障したことですべての制御タスクを残りの 1 つのプロセッサで実行しなければならない場合があげられる．

過負荷状態を回避する手法は，これまで多く提案されてきた [4], [5], [6]．多くはリアルタイム性とリソース制約条件を満たしつつ，制御性能を最大化するサンプリング周期を決定する最適化問題へと帰着されており，Cervin らの研究が代表的である [7]．これらの手法は問題設定がより簡単であるものの，オンラインで逐次サンプリング周期を変更するため同時に A/D・D/A 変換器やアクチュエータといった機器の動作周期も変更しなければならないという欠点がある．

その他の過負荷状態を回避する手法の 1 つとして，ジョブスキッピングがあげられる [8]．ジョブスキッピングは，リリースされるジョブを受理（アクセプト）するジョブと破棄（スキップ）するジョブに選別し，受理されたジョブのデッドラインが満たされるようにスケジューリングする手法である．リリースされるジョブの受理・破棄を表したリストをジョブスキッピングパターンと呼ぶ．この手法では，サンプリング周期を変更せずに過負荷状態を回避できる．しかし，制御タスクの場合，ジョブを破棄することで対応する時刻での制御入力の更新が行われず，制御性能の劣化を招く．この手法に基づく場合，リアルタイム制御システムの設計者は制御対象の安定性が損なわれるほどの連続的なジョブスキッピングを避け，制御性能の劣化を最小化するようにジョブスキッピングパターンを決定しなければならない．そのようなジョブスキッピングパターンを決定するうえで有効な指標として，(m,k)-firm 保証が提案されている [9], [10], [11]．(m,k)-firm 保証とは連続してリリースされた k 個のジョブのうち，m 個のジョブの受理およびデッドラインまでの実行完了が保証されていることである．Ramanathan はプロセッサ故障にともなう過負荷状態を回避するための手法として，(m,k)-firm 保証に基づくジョブの実行優先度決定アルゴリズムを提案し，スケジューラの設計を行った [9]．

過負荷状態を回避し，受理されたすべてのジョブのデッドラインを満たすようなジョブスキッピングパターンを決定することは重要であるが，破棄されたジョブの分だけ制御入力の更新が減り，制御性能が劣化する．つまり，実行可能なジョブ数が減る代わりに制御性能を維持するような新たな制御器を提案することが必要である．しかし，これらを同時に達成する統合的な設計は非常に複雑となる．

本論文では，出力フィードバック制御系を対象に，(m,k)-firm 保証に基づく過負荷状態の回避から得られたジョブスキッピングパターンが制御タスクに転送され，そのジョブスキッピングパターンのもとで制御性能を最大化する最適出力フィードバック制御法を提案する．本手法を利用することで，ジョブスキッピングアルゴリズムと制御タスクとを並行的に設計でき，組込みシステム開発期間の短縮できるという利点がある．

本論文は以下のように構成される．2 章では本論文で考える過負荷状態を回避するためのリアルタイム制御システムの構成を述べる．3 章では制御対象となるプラントおよび制御器をサンプル値制御システムとして定式化し，提案する最適出力フィードバック制御器を設計するうえで必要な制御性能の評価関数を定義する．4 章では最適出力フィードバック制御器および状態推定オブザーバの設計を行う．5 章では数値シミュレーションにより，提案した最適出力フィードバック制御器の有効性を検証する．最後に，6 章で本論文の結論を述べる．

2. 過負荷状態を回避するリアルタイム制御システム

本章では，Ramanathan が提案した (m,k)-firm 保証に基づくジョブの実行優先度決定アルゴリズム [9] を基にした，過負荷状態を回避するスケジューラを用いたリアルタイム制御システムについて述べる．まず，本論文で考えるリアルタイム制御システムの構成図および処理の流れを**図 1** に示す．複数の制御タスクが存在し，それぞれあらかじめ設定された固定周期でジョブをリリースする．リリースされたジョブはスケジューラによって管理され，逐次実行される．スケジューラは主に以下のような 2 つの機能を果たす．

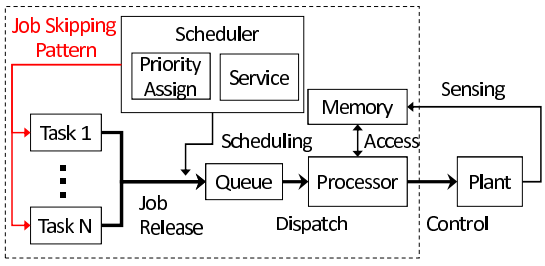


図 1 リアルタイム制御システムの構成と処理の流れ

Fig. 1 An architecture and a process of the real-time control system.

(1) Priority Assign
リリースされたジョブを mandatory ジョブと optional ジョブの 2 つに分類する．分類アルゴリズムは文献 [9] の図 4 を参照されたい．mandatory ジョブは optional ジョブよりも実行優先度が高い．なお，mandatory ジョブどうしおよび optional ジョブどうしの実行優先度はリリース周期が短いタスクほど実行優先度が高くなる RM スケジューリングアルゴリズムに基づいて決定する．実行優先度順に並べられたジョブは待ち行列（Queue）に格納される．実行優先度に従って実行可能なジョブと実行不可能なジョブが決定され，ジョブスキッピングパターンが得られる．そして，その情報は制御タスクに転送される．

(2) Service
プリエンプション可能な待ち行列内のジョブを実行優先度順に逐次実行する．

スケジューラによって決定された実行優先度に従い，受理されるジョブと破棄されるジョブが決定し，受理されたジョブはそれぞれのデッドラインまでに実行される．なお，すべての mandatory ジョブが実行完了することで，(m,k)-firm 保証が満たされることが証明されている [9]．プロセッサ上でジョブが実行されることにより，制御入力の更新が行われ，プラントは安定化制御される．なお，プラントの出力は制御タスクのリリース周期と同様，固定されたサンプリング周期でサンプリングされ，逐次メモリに格納される．受理されたジョブはプラントの出力履歴を利用し，次の制御入力をデッドラインまでに計算する．

3. 制御対象の定式化

3.1 計算遅延の存在するサンプル値制御システム

図 1 に示されるリアルタイム制御システム上に実装される周期的サンプル値制御システムを対象に，制御入力の計算時間にともなう遅延を考慮した最適制御問題を考える．制御器は周期的にサンプリングされるプラントの出力を用いて制御入力を更新する．更新のための計算時間を考慮した場合，サンプリング時刻から制御入力のアクチュエーション時刻まで遅延が存在する．本論文では，問題を簡単化するために，制御入力のアクチュエーション時刻は次のサンプリング時刻と等しいものとする．つまり，制御入力の更新遅延はサンプリング周期と等しい．制御入力がゼロ次ホールド回路により更新時刻間で一定値に維持される場合，プラントは次のような離散時間状態方程式で表される．

$$x(k+1) = Ax(k) + Bu(k-1), \tag{1}$$

$$y(k) = Cx(k). \tag{2}$$

ただし，$k \in \mathbb{N}$ は離散時刻，$x(k) \in \mathbb{R}^{n_x}$ は時刻 k での n_x 次元状態ベクトル，$u(k) \in \mathbb{R}^{n_u}$ は n_u 次元制御入力ベクトル，$y(k) \in \mathbb{R}^{n_y}$ は n_y 次元出力ベクトルを表す．さらに，各係数行列 $A \in \mathbb{R}^{n_x \times n_x}$，$B \in \mathbb{R}^{n_x \times n_u}$，$C \in \mathbb{R}^{n_y \times n_x}$ は定数行列である．制御対象は可制御，可観測とする．式 (1) のように，時刻 $k-1$ でサンプリングされたプラントの出力を基に計算した制御入力 $u(k-1)$ は時刻 k で印加される．

制御性能の評価指標として，次のような有限時間区間 L の線形二次評価関数を考える．

$$J = \sum_{k=0}^{L-1} \left\{ x^T(k+1)Qx(k+1) + u^T(k)Ru(k) \right\}. \tag{3}$$

ただし，行列 $Q \in \mathbb{R}^{n_x \times n_x}$，$R \in \mathbb{R}^{n_u \times n_u}$ は正定行列である．先述のとおり，制御入力は 1 ステップ分の遅延があり，制御入力 $u(k)$ は時刻 k でサンプリングされたプラントの状態に基づいて計算された制御入力であり，時刻 $(k+1)$ で印加される．よって，評価関数 (3) における制御入力の項は 1 ステップ遅れた値が適用される．

3.2 ジョブスキッピングを考慮したサンプル値制御システムのモデル

本節では，ジョブスキッピングを考慮したサンプル値制御システムの定式化を行い，非周期サンプル値制御問題へと帰着させる．ジョブが破棄されたとき，対応する制御入力の更新処理は行われないため，次のジョブが受理されるまで同じ制御入力が印加される．つまり，非周期的にジョブスキッピングが発生する場合，制御入力の更新間隔も非周期的となる．これを考慮して，各時刻における制御入力の値を適切に決定することで制御性能の劣化を最小限に抑えることを考える．なお，ジョブスキッピング決定機構で決定されたジョブスキッピングパターンは各制御器にも伝えられており，それを利用して最適な制御入力を決定する．

まず，時間区間 L の間に $\hat{L}$ 個のジョブが受理されるとする．$\hat{k}$ 番目（$\hat{k} = 0, 1, \ldots, \hat{L}-1$）のジョブによって制御入力が更新された時刻を $k(\hat{k})$ とおく．さらに，$\hat{k}$ 番目と $\hat{k}+1$ 番目に受理されたジョブの時間ステップ間隔を $f_{\hat{k}}$ とおくと，$\hat{k}$ は次式のように表される．

$$k(\hat{k}) = \begin{cases} \sum_{j=0}^{\hat{k}-1} f_j + 1 & \text{if } \hat{k} \geq 1, \\ 1 & \text{if } \hat{k} = 0. \end{cases}$$

ただし，初期時刻 $k=0$ と終端時刻 $k=L$ にリリースされるジョブは受理されるものとする．なお，**図 2** に時刻係数 k と $\hat{k}$ の対応関係を示す．受理されたジョブのリリース時刻 $k-1$ と $k-f_{\hat{k}-1}-1$ との間でリリースされた $f_{\hat{k}-1}-1$ 個のジョブは破棄されている．リリース時刻（サンプリング時刻）とアクチュエーション時刻はサンプリング周期と等しい遅延が存在し，この時刻間で新たな制御入力を計算する．そこで，アクチュエーション時刻におけるプラントの状態，出力は次式のような離散時間制御システム (1), (2) で記述できる．

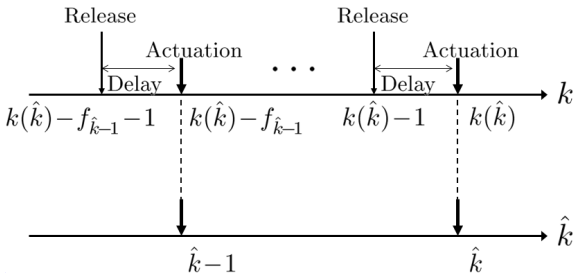


図 2 時刻係数 k と $\hat{k}$ が示す時刻の対応関係

Fig. 2 The relationship between the discrete-time instant indices k and $\hat{k}$.

$$\hat{x}(\hat{k}+1) = A(\hat{k})\hat{x}(\hat{k}) + B(\hat{k})\hat{u}(\hat{k}), \tag{4}$$

$$\hat{y}(\hat{k}) = C\hat{x}(\hat{k}). \tag{5}$$

ただし，各係数行列および変数は $A(\hat{k}) = A^{f_{\hat{k}}}$，$B(\hat{k}) = \sum_{j=0}^{f_{\hat{k}}-1} A^j B$，$\hat{x}(\hat{k}) = x(k(\hat{k}))$，$\hat{y}(\hat{k}) = y(k(\hat{k}))$，$\hat{u}(\hat{k}) = u(k(\hat{k}))$ である．同様にして，評価関数 (3) も変数 $\hat{k}$ を用いて以下のように記述できる．

$$J = \sum_{\hat{k}=0}^{\hat{L}-1}\Bigg\{\sum_{j=1}^{f_{\hat{k}}-1}\left(x_{\hat{k},j}^T Q x_{\hat{k},j} + u_{\hat{k},j}^T R u_{\hat{k},j}\right) + x_{\hat{k},0}^T Q x_{\hat{k},0} + u_{\hat{k},0}^T R u_{\hat{k},0}\Bigg\}.$$

ただし，変数 $x_{\hat{k},j}$，$u_{\hat{k},j}$ は以下のようになる．

$$\begin{aligned} x_{\hat{k},j} &= x(k(\hat{k})+j) \\ &= A^j x(k(\hat{k})) + \sum_{l=0}^{j-1} A^l B u(k(\hat{k})) \\ &= A^j \hat{x}(\hat{k}) + \sum_{l=0}^{j-1} A^l B \hat{u}(\hat{k}), \\ u_{\hat{k},j} &= u(k(\hat{k})+j-1). \end{aligned}$$

ジョブが破棄された場合，制御入力は更新されないことから次式も成り立つ．

$$u_{\hat{k},j} = u(k(\hat{k})-1) = \hat{u}(\hat{k}) \quad \text{for } j = 1, \ldots, f_{\hat{k}} - 1.$$

よって，制御システム (4)，(5) に対する制御性能評価関数は次式のように表される．

$$J = \sum_{\hat{k}=0}^{\hat{L}-1}\left\{\Theta(\hat{k}) + \hat{x}^T(\hat{k})Q\hat{x}(\hat{k}) + f_{\hat{k}}\hat{u}^T(\hat{k})R\hat{u}(\hat{k})\right\}. \tag{6}$$

ただし，

$$\Theta(\hat{k}) = \begin{cases} \sum_{j=1}^{f_{\hat{k}}-1} \Psi_{j,\hat{k}}^T Q \Psi_{j,\hat{k}} & \text{if } f_{\hat{k}} > 1, \\ 0 & \text{if } f_{\hat{k}} = 1, \end{cases} \tag{7}$$

$$\Psi_{j,\hat{k}} = \Phi(j)\hat{x}(\hat{k}) + \Gamma(j)\hat{u}(\hat{k}), \tag{8}$$

$$\Phi(j) = A^j, \tag{9}$$

$$\Gamma(j) = \sum_{l=0}^{j-1} A^l B. \tag{10}$$

以上より，ジョブスキッピングと計算遅延が存在する周期サンプル値制御系の最適制御問題は，時変離散時間制御システム (4)，(5) を対象とし，評価関数 (6) を最小化するような制御入力 $u(\hat{k})$ を決定する最適化問題として定式化できる．

4. 最適出力フィードバック制御器の設計

4.1 評価関数を最小化する制御入力

動的計画法と最適性の原理 [12] を適用することにより，評価関数 (6) を最小化する制御入力は次式のような時変状態フィードバック形式で求められる．

$$\hat{u}^0(\hat{L}-m) = -K(\hat{L}-m)\hat{x}(\hat{L}-m). \tag{11}$$

ただし，変数 m は $1 \le \forall m \le \hat{L}$，$m \in \mathbb{Z}_+$ を満たす．なお，最適状態フィードバックゲイン K は次式のように求められる．

$$K(\hat{L}-m) = \begin{cases} K_1 & \text{if } f_{L-m} > 1, \\ K_2 & \text{if } f_{L-m} = 1, \end{cases} \tag{12}$$

$$\begin{aligned} K_1 = &\Bigg\{\Gamma^T(f_{\hat{L}-m})P(\hat{L}-m+1)\Gamma(f_{\hat{L}-m}) \\ &+ \sum_{j=1}^{f_{\hat{L}-m}-1} \Gamma^T(j)Q\Gamma(j) + f_{\hat{L}-m}R\Bigg\}^{-1} \cdot \\ &\Bigg\{\Gamma^T(f_{\hat{L}-m})P(\hat{L}-m+1)\Phi(f_{\hat{L}-m}) \\ &+ \sum_{j=1}^{f_{\hat{L}-m}-1} \Gamma^T(j)Q\Phi(j)\Bigg\}, \\ K_2 = &\left\{B^T P(\hat{L}-m+1)B + R\right\}^{-1} \cdot B^T P(\hat{L}-m+1)A. \end{aligned}$$

なお，行列 P は以下のような差分方程式で与えられる．

$$P(\hat{L}-m) = \begin{cases} P_1 & \text{if } f_{L-m} > 1, \\ P_2 & \text{if } f_{L-m} = 1, \end{cases} \tag{13}$$

$$\begin{aligned} P_1 = &\bar{\Gamma}_{f_{\hat{L}-m},m}^T P(\hat{L}-m+1)\bar{\Gamma}_{f_{\hat{L}-m},m} + \sum_{j=1}^{f_{\hat{L}-m}-1} \bar{\Gamma}_{j,m}^T Q \bar{\Gamma}_{j,m} \\ &+ Q + f_{\hat{L}-m}K^T(\hat{L}-m)RK(\hat{L}-m), \\ \bar{\Gamma}_{j,m} = &\Phi(j) - \Gamma(j)K(\hat{L}-m), \\ P_2 = &\bar{A}^T P(\hat{L}-m+1)\bar{A} + Q + K^T(\hat{L}-m)RK(\hat{L}-m), \\ \bar{A} = &A - BK(\hat{L}-m). \end{aligned}$$

ただし，$P(\hat{L}) = 0$ である．以上のように，時変状態フィードバックゲイン K は，終端時刻 $\hat{L}$ から初期時刻 0 へと時間逆方向に逐次計算することができる．加えて，有限時間区間内の状態フィードバックゲイン K は受理されたジョブの時間ステップ間隔 $f_{\hat{k}}$ に依存する．つまり，ジョブスキッピングパターンに応じてフィードバックゲインが決定され，制御入力は時変の状態フィードバック形式で求めら

れるため，同じジョブスキッピングパターンが繰り返されれば制御則を変更する必要がない．ここで，制御入力に関して以下が成り立つ．

$$u(k(\hat{k})-1)=\hat{u}(\hat{k})=-K(\hat{k})\hat{x}(\hat{k}). \tag{14}$$

つまり，制御器は1ステップの計算遅延を考慮したとき，次のアクチュエーション時刻における制御系の状態 $\hat{x}(\hat{k})$ を推定しなければならない．よって，次節では計算遅延とジョブスキッピングを考慮したときの状態推定オブザーバの設計を行う．

4.2 修正オブザーバ

入出力値からプラントの状態を推定するために，以下の最小次元オブザーバを考える [13]．

$$z(k+1)=Fz(k)+Gy(k)+Hu(k-1), \tag{15}$$
$$w(k)=Wz(k)+Vy(k). \tag{16}$$

ただし，ベクトル $z\in\mathbb{R}^{n_x-n_y}$，$w\in\mathbb{R}^{n_x}$ はそれぞれオブザーバの状態，出力である．各係数行列は制御システムの係数行列 A，B，C やオブザーバゲインから決定される [13]．

ジョブが破棄されたとき，制御入力の更新が行われないので同様にオブザーバによる状態推定も行われない．もし時刻 $k-2$ においてリリースされるジョブが破棄されたとすると，オブザーバの推定値 $z(k-1)$ の更新も行われず，前の値 $z(k-2)$ がそのまま引き継がれる．この場合，状態推定則 (15)，(16) によって計算される次の推定値は $z'(k)=Fz(k-2)+Gy(k-1)+Hu(k-2)$ である．一方で，時刻 $k-2$ においてリリースされるジョブが受理された場合，計算される次の推定値は $z(k)=Fz(k-1)+Gy(k-1)+Hu(k-2)$ である．つまり，オブザーバ (15)，(16) はジョブが破棄されたことで，推定誤差 $z'(k)-z(k)=F(z(k-2)-z(k-1))$ を発生させてしまう．本節では，ジョブが破棄された時刻でもセンサはプラントの出力を観測し，リアルタイム制御システム内のメモリに格納していると仮定したうえで，推定誤差を解消する修正オブザーバの設計を行う．修正オブザーバは格納された過去の出力データを用いることで，正確な状態推定を行うことができる．

修正オブザーバを設計するにあたり，時刻 $k(\hat{k})-f_{\hat{k}-1}-1$ と $k(\hat{k})-1$ においてリリースされるジョブは受理され，その間 $k(\hat{k})-f_{\hat{k}-1}$ から $k(\hat{k})-2$ においてリリースされるジョブはすべて破棄される状況を考える．ここで，ジョブが破棄されたとき制御入力の更新は行われないので，以下の式が成り立つ．

$$\begin{aligned}u(k(\hat{k})-2) &= u(k(\hat{k})-3)\\ &\vdots\\ &= u(k(\hat{k})-f_{\hat{k}-1}-1)\\ &= -K(k(\hat{k})-f_{\hat{k}-1})w(k(\hat{k})-f_{\hat{k}-1}).\end{aligned} \tag{17}$$

さらに，時刻 $k(\hat{k})-f_{\hat{k}-1}$ と $k(\hat{k})-1$ の間においてオブザーバ (15)，(16) は次のような更新が行われる．

$$\begin{aligned}&z(k(\hat{k}))=Fz(k(\hat{k})-1)+Gy(k(\hat{k})-1)+Hu(k(\hat{k})-2),\\ &z(k(\hat{k})-1)=Fz(k(\hat{k})-2)+Gy(k(\hat{k})-2)+Hu(k(\hat{k})-3),\\ &\qquad\vdots\\ &z(k(\hat{k})-f_{\hat{k}-1}+1)=Fz(k(\hat{k})-f_{\hat{k}-1})\\ &\qquad\qquad +Gy(k(\hat{k})-f_{\hat{k}-1})+Hu(k(\hat{k})-f_{\hat{k}-1}-1).\end{aligned}$$

ここで，上記の式に式 (17) を代入することにより，次式のような修正オブザーバが得られる．

$$\hat{z}(\hat{k})=\hat{F}(f_{\hat{k}-1})\hat{z}(\hat{k}-1)+\hat{G}(f_{\hat{k}-1}), \tag{18}$$
$$\hat{w}(\hat{k})=W\hat{z}(\hat{k})+V\hat{y}(\hat{k}). \tag{19}$$

ただし，各変数は以下のように定義される．$\hat{z}(\hat{k})=z(k(\hat{k}))$，$\hat{w}(\hat{k})=w(k(\hat{k}))$，$\hat{z}(\hat{k}-m)=z(k(\hat{k})-f_{\hat{k}-m})$，$\hat{w}(\hat{k}-m)=w(k(\hat{k})-f_{\hat{k}-m})$ $(\forall m\geq 1)$．係数行列 $\hat{F}$，$\hat{G}$ は以下のように定義される．

$$\hat{F}(f_{\hat{k}-1})=F^{f_{\hat{k}-1}}-\sum_{j=0}^{f_{\hat{k}-1}-1}F^jHK(\hat{k}-1)W,$$

$$\begin{aligned}\hat{G}(f_{\hat{k}-1})=\sum_{j=0}^{f_{\hat{k}-1}-1}F^j\Big(&Gy(k(\hat{k})-j-1)\\ &-HK(\hat{k}-1)V\hat{y}(\hat{k}-1)\Big).\end{aligned}$$

以上のように，修正オブザーバはジョブスキッピングパターンとメモリに格納された過去のプラントの出力値を用いることにより，状態を推定する．そして，最適状態フィードバック制御則 (11) と修正オブザーバ (18)，(19) を結合させることにより，次式のようなジョブスキッピングと計算遅延を考慮した最適出力フィードバック制御器が得られる．

$$\hat{u}^0(\hat{k})=-K(\hat{k})\hat{w}(\hat{k}). \tag{20}$$

次章では修正オブザーバおよび最適出力フィードバック制御器の有効性を示すため，クアッドロータヘリの姿勢制御問題を例に数値シミュレーションを行う．

5. シミュレーション

本章ではクアッドロータヘリの姿勢制御を例題に，与えられたジョブスキッピングパターンのもとで，提案した最適出力フィードバック制御器によって制御性能の劣化が抑えられることを数値シミュレーションにより示す．

5.1 シミュレーション設定

本節では，制御対象となるクアッドロータヘリ，リアルタイムシステムおよびシミュレーションの数値設定を行う．まず，制御対象となるクアッドロータヘリが水平にホバリングしている状態では，以下の線形状態方程式でモデル化できる [14]．

$$\dot{x}(t) = A_c x(t) + B_c u(t),$$
$$y(t) = Cx(t),$$
$$x = (\phi\ \theta\ \psi\ \dot{\phi}\ \dot{\theta}\ \dot{\psi})^T,\ A_c = \begin{pmatrix} O_{3,3} & I_3 \\ O_{3,3} & O_{3,3} \end{pmatrix},$$
$$B_c = \begin{pmatrix} O_{3,3} \\ \mathrm{diag}(\frac{1}{I_{f_\phi}}, \frac{1}{I_{f_\theta}}, \frac{1}{I_{f_\psi}}) \end{pmatrix},\ C = \begin{pmatrix} I_3 & O_{3,3} \end{pmatrix}.$$

なお，ϕ, θ, ψ はそれぞれロール角，ピッチ角，ヨー角 (rad) を表し，I_n，$O_{m,n}$ はそれぞれ $n \times n$ の単位行列，$m \times n$ の零行列を表し，I_{f_ϕ}，I_{f_θ}，I_{f_ψ} はそれぞれロール角，ピッチ角，ヨー角方向に対応する慣性モーメントを表す．さらに，以上の連続時間状態方程式をサンプリング周期 h で離散化すると，次式のようなプラントの離散時間状態方程式表現が得られる．

$$x(k+1) = Ax(k) + Bu(k),$$
$$\begin{pmatrix} A & B \\ O_{3,6} & I_3 \end{pmatrix} = \exp\left\{ \begin{pmatrix} A_c & B_c \\ O_{3,6} & O_{3,3} \end{pmatrix} h \right\}.$$

なお，本シミュレーションではロール角，ピッチ角，ヨー角がセンサによって測定され，それぞれの角速度 $\dot{\phi}$，$\dot{\theta}$，$\dot{\psi}$ をオブザーバが推定すると仮定する．さらに，サンプリング周期を $h = 0.01$ [sec]，慣性モーメントを $I_{f_\phi} = 0.0717$，$I_{f_\theta} = 0.0717$，$I_{f_\psi} = 0.135$ と設定したとき，修正オブザーバの各行列は以下のように求められる．

$$F = \mathrm{diag}(0.1, -0.1, 0.1),$$
$$G = \mathrm{diag}(-81, -121, -81),$$
$$H = \mathrm{diag}(0.0767, 0.0628, 0.0407),$$
$$W = \begin{pmatrix} O_{3,3} \\ I_3 \end{pmatrix}, V = \begin{pmatrix} I_3 \\ \mathrm{diag}(90, 110, 90) \end{pmatrix}.$$

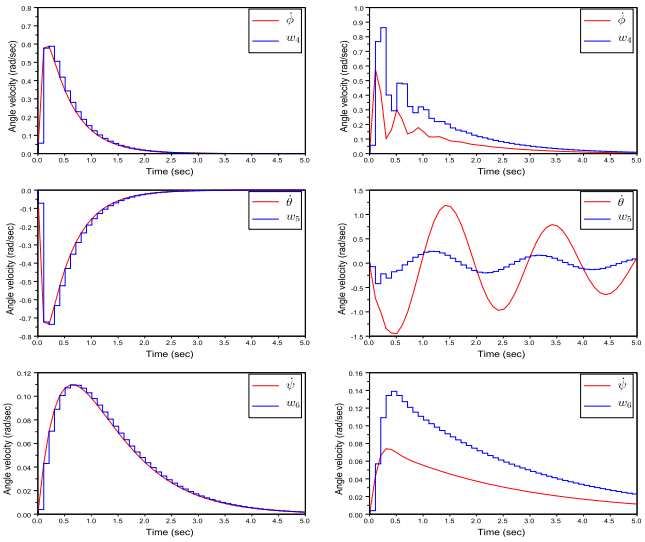
最後に，最適出力フィードバック制御器が 1 回の試行で計算する有限時間区間を $L = 500$ とし，評価関数の係数行列は $Q = \mathrm{diag}(4, 4, 0.1, 1, 1, 0.1)$，$R = I_3$ と設定する．

5.2 修正オブザーバの有効性

本節では，いかなるジョブスキッピングパターンが発生しても修正オブザーバが正確にクアッドロータヘリの状態を推定できることを示す．ここで，次のような 3 通りのジョブスキッピングパターンに基づいて修正オブザーバが状態推定を行う状況を考える．

- $(1, 10)$-firm を保証するジョブスキッピングパターン {1000000000}.
- $(5, 10)$-firm を保証するジョブスキッピングパターン {1010101010}，{1111100000}.
- $(10, 10)$-firm を保証するジョブスキッピングパターン {1111111111}.

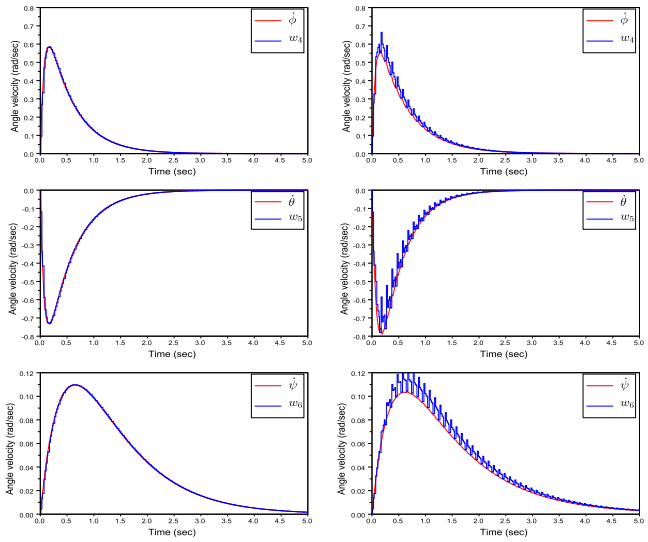
ジョブスキッピングパターンにおけるバイナリ値 ‘1’ と ‘0’ はそれぞれジョブの受理と破棄を意味し，シミュレーション時間区間において同じジョブスキッピングパターンを繰り返すものとする．なお，$(5, 10)$-firm 保証の場合，ジョブの受理と破棄が交互に行われる {1010101010} と最大で 5 回連続ジョブの破棄が起こる {1111100000} の 2 通りのジョブスキッピングパターンについて比較する．前者は $(5, 10)$-firm 保証を最低限満たすジョブスキッピングパターンの中で最も均等にジョブの受理と破棄を行っており，後者は $(5, 10)$-firm 保証を満たすものの，連続的なジョブの破棄が発生している．よって，直感的に前者の方が後者よりも制御性能が良くなると考えられる．$(10, 10)$-firm を保証する場合はジョブスキッピングが起こらない，つまりプロセッサ故障が起こらずに制御できていた場合を表す．制御器のパラメータは制御則 (12)，(13)，(20) に基づいてあらかじめ計算されているものとする．



(a) 修正オブザーバ (The modified observer is used)．　(b) 従来のオブザーバ (The conventional observer is used)．

図 3 {1000000000} の場合のシミュレーション結果
Fig. 3 Simulation results under {1000000000}.

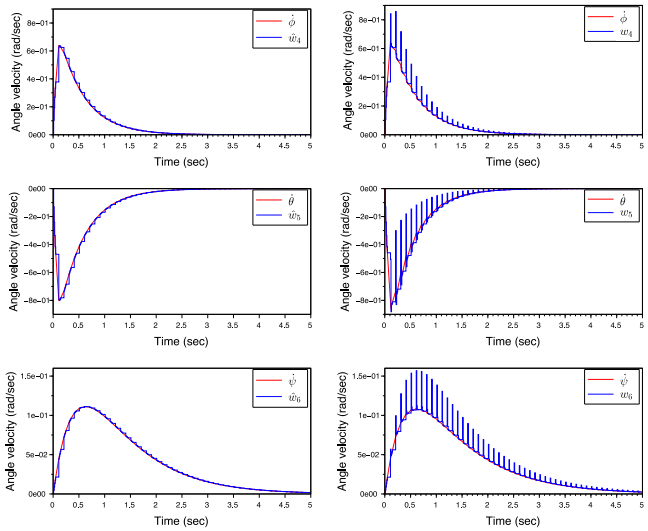
図 3，**図 4**，**図 5** において，時間区間 [0, 5] (秒) における各ジョブスキッピングパターンを用いた場合のプラントの状態シミュレーション結果を示す．ただし，図 (a) は修正オブザーバを用いた場合，図 (b) は従来のオブザーバ (15)，(16) を用いた場合を示す．オブザーバの出力ベクトルを $\hat{w} = (\hat{w}_1, \hat{w}_2, \hat{w}_3, \hat{w}_4, \hat{w}_5, \hat{w}_6)^T$ とし，グラフはクアッドロータヘリの角速度 $\dot{\phi}$，$\dot{\theta}$，$\dot{\psi}$ とオブザーバによる推定値 $\hat{w}_4$，$\hat{w}_5$，$\hat{w}_6$ を示している．シミュレーション結果より，修正オブザーバはどのジョブスキッピングパター



(a) 修正オブザーバ (The modified observer is used).　(b) 従来のオブザーバ (The conventional observer is used).

図 4　{1010101010} の場合のシミュレーション結果

Fig. 4　Simulation results under {1010101010}.



(a) 修正オブザーバ (The modified observer is used).　(b) 従来のオブザーバ (The conventional observer is used).

図 5　{1111100000} の場合のシミュレーション結果

Fig. 5　Simulation results under {1111100000}.

ンにおいても従来のオブザーバと比較して高精度な状態推定が行えていることが分かる．前章で述べたとおり，従来のオブザーバは式 (15)，(16) に従って状態推定値を更新するものの，ジョブスキッピングパターンを考慮しないために推定誤差が生じている．特に $(1,10)$-firm 保証の場合，状態が激しく振動しており，状態推定がまともに行えていないことが分かる．ジョブスキッピングパターン {1010101010} と {1111100000} の場合を比較すると，同じ $(5,10)$-firm 保証のジョブスキッピングパターンでも 5 回連続したジョブの破棄が発生するジョブスキッピングパターン {1111100000} の場合に従来のオブザーバでは大きく振動し，スキッピングパターンによって状態推定性能に差異

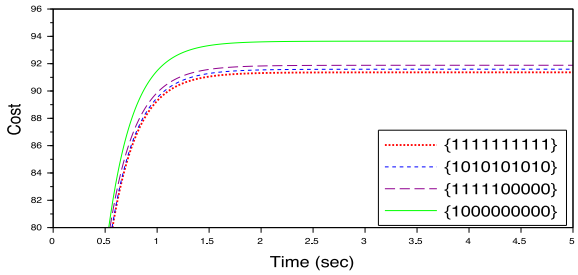


図 6　各 (m,k)-firm 保証が与えられた場合の評価関数の比較

Fig. 6　Comparison of the cost function under each (m,k)-firm constraint.

が発生する．一方，提案した修正オブザーバでは，スキッピングパターンの違いによる状態推定性能の差異が少なく，ジョブスキッピングパターンに寄らず正確な状態推定が行えていることが分かる．なお，$(10,10)$-firm 保証の場合は 2 つのオブザーバはまったく同じ推定を行うのは明らかであるため，シミュレーション結果は示していない．

5.3　最適フィードバック制御器の有効性

本節では，前節のシミュレーションで用いたジョブスキッピングパターンを再び用い，修正オブザーバを組み込んだ最適フィードバック制御器の有効性について検証する．そこで，以下のような各ジョブスキッピングパターンを用いた場合の時刻 k までの評価関数の時間推移を**図 6** に示す．

$$J(k) = \sum_{k'<k} \left\{ x^T(k'+1)Qx(k'+1) + u^T(k')Ru(k') \right\}.$$

なお，時刻 0 秒におけるコスト関数の値は 0 であり，時間推移とともに上昇し，時刻 2 秒付近で収束している．ここで，ジョブスキッピングによる制御性能の劣化度を評価するために次式のような指標 $D_{m,k}$ を導入する．

$$D_{m,k} = \frac{J_{m,k} - J_{10,10}}{J_{10,10}}.$$

ただし，$J_{m,k}$ は (m,k)-firm 保証の場合の評価関数 $J(k)$ の収束値を表す．なお，$(5,10)$-firm の場合はジョブスキッピングパターン {1010101010} の $D_{5,10}$，$J_{5,10}$ をそれぞれ $D^1_{5,10}$，$J^1_{5,10}$，{1111100000} のとき $D^2_{5,10}$，$J^2_{5,10}$ と記す．今回のシミュレーション結果では，$J_{10,10} = 91.366$，$J^1_{5,10} = 91.639$，$J^2_{5,10} = 91.884$，$J_{1,10} = 93.65$ となった．したがって，制御性能劣化度は $D^1_{5,10} = 0.00299$，$D^2_{5,10} = 0.00567$，$D_{1,10} = 0.025$ となり，それぞれ 50%，90%のジョブが破棄されているのに対して制御性能の劣化度が抑えられていることが分かる．同じ (m,k)-firm 保証でもジョブスキッピングパターンが異なれば制御性能も異なることは $(5,10)$-firm の結果から分かる．しかしながら，すべての $1 \le m \le k$ に対して (m,k)-firm 保証を満たすジョブスキッピングパターンは $(1,k)$-firm 保証のジョブスキッピングパターンよりも制御性能が良いことが分かる．つまり，提案手法を用

いることで，どの $(m, 10)$-firm 保証を満たすジョブスキッピングパターンでも制御性能の劣化度を 2.5%以下に抑えられる．

6. おわりに

本論文では (m, k)-firm 制約に基づいて過負荷状態を回避するリアルタイム制御システム上で，ジョブスキッピングパターンを考慮した最適出力フィードバック制御器の設計を行った．評価関数を最小化する最適入力は状態フィードバック形式で得られ，ジョブスキッピングパターンを考慮した修正オブザーバと組み合わせることにより最適出力フィードバック制御器を実現した．シミュレーションでは (m, k)-firm 制約に基づく複数のジョブスキッピングパターンに対して制御性能を比較し，制御性能の劣化が抑えられることを示した．なお，提案手法は可制御・可観測な線形システムに対して適応可能である．

本来，本手法における最適制御入力の計算時間は連続的なジョブスキッピング回数に依存する．つまり，各ジョブの実行時間はジョブスキッピングパターンに依存し，連続的なジョブスキッピング回数が多くなることでジョブスキッピングパターンどおりにデッドラインを満たせない可能性もある．そこで，本手法の計算量を見積もり，どのスキッピングパターンに対してもデッドラインを保証する制御則を設計することが今後の課題である．

謝辞 本研究は JSPS 科研費基盤研究（B）24360164 の助成によるものである．ここに記して謝意を表す．

参考文献

[1] Årzén, K.E. and Cervin, A.: Control and embedded computing: Survey of research directions, *Proc. 16th IFAC World Congress*, Prague, Czech Republic (2005).
[2] Liu, C.L. and Layland, J.W.: Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment, *J. ACM*, Vol.20, No.1, pp.46–61 (1973).
[3] Buttazzo, G.C.: *Hard real-time computing systems: Predictable scheduling algorithms and applications, 3rd Eddition*, Springer (2011).
[4] Buttazzo, G., Lipari, G., Abeni, L. and Caccamo, M.: *Soft Real-Time Systems: Predictability vs. Efficiency*, Springer (2005).
[5] Martí, P., Lin, C., Brandt, S.A., Velasco, M. and Fuertes, J.M.: Draco: Efficient resource management for resource-constrained control tasks, *IEEE Trans. Computers*, Vol.58, No.1, pp.90–105 (2009).
[6] Zhang, F., Szwaykowska, K., Wolf, W. and Mooney, V.: Task scheduling for control oriented requirements for cyber-physical systems, *Proc. 29th IEEE Real-Time Systems Symposium*, pp.47–56 (2008).
[7] Cervin, A., Eker, J., Bernhardsson, B. and Årzén, K.E.: Feedback-feedforward scheduling of control tasks, *Real-Time Systems Journal*, Vol.23, No.1, pp.25–53 (2002).
[8] Liu, J.W.S.: *Real-time Systems*, Prentice Hall (2000).
[9] Ramanathan, P.: Overload management in real-time control applications using (m, k)-firm guarantee, *IEEE Trans. Parallel and Distributed Systems*, Vol.10, No.6, pp.549–559 (1999).
[10] Bernat, G. and Burns, A.: Combining (n,m)-hard deadlines and dual priority scheduling, *Proc. 18th IEEE Real-Time Systems Symposium*, pp.46–57 (1997).
[11] Hamdaoui, M. and Ramanathan, P.: A dynamic priority assignment technique for streams with (m, k)-firm deadlines, *IEEE Trans. Comput.*, Vol.44, No.12, pp.1443–1451 (1995).
[12] Bellman, R.: *Dynamic Programming*, Princeton University Press (1957).
[13] Åström, K.J. and Wittenmark, B.: *Computer-Controlled Systems: Theory and Design, 3rd Edition*, Prentice Hall (1997).
[14] Aubrun, C., Simon, D. and Song, Y.Q.: Implementation: Control and Diagnosis for an Unmanned Aerial Vehicle, *Co-design Approaches for Dependable Networked Control Systems*, ISTE - Wiley, pp.267–304 (2013).

吉本 達也 （学生会員）

2012 年大阪大学大学院修士課程修了．同年同博士課程入学．リアルタイム制御システム，スケジューリング，最適制御の研究に従事．IEEE，電子情報通信学会，計測自動制御学会各会員．

潮 俊光 （正会員）

1985 年神戸大学大学院自然科学研究科博士課程修了．同年 4 月カリフォルニア大学バークレー校研究員．1997 年大阪大学大学院基礎工学研究科教授となり，現在に至る．離散事象システム，リアルタイムシステム，非線形現象の研究に従事．学術博士．電子情報通信学会フェロー，IEEE，計測自動制御学会等各会員．

安積 卓也 （正会員）

大阪大学大学院基礎工学研究科助教．2009 年名古屋大学大学院情報科学研究科情報システム学専攻博士後期課程修了．2008～2010 年日本学術振興会特別研究員．2010～2014 年立命館大学情報理工学部助教．2011～2012 年カリフォルニア大学アーバイン校客員研究員．2014 年より現職．リアルタイムシステム，組込み向けコンポーネントシステムの研究に従事．博士（情報科学）．IEEE，日本ソフトウェア科学会，電子情報通信学会各会員．