芸術科学会論文誌 Vol. 20, No. 1, pp. 1 – 9 (2021)

波動方程式による波伝播作用を利用した追跡行動アルゴリズム

渡辺大地 †(正会員)

† 東京工科大学メディア学部

Tracking Behavior Algorithm using Wave Propagation by Wave Equation

Taichi Watanabe†

† School of Media Science, Tokyo University of Technology

earth @ gamescience.jp

概要

近年コンピューターゲームの発展により，ゲーム AI が盛んに研究されている．ゲーム AI の重要な機能として経路探索や追跡がある．本研究は，任意のマップ中でキャラクター AI の追跡アルゴリズムを波動方程式を利用して実現する手法を提案する．ゲーム AI における追跡アルゴリズムは，経路探索のための各種理論を応用して実現することが多いが，経路を構成するグラフネットワークの構築や各ノードのコスト算出に手間や時間がかかってしまうという問題がある．本手法では，準備としては空間中の格子点において浸入の可不可のみを設定すればよく，容易に目的地への最短経路を求めることが可能である．さらに，動的なマップの変更の際にも追加処理を一切必要としないという利点がある．

Abstract

In recent years, with the development of computer games, game AI has been actively studied. The path search and tracking are important functions of game AI. In this research, we propose a method to realize tracking algorithm of character AI in arbitrary map using wave equation. The tracking algorithm in the game AI is often realized by applying various theories for route search. However, there is a problem that it takes time and effort to construct a graph network constituting the route and calculate the cost of each node. In this method, it is possible to set only the penetration possibility at grid points in space as preparation, and it is possible to easily find the shortest path to the destination. Furthermore, it has the advantage of not requiring any additional processing when dynamically changing the map.

1 はじめに

近年，ビデオゲームでのゲーム AI 技術の発展は目覚ましいものである．三宅 [1][2] は，ゲーム AI は大きく「キャラクター AI」,「メタ AI」,「ナビゲーション AI」の 3 種類に分類できることを述べている．このうち，キャラクター AI やナビゲーション AI の実現に重要となる技術が経路探索である．経路探索アルゴリズムはゲーム AI のみならず，カーナビゲーションシステムやロボット制御などにも用いられる重要な理論である．近年のゲームではオープンワールド型が流行しており，ゲーム中の Non Player Character(以下「エージェント」) は常に個々の目的に従って移動を行うが，ゲーム中の状況は刻々と変化していくため，行動制御はパターン化によって実現することができない．そのため，エージェントは現状から動的に経路を判断し，移動制御を行う必要がある．

このようなエージェントの行動を制御する方法として,「Killzone」[3] というゲームではマップ上に等間隔にポイントを敷き詰め，そのポイントの性質を分析・評価しエージェントを移動する手法を用いている．他にも，桑谷ら [4] や佐藤ら [5] はシューティングゲームにおいて敵の弾をよけるルートを経路探索を用いて求める手法を発表している．これらに共通しているのはエージェントを目的対象まで移動することである．目的対象までは基本的に障害物があり，また目的対象が動的に変化することもある．前述の Killzone の例は，あらかじめ各ポイントに様々な情報を蓄積しなければならないため，マップが広大になるにつれ蓄積する情報も膨大なものになる．経路探索を用いる手法では，マップが複雑化するほど処理が重くなる．必要領域すべてを探索せずに一定まで探索し，その時点での最適解を求める手法もある．しかし，一定までしか探索しないため精度が落ちてしまい，場合により望む結果が出ない場合がある．

一般的なグラフネットワーク上での経路探索は，古くから多くの手法が提案されている [6]．コンピューターゲーム内で利用される経路探索の理論として代表的なものは，ダイクストラ法 [7] やそれを改良した A*アルゴリズム [8] である．これらのアルゴリズムはエージェントから目的対象までにある各地点間の時間や距離などのコストを計算し，コストマップを生成する．そして，コストマップを基に最適な経路を計算する．しかし，問題点として，マップ全体のコストマップを参照するため処理が重く，また目的対象が移動するとコストマップの再構築が必要になる．

本論文では，2 次元波動方程式を用いてエージェントによる追跡行動を実現する手法を提案する．波動方程式は「シュレディンガー方程式」とも呼ばれ，空間中の振動や波動の現象を表現する．この理論を用いることで，空間全般に波が伝搬する様子を簡易に実現することができる．また，この方程式を用いたシミュレーションでは波の持つ反射や回折などの現象も実現できるため，複雑に入り組んだマップにおいても最終的には伝搬する．目的地から発生させた波が最終的にエージェントに到着した際，波が向かってきた方向が目的地への最短経路となる可能性が高いため，その方向に移動することにより追跡アルゴリズムを実現するというものである．本手法は，マップを格子点で区切り，波動方程式による伝搬作用を利用して各格子点に仮想的な波の高さを算出する．本手法では，他の経路探索に必要なグラフネットワークの構築やコストマップ計算は一切必要なく，そのため事前準備が簡易であることや，マップが変更した際のコストマップ再計算の必要がないという利点がある．

我々は，本研究の初期の成果を芸術科学会東北支部研究会にて発表した [9]．その時点では追跡エージェントが 1 体の場合にしか対応しておらず，複数エージェントへの適用の必要性が論じられた．NICOGRAPH2019 において，多重波による複数エージェントに対応した手法 [10] を提案した．本論文では，さらに従来手法である Lee のアルゴリズムと追跡効率に関する検証を追加している．

2 格子マップ上での波の伝搬

本手法では，まずプレイヤーキャラクターを含む各エージェントが移動可能な領域を規定するマップデータを入力とする．次にマップ上に格子点を配置し，各格子点がマップ内で移動可能領域にあるかどうかの情報を格納しておく．以降，マップ内でエージェントが移動できない領域を「壁領域」と呼称する．また，各格子点には波の振幅値を表す実数値を格納するための配列を確保しておく．このとき，格子点ごとの振幅値格納数は第 5 章で述べる多重波の波数の 3 倍である．

マップ内でエージェントが追跡行動を行う際には，まず追跡対象エージェント (プレイヤーキャラクター) から最も近い格子点上で振幅値に正の一定数を設定する．これは，その格子点で波が発生したことを意味する．発生した波は徐々に他の格子点に対し振幅値が波及していくことにより，マップ全体に広がっていく．

このとき，振幅値を真偽のみの論理値とし，隣接する格子点で真の場合に自身も真にするという処理だけでも波の伝播作用を実現することは可能であり，いわゆる決定性有限オートマトン [11] として記述することができ，これは Lee[12] のアルゴリズムという名称で知られている．この方法であればかなり高速な処理で伝播が実現できるという利点があるが，波到達時間はユークリッド距離ではなくマンハッタン距離に比例するものとなり，波の進行に異方性が生じてしまうという問題がある．また，波の進行方向も 8 方向のいずれかのみでしか検出できないため，エージェントの追跡経路が効率的にならない点も問題となる．図 1 は，オートマトンによる伝播の様子の模式図である．

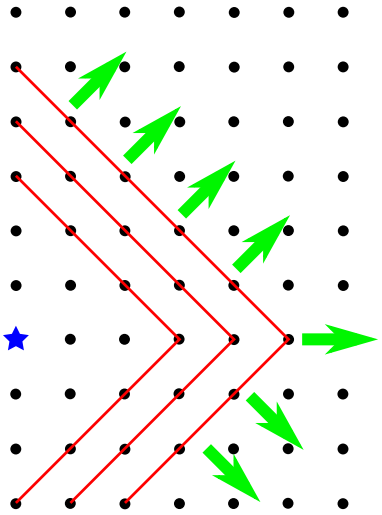

図 1　オートマトンによる伝播の様子

本手法では，波動方程式の伝播作用を利用することでこれらの問題を解決した．波動方程式による伝播速度は近似的であるがユークリッド距離に近く，全方角での等方性が保たれている．また，波が到着した際の方向算出もより正確なものとなる．図 2 は，波動方程式による伝播の様子の模式図である．

次章では，波動方程式を利用した具体的な算出方法について述べる．

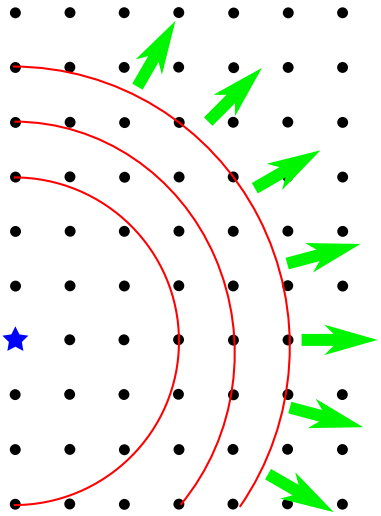

図 2　波動方程式による伝播の様子

3　波動方程式の離散化

関数 $z = z(x, y, t)$ で表される曲面において，2 次元波動方程式 [13][14] は以下の式で与えられることが知られている．

$$\frac{\partial^2 z}{\partial t^2} = c^2 \left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2} \right) - \mu \frac{\partial z}{\partial t} \tag{1}$$

ここで，c は波が曲面上を進む速さで，μ は粘性による減衰係数である．ただし，本手法では減衰現象を表現する必要がないので，計算の簡略化のため $\mu = 0$ とし，

$$\frac{\partial^2 z}{\partial t^2} = c^2 \left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2} \right) \tag{2}$$

を用いる．

次に，波動方程式を離散的に表して，プログラムで実装できる形式に改めてみる．まずは式 (2) の左辺から考えてみる．偏微分の定義から

$$\frac{\partial z}{\partial t} = \lim_{\Delta t \to +0} \frac{z(x, y, t + \Delta t) - z(x, y, t)}{\Delta t} \tag{3}$$

である．また，$z(x, y, t)$ が t で 2 階偏微分可能と仮定すると，$z(x, y, t)$ の変化量は，t の正方向と負方向で同一値とみなせることになる．従って，

$$\begin{aligned} \frac{\partial z}{\partial t} &= \lim_{\Delta t \to -0} \frac{z(x, y, t + \Delta t) - z(x, y, t)}{\Delta t} \\ &= \lim_{\Delta t \to +0} \frac{z(x, y, t) - z(x, y, t - \Delta t)}{\Delta t} \end{aligned} \tag{4}$$

も正しいということになる．

さて，Δt を十分小さな値と考え，

$$\frac{\partial z}{\partial t} \approx \frac{z(x,y,t) - z(x,y,t-\Delta t)}{\Delta t} \tag{5}$$

とする．$(\Delta t)^2$ を「Δt^2」と表記するものとしたとき，

$$\begin{aligned}\frac{\partial^2 z}{\partial t^2} &\approx \frac{\frac{z(x,y,t+\Delta t)-z(x,y,t)}{\Delta t} - \frac{z(x,y,t)-z(x,y,t-\Delta t)}{\Delta t}}{\Delta t} \\ &= \frac{z(x,y,t+\Delta t) - 2z(x,y,t) + z(x,y,t-\Delta t)}{\Delta t^2}\end{aligned} \tag{6}$$

という式で $\frac{\partial^2 z}{\partial t^2}$ は近似することができる．

次に式 (2) の右辺を考えてみる．式 (6) と同じように考えると，

$$\begin{aligned}\frac{\partial^2 z}{\partial x^2} &\approx \frac{z(x+\Delta h,y,t) - 2z(x,y,t) + z(x-\Delta h,y,t)}{\Delta h^2} \\ \frac{\partial^2 z}{\partial y^2} &\approx \frac{z(x,y+\Delta h,t) - 2z(x,y,t) + z(x,y-\Delta h,t)}{\Delta h^2}\end{aligned} \tag{7}$$

となる．

結果として，

$$\begin{aligned}z^*(x,y,t) = {} & z(x+\Delta h,y,t) + z(x-\Delta h,y,t) + \\ & z(x,y+\Delta h,t) + z(x,y-\Delta h,t) - \\ & 4z(x,y,t)\end{aligned} \tag{8}$$

としたときに，

$$c^2\left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2}\right) \approx c^2\left(\frac{z^*(x,y,t)}{\Delta h^2}\right) \tag{9}$$

となる．さらに，式 (2) に式 (6) と式 (9) を代入することで，

$$\begin{aligned}&\frac{z(x,y,t+\Delta t) - 2z(x,y,t) + z(x,y,t-\Delta t)}{\Delta t^2} \\ &\quad \approx c^2\left(\frac{z^*(x,y,t)}{\Delta h^2}\right)\end{aligned} \tag{10}$$

となる．

次に，式 (10) を離散式に変形する．まず，格子状にならんだメッシュ $z_{i,j}$ に対し，格子点間の距離を Δh と考える．また，z の時刻 k での位置を $z^k_{i,j}$ として表し，$z^{k+1}_{i,j}$ と $z^k_{i,j}$ の間の時間差 (タイムステップ) を Δt とする．

このとき，式 (10) は以下のように書き換えることができる．

$$\begin{aligned}&\frac{z^{k+1}_{i,j} - 2z^k_{i,j} + z^{k-1}_{i,j}}{\Delta t^2} \\ &\quad = c^2\left(\frac{z^k_{i+1,j} + z^k_{i-1,j} + z^k_{i,j+1} + z^k_{i,j-1} - 4z^k_{i,j}}{\Delta h^2}\right)\end{aligned} \tag{11}$$

これを $z^{k+1}_{i,j}$ について解くことで，

$$\begin{aligned}z^{k+1}_{i,j} = {} & \frac{c^2\Delta t^2}{\Delta h^2}(z^k_{i+1,j} + z^k_{i-1,j} + z^k_{i,j+1} + z^k_{i,j-1}) + \\ & \left(2 - \frac{4c^2\Delta t^2}{\Delta h^2}\right) z^k_{i,j} - z^{k-1}_{i,j}\end{aligned} \tag{12}$$

を得る．$c, \Delta t, \Delta h$ をいずれも固定値としたとき，式 (12) 中の $\frac{c^2\Delta t^2}{\Delta h^2}$ も固定値となるため，これを δ とおくことにより式 (12) は

$$\begin{aligned}z^{k+1}_{i,j} = {} & \delta(z^k_{i+1,j} + z^k_{i-1,j} + z^k_{i,j+1} + z^k_{i,j-1}) + \\ & (2 - 4\delta) z^k_{i,j} - z^{k-1}_{i,j}\end{aligned} \tag{13}$$

という簡素な式で表現できる．

式 (13) は隣接する格子点が 4 点あることが前提の式であり，メッシュの端の点には適用できない．3 点に接続する格子点の場合は，領域外にあたる格子点は全て常に 0 となると想定して式を変形する．例えば x 方向右端の点では

$$\begin{aligned}z^{k+1}_{i,j} = {} & \delta(z^k_{i-1,j} + z^k_{i,j+1} + z^k_{i,j-1}) + \\ & (2 - 3\delta)\, z^k_{i,j} - z^{k-1}_{i,j}\end{aligned} \tag{14}$$

を適用する．他の端の点についても，(14) 式中の $(z^k_{i-1,j} + z^k_{i,j+1} + z^k_{i,j-1})$ の部分を隣接 3 点の合計値にすればよい．同様に，隣接点が 2 点の場合も例えば右奥側の格子点の場合は

$$z^{k+1}_{i,j} = \delta(z^k_{i-1,j} + z^k_{i,j-1}) + (2 - 2\delta)\, z^k_{i,j} - z^{k-1}_{i,j} \tag{15}$$

とすればよい．

マップ上でエージェントが移動できない領域にある格子点については，z 値を常に 0 に固定するものとする．これにより，伝播してきた波は反射現象が生じるようになる．

式 (13)〜(15) は非常に高速に演算できる式であり，実装も極めて容易である．さらに，全格子点要素を並列に

演算することが可能であるため，マルチスレッドプログラミングや GPGPU との相性も良く，リアルタイムな表示にも十分対応可能なものである．

通常の波シミュレーションの場合，波全体のエネルギーが計算誤差により増大していくという問題があり，上記の式に減衰係数を積算することが多い．しかしながら，本研究においては波の波及効果は全体に行き渡る前に初期化してしまうため，減衰効果については考慮しないものとした．

4 追跡アルゴリズム

本研究における追跡アルゴリズムは，以下のような方針で実現する．

1. 追跡対象キャラクター (プレイヤー) を発生源とする波を随時発生する．
2. 波動方程式により波が空間中を伝播する．
3. 追跡行動を行うエージェントに波が到着した際，波が伝播してきた方向へ向かう．

第 3 章で述べたスカラー関数 $z = z(x, y, t)$ に対し，離散化によって格子点 $z^k_{i,j}$ については z 値が決定する．これに対し，任意の位置での z 値はバイリニア補間 [15] によって求めた．S を空間全体の辺長の半分とし，$i = \lfloor \frac{x-S}{\Delta h} \rfloor, j = \lfloor \frac{y-S}{\Delta h} \rfloor, u = \frac{x-i}{\Delta h}, v = \frac{y-i}{\Delta h}$ としたとき，(x, y) における z 値は以下の式によって求められる．

$$z(x,y,k) = (1-u)(1-v)z^k_{i,j} + u(1-v)z^k_{i+1,j} + (1-u)vz^k_{i,j+1} + uvz^k_{i+1,j+1} \tag{16}$$

この $z(x, y, t)$ の絶対値がある閾値を超えた場合，エージェントに波が到着したものとした．

到着した時点で，波の伝播方向を検出する．波の到着した瞬間において，エージェントの周囲では波が向かってきた方向の z 値が最も高い $z(x, y, t)$ が 1 階偏微分連続であると仮定すると，その方向は勾配ベクトル

$$\nabla z = \left(\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y} \right) \tag{17}$$

によって求めることができる．

まず，各格子点上での勾配ベクトルを求める．ここで，偏微分値は「正側と負側の傾きの平均値」で近似するものとし，以下のように算出した．

$$\frac{\partial}{\partial x} z^k_{i,j} \approx \left(\frac{z^k_{i,j} - z^k_{i-1,j}}{\Delta h} + \frac{z^k_{i+1,j} - z^k_{i,j}}{\Delta h} \right) /2 = \frac{z^k_{i+1,j} - z^k_{i-1,j}}{2\Delta h} \tag{18}$$

$$\frac{\partial}{\partial y} z^k_{i,j} \approx \left(\frac{z^k_{i,j} - z^k_{i,j-1}}{\Delta h} + \frac{z^k_{i,j+1} - z^k_{i,j}}{\Delta h} \right) /2 = \frac{z^k_{i,j+1} - z^k_{i,j-1}}{2\Delta h} \tag{19}$$

任意の点 $z(x, y, t)$ における勾配ベクトルは，式 (16) と同様にバイリニア補間によって求めた．

$$\nabla z(x,y,k) = (1-u)(1-v)\nabla z^k_{i,j} + u(1-v)\nabla z^k_{i+1,j} + (1-u)v\nabla z^k_{i,j+1} + uv\nabla z^k_{i+1,j+1} \tag{20}$$

以上から追跡アルゴリズムは，エージェントの位置における $z(x, y, t)$ の絶対値が閾値以上となったときに，$\nabla z(x, y, t)$ の値方向に移動することで実現できると換言できる．

なお，波は到着した後もそのまま継続して計算を行った場合，波の上下によって勾配ベクトルがかなり振動してしまうため，そのままでは追跡に利用することはできない．そのため，エージェントに第一波が到着した時点で ∇z を計算した後，波の z 値をすべて 0 に初期化するまではエージェントと波は反応しないものとし，波初期化後に改めてプレイヤー位置から波を発生するという処理を行うことで振動を回避した．

5 多重波による応答性の向上

多数のエージェントによる同時追尾を行う場合に，各エージェントごとに個別に波データを持つことは記憶量，処理速度のいずれの観点でも効率が悪い．そのため，複数のエージェントに対し共通の波データによって追尾処理を行うことが望ましい．しかしながら，波の伝播が全てのエージェントに行き渡るのに時間がかかってしまう場合に，エージェントが方向転換を行う機会が少なくなってしまい，結果的に追尾性能がかなり悪くなってしまうという問題が生じる．

そこで，本手法では単一の波ではなく，複数の波を並列に処理することにより上記の問題を解決した．以下，n 個の波を $\{W_0, W_1, \ldots, W_{n-1}\}$ と表現する．

第 4 章で述べたように，波伝播を実現するには各格子点が 3 世代分の振幅値を記録しておく必要がある．これを複数にするため，まず各格子点での振幅値記憶領域を $3n$ 個分確保しておく．例えば，波の個数が 4 である場合，それぞれの 3 世代分の振幅値を記録するため，格子点は $4 \times 3 = 12$ 個の振幅値保存領域を保持することとなる．これらの波は全エージェントで共通となるため，追跡エージェントの個数には依存しない．n が大きいほど追跡エージェントの追従性能は高くなるが，より多くの記憶容量と演算が必要となるため，状況に応じて適切な調整を行う必要がある．

波 W_0 をプレイヤーの位置を起点に発生した後，一定のタイミングを経てから同様に W_1 を発生する．このように，波 W_i を発生してから一定タイミングの後に W_{i+1} が発生するように設定しておく．その際，各波は独立して第 3 章で述べた演算を行うものとし，別の波の振幅値には一切関与しないものとする．これにより，後発の波が干渉作用で先発の波によって消失することを防ぐことができる．

各エージェントは W_i の第一波が到着した時点で勾配ベクトルを用いて方向を修正し，その直後から W_i には反応しないように設定する．従って，次にこのエージェントが方向を修正するのは W_{i+1} が到着したときとなる．

各波は共通の寿命 T を持つものとし，一度発生してから一定のタイミングの時間が T を経た波は初期化を行い改めてプレイヤーを起点に発生する．その際，各エージェントは改めてこの波の第一波に反応し方向を修正するように設定しておく．T の値を大きくしすぎると波伝播の誤差が激しく誤動作を起こしやすくなるが，小さい場合は追跡可能範囲が狭くなるため，事前に試行錯誤により適切な値を調整する必要がある．

波 W_i を発生してから W_{i+1} が発生するまでの時間は固定値 $\frac{T}{n}$ とすることで一定周期を保つものとした．

以上を繰り返すことにより，多重波を用いた連続的な伝播作用が可能となり，追尾性能を向上することが可能となる．

6 結果

6.1 速度検証

本手法を，我々が開発を進めている Fine Kernel ToolKit [16] (以下「FK」) 上に実装した．FK は内部で OpenGL を用いているリアルタイム 3DCG の研究開発用ライブラリである．今回はマルチスレッドや GPGPU は利用せず，シングルスレッド上で全て処理を行い，波動方程式関連の演算は全てシェーダーや GPGPU を用いず C++ 上で行った．また，多重波は 3 個とした．以下の表 1 が実行速度検証用の環境である．

表 1　実行速度評価環境

機種	Mac Pro (Late 2013)
CPU	Intel Xeon E5 2.7GHz 12-Core
メモリ	64G 1866-DDR3
GPU	AMD FirePro D700 6144M
OS	macOS Mojave 10.14.5

上記の環境で格子点が $100 \times 100 = 10000$ 個の場合のフレームレート (FPS) は描画なしで 1423.49, 描画ありで 416.23, $200 \times 200 = 40000$ 個の場合は描画なしで 547.31, 描画ありで 159.23 であり，リアルタイム 3DCG での実行には十分な実行速度を確保できていると言える．本手法は多重波の処理を並列に行うことが可能であり，マルチスレッドや GPGPU を用いることでさらなる処理速度の向上も期待できる．

6.2 動作検証

波の個数 n を 3，波の寿命 T を 300 ステップとし，動作検証を行った．図 3 は，赤い円錐型であるエージェントが，障害物を避けてプレイヤーキャラクターである青い球を追跡する軌跡を描画したものである． 今回の検証では，追跡 AI の挙動がわかりやすくなるように次の波が到着するまで等速度直線運動を行う単純な挙動アルゴリズムを用いた．そのため，折れ曲がっている部分が波が到着した時点となる．この軌跡から，障害物を回避して追跡行動を実現できていることがわかる．

図 4，図 5 はプレイヤーキャラクターが移動を行った際に，追跡 AI が改めて追尾を行っている様子を示したものである． 追尾対象が移動した際に変化することは波の発生源が異なるのみであり，データの再構築を行うことなく追尾が実現できていることがわかる．

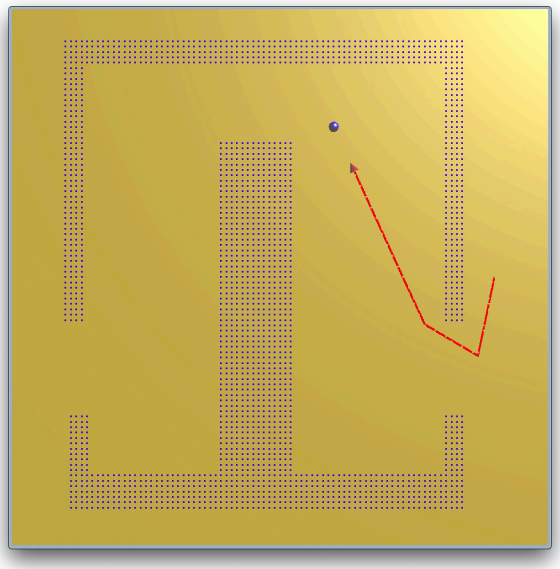

図 3　障害物回避の様子

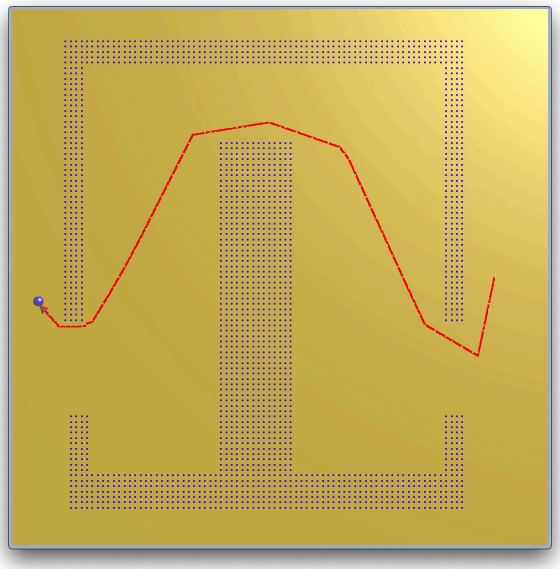

図 5　追尾の終盤段階

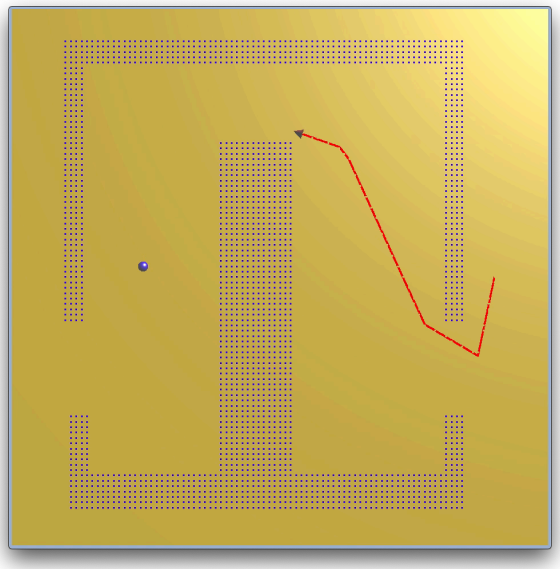

図 4　追尾の初段階

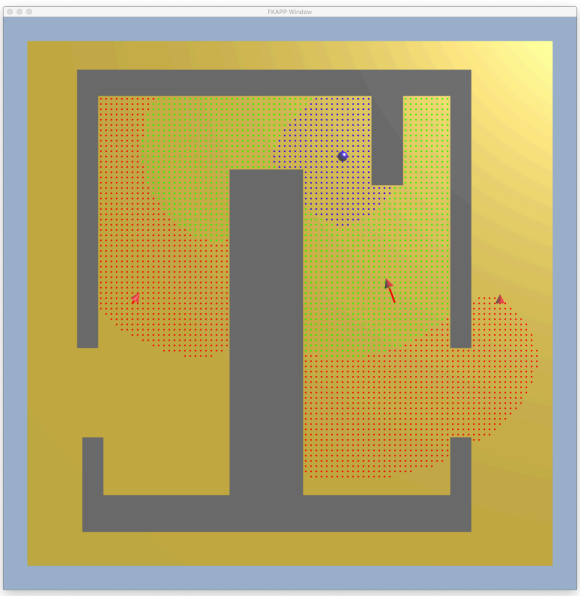

図 6　複数エージェント追尾の初段階

図 6，図 7 は，多重波による伝播作用と複数エージェントによる追尾処理の様子を示したものである．　多重波が波状的に伝播作用が行われていることや，複数エージェントによる追尾が効率の良い経路を選択できていることがわかる．

表 2 は，本手法と Lee[12] のアルゴリズムにおいて同状況からプレイヤーキャラクターに対する追跡を完了するまでのステップ数を示したものである．「A」「B」「C」は初期位置が異なる各エージェントである．「静止」と「ランダム動作」はプレイヤーキャラクターを静止していた場合とランダムに動き回るよう設定した場合を示す．

本手法による追跡経路が，従来の手法と比べてより効率の良い経路を算出できていることがわかる．

6.3　問題点

実際に実装してみたことで，本手法にいくつかの問題点が見受けられた．

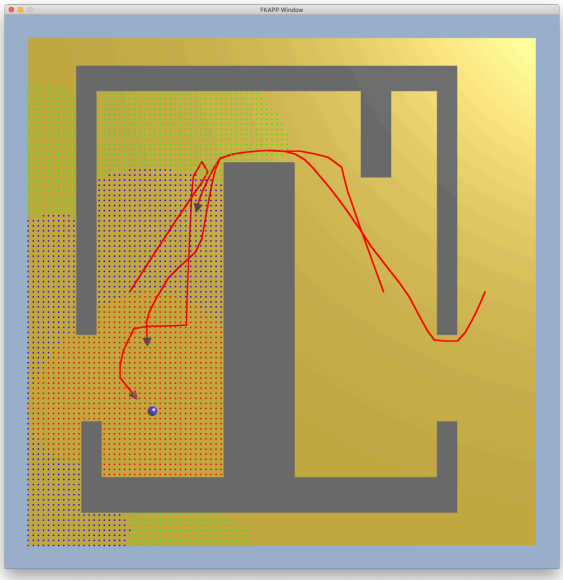

図 7　複数エージェント追尾の終盤段階

表 2　本手法と Lee 手法の効率比較

		A	B	C
静止	Lee	769	406	745
	本手法	632	313	584
ランダム動作	Lee	1104	747	450
	本手法	905	654	439

波動方程式による伝播では，その特性上発生源より距離が離れている場合や，反射波や回折波では波自体の振幅値が小さくなる．そのような箇所では波が到着した時点では閾値を超えずに到着を検出せず，周囲からの反射波や回折波を合計して初めて振幅が閾値を超えるケースがある．このような場合，勾配ベクトルは必ずしも波の到着方向とは一致しないため，結果的に追跡 AI が望ましくない方向へ移動するという現象が発生した．現在は，第一波の勾配が乱れてしまう前に初期化を行うように波の寿命を調整しているが，経路的に遠いエージェントには波が届かなくなってしまうという問題が残っている．このことを回避するには，第一波の振幅が保たれるように波動方程式を改変するということが考えられる．

また，波動方程式の離散化に伴い計算誤差が発生することから，波動速度や減衰係数などの値によっては波自体が発散してしまうことがあり，適切な係数設定の調整が必要である．

7　まとめと今後の展望

本論文では，波動方程式を用いた追跡行動アルゴリズムを提案した．本手法を用いた実装を検証し，AI が障害物を自動的に回避しつつ追跡行動を行えたことを確認した．今後，6.3 節で述べた問題を解決するために理論の改変を行っていく予定である．

参考文献

[1] 三宅陽一郎．ディジタルゲームにおける人工知能技術の応用．人工知能学会論文誌, Vol. 23, No. 1, pp. 44–51, 2008.

[2] 三宅陽一郎．ディジタルゲームにおける人工知能技術の応用の現在．人工知能学会論文誌, Vol. 30, No. 1, pp. 45–64, 2015.

[3] 三宅陽一郎．Killzone における NPC の動的な制御．https://www.slideshare.net/youichiromiyake/killzonenpc. 参照:2019.7.12.

[4] 桑谷拓哉, 橋本剛．熟練プレイヤーレベルを目指す弾幕シューティング AI の開発．情報科学技術フォーラム, Vol. 12, No. 2, pp. 383–384, 2013.

[5] 佐藤直之, Sila Temsiririrkkul, Luong Huu Phuc, 池田心．Influence Map を用いた経路探索による人間らしい弾避けのシューティングゲーム AI プレイヤ．情報処理学会ゲームプログラミングワークショップ 2016 論文集, Vol. 2016, pp. 57–64, 2016.

[6] 森畑明昌, 松崎公紀, 武市正人．領域限定言語に基づく最適経路問合せ．情報処理学会論文誌 プログラミング, Vol. 4, No. 2, pp. 116–133, 2011.

[7] E.W.Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, Vol. 1, pp. 269 – 271, 1959.

[8] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimal cost paths. *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 2, pp. 100 – 107, 1968.

[9] 渡辺大地, 西川孝亮．波動方程式伝搬作用による追跡行動アルゴリズム．平成 30 年度 第 4 回芸術科学会東北支部研究会, 2019.

[10] 渡辺大地, 西川孝亮．波動方程式による波伝播

作用を利用した追跡行動アルゴリズム. *NICOGRAPH2019*, pp. 91 – 98, 2019.

[11] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 2006.

[12] C.Y. Lee. An algorithm for path connections and its applications. *IRE Transactions on Electronic Computers*, Vol. EC-10, No. 2, pp. 346–365, 1961.

[13] 恒藤敏彦. 弾性体と流体. 岩波書店, 1994.

[14] Eric Lengyel. ゲームプログラミングのための 3D グラフィックス数学. ボーンデジタル, 2002.

[15] W.H.Press, B.P.Flannery, S.A.Teukolsky, and W.T.Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[16] Fine Kernel Project. Fine kernel toolkit. `https://gamescience.jp/FK`. 参照:2019.7.12.

渡辺 大地

1994 年慶応義塾大学環境情報学部卒業．1996 年慶応義塾大学政策・メディア研究科修士課程修了．2016 年岩手大学工学研究科デザイン・メディア工学専攻博士後期課程修了．博士 (工学)．1999 年より東京工科大学メディア学部講師．2017 年より同准教授，2020 年より同教授，現在に至る．コンピュータグラフィックスやゲーム制作に関する研究に従事．芸術科学会，情報処理学会，画像電子学会，人工知能学会会員．