情報処理学会論文誌　数理モデル化と応用　Vol. 3　No. 3　44–53 (Oct. 2010)

時間オートマトンによるフェースディスプレイの上位設計と形式的検証

田川聖治[†1]　高橋佑輔[†1]　加藤暢[†1]

本稿では，フェースディスプレイの設計と検証の手法を提案している．フェースディスプレイとは，対象システムに同期して表情アニメーションを実行するものである．まず，フェースディスプレイのモデルは感性モデルと表現モデルの 2 階層により構成される．感性モデルは人間の心理状態の変化を記述し，表現モデルは人間の表情の変化を記述している．また，感性モデルと表現モデルは，それぞれ時間オートマトンにより定義している．さらに，対象システムのモデルについても，時間オートマトンにより構築している．最後に，CTL（Computation Tree Logic）時相理論に基づくモデル検証ツールである UPPAAL を用いて，フェースディスプレイと対象システムを並列合成したシステム全体の到達可能性，安全性，および，活性を判定している．

High-level Design and Formal Verification of Face Display Using Timed Automata

Kiyoharu Tagawa,[†1] Yusuke Takahashi[†1] and Toru Kato[†1]

A high-level design and formal verification method of the face display system is proposed. The face display system performs the animation of human's faces synchronizing with the states of an arbitrary target system. The model of the face display system is composed of two hierarchical parts called the kansei-model and the hyougen-model respectively. The kansei-model represents the transition of human's feelings. On the other hand, the hyougen-model represents the transition of human's faces. Both the kansei-model and the hyougen-model are defined by using timed automata. Furthermore, the target system is also modeled by using timed automata. Then a model checking tool based on the temporal logic of CTL (Computation Tree Logic), i.e., UPPAAL, is used to verify the reachability, the safety and the liveness properties of a whole system into which the face display system and the target system are integrated.

1. はじめに

表情は重要な非言語コミュニケーションの手段であり，認知科学の分野では古くから研究されている[1)]．また，情報工学の分野においても，表情を含む「顔」に関連する研究は活発に行われている[2)]．ここで，情報工学における「顔」の研究は，認識技術と合成技術に大別される．さらに，顔の合成技術の応用研究として，人と機械のコミュニケーション支援を目的としたヒューマンインタフェースの開発があり，数多くの事例が報告されている．

たとえば，Chernoff が考案したフェース法[3)] は，多変数データを表現する顔図形の作成手法である．フェース法では，多変数データの各成分を顔図形の目，口，眉などのパーツの変化に割り当てることで，多変数データを 1 つの表情として提示する．近年，情報処理技術の進歩にともなって，フェース法にはアニメーションの技法が導入され，原子力発電所など大規模で複雑な対象システムの状態を瞬時に把握するために利用されている[4)–6)]．

そのほか，顔の合成技術の応用例として擬人化エージェントがある．ユーザの作業を代行するソフトウェアをエージェントと呼び，それらの中でコンピュータグラフィックス（CG）などで生成された「顔」を持つものが擬人化エージェントである．擬人化エージェントにおける顔の役割は，ユーザに対して親しみやすさを演出することである．さらに，手話アニメーションにより文章を伝達するシステム[7)] では，手話を演じるキャラクタの表情が，感情を視覚的に伝えるとともに，ユーザの文章内容の理解度を高める効果を発揮している．

通常，コンピュータによる表情アニメーションは，原理的に複数の顔画像を臨界フリッカ周波数[8)] に基づく時間間隔で連続的に表示することで実現される．また，各顔画像の表情の記述形式としては，エクマンらが開発した FACS（Facial Action Coding System）における AU（Action Unit）などがある[9)]．さらに，表情アニメーションにおける顔画像の時間的な推移の記述形式としては，一連の顔画像を時系列の順番に並べて記録する手法[2)] のほか，顔パーツを単位として顔画像の変化を記録する表情譜[10)] が提案されている．しかし，アプリケーションによっては，顔画像は写実的である必要はなく，むしろ誇張された表情の方が効果的な場合もある．また，システムの状態に同期させて顔画像の表示パターンを動的に切り換える場合は，表情アニメーションの上位にあるシナリオの設計が重要となる．

本稿では，時間オートマトンを用いたフェースディスプレイの設計と検証の手法を提案

†1 近畿大学理工学部
School of Science and Engineering, Kinki University

する．まず，フェースディスプレイとは，様々な対象システムに「顔」を提供するものであり，その状態に応じた表情アニメーションをリアルタイムで実行する．また，フェースディスプレイのような実時間システムの開発では，適切なモデルを用いた上位設計やモデル検証を行うことで，設計の後戻りが少なくなり，開発後の保守や拡張も容易となる[11]．

時間オートマトン[12]とは，有限個のロケーション[⋆1]にクロックを付加したBüchiオートマトンであり，時間制約のある実時間システムのモデルとして利用されている[13]．ここで，フェースディスプレイのモデルは，表情アニメーションにおける顔画像の時間的な遷移を記述した表現モデルと，人間の表情の背景にあると考えられる心理状態の時間的な推移を記述した感性モデルの2層から構成され，それぞれが時間オートマトンで表される．さらに，感性モデルでは心理状態をロケーションとし，表現モデルでは顔画像の表示期間をロケーションとしている．感性モデルと表現モデルを分けることで，対象システムの状態に対応したシナリオの設計と，意匠を凝らした顔画像や表情の微妙な調整が独立に行える．

フェースディスプレイを時間オートマトンによりモデル化する利点としては，イベントの発生や時間制約による心理状態の複雑な遷移規則のほか，各心理状態に対応した顔画像の時間的な推移を詳細に指定できることがあげられる．また，顔画像の表示パターンの全体像を視覚的に把握できるため，表情アニメーションのシナリオの設計も容易となる．さらに，対象システムについても時間オートマトンによりモデル化することで，時間オートマトンの代表的なモデル検証ツールであるUPPAAL[14]を用いて，フェースディスプレイと対象システムを並列合成したシステム全体のモデル検証が可能となる．ここで，時間オートマトンは実時間システムのモデルとして広く利用されているため，様々な対象システムのモデルを構築することが可能であると考えられる．UPPAALによれば，CTL（Computation Tree Logic）時相理論[13]の論理式に基づき，時間オートマトンの到達可能性や活性のほか，デッドロックフリーなどの安全性を検証することができる．とくに，フェースディスプレイの設計では，対象システムにおける処理時間や顔画像の表示時間などを変えて，対象システムの状態と表情アニメーションのタイミングの整合性を事前に検証できるメリットは大きいと思われる．ちなみに，SPINなどソフトウェアのモデル検証ツールでは，時間オートマトンの検証は行えず，モデルをコードで記述する必要があるために習熟を必要とする[15]．

本稿では，具体的な対象システムとしてプログラムの教育支援システムを取り上げ，時間オートマトンを用いたフェースディスプレイの設計と検証の手法について説明する．ここで，フェースディスプレイに期待される効果は，教育支援システムを利用する学生の学習意欲の向上である．一般的に，プログラムの教育では，指導者の人数に比べて学生数が圧倒的に多い．このため，独自の教育支援システムを開発し，授業や演習で利用する試みがいくつかの教育機関で行われている[16]．教育支援システムを利用することで，学生が作成したプログラムの評価などを自動化し，授業や演習の効率化を図ることができる．しかし，学生は自主的に学習を進める必要があるため，学生に学習意欲を維持させることが課題となる．

教育支援システムに搭載されたフェースディスプレイが実行する表情アニメーションの設計では，前述のフェース法のように対象システムの状態を表示したり，擬人化エージェントのように親しみやすさを演出したりするだけでなく，人間の表情が持つ感情の同調性の効果も期待している．私たちは，嬉しそうな顔を見ることで喜びの感情が沸き起こり，悲しそうな顔を見ることで気分が沈むことを体験したことがある．すなわち，表情には送り手の感情を受け手に伝える機能のほか，送り手と同じ感情を受け手にいだかせる同調性の機能がある[17]．さらに，他人の表情を観察するだけで喜怒哀楽の感情が発現する現象には，ミラー・ニューロンが深く関与していることが，脳科学の分野において指摘されている[18]．このため，学習内容に応じた適切な表情アニメーションを学生に見せて，人間の意欲の根源である感情を刺激すれば，学生がやる気を起こし，学習内容に興味を持つ可能性もあると考えられる．

2. 時間オートマトンとモデル検証

2.1 時間オートマトン

時間オートマトンは6項組 $(L, l_0, \Theta, A, E, I)$ で定義される[12]．ここで，L はロケーションの集合，$l_0 \in L$ は初期ロケーション，Θ はクロックの集合，A はアクションの集合，E はロケーション間の遷移の集合であり，クロックに関する制約条件の集合を $B(\Theta)$ とすると，$E \subseteq L \times A \times B(\Theta) \times 2^{\Theta} \times L$ が成り立つ．$I : L \to B(\Theta)$ はロケーションに対してインバリアントと呼ぶ時間の制約条件を割り当てる．また，時間オートマトンはBüchiオートマトンと同様に永遠の動作を記述するため，受理状態に相当するものはない．

2.2 UPPAALの時間オートマトン

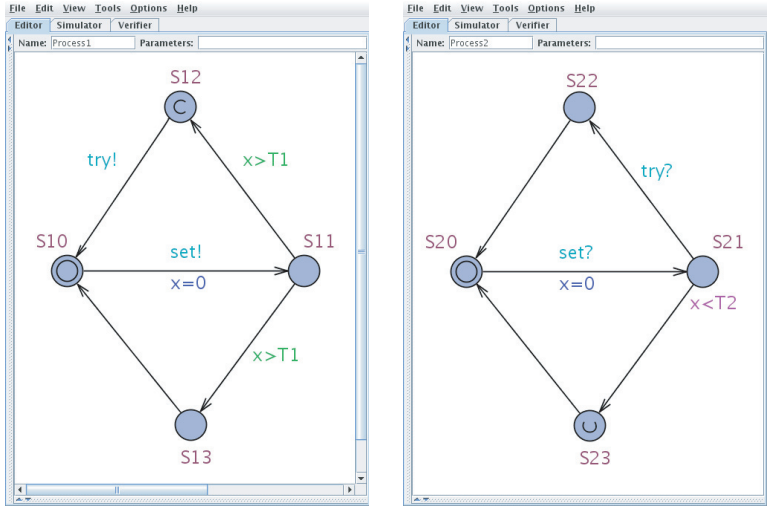
時間オートマトンのモデル検証ツールであるUPPAAL[14]は，スウェーデンのUppsala大学とデンマークのAalborg大学によって開発された．また，UPPAALはグラフィカルなエディタによる実時間システムのモデリング，シミュレーション，および，CTL時相論理の論理式に基づくモデル検証が行える総合開発環境でもある．ここで，前述の時間オート

⋆1 ロケーション（location）は有限オートマトンの状態に相当する．

マトンを用いても，抽象化された実時間システムのモデルは構築できる．しかし，フェースディスプレイのように本格的な実時間システムの振舞いを正確に記述するためには，6 項組の標準的な時間オートマトンでは表現能力に限界がある．このため，UPPAAL では標準的な時間オートマトンに拡張を加えた拡張時間オートマトンをモデル検証の対象とする．

UPPAAL が対象とする時間オートマトンでは，クロックのほか，整数の変数や配列が扱える．時間に関する制約条件には，ロケーションに付随するインバリアントのほか，遷移に付随するガードがある．また，特別なロケーションとして，時間の経過が許されない Urgent（U）と Committed（C）がある．ここで，後者のロケーション（C）は前者（U）よりも制約が厳しく，ロケーション（C）からの遷移はつねに最優先される．さらに，複数の時間オートマトンが，チャネルを介したイベントの送受信により同期をとることもできる．

UPPAAL の時間オートマトンの具体例として，**図 1** に 2 つの実時間システム Producer と Consumer のモデルを示す．丸はロケーションを表し，二重丸は初期ロケーションである．ロケーション間を結ぶ矢印は遷移を示している．ここで，各時間オートマトンはそれぞれクロック `x` を持ち，それらの時間は `S12` と `S23` を除くロケーションにおいて一様かつ



(a) Producer　　(b) Consumer

図 1　時間オートマトン
Fig. 1　Timed automata.

稠密に経過する．また，Consumer の `S21` に付随する時間制約式 `x<T2` がインバリアント，Producer の `S11` からの遷移に付随する時間制約式 `x>T1` がガードである．さらに，Producer と Consumer の間で送受信されるイベントは `set` と `try` の 2 種類であり，Producer が `set!` と `try!` で送信した各イベントを，Consumer はそれぞれ `set?` と `try?` で受信する．

時間オートマトンで遷移が起きる条件は，その遷移に付随するガードと遷移先のインバリアントを満たし，イベントが同期することである．このため，Consumer は Producer から `set` を受けて初期ロケーション `S20` から `S21` に遷移するとともにクロックを `x=0` とリセットする．その後，クロック `x` の時間が `T2` に至るまでの任意の時刻に `S23` に遷移するか，Producer のガードとインバリアントの時間制約式を満たす期間内（`T1` $<$ `x` $<$ `T2`）に `try` を受けて `S22` に遷移する．一方，Producer は `S11` において時間 `T1` が経過した後，任意の時刻で `S12` か `S13` に遷移する．ここで，`S12` に推移した場合はただちに `try` を送信する．

2.3　UPPAAL によるモデル検証

モデル検証ツールである UPPAAL によれば，時間オートマトンでモデル化された実時間システムに対して，CTL 時相理論に基づく以下のような論理式を検証できる．

(1)　E◇ ϕ
(2)　A□ not ϕ
(3)　A□ (ω imply (A◇ ϕ))
(3′)　ω --> ϕ

論理式 (1) は到達可能性と呼ばれ，いつかは項 ϕ により記述された状況に到達することを検証する．また，論理式 (2) は安全性と呼ばれ，実時間システムがどのように振る舞っても，項 ϕ によって記述された状況に至らないことを検証する．さらに，論理式 (3) は活性と呼ばれ，項 ω により記述された状況が発生すれば，必ずいつかは状況 ϕ に至ることを検証する．ただし，UPPAAL の文法に従えば，論理式 (3) は論理式 (3′) のように記述する．

たとえば，図 1 の時間オートマトンに対しては，以下のような論理式が検証できる．

(4)　E◇ `Consumer.S22`
(5)　A□ not (`Consumer.S22`)

論理式 (4) は到達可能性であり，いつかは Consumer のロケーションの `S22` に至ることを検証する．また，論理式 (5) は安全性であり，Consumer の `S22` に至ることはないことを検証する．ここで，ガードとインバリアントの時間が `T2` $\leq$ `T1` であるとき，Consumer と Producer が `try` で同期することはなく，論理式 (5) は成立するが論理式 (4) は成立しない．一方，`T2` $>$ `T1` ならば，論理式 (4) は成立するが論理式 (5) は成立しない．

3. ロボット TA の設計

3.1 ロボット TA の機能

著者の 1 人が開発したロボット TA（Teaching Assistant）は教育支援システムの一種であり，画像処理のプログラミング実習において，実際に学生たちに使用させている．そこで，本稿では，ロボット TA に「顔」を提供するフェースディスプレイを開発する．

学生は講義で画像処理のアルゴリズムを学んだ後，演習で画像処理プログラムを作成する．その後，学生が作成した画像処理プログラムをロボット TA が点検し，その合否を学生に通知する．高性能なサーバとネットワークを必要とする e-Learning などとは異なり，ロボット TA はスタンドアロンのソフトウェアである．このため，学生は Web ページからロボット TA をダウンロードして，各人のパソコン上で自由に繰り返し使用できる．

ロボット TA の原理を**図 2** のフローチャートに基づき説明する．はじめに，学生は課題の画像処理プログラムを作成した後，与えられたテスト画像を入力画像（img1）として画像処理を施した出力画像（img2）を生成する．次に，学生はロボット TA を起動して，入力画像と出力画像のファイルをロボット TA に読み込ませる．ロボット TA は正解の画像処理プログラムを内蔵しており，入力画像を自動的に処理して正解画像（img3）を生成する．さらに，ロボット TA は正解画像と出力画像をピクセルごと比較して，両方の画像が完全に一致すれば，その旨を文字表示により学生に通知する．一方，出力画像と正解画像が異なる場合，誤り箇所を明示した添削画像（img4）を作成し，正解画像と添削画像をディスプレイ上に並べて表示する．もちろん，正解画像は表示されるだけで，画像ファイルとして取り出すことはできない．**図 3** はロボット TA が 2 枚の画像を表示した様子であり，ロボット TA の上端には画像ファイルの読み込みボタンと，点検のためのテストボタンがある．

3.2 ロボット TA のモデル

時間オートマトンによりロボット TA のモデルを構築する．まず，UPPAAL のグラフィカルなエディタを用いて描いたロボット TA の時間オートマトンを**図 4** に示す．ロケーションはアイドル「`Idle`」，画像を開く「`Open`」，画像を閉じる「`Close`」，画像を点検中「`Check`」，合格「`Pass`」，不合格「`Fail`」の 6 個であり，初期ロケーションは `Idle` とする．また，時間オートマトンは，1 つのクロック `x` と画像の枚数を表す変数 `img` を持つ．

まず，学生がロボット TA を起動したとき，図 4 の時間オートマトンは初期ロケーション `Idle` に位置する．次に，学生がロボット TA を操作することで，その内容に応じたロケーションに `Idle` から遷移するとともにイベントを送信する．ここで，学生がロボット TA を続けて操作する最短の間隔を `Tu` とし，`Idle` からの遷移のガード（`x` > `Tu`）とする．

学生が入力画像（img1）か出力画像（img2）を開くと，図 4 の時間オートマトンはイベント `open!` を送信してロケーション `Idle` から `Open` に遷移するとともに，変数 `img` をインクリメントする．その後，`Open` で画像ファイルを読み込むための時間を費やして `Idle`

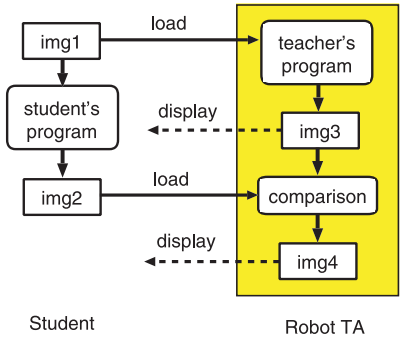


図 2　ロボット TA の機能
Fig. 2　Function of robot TA.

図 3　ロボット TA による画像表示
Fig. 3　Dislay of image on robot TA.

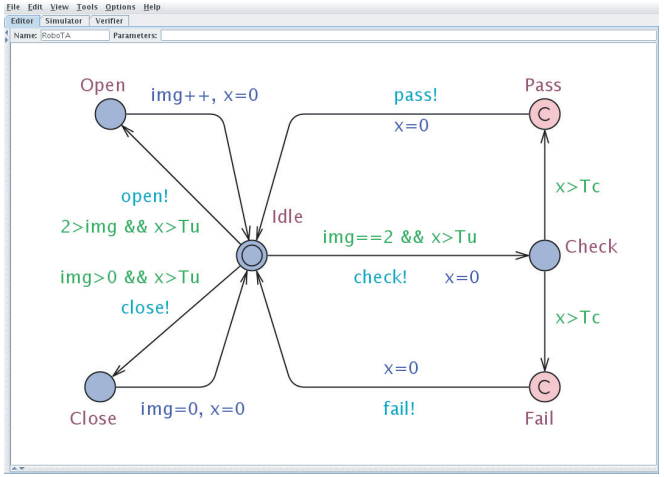


図 4　ロボット TA のモデル（Robot）
Fig. 4　Model of robot TA (Robot).

に戻る．同様に，画像を閉じた場合は，イベント close! を送信して Idle から Close に遷移するとともに，変数 img の値を零とする．画像の点検は入出力画像を開いた img==2 の状態でのみ実行でき，Idle からイベント check! を送信して Check に遷移する．ここで，画像の点検時間の下限値を Tc として，Check からの遷移のガード（x > Tc）とする．Check における点検の結果，合格の場合は Pass を，不合格の場合は Fail を経由し，それぞれイベント pass! または fail! を送信して初期ロケーションの Idle に戻る．

4. フェースディスプレイの設計

4.1 フェースディスプレイの機能

フェースディスプレイは，ロボット TA の状態に応じて表情アニメーションを実行する．顔画像は 6 基本表情[9]（驚き，恐怖，嫌悪，怒り，幸福，悲しみ）と無表情，および，それらの表情を補間する計 28 枚の顔画像を準備した．顔画像のキャラクタとしては，学生たちの大半が男性であることから，表情が持つ同調性の効果を期待して，多くの学生が自らを投影できる若い男性とした．また，各顔画像は AU[9] に基づく感情の表情による表現方法なども参考にして，市販のグラフィック・ソフトウェア Poser7®によって作成した．

まず，ロボット TA を起動すると無表情な男性の顔が現れる．このとき，**図 5** のような目を開いた顔画像と閉じた顔画像の 2 枚を適当なタイミングで交互に表示することで，無表情な顔に「瞬き」の動作を繰り返させる．無表情な顔もつねに動かすことで，不自然さが軽減されるとともに，フェースディスプレイが正常に機能していることを確認できる．

次に，**図 6** に示すような「怒り」や「悲しみ」の感情を表現する顔画像が，ロボット TA の状態の変化や学生による操作に同期して表示される．その後，フェースディスプレイは，適当な時間制約に従っていくつかの顔画像を経由し，途中でロボット TA が操作されなければ，自動的に無表情な「瞬き」の動作に戻るものとする．ただし，表情アニメーションの具体的なシナリオの詳細については，そのモデルと設計の方法を含めて後述する．

図 5　無表情な瞬き
Fig. 5　Blinking of poker face.

図 6　「怒り」と「悲しみ」
Fig. 6　Anger and sorrow.

4.2 フェースディスプレイの階層モデル

表情アニメーションの設計では，ロボット TA の状態や時間の経過に応じて表示する顔画像と順序，および，各顔画像の表示時間を決めればよい．したがって，時間オートマトンの各ロケーションを顔画像とすれば，自然にフェースディスプレイのモデルが得られる[19]．しかし，意匠を凝らした表情アニメーションでは，時間オートマトンの構造が複雑となり，その描画や理解が困難となる．また，ロケーションの数が顔画像の枚数に比例するため，顔画像の枚数が多くなると UPPAAL を用いたモデル検証に要する時間も長くなる．

本稿では，上記の課題を解決するために，人間の表情の背景にあると考えられる感性をともなった感情に着目し，フェースディスプレイの表情アニメーションに含まれる顔画像の集合を構造化する．ここで，様々な情報メディアを使った人間同士のコミュニケーションに対し，感性メディア，表現メディア，通信メディアの 3 層からなるモデルが提案されている[8]．まず，最上層に位置する感性メディアは，人間の脳の中に感性をともなって存在する情報である．喜怒安楽のような人間の感情も，「激しく」や「静かに」などの形容詞で表現される感性をともなった情報である．次に，表現メディアとは画像や音声などの表現パターンであり，人間同士のコミュニケーションでは，感性をともなった情報が表現メディアを通じて表出される．この際，同じ情報であっても付随する感性によってその表現パターンは異なる．最後に，最下層の通信メディアは，表現パターンを伝える空気や電波などの物理的な媒体である．

冒頭でも述べたとおり，フェースディスプレイには表情の持つ感情の同調性の効果を期待している．したがって，上記のコミュニケーションのモデルにフェースディスプレイをあてはめれば，感性メディアはロボット TA を操作する学生自身の心理状態の変化である．ただし，本稿では様々な感性をともなった感情を心理状態と呼ぶものとする．次に，表現メディアは一連の顔画像の時系列による表情アニメーションである．さらに，通信メディアはパソコンのディスプレイとなる．そこで，**図 7** のようにフェースディスプレイを感性モデルと表現モデルの 2 層に分け，それぞれのモデルを時間オートマトンを用いて構築する．

図 7 において，実線の楕円は 1 つの時間オートマトンを表し，右端には図 4 のロボット TA のモデルを示している．また，左側の二点鎖線で囲まれた部分がフェースディスプレイのモデルであり，1 つの時間オートマトンで表された感性モデルと，一点鎖線で囲まれた表現モデルの 2 層で構成されている．さらに，表現モデルは複数の時間オートマトンの集合であり，各時間オートマトンはショットと呼ぶ短編の表情アニメーションをモデル化している．まず，ロボット TA が発生する各種イベント（events）と時間制約による遷移規則に

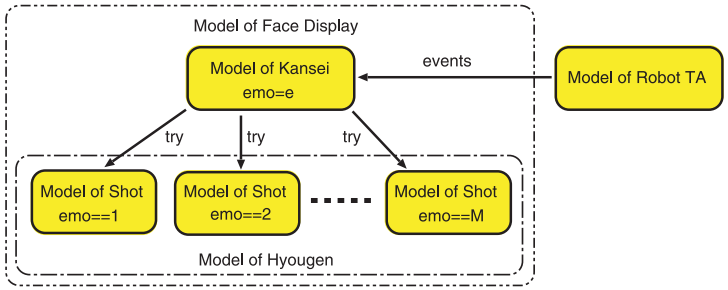


図 7　システム全体のモデル
Fig. 7　Model of a whole system.

基づき，感性モデルのロケーションである心理状態が変化する．各心理状態には表現モデルの 1 つのショットが対応している．したがって，感性モデルの心理状態の総数を M とすると，M 個のショットが存在する．ある時点で活性化した心理状態は，対応するショットの表情アニメーションを起動するため，グローバルな変数 `emo` にそのショットの識別番号 `e` を代入し，イベント `try` をブロードキャストする．すべてのショットは `try` を受信してそれぞれの識別番号を確認し，該当するショットのみが表情アニメーションを開始する．

4.3　フェースディスプレイの感性モデル

本稿では，フェースディスプレイの表情アニメーションについて，以下のようなシナリオを考えた．ただし，シナリオの設計は顔画像ではなく，心理状態に基づき行うものとする．まず，ロボット TA に入出力画像のファイルを読み込むと，フェースディスプレイは図 5 の無表情に対応する心理状態から期待を込めた「幸福」に変化する．一方，画像を閉じた場合は「嫌悪」となる．また，ロボット TA が画像を点検している最中は，不安な意味での「恐怖」とする．さらに，点検の結果が合格となった場合，心理状態は「驚き」から「幸福」に変化する．一方，不合格となった場合は，「怒り」から深い「悲しみ」に変化する．

上記のシナリオを時間オートマトンで表現した感性モデルを**図 8** に示す．図 8 にはクロック `x` と 8 種類の心理状態 `K0` ～ `K7` が含まれており，各心理状態は持続時間 `Tm` をガードとしている．図 8 の感性モデルは，図 4 のロボット TA から 5 種類のイベント `check`，`pass`，`fail`，`open`，`close` を受信して心理状態を遷移させるとともに，グローバル変数 `emo` に適切なショットの識別番号を代入し，表現モデルに対してイベント `try` を発信する．たとえば，図 8 の初期ロケーション `K0` は無表情な心理状態に対応し，ロボット TA で画像の点検が行われると `check` を受信して「恐怖」の心理状態 `K1` に遷移する．この際，`try` を発信

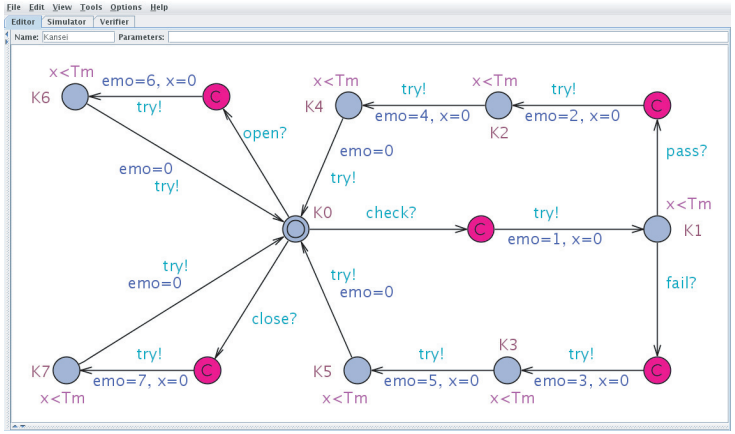


図 8　感性モデル（Kansei）
Fig. 8　Model of kansei (Kansei).

するとともに `emo=1` と設定する．ここで，`check` の受信と `try` の発信を同時に行うため，ロケーション（C）が `K0` と `K1` の間に挿入されている．次に，点検の結果が合格となると `pass` を受信して「驚き」の心理状態 `K2` に遷移する．さらに，`K2` で `Tm` 時間が経過すると「幸福」の心理状態 `K4` に遷移して `Tm` 時間とどまった後，初期ロケーション `K0` に戻る．

4.4　フェースディスプレイの表現モデル

表現モデルに含まれる各ショットは，感性モデルの心理状態に対応した単純な構造を持つ短編の表情アニメーションである．図 8 の「驚き」の心理状態 `K2` に対応する表情アニメーションの具体例を**図 9** に示す．図 9 のショットは 5 枚の顔画像から構成され，驚きから幸福な表情に変化する．図 9 の各顔画像の下に示した数値は，顔画像の識別番号 `f[k]` と時間 `Ts` を単位とした表示時間 `t[k]` である．ここで，図 9 のショットをモデル化した時間オートマトンとして，各顔画像をロケーションに対応させたものが考えられる[19]．しかし，ショットに含まれる顔画像の枚数が増えると，時間オートマトンの描画に手間がかかる．また，顔画像の枚数が異なるショットごと，新たに時間オートマトンを描画する必要がある．

本稿では，表現モデルに含まれる各ショットを 4 個のロケーションからなる時間オートマトンによりモデル化する．図 9 に示したように顔画像を順番に表示して終了するショットをフロー型ショットと呼び，**図 10** の時間オートマトンによってモデル化する．図 10 の時間オートマトンは，ショットに含まれる顔画像の識別番号の配列 `f[k]`，その表示時間の配列

図 9　「驚き」から「幸福」のショット
Fig. 9　Shot from surprise to joy.

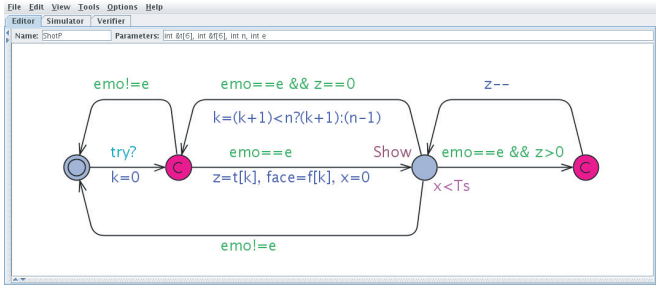


図 10　フロー型ショットのモデル（ShotF）
Fig. 10　Model of flow type shot (ShotF).

`t[k]`，配列の長さ `n`，および，ショットの識別番号 `e` を引数として与えられ保持している．まず，図 10 の時間オートマトンでは，図 8 に示した感性モデルからのイベント `try!` に `try?` で同期し，変数 `k` を初期化してロケーション（C）に至る．ここで，グローバルな変数 `emo` の値がショットの識別番号 `e` と異なれば，初期ロケーションに戻り待機する．一方，変数 `emo` の値が識別番号 `e` に等しければ，時間 `Ts` を単位とする顔画像の表示時間 `t[0]` を変数 `z` に代入し，顔画像の識別番号 `f[0]` をグローバルな変数 `face` に代入する．このとき，変数 `face` に顔画像の識別番号を代入できるショットは，識別番号 `e` が変数 `emo` と一致する 1 つだけであり，変数 `face` で指定された顔画像がディスプレイ上に表示される．

次に，図 10 の時間オートマトンはクロック `x` を初期化してロケーション `Show` に至り，変数 `emo` が `e` に等しい限り，変数 `z` の値を更新しながら，識別番号 `f[0]` の顔画像を最長で `t[0]` × `Ts` 時間表示する．さらに，変数 `k` の値を更新し，識別番号 `f[1]` から `f[n-1]` の顔画像を順番に `face` に代入して表示する．その後，変数 `k` の値は `n-1` に固定され，識別番号 `f[n-1]` の顔画像のみが表示される．ただし，途中で変数 `emo` の値が変わり別のショットが起動された場合は初期ロケーションに戻り，次に `try` を受信するまで待機する．

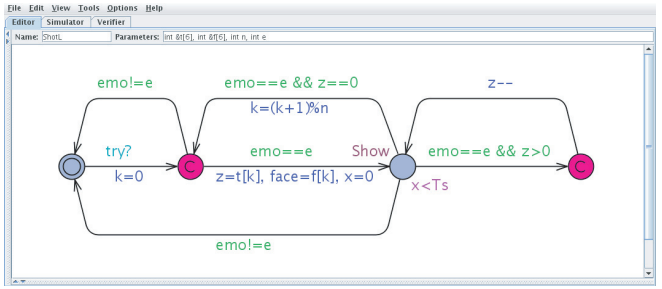


図 11　ループ型ショットのモデル（ShotL）
Fig. 11　Model of loop type shot (ShotL).

図 9 のフロー型ショットの表情アニメーションに対して，図 5 に示したように一連の顔画像を繰り返し表示する表情アニメーションをループ型ショットと呼ぶ．ループ型ショットの時間オートマトンによるモデルを**図 11** に示す．図 10 と図 11 の時間オートマトンの違いは，変数 `k` の更新方法だけある．また，表現モデルに含まれる M 個のショットは，フロー型かループ型のいずれかに属するものとする．したがって，図 10 と図 11 に示した 2 種類の時間オートマトンの引数を変えるだけで，すべてのショットのモデルが得られる．

5. フェースディスプレイのモデル検証

5.1 感性モデルの検証

複数の時間オートマトンによってフェースディスプレイをモデル化したことで，その検証についても分割して行うことができる．以下では，感性モデルとロボット TA の時間オートマトンのみを並列合成した実時間システムのモデルに対して，UPPAAL を用いて検証した論理式の一例を紹介する．ただし，図 4 に示したロボット TA の時間オートマトンを「Robot」とし，図 8 の感性モデルの時間オートマトンを「Kansei」とする．

(6)　E◇ (`Robot.Check && Kansei.K1`)
(7)　A□ not (`Robot.Fail && Kansei.K4`)
(8)　`Robot.Pass --> Kansei.K2`
(9)　A□ not deadlock

論理式 (6) は到達可能性であり，ロボット TA で画像を点検中であるとき，感性モデルが「恐怖」の心理状態になる状況が，いつかは起きることを検証する．論理式 (7) は安全性であ

り，ロボット TA が不合格と判定したとき，感性モデルでは絶対に「幸福」とはならないことを検証する．論理式 (8) は活性であり，ロボット TA が合格と判定したならば，感性モデルが「驚き」の心理状態に至ることを検証する．論理式 (9) も安全性の一種であり，感性モデルとロボット TA がデッドロックに陥らないことを検証する．ここで，ロボット TA における画像の点検時間の下限値 `Tc` と感性モデルにおける心理状態の持続時間 `Tm` が `Tm` > `Tc` の関係にあるとき，上記の論理式 (6) から (9) はすべて成立した．しかし，`Tm` ≤ `Tc` とすると論理式 (9) が成立せず，デッドロックに陥ることが判明した．したがって，システム全体がデッドロックに陥らないためには，事前に `Tc` の値を計測して，`Tm` > `Tc` となるように `Tm` の値を決めるか，`K1` からガードの `x<Tm` を外し，無表情な瞬きに対応する心理状態 `K0` と同様，`K1` に対応する表情アニメーションをループ型のショットとする必要がある．

5.2　表現モデルの検証

上記のロボット TA「Robot」と感性モデル「Kansei」のほか，感性モデルの心理状態 `K2` に対応したフロー型ショットの時間オートマトン「ShotF2」を並列合成した実時間システムのモデルに対して，UPPAAL を用いて検証した論理式の一例を以下に紹介する．

(10)　E◇ (`Robot.Idle && face==21`)

(11)　A□ not (`Robot.Fail && ShotF2.Show`)

(12)　`Robot.Pass --> face==21`

(13)　A□ not deadlock

論理式 (10) は到達可能性であり，フェースディスプレイは識別番号 21 の「驚き」の顔画像を表示しているが，ロボット TA のロケーションは「`Idle`」にある状況を検証している．すなわち，ロボット TA は画像の点検を終了しているが，表情アニメーションの実行には時間を要するため，ロボット TA の状態に対して顔画像の表示には遅れを生じる．それが容認できる事態であるか否かを評価できることも，時間オートマトンを用いたモデル検証の利点である．論理式 (11) は安全性であり，ロボット TA における判定結果が不合格ならば，上記のフロー型ショット「ShotF2」は絶対に実行されないことを検証する．論理式 (12) は活性であり，ロボット TA が合格と判定したならば，識別番号 21 の「驚き」の顔画像が表示されることを検証する．学生によるロボット TA の操作の間隔 `Tu` が心理状態の持続時間 `Tm` よりも十分に大きい場合，論理式 (12) は成立するが，`Tu` < `Tm` とすると論理式 (12) は成立しない．同様の検証を行った結果，学生が素早くロボット TA を操作した場合，表情アニメーションに含まれるいくつかの顔画像は表示されないことが判明した．

論理式 (13) は論理式 (9) と同じであり，システム全体がデッドロックに陥らないことを検証する．ロボット TA と感性モデルの時間オートマトンに加え，1 つのショットの時間オートマトンを並列合成した場合，UPPAAL により論理式 (14) の検証に要する時間は数秒であった．したがって，計 8 個のショットを個別に検証すれば，所要時間の合計は 1 分程度となる．一方，すべてのショットを並列合成した場合，論理式 (14) の検証には約 20 時間以上を要した．これより，分割可能な複数の時間オートマトンにより構成されたフェースディスプレイのモデルは，モデル検証に要する時間の節約に有効であることが確認できた．

6.　フェースディスプレイとロボット TA の実装と評価

提案手法の具体例として取り上げたフェースディスプレイとロボット TA は，Java により並行プログラムとして実装した．その並行プログラムの UML（Unified Modeling Language）によるクラス図を**図 12** に示す．ロボット TA とフェースディスプレイのプログラムは，それぞれパッケージ「robotTA」と「faceDisplay」にまとめてある．ただし，パッケージ「robotTA」ではメイン・メソッドを含む「RoboTA」以外のクラスは省略している．

パッケージ「faceDisplay」において，クラス「FaceDisplay」は図 8 の感性モデルに基づき実装されており，そのオブジェクトは「RoboTA」のメイン・メソッドからスレッドとして実行される．「Event」はイベントのクラスであり，ロボット TA が発生したイベントを変数 `event` に保持している．ここで，ロボット TA はメソッド `setEv(event)` で `event` に値を設定し，フェースディスプレイは `getEv()` により `event` の値を読み取る．

図 10 と図 11 に示したフロー型とループ型のショットは，クラス「FaceDisplay」のメソッドの `shotF()` と `shotL()` により実装されている．図 10 のフロー型ショットを実装し

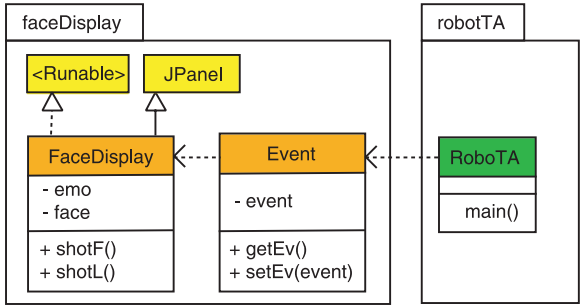


図 12　フェースディスプレイとロボット TA のクラス図
Fig. 12　Class diagram of face display and robot TA.

```
for(int k=0; k<n; k++){
  int z = t[k];
  face = f[k];
  while(z > 0 && emo == e){
    repaint();
    try{
      Thread.sleep(Ts);
    }
    catch(InterruptedException ee){};
    z--;
  }
  if(emo != e) break;
}
```

図 13　フロー型ショットのプログラム
Fig. 13　Program of flow type shot.

たプログラムの擬似コードを**図 13** に示す．図 13 のプログラムでは，`for` 文内において，`k` 番目の顔画像の表示時間 `t[k]` が変数 `z` に設定され，その顔画像の識別番号 `f[k]` が変数 `face` に設定される．次に，`while` 文内において，変数 `face` に設定された顔画像がメソッド `repaint()` によって表示される．ここで，1 枚の顔画像が表示される単位時間は `Ts` である．したがって，変数 `face` に設定された 1 枚の顔画像の表示時間は最長で `t[k]` × `Ts` となる．ただし，途中で `emo!=e` となると，`break` によって `for` 文から抜け，ショットの処理は終了する．図 10 の時間オートマトンと図 13 のプログラムを比較すると，両者の構造はよく対応しているため，時間オートマトンに基づくプログラムの実装や保守は容易である．

最後に，学生 11 名を被験者としたアンケート調査により，開発したフェースディスプレイがおよぼす心理的な影響を評価した．その結果，従来のフェースディスプレイを搭載していないロボット TA と比較し，全員がロボット TA に「顔」があった方が良いと答えた．

7. おわりに

本稿では，時間オートマトンを用いたフェースディスプレイの汎用的な設計と検証の手法を提案した．また，画像処理のプログラミング実習の教育支援システムであるロボット TA を具体例とし，提案したフェースディスプレイの設計と検証の手法について説明した．

フェースディスプレイの開発では，対象システムと同期したシナリオの設計や，対象システムの状態と表情アニメーションのタイミングの調整が重要となる．提案したフェースディスプレイの設計と検証の手法によれば，シナリオと表情アニメーションの設計を独立に行えるほか，時間オートマトンのモデル検証ツールである UPPAAL を用いて，フェースディスプレイと対象システムの挙動の時間的な整合性が検証できる．また，フェースディスプレイの開発に先行して，そのモデル化と検証を行うことで，フェースディスプレイのプログラムを機械的に実装できるほか，将来の保守や拡張も容易になることが期待される．さらに，時間オートマトンの遷移規則に基づき顔画像を表示する技法は，動画像や本格的な CG アニメーションに比べて，開発コストが廉価であり，コンピュータに与える負荷も小さい．このため，小型のパソコン上で実行させる様々なアプリケーションに応用可能である．

今後の課題として，フェースディスプレイに学習機能を付加することがあげられる．たとえば，学習支援システムなど操作の目標が明確な場合，感性モデルにおけるロケーションの遷移規則に強化学習の手法[20] を導入することが考えられる．また，フェースディスプレイが学生に与える影響については，より大規模な調査と統計的な分析が必要である．

参 考 文 献

1) 吉川左紀子，益谷　真，中村　真：顔と心：顔の心理学入門，サイエンス社 (1993).
2) 長谷川修，森島繁生，金子正秀：「顔」の情報処理，電子情報通信学会論文誌 D-II, Vol.J80-D-II, No.8, pp.2047–2065 (1997).
3) Chernoff, H.: The use of faces to represent points in k-dimensional space graphically, *Journal of the American Statistical Association*, Vol.68, No.342, pp.361–368 (1973).
4) 野口典正，安居院猛，中島正之：表情アニメーションによる多変量データ表示システム，電子情報通信学会論文誌 D，Vol.J70-D, No.11, pp.2177–2181 (1987).
5) Pflughoeft, K.A., Zahedi, F.M. and Soofi, E.: Data visualization using figural animation, *Proc. Americas Conference on Information Systems*, pp.1701–1705 (1990).
6) Wyatt, R.: Face charts: A better method for visualizing complicated data, *Proc. IADIS International Conference Computer Graphics and Visualization*, pp.51–58 (2008).
7) 児玉哲彦，安村通晃：表情の表現を含む手話アニメーションの試作，情報処理学会研究報告，Vol. 2003-HI-103, No.47, pp.23–29 (2003).
8) 美濃導彦，西田正吾：情報メディア工学，オーム社 (1999).
9) エクマン，P.，フリーセン，W.V.（著），工藤　力（訳編）：表情分析入門，誠信書房 (1987).
10) 平山高嗣，川嶋宏彰，西山正紘，松山隆司：表情譜：顔パーツ間のタイミング構造に基づく表情の記述，ヒューマンインタフェース学会論文誌，Vol.9, No.2, pp.201–211 (2007).
11) 岡野浩三：上位設計におけるシステムの振る舞い検証技術，システム/制御/情報，

Vol.52, No.9, pp.328–333 (2008).
12) Alur, R. and Dill, D.L.: A theory of timed automata, *Journal of Theoretical Computer Science*, Vol.126, No.2, pp.183–235 (1994).
13) Clarke, E.M.J., Grumberg, O. and Peled, D.A.: *Model Checking*, The MIT Press (1999).
14) Behrmann, G., David, A. and Larsen, K.G.: A tutorial on uppaal (2004). http://www.uppaal.com
15) 高橋孝一：モデル検証入門，システム検証の科学技術予稿集，pp.3–13 (2004).
16) 島川博光：支援システム導入によるプログラミング教育改善の試み：実授業への適用結果を中心に，システム/制御/情報，Vol.53, No.11, pp.453–458 (2009).
17) 大坊郁夫：しぐさのコミュニケーション：人は親しみをどう伝えあうか，サイエンス社 (1998).
18) Oztop, E., Kawato, M. and Arbib, M.: Mirror neurons and imitation: A computationally guided review, *Neural Networks*, Vol.19, No.3, pp.254–271 (2006).
19) 田川聖治，高橋佑輔：時間オートマトンによるフェースディスプレイのモデル化と検証，第 22 回自律分散システム・シンポジウム資料，pp.309–314 (2010).
20) Richard, S.S. and Barto, A.G.: *Reinforcement Learning*, The MIT Press (1998).

(平成 22 年 2 月 2 日受付)
(平成 22 年 4 月 1 日再受付)
(平成 22 年 4 月 27 日採録)

田川　聖治（正会員）

1962 年生．1991 年神戸大学大学院工学研究科修了．神戸大学工学部助手，助教授を経て，現在，近畿大学理工学部教授．博士（工学）．進化的計算とその応用に関する研究，および教育支援システムの開発に従事．モデル検証とシステム設計に興味を持つ．電気学会，計測自動制御学会，システム制御情報学会，システム設計検証技術研究会，IEEE 各会員．

高橋　佑輔

1988 年生．株式会社ケイ・オプティコム勤務．近畿大学理工学部情報学科在学中，ロボット TA とフェースディスプレイの開発に取り組む．

加藤　　暢（正会員）

1965 年生．1997 年岡山大学大学院自然科学研究科修了．日本学術振興会特別研究員として並行計算理論に関する研究に従事．2000 年より近畿大学理工学部講師．博士（工学）．並行論理型言語の意味論，モデル検証，およびプロセス代数に関する研究に従事．電子情報通信学会，ソフトウェア科学会各会員．