インタラクティブシステム設計のための時区間表現によるスクリプト記述法

間瀬健二[†] クラウディオ S. ピンヤネス[††] アーロン F. ボビック[††]

没入感や臨場感があるインタラクティブなシステムでは，多数のセンサを組み合わせてセンサ統合したマルチモーダルなインタフェースがよく用いられる．このようなインタフェースを実現する際に，イベントの発生や表示処理などに関する時間的な自由度を持ち，多重のイベントを記述しやすいプログラミングパラダイムが必要である．本文では，インターバルスクリプトと呼ぶ時区間の概念を導入したイベントや動作の記述法と，それに基づく制御法を提案する．インタラクションの展開は，システムデザイナがセンサや動作のためのルーチンプログラムを時区間に対応付けて，さらにその時区間の並びを定義することによって記述される．本手法に基づく *SingSong* と呼ぶ実験システムを制作し，その2つの場面を取り上げて，スクリプトの記述法を詳しく述べる．

Scripting Method Based on Temporal Intervals for Designing Interactive Systems

KENJI MASE,[†] CLAUDIO S. PINHANEZ[††] and AARON F. BOBICK[††]

Multimodal interfaces are often used in immersive interactive systems, which integrates many sensors and actuators, to provide situated and robust interactions. This paper presents a novel programming paradigm, *Interval Scripts*, which has the temporal flexibility in designing event handling and actuator control. In such interval scripts, the system's designer attaches sensor and actuator routines to time intervals and the development of the interaction is defined by local precedence relations provided by the designer of the system. Interval scripts have the potential to express more clearly the complex structure of time events in immersive environments where the user's actions span non-punctual intervals of time, making the use of event-based interaction design quite difficult. This paper examines in detail two scenes implemented in a prototype interactive system called *SingSong*.

1. はじめに

没入感や臨場感を与えるインタラクティブなシステムにおいては，インタラクションの流れの制御の良し悪しが，利用者のシステムに対する評価に大きく影響する．ユーザに快適なインタフェースを提供するためのジェスチャなどを使ったマルチモーダルインタフェースの研究がさかんに行われているが，インタラクションの流れの制御や，インタラクションを記述するスクリプトについての直接の議論は少ない[1]．わずかに，インタラクションの展開に演劇理論を適用する提案や[2]，コンピュータどうしのインタラクションの一例であるソフトウェアエージェント間での通信プロトコルとしてのスクリプト言語仕様がいくつか提案されている程度である[3),4]．一方，マルチメディアシステムではメディアの表現に時間がかかるため，流れの制御が重要であり，オーサリングシステムにおけるメディアオブジェクトの時間的配置の問題が議論されているが[5]，ユーザとのインタラクションについては十分に表現できていない．

従来型のインタフェースデザインでは，つねにセンサ装置からのデータを処理して，その結果をアプリケーションで利用するためのプログラミングが行われてきた．しかし，マルチモーダルなインタフェースを設計する際には，情報統合によりメッセージ理解の頑健性を達成することなどを目的として，複数のセンサを状況に従って活性化させる場合がある[1]．福本ら[6]は手動作センサ，および音声コマンドセンサを常時活性化しながら同期をとることで，1つのマルチモーダ

† 株式会社 ATR 知能映像通信研究所
ATR Media Integration & Communications Research Laboratories
†† MIT メディア研究所
MIT Media Laboratory

ルインタフェースを実現している．

マルチモーダルインタフェースの状況依存性などの利点を活用しようとすると，これらのセンサを時間的・空間的に複雑に組み合わせる必要が生ずる．しかしながら，センサの状態をつねにチェックして，キューにためて処理をする従来型のイベントハンドリングでは，状況に応じた解釈をシステムにさせるために，プログラミングが困難になる場面が生じる．たとえば，マウスボタンのクリックで誤ってダブルクリックに解釈されることがあるように，あるジェスチャで1つのコマンドを指定したつもりだったのに複数のイベントがキューイングされてしまうと，困ったことが生ずる．

すなわち，高度なマルチモーダルインタフェースを設計するためには，イベント発生や処理に関する時間的自由度を持ち，多重のイベントを記述しやすい高次のプログラミングパラダイムが必要である．とりわけ，ジェスチャのように1つのメッセージ伝達のためにある時間幅を必要とし，その長さが個人や状況によって変動するものでは，そのインタラクション記述と解釈に時間的な自由度を与えることが必要になる．すなわち，本論文の基本的な目的に，明示的な時刻の参照なしにスクリプトを記述できるようにすることがある．今後ジェスチャ認識や音声認識などの技術が進むにつれ，個別の動作単位に対する検出処理や認識処理の論理センサがソフトウェアツールキットとして整備されてくると考えられる．これらの論理センサが充実してきたとき，それらをルーチンプログラムとして組み合わせる際のプログラミング手法が必要である．

そこで本論文では，ユーザとシステムのインタラクションをスクリプト表現で記述し，そのスクリプトに従ってインタラクションを管理する方法を提案する．具体的には，インターバルスクリプト（Interval Script）と呼ぶ時区間[7]の概念を利用したイベントや動作の制御記述法を提案する．従来のイベント制御法においてイベントは“一瞬の点”のように解釈されていたが，インターバルスクリプトでは時間幅を持った“区間”まで解釈を拡張して扱うことができるようになる．さらに，そのコンセプトに基づいた“*SingSong*”という，コンピュータ俳優が登場する臨場感あるインタラクティブシステムの設計制作の実例により，スクリプト記述法の実際をしめす．予備実験によって，ジェスチャ動作の長さやタイミングの記述に関する緩やかな制約がもたらす効果を検証する．

以下，2章では関連する研究を概観する．そして，3章でインターバルスクリプトの記述法とそれを動かすためのインタラクションマネージャの構造を提案し，4章で具体的なアプリケーションシステム *SingSong* を例にとりシステムの構成とスクリプトの具体的な記述例を説明する．最後に5章でスクリプト記述について考察し，今後の課題を述べる．

2. 関連研究

佐藤ら[8]はリアクティブなアニメーション記述のために，イベントと動作の関係を宣言的に記述する方法として，時限タプルの考えを導入した．時限タプル間の時間関係演算は本論文で利用する時区間演算のサブセットと考えられるが，タプル間の条件文表現が可能となっている．なおインタラクティブでもリアクティブでもないバッチ処理で生成するアニメーションは，時空間上のオブジェクトどうしの関係として記述することが有効で，時空間制約（たとえば，文献9)）が有効な方法として用いられている．

また本論文で紹介するスクリプト記述法は，映像や音楽を素材とするマルチメディアタイトルのオーサリングと実行時の制御に関する方法にも密接に関連する．マルチメディアオブジェクトや操作イベントもやはり表現やタイミングに時間的な曖昧さを有しているため，その記述法がいろいろと提案されており，文献5) は時空間メディアの整形システムにおける課題と要求項目についてこれまで提案されている手法を分類整理している．たとえば，TEX の glue モデルを時空間オブジェクトに拡張適用してコンパイル時の時空間における配置の定義を容易にしている方法[10]や，simultanious や before などの時間制約の記述ができる Firefly というシステム[5]がある．

本論文では，ユーザとシステムの相互作用のための外部入力として，ジェスチャなどのセンサからのイベントを用いることを前提としたインタラクティブなシステムの記述法について議論している．ジェスチャなどの場合はイベントの認識のタイミングが変動するので，イベントにも時区間を導入して記述する方法を本論文では提案する．またそのことによって，イベントと動作が同じ表現で記述でき，時区間論理を導入することが可能となった．

3. インターバルスクリプト

インターバルスクリプト（Interval Script）は各種イベントに**時区間**（temporal interval：以降，単に区間と呼ぶこともある）の概念を導入したインタラクティブシステム記述のためのスクリプトである．システム側の表出動作あるいはセンサ動作などのイベントを，不定長の時間長さを有する区間単位に対応付け

て，動作制御を記述する方法である．すなわち，時区間どうしの関係を明示的に表してインラクションを記述する．以下，本文では，CG キャラクタの動作や音の生成など個々の論理的なシステム動作単位を**動作子**（**actuator**），ジェスチャイベント入力などのセンサ動作単位を**感覚子**（**sensor**）と呼ぶことにする．またこれらをまとめて**外部結合子**（**external**）と呼ぶ．インターバルスクリプトの記述法を用いると，インタラクティブシステムのデザイナは，まずいろいろな動作子や感覚子を実現するプログラム（以下，ルーチンと呼ぶ）を用意し，それぞれを1つの時区間と結合させ，次に時区間どうしの時間的な関係を記述する．ここで時間的な関係とは，2つの時区間が，「連続して」いるとか「重なって」いるとかの表現である．時区間の表現では，区間だけがあり，開始時刻，終了時刻，および時間長さなどの，絶対的な時間に関する表現が必要でないことが特徴である．

3.1 Allen の時区間代数

2 つの時区間の関係を定義するために，我々は Allen[7] による，時区間に基づく時間表現モデルを導入する．**図1** に示すように，このモデルでは2つの区間の関係が13種のプリミティブな関係ですべて言い尽くされる．すなわち，実世界において2つの事象があるとき，その事象を時区間であらわすと，事象の時間的な関係はこれらのプリミティブな関係のどれかに当てはまる．たとえば，車を運転するという動作はエンジンがかかっているとき起すことができる．これを Allen の時間表現モデルのプリミティブを使って表現すると，「車の“運転区間”」と「エンジン状態“ON 区間”」との関係は，「運転区間は，ON 区間（中）に，“始まる（START）”か，または“終わる（FINISH）”か，あるいは“間（DURING）”に生じるか，“等しい（EQUAL）”」と記述できる．すなわち，運転区間と ON 区間の関係は {**START**, **DURING**, **FINISH**, **EQUAL**} のいずれかで表現できる．もちろん，実際には，これらのうちの1つの関係だけが実際に起こる．なお，以降この時区間の関係を記述する際には (区間 A, **START** or **DURING**, 区間 B) のような疑似コード記法を使うことがある．

時区間の関係を Allen 式に記述すると，2つの区間の関係が全集合に波及するメカニズムを利用することができる．たとえば (区間 A, **BEFORE**, 区間 B) の関係と，(区間 B, **BEFORE**, 区間 C) の関係があるとき，時区間代数を適用すると，(区間 A, **BEFORE**, 区間 C) が成立するという結果を得ることができる．これを利用すると，時区間の二項関係を記述した集合から時区間全体の前後関係を知ることができ，Pinhanez and Bobick はこの計算を高速に行う PNF 計算法を提案した[11]．本文では PNF 計算法をベースにしたインタラクションマネージャを開発し，インターバルスクリプト記述法を実現した．

3.2 インターバルスクリプトの制御

インターバルスクリプトを入力とするインタラクティブシステムの制御部（Interval Script Controller）の内部構造を**図2** に示す．図にしめすように，時区間関係テーブル（Interval Relationship Table），時区間の PNF 状態テーブル（Interval PNF States Table），PNF 計算を行うインタラクションマネージャ（Interaction Manager），および外部結合子テーブルから構成される．デザイナは，スクリプトエディタにより，インターバルスクリプト本体である時区間の関

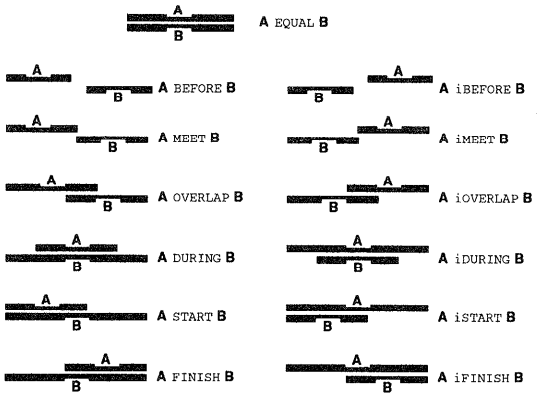


図1 Allen の時区間代数：2つの区間における13個のプリミティブな関係[7]

Fig. 1 Allen's interval algebra: the possible 13 primitive time relationships between 2 intervals [7].

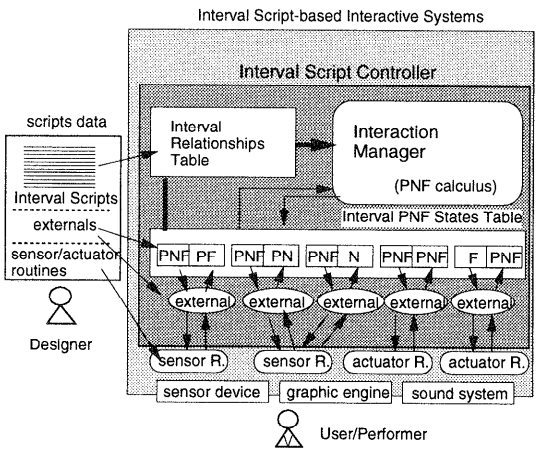


図2 インターバルスクリプト制御部のダイアグラム

Fig. 2 Diagram of the interval script controller.

係，感覚子および動作子ルーチン，および外部結合子からなるスクリプトデータを作成，記述して，この制御部へ入力する．

3.3 PNF 計算法

PNF 計算法は，ある瞬間において個々の時区間に過去（P），現在（N），未来（F）の3要素からなる状態を割り当てる操作を基本にしている．たとえば，2つの時区間に(区間 A, MEET, 区間 B) という関係があったとき，もし区間 A が過去（P）の事象であれば，区間 B の状態は“過去または現在（PN）”であることが分かる．2つの区間は MEET によって接続しているので，区間 A が過去（P）となってしまった時点で，区間 B が未来（F）という状態はありえない．このように時区間代数で関係付けられた時区間どうしは，事象の過去・現在・未来の状態を伝搬することによって，たとえば，どの区間がこれから起こりうるかということを調べることができる．なお，そのほかに，P と N と F の OR の組合せで，PN，NF，PF，PNF の状態がある．たとえば PN はある事象が過去に開始したが終わったかどうか不定の状態を，PNF は事象の状態を決める情報がないことを示す．

PNF 計算法は，どれかの時区間の PNF の状態をデバイスのルーチンなどから入手できることを前提としている．この計算法における PNF 制約アルゴリズムを使うと，この限られた情報から他の動作子などの区間まで PNF 状態を伝搬させることができる．そうして，状態が現在（N）になっている動作子があればそれを働かせ，状態が過去（P）の動作子は停止させるように働く．PNF 計算法の詳細は文献 11) にゆずる．

3.4 実世界との結合

感覚子や動作子という外部結合子は，ソフトウェアプログラムツールやデバイスとして提供される異なるルーチンを実際に動かすために，内部構造に論理的にマッピングする概念である．外部結合子によってデザイナの用意したルーチンと区間構造を結び付け，シームレスにカプセル化する役目を果たす．後で例で示すように1つの感覚子（あるいは動作子）ルーチンは，外部結合子によって定義された1つ以上の区間に結合される．

3.4.1 感 覚 子

インタラクティブな環境では，感覚子はセレクタ（リストからの選択），ロケータ（位置の指定），バリュエータ（量の指定）などの役割を果たす[6]．本論文のシステムでは，現在のところ2値のセレクタ感覚子のみを実現している．この感覚子は何かが起こっているかいないかを検出することができる．なお，すべての感覚子は少なくとも次の2つの区間に関連付けをする必要がある．感覚子がアクティブになっている状態を示す activity 区間と，感覚子が目的としている状態を検出している状態に対応付けられる event 区間である．

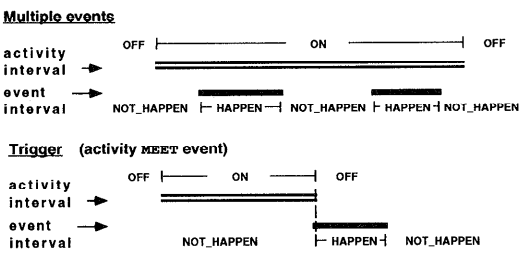


図3 感覚子に関係付けられた区間の2つの例
Fig. 3 Intervals associated with a sensor, in two different configurations.

2 値の選択を行うセレクタ感覚子の場合には，デザイナはスイッチングのコマンドである ON, OFF, reset 区間を入力として受け付け，HAPPENing, NOT-HAPPENing, UNKNOWN の3つのうちのどれかを値として返すようなルーチンを用意する．いまちょうど activity 区間が働いているときならば，インタラクションマネージャはデザイナのルーチンに ON コマンドを送り，それ以外のときは OFF コマンドを送るように動作する．そして，ルーチンがイベントを検出すると event 区間に HAPPENing を返す．

図3 は2つの異なる場合について感覚子を区間に関連付けた場合の図式表現である．複数イベントの例では，1つの activity 区間の間に event 区間が2度割り付けられている．また，図3の下の図は，感覚子の特別な場合であるトリガ感覚子を定義している．このとき event 区間を1個だけ用いる．トリガ感覚子を定義するには，あるトリガとして用いたい感覚子ルーチンに対して (activity 区間, MEET or BEFORE, event 区間) の関係をスクリプトに追加記述することで達成される．

3.4.2 動 作 子

動作子に関しては，デザイナは，ON, OFF, reset 区間の各コマンドを入力として受け付けて NOT-DOING, DOING, DONE という状態のいずれかを出力するルーチンを準備する必要がある．このように，ルーチンからのフィードバックを使えるようにしておくことが重要である．というのは，実世界でのいろいろな動作はそれ自体が持っている優先度やタイミングがあって，いつもデザイナの思いどおりに動いてくれるとは限らないからである．たとえば，ある場面で音を鳴らすようにスクリプトを記述したとしても，ネットワーク

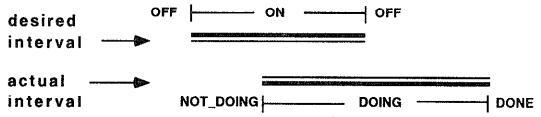


図4 動作子に関係付けられた区間の例
Fig. 4 Intervals associated with an actuator.

の問題で音を鳴らすタイミングが遅れてしまったり，ほかの動作子が必要なハードウェアの資源を掴んでしまって音を鳴らせないことが起こりうるからである．

本論文では，動作子が本来持っているこのような性質を，**desired 区間**と **actual 区間**という 2 個の区間動作子に関連付けることによって解決している．図 4 は区間と動作子ルーチンの関係を図示している．**desired** 区間の間，インタラクションマネージャは **ON** コマンドをルーチンに送る．そして，動作子が働いて **DOING** のメッセージが **actual** 区間に戻ってきたら，それは，そのとき動作が確かに発生していることを示すものである．さらに，**desired** 区間が終わったときには，インタラクションマネージャは **OFF** コマンドを動作子ルーチンに送りはじめる．しかしながら **actual** 区間の終了はそのルーチン自身によって決められる．その時点はルーチンが **OFF** メッセージを受け取る前か，同時か，あるいは後かもしれない．それは制御しているデバイスの特性に依存することになる．2 個の区間を用いることによってこのようなデバイスの特性に依存しないスクリプト記述をすることができる．

3.4.3 タ イ マ

本論文での提案の基本的な目的は，明示的な時刻の参照なしにスクリプトを記述することであるが，しばしばある動作やセンシングの時間長に絶対的な制約を加えたいときがある．そこで，非常に特殊な場合として，時間長を属性として持つ**タイマ動作子**を導入する．スクリプトの記述は一般の動作子と同様に **desired** 区間と **actual** 区間を定義し，タイマの時間長を指定する．**desired** 区間が呼ばれるとタイマルーチンがスタートし，設定時間長が経過するとタイマルーチンは **DONE** を出力し **actual** 区間を終了させるように働く．すなわち，**desired** 区間はタイマ動作子をオン・オフし，**actual** 区間の終了がタイマ動作子の終了を他の区間に伝える働きをする．

4. インターバルスクリプトによる応用システムのデザイン

SingSong と呼ぶ物語ベースのインタラクティブシステムを作成して，本文で提案した手法を検証した．図 **5** に基本的なシステム構成をしめす．すなわち，大型スクリーンに 4 人のコンピュータグラフィックス（CG）で生成したキャラクタを表示させ，そのキャラクタたちに音楽を演奏させるという設定である．音楽生成には MIDI シンセサイザを用いる．スクリーンの前に立った人間の演技者のジェスチャを認識するためビデオカメラをスクリーン上部に設置してある．ジェスチャの認識には Pfinder と呼ぶ頭，両手，足，胴の位置を検出するプログラムを用いている[12]．

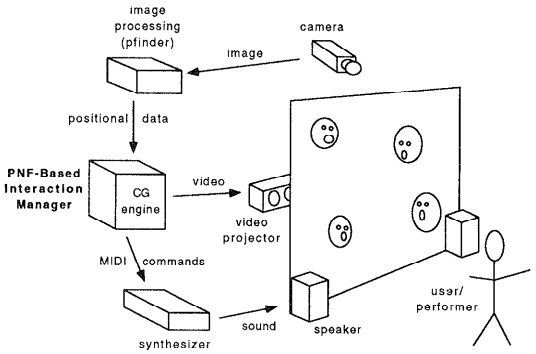


図5 *SingSong* のシステム構成
Fig. 5 The basic setup of *SingSong*.

表1 *SingSong* の基本ストーリー
Table 1 The basic story of *SingSong*.

場面は歌手たちのおしゃべりから始まる．
指揮者が入ってきて，腕をあげて歌手たちを静粛にさせる．3 人の歌手は静かにするが，歌手#1 はいうことを聞かない．もう一度言うと悪態をついてやっと静かになる．
次は，音合わせである．音叉が現れ，指揮者が 1 人ずつ指し示して音叉をたたくようにすると，音合わせの音がなる．歌手はそれにあわせて同じ音をだす．音合わせを順番にやっていくが，しかし，また問題の歌手#1 がいる．何度やっても音があわない．しかたなしに指揮者が膝を折って懇願するとやっということを聞いて音合わせが終わった．
準備が整ったら，手をあげて演奏開始である．指揮者の指揮にあわせて調子をとる．演奏が終わると，拍手が聞こえ，指揮者の礼にあわせて歌手も礼をしておしまい．指揮者が退場しようとすると，またあの歌手#1 がぶつぶつ文句を言う．

SingSong は，**表 1** に示すようなストーリーであり，そこではユーザは 4 人の CG キャラクタ歌手による合唱の指揮をする役になる．すべてのインタラクションはノンバーバルであり，ユーザのジェスチャと CG キャラクタの出す音と動作が用いられている．

SingSong は一般のユーザが体験するシステムとしても，あるいは俳優ユーザがコンピュータ劇を演ずる場としても，楽しめるように設計した．後者に関しては文献 13) でも述べられているように，人間の演技者にコンピュータで生成した仲間を与え，ユーザはシステムに単に反応して動作するのではなく，あらかじめ

決められた台本に従って，協力して演ずることができるようになっている．この演技モードとユーザによる体験モードはシームレスになっており，たとえば，一般ユーザは俳優の生の演技を見ながらストーリーを理解し，自分も気軽にストーリーの中に入っていくことができる．

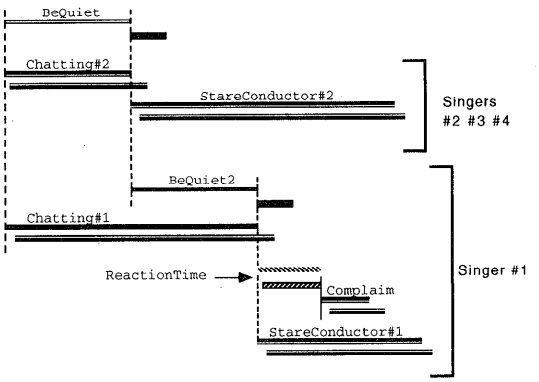
以下，2つの場面を取り上げて，具体的にインターバルスクリプトを使ってどのようにインタラクションを記述するかを説明する．

4.1 場面1のスクリプト

場面1は以下に示すいろいろな外部結合子を用いて記述されている．**おしゃべり（Chatting）**とよぶ動作子は，おしゃべりをまねる音と口の動きを生成し，それぞれの歌手のおしゃべり動作を制御する．**静かに（BeQuiet）**と**静かに2（BeQuiet2）**は，ユーザが両手を頭の上にあげたときに反応するトリガ感覚子である．**指揮者に注意（StareConductor）**という動作子により，歌手の視線が指揮者を追うようにしている．指揮者のジェスチャと歌手#1の文句の間の一瞬の間（ま）を記述するためには，**反応時間（ReactionTime）**という3秒のタイマ動作子を定義した．さらに，歌手#1だけに使われる**文句（Complain）**という動作子を定義し，文句をいっている動作になるような音とグラフィックスを制御する．

図6に場面1における時区間の関係を図示し，具体的なスクリプトの記述をしめす．図中では歌手の#1と#2を例にとって記述してあるが，歌手#3，#4は歌手#2と同じ動作をするので省略してある．なお，スクリプトは実際には C++ のプログラムコードの中で表現してあるので，記法として括弧やコンマが使われているが，簡単のためここでは省略した．また，中央にボールド体で記述してある関係プリミティブが複数あるのはORの関係であることを示している．外部結合子間の関係を完全に記述してあることに注目いただきたい．

まず，Chatting 動作子の desired 区間と BeQuiet 感覚子の activity 区間が同時に始まるように定義する．図6のスクリプトの第1，第2行にあるように，BeQuiet 感覚子の activity 区間と，Chatting#1 動作子の desired 区間および Chatting#2 動作子の desired 区間との関係を，START または EQUAL または iSTART であると記述すればよい．言い換えると (BeQuiet.activity 区間，START or EQUAL or iSTART，Chatting#1.desired 区間) (BeQuiet.activity 区間, START or EQUAL or iSTART, Chatting#2.desired 区間) という関係で記述される．この関係は図6の図式表現において，3つの区間の開始点が点線で並んでいることでも知ることができる．



BeQuiet activity	START EQUAL iSTART	Chatting#1 desired
BeQuiet activity	START EQUAL iSTART	Chatting#2 desired
Chatting#2 desired	MEET BEFORE	BeQuiet event
BeQuiet activity	MEET BEFORE	BeQuiet event
BeQuiet event	START EQUAL iSTART	StareConductor#2 desired
BeQuiet event	START EQUAL iSTART	BeQuiet2 activity
BeQuiet2 activity	MEET BEFORE	BeQuiet2 event
BeQuiet2 event	START EQUAL iSTART	ReactionTime desired
BeQuiet2 event	START EQUAL iSTART	StareConductor#1 desired
Reaction Time event	MEET	Complaim desired

図6 場面1における時間関係の図式表現とそのインタバールスクリプト

Fig. 6 Diagram of temporal relations and interval script corresponding to the first scene of *SingSong*.

次は BeQuiet 感覚子の event 区間にあたる BeQuiet.event 区間が Chatting#2 動作子の desired 区間を終了させるよう定義する．すなわち，歌手は指揮者が腕をあげたときにおしゃべりをやめるよう命令されることになる．BeQuiet.event 区間はさらに BeQuiet 感覚子の activity 区間を終了させて，この外部結合子がトリガ感覚子のように振る舞い，それ以降はセンサ機能をいったん停止させることができる．また，この event 区間は StareConductor#2.desired 区間を開始させる．この，区間を開始させたり終了させたりする記述には，図6に示すように，開始には START or EQUAL or iSTART を，終了には MEET or BEFORE の関係を用いる．ところで，歌手#1は指揮

者が腕を再度あげるまでおしゃべりをやめない．したがって，**BeQuiet.event** 区間は **Chatting#1** 動作子を終了させることも **StareConductor#1** 感覚子を起動させることもしない．そのかわりに（**BeQuiet.event** 区間，**START** or **EQUAL** or **iSTART**, **BeQuiet2.activity** 区間）というトリガ感覚子の関係で記述してある．

BeQuiet2 感覚子による **event** 区間の検出は，その感覚子を終了させ，**StareConductor#1** 動作子を開始させ，さらに **ReactionTime** タイマ動作子の **desired** 区間を開始させる．**Complain** 動作子を起動して，場面 1 は終了するが，そのきっかけは **ReactionTime.actual** 区間の終了による．

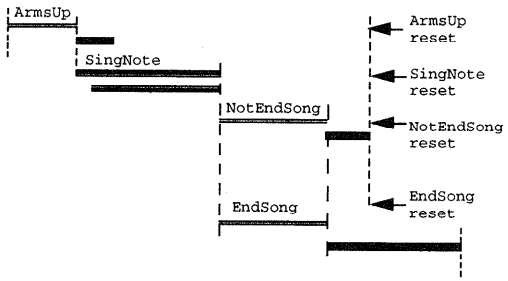
以上のように，すべての構造は時区間の間の時間的な相互前後関係で記述されており，明示的な時区間の長さを用いていない．この場面 1 は 1 つのイベントから複数のアクションが生じている非常に良い例である．

4.2 歌唱場面におけるスクリプト

SingSong における歌唱演奏の場面は，基本的に短いインタラクションのループになっている．歌手は指揮者の腕をあげるというトリガによって次の音を出すようになっている．**図 7** に時間的な図解とそれを定義しているスクリプトを示す．

個々の音の演奏は **ArmsUp** 感覚子の活性化（**activity** 区間）から始まる．**ArmsUp** 感覚子は指揮者の両腕があがったときにルーチンから **HAPPENing** の戻り値を受ける．**ArmsUp.event** 区間が開始すると，それに対応するセンサは止められ，**SingNote** 動作子が開始させられる．この制御は図 7 のスクリプトの最初の 2 行に書かれている．動作子はつねに **desired** 区間によってスタートさせられることに注意していただきたい．動作子の **actual** 区間の開始タイミングは，その動作子に対応する（表示や音生成をする）ルーチンの出力のしかたによって，決まっている．

すなわち，**SingNote.desired** 区間の間，**ON** コマンドが歌の次の音を出すルーチンに送られている．しかし **actual** 区間は，そのルーチンが確かにシンセサイザに MIDI の **NoteOn** コマンドを送って，はじめて有効になる．**SingNote.actual** 区間はシンセサイザが音を出し終わったときに，**SingNote.desired** 区間の終了の引き金となり，2 つの感覚子 **NotEndSong** および **EndSong** をスタートさせる．**NotEndSong** と **EndSong** は 2 つの相補的な感覚子で，曲のすべての音が歌われたかどうかをチェックしている．これらは全曲を歌うためのループの終了条件になっている．そして，これがいったん活性化されると，感覚子は，曲の進行にあわせてつねにアップデートしている内部共



ArmsUp activity	MEET BEFORE	ArmsUp event
ArmsUp event	START EQUAL iSTART	SingNote desired
SingNote actual	FINISH EQUAL iFINISH	SingNote desired
SingNote actual	MEET	NotEndSong activity
SingNote actual	MEET	EndSong activity
EndSong activity	MEET BEFORE	EndSong event
NotEndSong activity	MEET BEFORE	EndSong event
EndSong activity	MEET BEFORE	NotEndSong event
NotEndSong activity	MEET BEFORE	NotEndSong event
NotEndSong event	MEET BEFORE	ArmsUp reset
NotEndSong event	MEET BEFORE	SingNote reset
NotEndSong event	MEET BEFORE	NotEndSong reset
NotEndSong event	MEET BEFORE	EndSong reset

図 7 歌唱場面における時間関係の図式表現とそのインターバルスクリプト

Fig. 7 Diagram of temporal relations and interval script corresponding to the singing scene of *SingSong*.

通変数をチェックする．もし，曲の終わりに達したら，**EndSong.event** 区間が起こり，両方の感覚子を止め，次の場面を開始させる（この部分は図にはない）．

曲が終わりに達していなければ，**NotEndSong.event** 区間は両方の感覚子を止める．そして，関連付けられているすべての外部結合子に，特別な **reset** と呼ばれる区間を働かせる．**reset** 区間が働くとその外部結合子に関連付けられたすべての区間は，たとえいったん起こった区間も含めて，未決定状態にもどされる．たとえば，ある感覚子の **reset** 区間が働くと，その **activity** 区間と **event** 区間に加え **reset** 区間自身も未決定状態にもどされる．こうして，時間的に逆

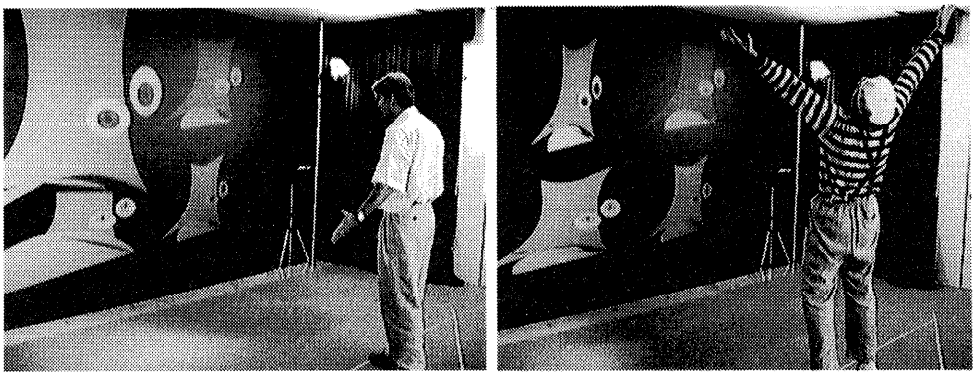


図8 *SingSong* の2つのシーン：左はユーザがシステムで遊んでいる様子．右は俳優による演技の様子．
Fig. 8 Two scenes from *SingSong*. The left picture shows a user playing with the system, while the right picture portrays a performance case.

戻りすることができるようになり，ループ終了の感覚子と組み合わせて，ループを構成することができるようになる．

このように *SingSong* のスクリプトは，時区間を関係付けるパラダイムを用いて，いろいろ異なる構成を実現している．

5. 試行実験と考察

SingSong は一般のユーザと俳優のような演技者の両者を想定しながら，表1に与えられたストーリーに従って普通に演技すると，3〜5分間で終わるように設計されている．**図8** はその様子を示しているが，左は一般ユーザが歌手#1（左上）の文句に反応している場面で，右はピエロをまねした俳優ユーザが合唱の指揮をとっている様子である．画面は170インチの大きさで，前に立つと70〜100度くらいの視野角があり，没入感を提供する．著者の1人であるPinhanezが演技者ユーザとして，また研究室内の数名に一般ユーザとなって実験に協力してもらった．

ストーリーとスクリプト記述の説明で示したように，**表2** のようなジェスチャがあり，ストーリーの場面に合わせて解釈される．

5.1 試行実験の観察

一般ユーザには，ストーリーの概略とコマンドとなるジェスチャを簡単に教え，演技者ユーザがまず演技をしてみせる．短いストーリーなので，一般ユーザはストーリーと使えるジェスチャをすぐに理解する．そこで，交替して，一般ユーザに実験してもらった．

実験を進めると，一般ユーザは指揮者の役回りをごく自然にするようになる．ユーザには，場面の詳細の進行に関してある程度自由度があり，たとえば，歌手の順番や回数（1回以上）は任意にできる．とくに指揮者として曲の演奏の場面に入ると，自由に体を振って，教えられたジェスチャ以外を試しながら演奏を楽しみ始めた．インタフェースの単純さと楽しい音楽の生成の喜びがうまくつながってシステム全体の利用が楽しい体験をもたらしていると考えられる．

表2 *SingSong* で定義されている主なジェスチャ
Table 2 Gestures used in *SingSong*.

ジェスチャ	システム動作	
立つ位置	歌手の目線	
両手をあげる	静粛に	演奏開始
片手を上にあげる	上段歌手指示（左右）	テンポ
片手を横に伸す	下段歌手指示（左右）	テンポ
手を振る	音叉をたたく	
しゃがむ	歌手#1への懇願	

しかしながら，本システムではストーリーの大きな流れに従って演技することが要求され，特定の場面全部や大事なシーンを飛ばすことはできない．したがって，ユーザがストーリーを無視して意図的に次へ進もうとしても，次の場面に移れないことがしばしば生じた．また，ジェスチャ認識誤りにより大事なシーンを実行できずに，やはり次の場面に移れないというケースがあった．これにはある条件でシステムが強制的に次の場面へ展開するようにスクリプトに工夫をする必要があった．また，大事なシーンのインタラクションには認識しやすいジェスチャを用いることが，デザイナに求められることが分かった．

5.2 マルチモーダルインタフェースと時区間によるスクリプトの記述

3，4章で述べたように，感覚子と動作子を時区間に結合する際には，すべて2種類の時区間を用いている．たとえば，動作子の制御には `desired` 区間と `actual` 区間を，感覚子には `activity` 区間と `event` 区間という2つの時区間を持ち込まなければならなかった．こ

れは，ジェスチャベースのインタラクションを前提としたからにほかならない．

具体的な例として，ジェスチャを指揮のタイミングとして使って，順番にコンピュータ内で音楽の拍子をとる場面がある．フィードバックの遅れやユーザのゆっくりした表現など何らかの問題で，ユーザ自身が知らないうちに，連続したジェスチャから複数のイベント（コマンド）をセンサが認識しているかもしれない．しかし，システムが基本的に必要としているのは1個のトリガであり，複数のトリガになりうるイベントが入力されることはスクリプトのデザイナにとってスクリプティングを複雑にするばかりである．うまくプログラミングしておかないと，いくつか音をとばして演奏を指揮することになってしまうからである．トリガ感覚子のように2種類の時区間を組み合わせることで，このような問題を簡単に解決することができた．

マルチモーダルインタフェースにおいては上記の状況依存性の利用のほかに複数種類の入力の統合がしばしば行われる．インターバルスクリプトは，ジェスチャと音声などの異なる感覚子ルーチンの組合せ統合によるマルチモーダルなインタラクションの記述も可能である．しかしながら，*SingSong* ではまず，上記のように状況依存性におけるジェスチャの解釈のあいまいさを確認するため，感覚子ルーチンについてはジェスチャのみを実現し考察した．

5.3 インターバルスクリプト記述法の評価

Allen の時区間代数と時区間表現を使うと，インタラクティブなスクリプトの設計に次のようなメリットを見いだすことができる．

- 区間の長さや，区間の両端の関係を細かく記述する必要がない．
- Allen による時間制約の伝搬アルゴリズムを利用することで，デザイナは最も関連する区間相互の関連を定義するだけでよいから，スクリプトが単純になる．
- 区間の接続関係の記述法は，あるストーリーのなかの多重経路や多重インタラクションの定義に利用できる．すでに述べたように，あるインタラクションの記述が時区間どうしの関係をはっきりと決定する．したがって，インターバルスクリプトは，節点が区間で枝が時間構造として定義されるようなグラフ構造と見なすこともできる．インターバルスクリプトはストーリーとインタラクションのある空間を記述していると考えることができる．

インターバルスクリプトのパラダイムはまだプリミティブな表現にとどまっており，低次の記述しかできていない．ある見方をすれば時間軸におけるイベントを記述するアッセンブリ言語ということもできる．しかしながら，実際にインターバルスクリプトを書き，システムを作成した経験によれば，同じような時区間の結合のパターンがいろいろに異なる状況で現れるのを見ることができる．たとえば，連続したイベント，条件付き分岐，ループなどがある．将来，これらのパターンを，より高次の外部結合子にまとめて定義することによって，言語としてのアブストラクションが可能になると考えられる．

5.2 節でふれた複数種類の感覚子ルーチンを使うマルチモーダルなイベントを処理するためには，たとえば，区間 A と区間 B がほぼ同時に発生したときに AND 条件分岐をするような，論理演算をする処理が必要である．これは時区間表記を越えた演算であり，インターバルスクリプトにより直接記述することはできていない．現在は，アドホックにそれを実現するための論理演算や計数をする感覚子ルーチンを用意している．この感覚子ルーチンは他の感覚子イベントからのトリガで内部状態を変更し，設定した式を満たせば **HAPPENing** 信号を発生するように働く．この例として，表1のストーリの中で「何度やっても音があわない．しかたなしに指揮者が膝を折って …」という場面で，4人の歌手全員の音合わせが終わったかどうかの判定の処理がある．同じく，この場面では全員の音合わせを済まさないと次の場面へ移れないため，ストーリーを熟知していないユーザは困惑してしまう．そこでタイマ動作子を使って一定時間内に先へ進まないと強制的に音合わせが終わったことにしている．タイマの代わりに音合わせの回数を計数して強制的に次の場面に進めることも可能である．

また，今回はセンサとして2値のセレクタ感覚子だけを実装してあり，バリュエータ感覚子やロケータ感覚子は実装できていない．セレクタ感覚子は，時区間の「ある/なし」状態と相性が良いため，比較的簡単に結合できている．その類推から連続値を入出力とするようなセンサに対して，多値レベルを状態とする時区間を導入することが考えられるが，それを処理できる時区間代数はなく，PNF 計算法もない．これを解決するには，やはり上に述べた高次の外部結合子などの導入が必要になると考えている．

なお，各時区間表現は感覚子や動作子の状態を2値表現で記憶している．したがって，多重のイベント等の組合せ状態は時区間相互の関係として簡単に記述できる．一方，状態遷移図を用いても同等のストーリー展

開を記述することは可能であるが，フレキシビリティが著しく劣ってしまう．ストーリーが複雑化するにつれ，修正をするときに状態の組合せをすべてチェックすることは現実的に不可能となる．

さらに，*SingSong* の設計から制作までは，非常に短期間の間に行うことができた．これはインターバルスクリプトのパラダイムが本質的に持つ記述性の良さによるところが大きいと考えられる．新しいルーチンを導入したり，インタラクションをテストするなかで，スクリプトの変更は次々に行われたが，これもインターバルスクリプトだったから可能だったといえる．言語仕様としてはまだ低次ではあるが，時間的関係を扱う記述として十分な構造化を達成しているといえよう．

6. おわりに

本文で提案しているインターバルスクリプトのパラダイムは，さらに一般的なツールと，インタラクティブなスクリプト制作のパラダイムのための第1ステップにすぎない．すなわち，インターバルスクリプトは時区間の概念を導入したことにより，従来のプログラミングパラダイムに対して高次な概念を達成したといえるが，考察でも述べたように時区間操作の記法の構造化に関してはアセンブラレベルのものである．今後はいくつかのスクリプトパターンを統合したより高次の構造化表現を導入する必要がある．

将来，プログラマでない一般のストーリー制作者でもスクリプト制作ができるようになるためには，インターバルスクリプトに適した図式化表現を考えて，GUI によるスクリプトエディタを提供することが必要である．本文では説明のためにタイムライン（線表表現）を用いて図式化表現を試みた．これらは単純な例であり，実は，本質的に多重でかつ無関係なものも多く含む動作の関係をうまく記述しながら，スクリプト全体を図式化する手段はまだない．また，このような図式表現が与えられたときに，必要なスクリプトを残らず生成できるかどうかはいまは不明である．

今後，ジェスチャのような全身体的な動作によるインタラクションは，従来のキーボードやマウスや音声などを用いた単調なインタフェースとはまったく異なったものになると考えられる．それは “ask & tell” のインタフェースから “action & feel” のような没入感ある環境におけるインタフェースへと移行していくだろう．その際に，本論文で提案しているようなセンサとシステム動作の制御を概略的に記述したり，インタラクティブな動作の直接的な表現を記述したりする考え方が重要性を増すだろう．本論文ではコンピュータ劇のような娯楽性のあるシステムを例にとって議論を進めてきた．しかし，インターバルスクリプトが目指すのは，より一般的なアプリケーションのインタラクションにこのような考え方を持ち込むことによって，複雑なシステムであっても分かりやすく達成感をもたらすインタフェースを実現することである．

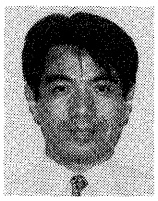
謝辞 本文における Interval Scripts のコンセプト案とインプリメンテーションの大部分は，第2著者の Pinhanez の contribution によるものである[14]．MIT-Japan プログラムの Starr 基金の援助によって，1996年夏，Pinhanez が ATR に滞在し，本研究が実施できたことに感謝する．また本研究遂行にあたり日頃ご支援をいただく（株）ATR 知能映像通信研究所酒井保良会長，中津良平社長ならびに有益な議論をしていただいた西本一志研究員はじめ第二研究室のメンバと土佐尚子研究員に感謝する．Pfinder プログラムは MIT Media 研究所の Alex Pentland 教授から提供を受けた．

参考文献

1) 宮里 勉，間瀬健二：ネットワーク利用者を支援するマルチモーダルヒューマンインタフェース，情報処理，Vol.38, No.1, pp.42–47 (1997).
2) Laurel, B.: *Computers as Theatre*, Addison-Wesley, Reading, MA (1991).
3) General Magic White Paper: *Telescript Technology*.
4) 西田豊明：ソフトウェアエージェント，人工知能学会誌，Vol.10, No.5, pp.704–711 (1995).
5) Buchanan, M.C. and Zellweger, P.T.: Automatic Temporal Layout Mechanisms, *ACM Multimedia '93*, CA, pp.341–350 (1993).
6) Fukumoto, M., Mase, K. and Suenaga, Y.: Finger-Pointer: Pointing Interface by Image Processing, *Comput. & Graphics.*, Vol.18, No.5, pp.633–642 (1994).
7) Allen, J.F.: Towards a general theory of action and time, *Artificial Intelligence*, Vol.23, pp.123–154 (1984).
8) 中村昌志，佐藤孝治，原田康徳：力学に基づいたリアクティブアニメーションの宣言的な記述，*WISS '95*, pp.93–101，ソフトウェア科学会 (1995).
9) Ngo, J.T. and Marks, J.: Spacetime Constraints Revisited, *SIGGRAPH 93*, Anaheim, CA, pp.343–350 (1993).
10) Hamakawa, R. and Rekimoto, J.: Object Composition and Playback Models for Handling Multimedia Data, *ACM Multimedia '93*, CA, pp.273–281 (1993).

11) Pinhanez, C.S. and Bobick, A.F.: PNF Calculus: A method for the representation and fast recognition of temporal structure, Technical Report 389, M.I.T. Media Laboratory Perceptual Computing Section (1996).
12) Wren, C.R., Azarbayejani, A., Darrell, T. and Pentland, A.: Pfinder: Real-Time Tracking of the Human Body, *IEEE trans. PAMI*, Vol.19, No.7, pp.780–785 (1997).
13) Pinhanez, C.S.: Computer theater, Technical Report 378, M.I.T. Media Laboratory Perceptual Computing Section (1996).
14) Pinhanez, C.S., Mase, K. and Bobick, A.F.: Interval Scripts: A Design Paradigm for Story-Based Interactive Systems, *CHI 97*, Atlanta, GA, pp.287–294 (1997).

(平成9年 6 月30日受付)
(平成9年12月 1 日採録)



間瀬 健二（正会員）

1979年名古屋大学工学部電気工学科卒業．1981年同大大学院修士（情報）課程修了．同年日本電信電話公社（現在NTT）入社．1988～1989年米国MITメディア研究所客員研究員．1995年より（株）ATR知能映像通信研究所第二研究室長．工学博士．コンピュータグラフィックス，画像処理とそのヒューマンインタフェース，コミュニケーション支援への応用が主な研究テーマ．訳書「ロボットビジョン」（朝倉書店，共訳）．IEEE，ACM，電子情報通信学会，日本情報考古学会各会員．

Claudio S. Pinhanez

Claudio Pinhanez is a Ph.D. candidate in Media Arts and Sciences at the Media Laboratory, the Massachusetts Institute of Technology (MIT). His research interests are action recognition, computer theater, high level computer vision, entertainment technology and AI. He received the B. Math and the M. Comp. Sc. degrees from University of Sao Paulo in 1985 and 1989 respectively. He was a lecturer at University of Sao Paulo, a researcher at Osaka University, and a visiting researcher at ATR, during 1987～1993, 1991～1992 and in summer 1996, respectively.

Aaron F. Bobick

Aaron Bobick is an Assistant Professor at the Media Laboratory of the Massachusetts Institute of Technology (MIT). His research interests include image understanding, artificial intelligence, and human perception. The current focus of his research is on dynamic scene understanding and the representation of actions. He received degrees from MIT of SB.Math, SB.CS and Ph.D. in Cognitive Science in 1981, 1981, and 1987 respectively. He was a computer scientist at the Artificial Intelligence Center, SRI in 1987～1991, and earlier a computer scientist at the Draper Laboratory in 1981.