数理モデル化と問題解決　18－10
（１９９８．３．２０）

ＮＰＣ動作アルゴリズムの自動生成に関する考察

長久 勝
フィルインカフェ
東京都新宿区天神町６村松ビル２Ｆ
e-mail:mnagaku@cap.bekkoame.or.jp

Abstract
コンピュータゲームにおいて、俗に言う敵の動作アルゴリズムを自動生成する方法論についての考察。本稿においては、特に対戦格闘型動作アルゴリズムを非線形最適化手法で自動生成する事について述べる。コンピュータゲームを数理モデルとして捉え、数理的な手法を実際の開発に導入する事を提案する。

Consideration Concerning Automatic Generation of NPC Operation Algorithm

Masaru NAGAKU
Ｆｉｌｌ ｉｎ Ｃａｆｅ
Muramatsu bldg. 2F, 6, Tenjin, Sinjyuku, Tokyo 162-0808 Japan
e-mail:mnagaku@cap.bekkoame.or.jp

Abstract
In the computer games, the methodology by which so-called enemy's operation algorithm is generated automatically is considered. In this text, the thing to generate the fighting grapple type operation algorithm automatically especially by the nonlinear optimization technique is described. The computer game is caught as a mathematical principle model and the thing to introduce the mathematical principle technique into an actual development is proposed.

1. はじめに

コンピュータゲームの制作を工学的な見地から俯瞰してみる時、そこに生じている問題を捉えるのに、その問題の質から次の２つのカテゴリに分類出来る。１つはゲームプログラム内部の実装に関する問題であり、もう１つはゲームに組み込まれるデータの生成に関する問題である。

前者の問題に関して、現在の開発現場では、開発対象となるプラットフォーム（ゲーム機）毎、開発しようとするゲームのジャンル毎に蓄積された、実装に関するノウハウを適用する事で問題解決にあたっている。いわば匠の技を必要とする現状は、問題解決の手法として未熟であると言わざるを得ない。しかしながら、他業種のプログラムに比較して、短期間で劇的に変化する技術背景を持ち、苛烈な競争にさらされている事を考慮すると、やむを得ないとも言える。

後者の問題に関して、現状どの程度効率的に行われているかと言えば、効率はあまり問題にされない。何故なら、ここでいうデータは画像、音楽、文章など多岐にわたるが、プログラムの質よりも、これらデータの質が商品価値を決定付ける要因として大きいからである。効率よりも、まず絶対的な質を求められる場合が多いのである。また、これらのデータの大半を制作するのが技術者ではない事からも、ソフトウェア工学の見地から効率を論じる事が容易でないと分かる。環境工学、人間工学といった分野からの適切なフィードバックがあれば、工学的な分析も可能と思われる。

後者の問題を解決する手法として、何らかのアルゴリズムを用いてデータを自動的に生成する事を考える。既に述べた様に、これは容易ではない。画像、音楽、文章を実用に足る水準で自動生成する事は現状ではほぼ不可能と言える。では自動生成し易いデータとはどの様なものかをまず考察してみる。

最初に、画像、音楽、文章のデータを自動生成困難とした前提条件として”実用に足る水準”と述べたが、評価対象となるデータに対して、定式化された評価基準に基づいて自動的に検証出来なければ、目的のデータを自動生成によって得る事は出来ない。複数のデータを一意的に順位付けられる指標が必要であり、その指標はアルゴリズムとして定式化され得るものでなければならない。

次に、不特定多数の候補から指標に基づいて最適なものを得ようとする時、実用に耐え得るシステムである為には、計算量が爆発しないシステムである事が求められる。従って一般的に、探索空間を小さくする為になるべく小さいデータ、評価時間を短くする為になるべく単純に求められる指標である事が必要である。

画像、音楽、文章は、実用に足る水準のデータがかなり大きい事や、簡単には指標を定式化出来ない事から、明らかにこうした条件を満たすデータではない。

本稿においては、この条件を満たすデータとして、ゲーム上でのキャラクタの動作アルゴリズムを取り上げる。

2. 対戦格闘型アクションゲーム

現在、コンピュータゲームは様々なジャンル分けがされている。例えば、有名なインベーダゲームはシューティングゲームと呼ばれるジャ

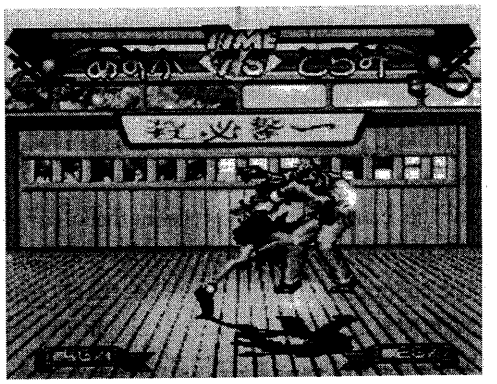

図1:対戦格闘型アクションゲーム

ンルに分類される。他にも、ロールプレイングゲーム、パズルゲーム、アクションゲームなどのジャンルがある。こうしたジャンル分けを考える場合、主にゲームのシステムによって分類がなされている。

本稿において具体的に取り上げるのは、対戦格闘型アクションゲームと呼ばれるジャンルについてである。(図1)

このジャンルのゲームシステムを要約すると、ほぼ同等の性質を持ったキャラクタが、1対1で戦い優劣を競う、といったものである。格闘の名が示す通り、格闘技の試合をそのままゲームにしたものと考えていただければ良い。

ゲームシステム上、重要な変数の1つが体力(図1画面上部左右の横棒状の表示)である。通常、ゲーム開始時の両者の体力表示は同じであるが、相手の攻撃を受けると減少していく。

プレイヤーは一方のキャラクタを操作し、もう一方のキャラクタを倒す事が目的である。具体的には、相手の攻撃を受けない様に、相手を攻撃する操作を行い

- 制限時間(図1上部中央)内に、攻撃によって相手の体力を0ポイントにする。
- 制限時間終了後の残体力が、相手のそれよりも多い。

のいずれかであれば、目的達成すなわち「勝ち」となる。

このゲームでは、2人のプレイヤーが対戦する場合と、動作アルゴリズムに基づいてコンピュータの操作するキャラクタと1人のプレイヤーが対戦する場合がある。本稿では、後者の場合における、コンピュータが操作するキャラクタの動作アルゴリズムを自動生成する事について考える。

3. NPC動作アルゴリズム

ゲーム上には、プレイヤーが操作するキャラクタ以外のキャラクタが登場する。これらNPC(ノン・プレイヤー・キャラクタ)を動作させる為の何らかのシステムが必要である。特にアクションゲームにおいては、通常1/60秒毎にNPCに動作指示を与えなければならない。こうしたNPC動作アルゴリズムについて、対戦格闘型アクションゲームを例に挙げて考察する。ここでは、NPC動作アルゴリズムの実装形態を考える。

このタイプのゲームにおいては、NPCもPC(プレイヤー・キャラクタ)と同様の動作を行うので、人間のプレイヤーがどの様に状況を判断して操作を行うのかを参考に、動作アルゴリズムを設計出来る。

ゲーム内で操作によって行える動作は、数種類の攻撃、移動など限られている。そこで可能な選択肢の中から、状況によって最適なものを選ぶわけであるが、ここでいう状況とは次の様なものから表現されると考えられる。

- 敵との水平距離
- フィールド上での位置
- 敵の動作状態
- 自キャラクタの動作状態
- 残体力
- 残時間
- その他

これら状況の各要素を以下、状況変数と呼ぶ事にする。

NPC動作アルゴリズムは、書き下された手順として実装される場合もあるが、アルゴリズムをデータ列で表現し、様々な操作を行い易くする。

まず、アルゴリズムが状況(状況変数群)を入力、操作を出力とする関数の形式で表現出来る事を確認しておく。(図2)

ここで、動作状態などの状況変数は、定義された数個の値しか取らない。例えば、動作状態では、移動中=1、攻撃動作中=2、ジャンプ中=3と定義されていれば、変数の値は1、2、3のいずれかである

また、コンピュータゲーム上で、整数で表現される距離などの状況変数も、実際には場合分

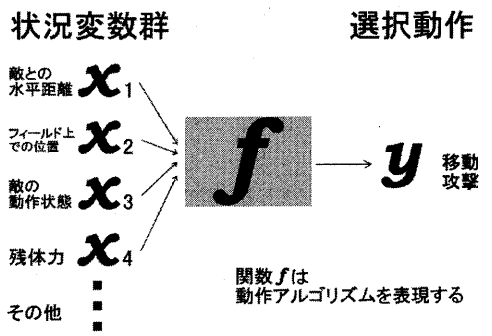

図2

け程度の表現で実用上問題がないと考えられる。例えば、攻撃の当たる距離であれば、実際の距離が1でも2でも同じである。

これら、状況変数が有限個の離散値で表現可能と考える事は、操作が数個の離散値しかとらない事からも妥当であると考えられる。

この様に考えれば、表現としては、分割された状態空間（全状況変数によって張られる空間）上の各領域に、出力値をマッピングした程度の表現で構わない事になる。(図3)

更に、状態空間の分割が、各状況変数について、座標軸に直交する様な分割であれば、この関数を多次元配列として、コンピュータ上で扱い易く表現出来る。

また、実際には、各状況変数によって空間を分割する幅などは異なるであろうし、正の領域しか持たないものもある。負に関しては正の領域の反転で良いものも考えられる。従って、状況変数による次元に比べて、かなりコンパクトな多次元配列で良いと考えられる。

こうした方法論によって、NPC動作アルゴリズムは、十分コンパクトな多次元配列で表現可能と考えられる。

本稿においては、多次元配列による表現を提案しているが、関数を目的の構造を持つものへと最適化する手法として、ニューラルネットワーク（NN）を用いた枠組みが良く知られている。NNを用いた枠組みでは、精度の高い表

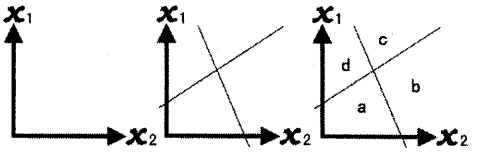

図3

現を、比較的容易に得る事が可能であると考えられているが、関数自体の演算量がある程度必要である。これはスペックに比べて高速な動作を要求されるコンピュータゲームの分野においては、かなり閾値が高く感じられる。それに比べて多次元配列による表現は、速度的に保証された実装が容易である。

コンピュータゲームにおける動作速度は、他の分野のソフトウェアに比べて重要度が高くなる傾向がある。この為、それを考慮したモデルを作成する事が必要である。

4. 動作アルゴリズムの最適化

多次元配列で表現された動作アルゴリズムを最適化して、実際に商品に組み込める水準のものを得なければならない。ここではその手法について述べる。

まず最初に、動作アルゴリズムとゲーム本体のプログラムの関係について整理する。プレイヤーがNPCと戦う場合、以下の通りである。

- プレイヤーの操作（動作選択）とNPC動作アルゴリズムの動作選択が発生する。
- 選択された動作は、ゲーム本体のプログラム内部に新しい状況を作り出す。
- 新しい状況に応じて、再び、プレイヤーとNPCの動作選択が行われる。

勝敗が判定されるまで、この繰り返しによってゲームの処理は進む事になる。(図4)

ここで、NPCの目的は、ゲームの枠組み内での十分な強さによって、プレイヤーの相手を務める事である。また、不自然な動きが目立つ

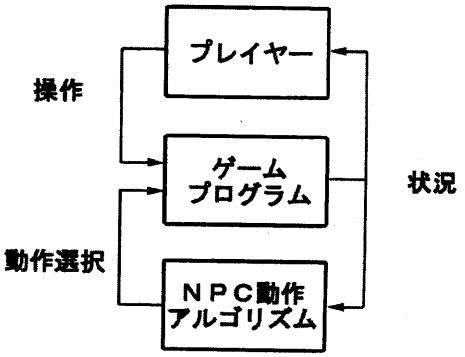

図４

と、プレイヤーに対して心理的なストレスを与える場合が多いので、極力、人間が操作している様に感じさせるアルゴリズムでなければならない。

こうしたＮＰＣに要求される能力から、動作アルゴリズムを最適化する際に必要となる、動作アルゴリズムの評価法を考える。

この評価法について、実際にはかなりの複雑さを伴う。例えば、不自然な動きを定式化し、評価法に加える事は極めて難しい。そこで、評価法を単純化する為、プレイヤーがゲームを遊んで上達した結果、非効率な不自然な動きは淘汰されているだろうと考える。同様に、十分に強い動作アルゴリズムは、自然な動きを行うだろうと考えるわけである。従ってここでは、強い動作アルゴリズムを評価する方法を考える。

強い動作アルゴリズムを相対的に評価する方法は至極簡単である。２つの動作アルゴリズムの優劣を付けたければ、実際に戦わせれば評価出来る。この評価を行うには、ゲームプログラムがそのまま流用可能である。特定のグループ内で順列が必要ならば、リーグ戦によるポイント計算で決める事が可能である。通常、ゲームプログラム内では乱数が使用される為、同一の組み合わせの対戦を複数回行うと良いだろう。

一般的に、データ表現に対して、評価法が定式化されていれば、様々な最適化手法を用いて目的のデータを得る事が可能である。従って、以降の操作に関しては、最適化手法について記された文献を参照して頂くものとし、ここでは簡単な例を示すにとどめる。

本稿のモデルに対して、遺伝的アルゴリズム（ＧＡ）を用いて、強い動作アルゴリズム得る手法について簡単に述べる。

動作アルゴリズムを表現した、多次元配列を染色体とするモデルを考える。配列内の各変数が取り得る値は、移動、攻撃などのあらかじめ定義された有限個の値でしかない為、それをそのまま対立遺伝子と出来る。

次に、ランダムに構成された染色体の集合を用意する。ＧＡの性質上、この集合が小さ過ぎると支障があるので、染色体の規模に応じた十分な大きさの集合である事が必要である。

染色体集合のメンバーに順列付けを行う。例えば、総当たり戦、同一組み合わせ３回とし、勝ち星のみをカウントしてポイントとするなどと考える。

評価の低い染色体を破棄して、集合を更新する。通常、一定の割合で染色体の破棄を行うが、このモデルにおいては、０ポイントのものを無条件に破棄するとしておけば、初期の世代の進化を速める事が可能ではないかと考えられる。新規に作成される染色体は、前の世代の生き残りの集団から、交差や突然変異によって得られる。

こうして、染色体の順列付け、世代交代を繰り返す事で、優秀な染色体の獲得が可能と考える。本稿のモデルにおける評価法は、絶対的な指標ではなく、集団内での相対的な地位を得るものでしかない。しかし、世代が進むことで、優秀なものが生き残り、より勝つ事が難しくなっていく為、絶対的な能力も向上すると考えられる。１世代目の１ポイントと 100 世代目の１ポイントは、絶対的な強さという意味で同じ重みではない。

ＧＡなど非線形最適化手法の多くは、漸近的に真の解を求めようとする枠組みである。従って、真の解に十分近づいたと思われる段階で操

作を終了しなければならない。通常、評価が絶対的な値で行われる場合、操作を打ち切る判断は容易である。実用上問題ないと考えられる評価値を持つものが現れた段階で終了すれば良い。

本稿のモデルの様な場合にはそうした判断は難しい。最も単純な判断方法は、適当な世代毎に優秀な動作アルゴリズムを取り出し、実際に人間がテストする事である。しかし、これでは自動生成による作業の効率化といった観点からは好ましくないので、次の様なアイデアを紹介しておく。

ＧＡにおいて、各世代の探索空間上での分布を考えてみる。真の解は複数存在する可能性があるが、世代が進めば、生き残る優秀な染色体の分布は真の解の近傍に集中する。従って、十分な世代を経た後、生き残る優秀な染色体の集団は、極めて似通った構造を持つ複数の染色体のグループから構成されると予測される。染色体の似通り具合は、ハミング距離を導入する事で定量的に判断出来るので、同一グループを判断する為、十分に小さい値を設定し、グループに分割する事が出来ると考えられる。この染色体集団上でのグループ分けを監視し、複数グループの寡占状態が現れれば、十分優秀な染色体が得られたものとして操作を終了する。

5. 最後に

現在、コンピュータゲーム制作の現場では、その最先端なイメージとは違い、先端技術の導入が行われているとは言い難い。特にソフトウェアのアルゴリズムに関しては、他分野に比べて遅れていると言って良いと思われる。

本稿においては、コンピュータゲーム制作に数理モデルを導入し、非線形最適化手法を応用する事を提案した。適切な数理モデルに定式化出来れば、様々な研究成果の恩恵に与かれる事が示せたと思う。

ここで数理モデルは、一種のインターフェイスとして機能している。現状、こうしたインターフェイスを設定して、コンピュータゲームを捉える事は一般的でない。この現状が改善されなければ、研究者にとってゲームはそれ以上の意味を持たない。

実はコンピュータゲームは、先端の研究成果をテストするのに最適の環境である。実装コストを自身の成果でペイ出来る可能性を持ち、不具合があっても周囲に与える影響は少ない。この側面は、もっと着眼されるべき価値があると思われる。

コンピュータゲームについて、工学的な見地から、活発な研究がされるように願いつつ、本稿を終わらせて頂く。

謝辞

本稿を発表する機会と、貴重なアドバイスを下さった、和歌山大学システム工学部の城和貴氏に感謝します。

参考文献

1）星野：遺伝的アルゴリズム，ｂｉｔ，24-9/10，1992

2）小林：遺伝的アルゴリズムの現状と課題，計測と制御，32-1，1993

3）畝見：ＧＡの制御への応用，計測と制御，32-1，1993

図１について

「図１：対戦格闘型アクションゲーム」
作品名：あすか１２０％
（C）Ｆｉｌｌ ｉｎ Ｃａｆｅ／他