研究速報

Takagi-Sugeno ファジィモデルを用いたリアルタイムシステムにおける QoS 適応制御

上滝　晴雄[†]　　潮　　俊光[†]（正員）

QoS Adaptive Control Based on Takagi-Sugeno Fuzzy Model in Real-time Systems

Haruo KOUTAKI[†], *Nonmember* and
Toshimitsu USHIO[†], *Member*

[†] 大阪大学大学院基礎工学研究科，大阪府
Graduate School of Engineering Science, Osaka University; 1-3 Machikaneyamacho, Toyonaka city, Osaka 560-8531, JAPAN

あらまし　QoS 適応制御のためのミドルウェアソフトウェアの開発が重要となってきている．Koliver らは，分散型マルチメディアシステムの QoS 適応制御を，ユーザの知覚の QoS とアプリケーションの QoS をファジィ制御器を用いてマッピングする制御モデルを提案している．しかし，“Standard” なファジィ制御を用いると計算負荷が大きく実用的でない．そこで本論文では，Takagi-Sugeno ファジィモデルを用いた QoS 適応制御法を提案し，その有効性を調べる．

キーワード　QoS 適応制御，リアルタイムシステム，Takagi-Sugeno ファジィモデル

1. ま え が き

計算機環境においてリアルタイムに送られてくるデータを処理する場合，ユーザはある一定の水準以上のサービス品質 (QoS:Quality of Service) が保証されることを期待する．ユーザに対する QoS 保証を実現するためには，単にネットワークやエンドシステムに対して個々に対処するのではなく，これらを統合した QoS 適応システムが必要となり，このようなフレームワークに関する研究が報告されている [1], [2]. また，QoS へのファジィ理論の応用が報告されている．ユーザレベルの QoS とシステムレベルの QoS を変換する QoS マッピングと呼ばれる QoS パラメータの交換にファジィ理論が応用されている [3]．最近，“Standard” なファジィモデルを用いた QoS 適応制御法 [4] が提案されたが，非ファジィ化処理でのオーバーヘッドが大きく効率がよくない．そこで本論文では，Takagi-Sugeno ファジィモデル [5] を用いることで大幅に計算負荷が軽減できることを示す.

2. Takagi-Sugeno ファジィモデル

本節では Takagi-Sugeno ファジィモデルについて概説する．Takagi-Sugeno ファジィモデルのルール $R^i(i=1,\cdots,n)$ は以下のような If 部からなる前件部と then 部からなる後件部から構成される．

$$R^i: \text{If } x_1 \text{ is } A_1^i,\ \cdots,\ \text{and } x_k \text{ is } A_k^i,\quad \text{then } y\ = g_i(x_1,\cdots,x_k) \tag{1}$$

ここで，$x_1, x_2, \cdots, x_k$ は前件部の変数，y は後件部の変数，$A_1, A_2, \cdots, A_k$ はメンバシップ関数のファジィ集合，g_i は入力 $(x_1, x_2, \ldots, x_k)$ が前件部の条件を満たすときの出力 y を与える関数である．本論文では関数 g_i を 1 次関数と仮定する．すなわち

$$g_i(x_1,\cdots,x_k) = p_0^i + p_1^i x_1 + \cdots + p_k^i x_k \tag{2}$$

とおく．但し，p_j^i $(j=0,\cdots,k)$ はオフラインで決定される定数である．ルールの前件部からの入力 $x_1,\cdots,x_k$ に対するメンバシップ関数値を $\mu_{A_1^i}(x_1),\cdots,\mu_{A_k^i}(x_k)$ とすると満足度 α_i は

$$\alpha_i = \mu_{A_1^i}(x_1) \wedge \mu_{A_2^i}(x_2) \wedge \cdots \wedge \mu_{A_k^i}(x_k) \tag{3}$$

である．ただし $a \wedge b = \min(a,b)$ とする．ファジィ推論値 y^* は以下のようになる．

$$y^* = \frac{\sum_{i=1}^n \alpha_i \left(p_0^i + p_1^i x_1 + \cdots + p_k^i x_k \right)}{\sum_{i=1}^n \alpha_i} \tag{4}$$

また関数 g_i のパラメータ $p_0^i, p_1^i, \ldots, p_k^i$ $(i = 1,2,\ldots,n)$ を

$$P = (\, p_0^1 \ \cdots\ p_0^n\ p_1^1 \ \cdots\ p_1^n \ \cdots\ p_k^1 \ \cdots\ p_k^n \,)^T \tag{5}$$

とおくと与えられた入出力データ $(x_{1j}, x_{2j}, \ldots, x_{kj}) \rightarrow y_j$ $(j=1,\ldots,m)$ に対して行列 X とベクトル Y を次式のように定義する．

$$X = \begin{pmatrix} B_1 & x_{11}B_1 & \cdots & x_{k1}B_1 \\ \vdots & \vdots & \ddots & \vdots \\ B_m & x_{1m}B_m & \cdots & x_{km}B_m \end{pmatrix} \tag{6}$$

$$B_j = (\beta_{1j} \cdots \beta_{nj}) \tag{7}$$

$$\beta_{ij} = \frac{\mu_{A_1^i}(x_{1j}) \wedge \cdots \wedge \mu_{A_k^i}(x_{kj})}{\sum_{i=1}^n \left(\mu_{A_1^i}(x_{1j}) \wedge \cdots \wedge \mu_{A_k^i}(x_{kj}) \right)} \tag{8}$$

$$Y = \begin{pmatrix} y_1 & \cdots & y_m \end{pmatrix}^T \tag{9}$$

このときパラメータベクトル P は $P = (X^T X)^{-1} X^T Y$ で得られる．

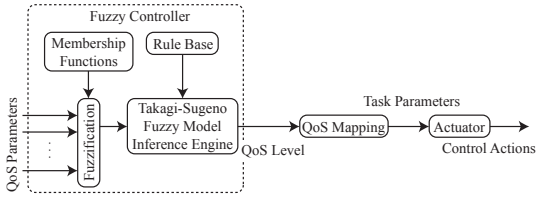

図 1 Takagi-Sugeno ファジィモデルの QoS 適応制御器
Fig. 1 QoS adaptive control mechanism of Takagi-Sugeno fuzzy model

3. QoS 適応制御

3.1 QoS パラメータとタスクパラメータ

各タスクで行われた処理結果の評価は QoS パラメータとして定量的に評価できるとする．QoS パラメータの例としては，CPU での計算時間などがある．また，タスクを実行するときにはタスクパラメータと呼ばれる変数によって処理が異なる．タスクパラメータの例としてはタスクのリリース時間，最大 CPU 利用時間などがある．

3.2 ファジィ QoS 適応制御器

本節では，各 QoS パラメータからファジィ推論を用いてリアルタイムシステム全体を評価する QoS レベル値を求め，この値を基にタスクパラメータを制御するファジィ QoS 適応制御器を提案する．本論文ではファジィ制御に Takagi-Sugeno ファジィモデルを応用する．本制御器は図 1 に示すように，ファジィ制御器，QoS マッピング，アクチュエータの 3 つのサブシステムから構成される．

- ファジィ制御器 (fuzzy controller)

各タスクの QoS パラメータをファジィ化し，ファジィ推論エンジンに送る．ファジィ推論エンジンでは，ファジィルールを適応し Takagi-Sugeno ファジィモデルを用いて QoS レベル値を計算する．QoS レベル値は QoS マッピングに送られる．

- QoS マッピング (QoS mapping)

ファジィ制御器で計算された QoS レベル値をもとに，各タスクのタスクパラメータ値を決定する．

- アクチュエータ (actuator)

アクチュエータでは，QoS マッピングのプロセスで決定された新しいタスクパラメータ値をもとにシステムで実行されるタスクのリリースを行う．

4. 画像トラッキングへの応用例

3 章で提案したファジィ QoS 適応制御の応用例として，リアルタイムに送られてくる画像のトラッキングを考える．送られてくる画像フレーム全てに対してトラッキングを行うこと，また，その 1 枚の画像フレーム全領域に対してトラッキングを行うことはリアルタイム処理を行う上で冗長な作業を含む．そこで CPU における計算負荷の軽減方法として，トラッキングしない画像フレームは処理を行わず，トラッキングする画像フレームのフレームレートを変更する．また，トラッキングする領域は前回トラッキングしたフレームの対象物体の重心を中心とした正方領域としてこの正方領域の大きさを変更する．

この応用例の場合，ユーザが求める QoS は対象物体を見失うことなくトラッキングすることである．ファジィ制御器の入力である QoS パラメータとして，トラッキングしたフレームの対象物体の重心間距離とトラッキングにかかった CPU 計算時間を設定する．重心間距離は，現在トラッキングしたフレームの重心座標と，この処理の 1 つ前にトラッキングしたフレームの重心座標のユークリッド距離で計算を行い，CPU 計算時間は，Windows API の timeGetTime 関数を用いて，トラッキングを開始して重心を求めるまでのプログラムにおける計算時間で測定する．また，タスクパラメータとして，トラッキングする画像のフレームレートと対象物体の重心を中心とした領域の大きさを設定する．これらのタスクパラメータは，ファジィ制御器の出力値である QoS レベル値の値をもとに 2 つ同時に決定する．QoS レベル値のある一定区間ごとに，フレームレートの値と領域の大きさの値の組を事前に与えておく．QoS マッピングにおいて，該当する範囲の QoS レベル値のフレームレートの値と領域の大きさの値の組を照合して，それを新しいフレームレートと領域の大きさとしてアクチュエータで反映する．

本論文に用いた 6 つのルールを以下に示す．ただし，重心間距離と CPU 計算時間の QoS パラメータをそれぞれ x_{dist}, x_{time} とおき，それぞれに対応するメンバシップ関数のラベルを $A_{\mathrm{CLOSE}}, A_{\mathrm{MEDIUM}}, A_{\mathrm{REMOTE}}$，および $A_{\mathrm{UNLOADED}}, A_{\mathrm{LOADED}}, A_{\mathrm{OVERLOADED}}$ とする．2 つのタスクパラメータの両方に対して以下の同じルールを適用している．

$$\mathrm{Rule}^1: \text{If } x_{dist} \text{ is } A_{\mathrm{REMOTE}} \text{ and } x_{time} \text{ is } A_{\mathrm{OVERLOADED}}, \text{ then } y = p_0^1 + p_1^1 x_{dist} + p_2^1 x_{time}.$$

$$\mathrm{Rule}^2: \text{If } x_{dist} \text{ is } A_{\mathrm{REMOTE}} \text{ and } x_{time} \text{ is } A_{\mathrm{LOADED}}, \text{ then } y = p_0^2 + p_1^2 x_{dist} + p_2^2 x_{time}.$$

$$\mathrm{Rule}^3: \text{If } x_{dist} \text{ is } A_{\mathrm{MEDIUM}} \text{ and } x_{time} \text{ is } A_{\mathrm{OVERLOADED}}, \text{ then } y = p_0^3 + p_1^3 x_{dist} + p_2^3 x_{time}.$$

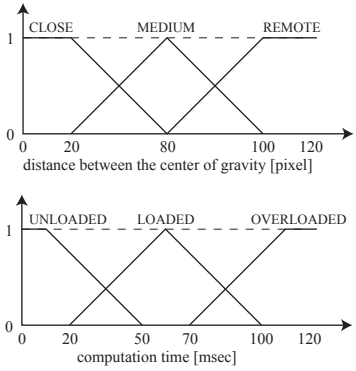

図 2　メンバシップ関数
Fig. 2　Membership functions

表 1　パラメータベクトル値 P
Table 1　Parameter vector P

i	1	2	3	4	5	6
p_0^i	1.2	0.9	1.0	1.6	1.0	0.6
p_1^i	1.3	1.2	1.5	1.7	0.5	1.3
p_2^i	1.1	1.1	1.0	1.0	0.9	1.1

Rule^4 : If x_{dist} is A_{MEDIUM} and x_{time} is A_{LOADED},
then $y = p_0^4 + p_1^4 x_{dist} + p_2^4 x_{time}$.

Rule^5 : If x_{dist} is A_{CLOSE} and x_{time} is A_{LOADED},
then $y = p_0^5 + p_1^5 x_{dist} + p_2^5 x_{time}$.

Rule^6 : If x_{dist} is A_{CLOSE} and x_{time} is A_{UNLOADED},
then $y = p_0^6 + p_1^6 x_{dist} + p_2^6 x_{time}$.

また，本論文で用いた 6 つのメンバシップ関数，およびパラメータベクトル値 P を図 2，表 1 に示す．

さらに，実行画面を図 3,4 に示す．2 つの対象物体をトラッキングしており，トラッキングする領域を四角で表示している．これらの物体のうち左の物体は常にゆっくり動き，右の物体は動くスピードが変動するとした．タスクパラメータを固定した場合 (図 3)，動くスピードの異なる 2 つの物体に対して，常に同じ大きさの領域をトラッキングし，トラッキングを行うフレームレートも一定である．一方，本制御法を用いた場合 (図 4)，早く動く右の物体に対してはトラッキングする領域を大きくし，フレームレートを小さくしている．ゆっくり動く左の物体に対しては領域を小さくしてフレームレートを大きくしている．

実験環境は以下のとおりである．

- CPU: ADM Athlon 951MHz
- OS: Windows 98 SE

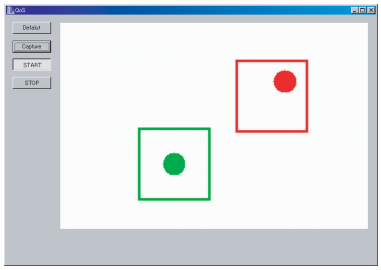

図 3　QoS を考慮しない場合
Fig. 3　Tracking without QoS control

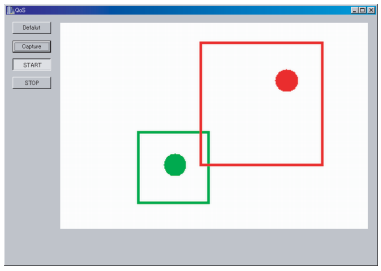

図 4　本制御法を用いた場合
Fig. 4　Tracking with the proposed method

表 2　制御器における CPU 計算時間の比較
Table 2　CPU calculation time in the controller

computation time	average (μsec)
Standard model	6.857×10^3
Takagi-Sugeno model	1.024

- メモリ: 192MB
- ソフトウェア言語: Borland C++ Builder 4
- キャプチャボード: PHOTRON FDM-PCI

4.1　計算時間の従来法との比較

文献 [4] で用いられたファジィ制御と本制御法を比較するために，ルール数を 6 とした場合のファジィ制御器における CPU 計算時間の平均を調べた結果を表 2 に示す．

文献 [4] で用いられた制御法では，非ファジィ化処理でオーバーヘッドが大きくなるが，本制御法では式 (4) で示した簡単な演算によって出力を生成できるため，ファジィ制御器における計算時間が大幅に軽減できることが分かる．また，文献 [4] ではルール数を 3 として実験を行っているが，本制御法では 6 に増やしても計算のオーバーヘッドが小さいことを確認した．

4.2　QoS に関する考察

次にトラッキングするフレーム数を 100 として画像トラッキングを行った．タスクパラメータであるトラッキングする画像のフレームレートと対象物体の重心を中心とした領域の大きさをある一定値に固定した

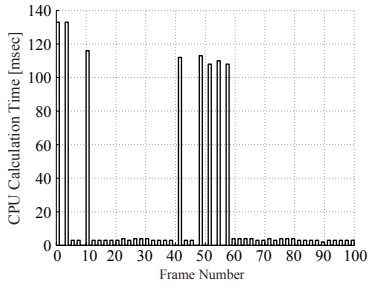

図 5　QoS を考慮しない場合の CPU 計算時間
Fig. 5　CPU calculation time when no control is used

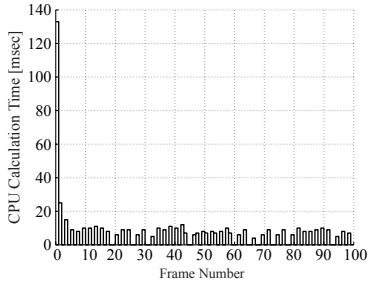

図 6　本制御法を用いた場合の CPU 計算時間
Fig. 6　CPU calculation time when the proposed method is applied

QoS 考慮しない場合と本制御法を用いた場合の CPU 計算時間をそれぞれ図 5,6 に示す．フレーム番号 0 では画像フレームの全領域に対してトラッキングを行うため CPU 計算時間が大きくなる．CPU 計算時間が 0 であるフレーム番号は，トラッキングをせずにフレームを飛ばしたことを意味している．

QoS を考慮しない場合，トラッキングする領域を固定しているため，CPU 計算時間はほぼ一定の値を取る．しかし，フレーム番号が 40 から 60 の間で CPU 計算時間が大きくバースト的に変化していることがわかる．これはトラッキングミスしたことにより画像全体に対して物体の検索を行ったためである．本制御法を用いた場合，トラッキングする領域の大きさをリアルタイムに計算し変更するため図 5 と比べると CPU 計算時間が多少変動している．しかし，CPU 計算時間が大きくバースト的に変化するフレームがなく，トラッキングミスなく処理しており．本制御法が有効であることが確認できた．

5. む す び

本論文では，リアルタイムシステムにおける QoS 適応制御法として Takagi-Sugeno ファジィモデルを用いる QoS 適応制御法を提案した．また，応用例として提案手法をリアルタイムに送られてくる画像のトラッキングに適用することによりその有用性を示した．

今後の検討課題として，通信ネットワークを介したリアルタイムシステムに拡張することがある．さらに本制御法を実装したミドルウェアの開発が挙げられる．

文　献

[1] Klara Nahrstedt and Jonathan M. Smith, “The QoS Broker”, IEEE Multimedia Magazine, vol.2, no.1, pp.53–67, Spring 1995.

[2] Baochun Li and Klara Nahrstedt, “A Control-Based Middleware Framework for Quality-of-Service Adaptations”, IEEE Journal on Selected Areas in Communications. vol.17, no.9, pp.1632–1650, Sept. 1999.

[3] 中岡 謙, 松田 潤, “ファジィクラアシファイアシステムによる QoS マッピングルールの獲得”, 電子情報通信学会論文誌, vol.J85–D–I, no.1, pp.69–78, Jan, 2002

[4] Cristian Koliver, Klara Nahrstedt, Jean Marie Fraga and Sandra Aparecida Sandri, “Specification, Mapping and Control for QoS Adaptation”, Real-Time Systems, vol.23, no.1–2, pp.143–174, 2002.

[5] Tomohiro Takagi and Michio Sugeno, “Fuzzy Identificaion of Systems and Its Applications to Modeling and Control”, IEEE Transactions on Systems, Man, and Cybernetics, vol. 15, no.1, pp.116–132, January/February, 1985.

（平成 xx 年 xx 月 xx 日受付）