K_031

状態遷移図を用いたキャラクタ定義に基づくゲーム記述環境

A Game Description Environment Based on Character Definition in Terms of State-Transition Diagram

下村 達也[†] 岡本 秀輔[‡] 鎌田 賢[†] 米倉 達広[†]
Tatsuya Shimomura Shusuke Okamoto Masaru Kamada Tatsuhiro Yonekura

1 まえがき

1990年代に流行したゲームソフトウェア The Incredible Toon Machine [1] では，自律的に動作するキャラクタを組み合わせて，アクションゲームを構成できる．様々なキャラクタはすでに用意されており，ユーザは，それらの初期配置を指定する．ゲームをスタートすると，キャラクタたちは相互に影響しあいながら自律的に振舞う．その結果として，ユーザが予定した，あるいは思いもよらなかったシナリオが展開される．このソフトウェアが創造的な子供たちの人気を得た理由は，ユーザがゲームを記述できる点にあった．そして，このソフトウェアで遊んだ創造的な子供たちは，さらに自分でキャラクタの動作を記述したいと願ったはずである．

ゲームにおけるキャラクタは，状態機械として表現できる．キャラクタの動作を有限状態機械として記述するアニメーション作成ツール [2, 3, 4] は既に実現されている．特に文献 [3] で示されたツールは，商品化され，Islay(アイラ) と呼ばれている．グラフィカルエディタで古典的な状態遷移図を編集する直感的なインタフェースを特徴としている．

Islay を用いて対話型のコンピュータアニメーションを作成することはできるが，コンピュータゲームの記述に用いるには不十分である．それは物理法則を表現する手段に欠けているためである．位置や速度などの物理状態は，物理法則にしたがって連続的に変化する．これに対して，Islay でのキャラクタ記述は，有限個の状態からなる状態機械である．物理状態の連続的な変化を近似的に表現することは可能ではあるが，膨大な数の状態が必要となる．むしろ，キャラクタのアイコンなどに対応する論理状態と物理状態は切り離して扱うほうが自然である．

そこで，本論文では，Islay に対する物理状態の自動更新機構の追加について述べる．これによって，直感的なインタフェースを特徴とするゲーム記述環境が実現される．論理状態から物理状態への関与として，力を作用させることだけを許し，位置や速度などの物理状態は，力学法則にしたがって定めるように，Islay を改良する．物理状態の更新機構はプラグインとして実装した．そのため，プラグインを取り換えるだけで，他の物理法則の導入も容易となっている．

*1 茨城大学 工学部 情報工学科
*2 成蹊大学 理工学部 情報科学科

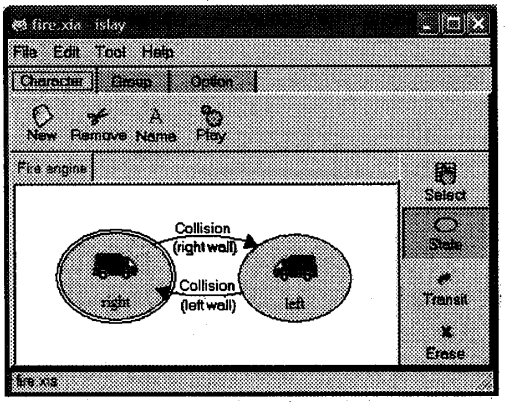

図1 状態遷移図を用いたキャラクタ記述

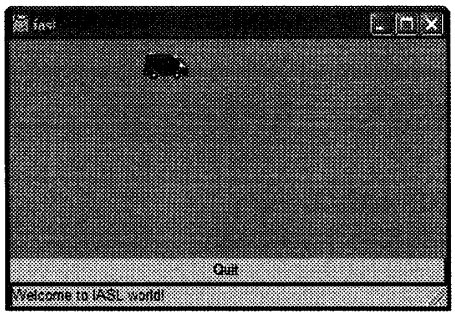

図2 アニメーションの実行画面

2 Islay の概要

Islay は，対話型アニメーションを作成するためのツールである．Islay による対話型アニメーションの記述では，キャラクタを図1のような状態遷移図を用いて記述する．1つの状態遷移図は1種類のキャラクタに対応する．キャラクタを記述するには，各状態にキャラクタの表示画像と動作を指定し，遷移の条件として壁との衝突などのイベントを指定する．また，遷移条件にはマウスやキーボードからの入力を指定することも可能であり，これによって，対話型アニメーションを実現できる．

図1は，消防車が画面内を左右にいったり来たりするアニメーションの，状態遷移図である．この状態遷移図には `right`,

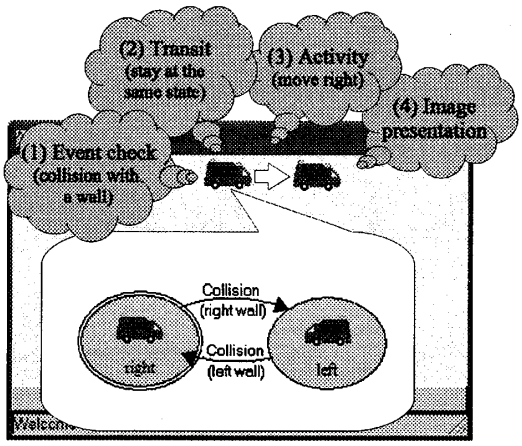

図3　キャラクタ更新処理

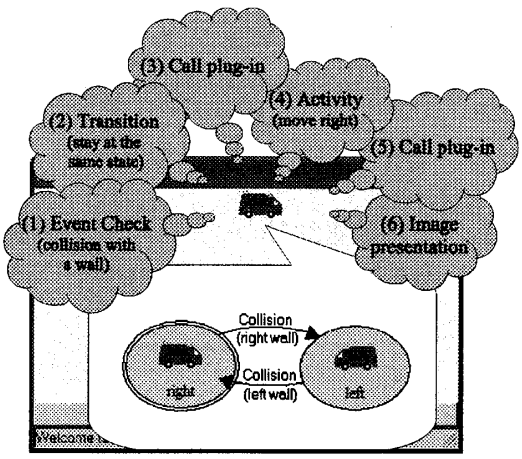

図4　改良されたキャラクタ更新処理

`left` という名前の2つの状態がある．これらの状態には，それぞれ向きの異なる消防車の画像と，左右方向への移動が指定している．状態 `left` と状態 `right` の間の遷移ではキャラクタと壁との衝突をトリガとして他の状態へ遷移するように指定している．このキャラクタの初期状態は，2重線の楕円で示された状態 `right` である．このように記述したアニメーションを実行すると，図2のように，壁にぶつかるまで同じ方向へ進み，ぶつかると逆向きに進んでいくアニメーションとなる．

Islay によるアニメーション実行では，一定の時間間隔毎にキャラクタの更新処理を行う．キャラクタの更新手順は，図3に例示するように

(1) イベントのチェック
(2) 状態遷移 (論理状態の更新)
(3) 動作の実行 (物理状態の更新)
(4) キャラクタの表示

の順で行われる．

3　物理法則の導入

Islay では，キャラクタは一定時間毎に状態に設定された方法で移動する．状態に設定できる移動方法は

- 絶対座標指定のジャンプ
- 相対座標指定の等速度移動
- ランダムジャンプ
- 他のインスタンスの追跡
- 最大移動量を指定したランダム移動

の5種類であり，いずれもキャラクタの位置情報を直接更新する．そのため，これらの移動方法を用いて物理法則にしたがったキャラクタの移動を記述しようとすると，膨大な数の状態が必要になり，状態遷移図が繁雑になる．物理法則にしたがってキャラクタの位置や速度を決定して移動するためには，物理状態を更新時に物理法則を考慮してキャラクタの移動先を決定すればよい．ただし，ゲームを作成する場合には，物理法則にしたがって動くキャラクタと，物理法則を無視して動くキャラクタが同時に存在することが考えられるので，キャラクタごとに法則の適用するか否かを切り替えられることが望ましい．

3.1　プラグインとしての物理法則の導入

物理法則にしたがった移動量を計算する処理をプラグインとして実装する．プラグインとは，実行時に動的に呼び出されるライブラリ群を利用して，特定の処理を外部ライブラリに任せる (委譲する) 機構である．従来の状態遷移図に加えて，プラグインの使用をオプションとして指定できるようにする．

具体的には，従来のキャラクタの更新処理を，図4に例示するように

(1) イベントのチェック
(2) 状態遷移 (論理状態の更新)
(3) プラグイン呼び出し
(4) 動作の実行 (物理状態の更新)
(5) プラグイン呼び出し
(6) キャラクタの表示

となる．

プラグインでは，物理法則にしたがってキャラクタの物理状態を更新する．

4　実装例

本論文では，プラグインの実装例として，万有引力の法則を実現するプラグインを実装した．このプラグインでは，キャラクタの更新時に画面内に存在する他のキャラクタとの間に働く万有引力を計算し，それに基づいて移動量を決定している．このプラグインの利用例を図5と図7に示す．

万有引力プラグインでは，キャラクタの作成時に初期位置，初速度と質量を設定する．例えば，図5では，図6のようにキャラクタを初期化している．

アニメが開始されると，2つのボールの間に働く万有引力によって，ボールが互いに近づいていく．この例では，キャラクタの動きをわかりやすくするために，軌跡を表示している．

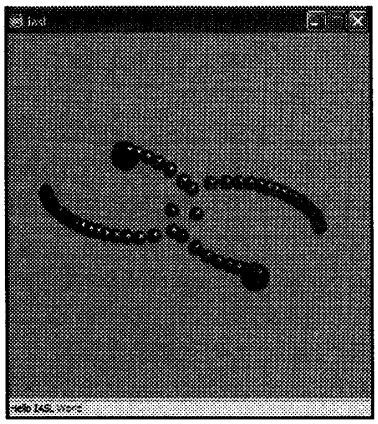

図5　万有引力プラグインを用いたアニメーションの例

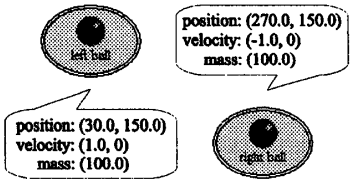

図6　図5の初期設定

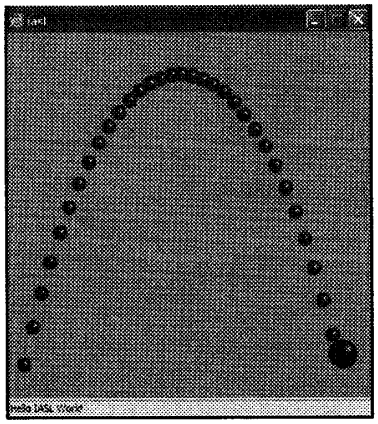

図7　放物運動の例

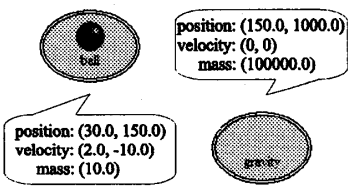

図8　図7の初期設定

図7は，放物運動の実現例である．この例では，図8のようにキャラクタを初期化している．画面外下の遠方に巨大な質量を持つキャラクタを重力源として配置することで，画面下向きに重力を発生させている．この方法を応用すると，例えば図9のような大砲ゲームが実現できる．

このゲームの記述は，図10-13のようになる．図10は，大砲の記述である．物理状態としては，大砲の初期位置が指定されている．論理状態としては，スペースキーの押下に応じて砲弾を生成することが記述されている．図11は，砲弾の物理状態として初期速度と質量が指定され，論理状態には何かと衝突すると爆発して消えることが記述されている．ターゲットとなる家の記述をしている図12では，物理状態として初期位置が与えられ，砲弾と衝突すると爆発して消えるという論理的遷移をたどる．図13は，図7で示した重力源と同じもので，画面下向きの重力を発生させている．この例では，物理的初期状態を指定し，論理的な遷移を記述するだけで，砲弾の運動などの物理状態の記述を気にすることなくゲームが構成できている．

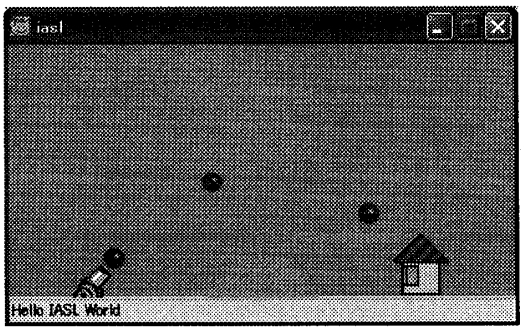

図9　万有引力プラグインを利用した大砲ゲーム

5　関連研究

この節では，アニメーション作成ソフトウェアやグラフィカルプログラミング環境などの例をあげる．

Squeak Etoys [5] は，タイルスクリプトを基本とした教育用のオブジェクト指向プログラミング環境である．主に小学校高学年の子供を教育の対象としている．エディタで描いた画像をオブジェクトとして扱う．オブジェクトの動作は，事前に用意された属性変更手続きをマウスで並べて定義する．Squeakでは，速度や加速度を指定してオブジェクトを移動させることができる．

ドリトル [6] は，タートルグラフィックスに基づく小学校高学年向けの教育用プログラミング言語である．画面上に作成した名前付きの図形に対して，日本語で命令を記述することで，プログラムを作成する．計算式を含む命令を作成することが可能なので，様々な運動を実現することができる．

びすけっと (Viscuit) [7] は，画面上の画像を動かすために，図形書き換えルールと呼ばれる，パターンにマッチした図形の位置や形などを更新する技術に基づいたプログラム作成ツールである．びすけっとでは，物理法則はサポートされていない．

Defart [8] は，AIBO [9] を自律的に動作させるプログラムを開発するための統合ソフトウェア開発環境である．Defartでは，状態遷移図によってAIBOの振舞いをプログラミングする．

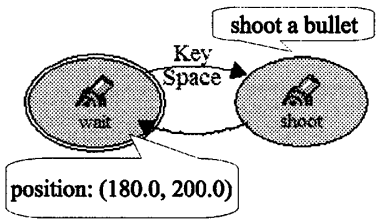

図 10　大砲の記述

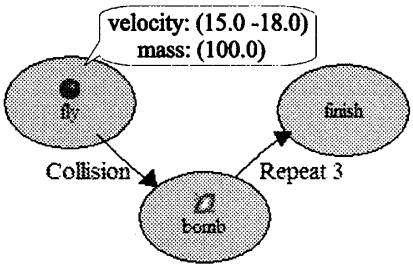

図 11　砲弾の記述

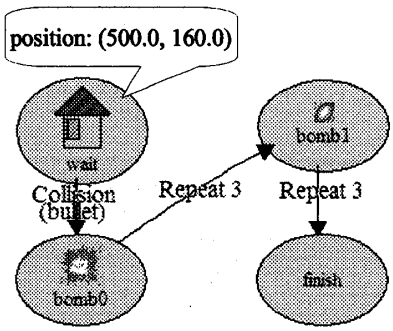

図 12　家の記述

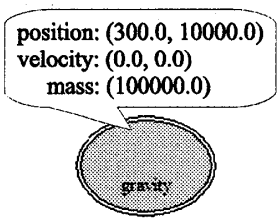

図 13　重力源の記述

6　むすび

対話型アニメ作成ツール Islay において，インスタンスの更新機構を論理状態の更新と物理状態の更新とに分離し，物理状態の更新は物理法則にしたがって行われるように改良した．物理状態を更新するための物理法則は，プラグインとしてプログラム本体から切り離されているので，プラグインを書き換えることによって，様々な物理法則を導入できるようになっている．

プラグインの実装例として，万有引力の法則にしたがってインスタンスの位置，速度を計算するプラグインを実装した．これにより，従来の Islay では膨大な数の状態を必要とした放物運動など，連続的に速度が変化するような動作を容易に実現することが可能となった．

今後の課題として，万有引力以外の物理法則，例えば，電気力や磁気力などの法則を実現するプラグインの作成が挙げられる．また，図 4 で示したように論理状態の状態遷移をトリガとしてプラグインを呼び出すこともできるので，遷移の瞬間に効果音を鳴らすようなプラグインの作成も可能である．そうすれば，キャラクタ同士の衝突や，ユーザからの入力に対応して効果音を再生することができる．このようなプラグインの検討も今後の課題である．

参考文献

[1] Sierra On-Line, Inc.: The Incredible Toon Machine 1994.

[2] Toshio Morioka *"A tool for authoring interactive animation based on state transition diagram"*, Master's thesis, Graduate School of Science and Engineering, Ibaraki University, Feb. 2002.

[3] 岡本 秀輔, 鎌田 賢, 中尾 隆司: "状態遷移図に基づく対話型アニメーション作成ツールの提案" 情報処理学会論文誌：プログラミング, Vol.46, No.SIG 1(PRO 24), pp.19-27, 2005 年 1 月.

[4] 有限会社ラーニングアイ, Islay アイラ 製品リリース, http://www.learning-i.jp/.

[5] Viewpoints Research Institute, I.: squeakland. http://www.squeakland.org/.

[6] 兼宗進: ドリトルとは. http://kanemune.cc.hit-u.ac.jp/dolittle/.

[7] 原田康徳: Viscuit. http://www.viscuit.com/.

[8] ITOLAB: Defart. http://www.itolab.com/.

[9] SONY: AIBO. http://www.jp.aibo.com/index.html.