社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－GI－16 (9)
2006／6／30

シューティングゲームの敵機攻撃弾発射アルゴリズムに関する考察

川野 洋
日本電信電話株式会社、NTTコミュニケーション科学基礎研究所

本研究では、コンピュータシューティングゲームの難易度の指標について、探索手法の観点から考察し、異なるシューティングゲーム間での難易度の比較を可能とする難易度指標を提案する。また、敵機の攻撃弾の発射速度決定の手法について考察し、将来の自機の移動を予測するための探索手法を攻撃弾発射速度決定の過程に導入することで、従来手法に比べて同じ数の攻撃弾で自機破壊確率を大幅に向上させることが可能であることを示す。提案手法は、攻撃弾発射時点から未来の各時間ステップにおいて、自機がゲーム画面内の各位置に存在する確率を評価するものであり、その計算にかかる時間は、従来の探索計算法に比べて大幅に短縮されるこれにより、提案手法は、シューティングゲームの要求する実時間性を満たすことが出来る。提案手法の有効性は、筆者が開発したシューティングゲームを用いた実験により検証される。

Study of Shooting Control Algorithm for an Enemy Character in Computer Shooting Games

Hiroshi Kawano
NTT Corporation, NTT Communication Science Laboratories

In this paper, an indicator of "difficulty" of computer shooting games is defined from the view point of artificial intelligence research. The defined indicator is described by the searching theory. This paper also proposes an algorithm for deciding the velocity of attacking bullets launched by the enemy character in shooting games. The algorithm is an application of searching technique, and it can be used in real time by adopting the efficient calculation method based on the prediction of existing probability of the player's character at each time step and position in the future. The destruction rate of the player's character can be increased by the proposed method. The performance of the algorithm is examined by the experiment using an actual shooting game program developed by the author.

1. はじめに

シューティングゲームとは、ゲームのプレイヤーが戦闘機を模したキャラクター（自機とする）を操作し、自機から攻撃弾を発射して、自機に対して攻撃を加えてくる敵キャラクター（敵機）と戦い、敵機を破壊することで、得点を得る形式のコンピュータゲームである。シューティングゲームは、1978年にTAITO社により、「スペースインベーダー」が発表されて以降、10年以上にわたってコンピュータゲームを代表するジャンルの一つであり続け、今日までに、発表されたシューティングゲームの本数は、５００本を超えている。

多くの場合、シューティングゲームの敵機行動は、シューティングゲームが実装された計算機上のアルゴリズムによって決定されるものである。すなわち、敵機行動制御のアルゴリズムの知能レベルの高低に、シューティングゲームの面白さと難易度は大きく依存していると言える。ところが、シューティングゲームの場合、複数の敵機がゲーム画面上に存在し、ある敵機が自機により破壊されても、常に新しい敵機がゲーム画面上に出現すること通常である。さらに、敵機は自機によって破壊されない攻撃弾を発射する場合が多い。このため、シューティングゲームの難易度は、むしろ敵機や、敵機の発射した攻撃弾の数に大きく左右されることが多く、これまでは、ゲームの難易度を上げるために

は、敵機や敵機攻撃弾の数を多くする手法に頼ることが多かった。その結果、敵機行動制御アルゴリズムの知的レベルの向上については、有効な手法が提案されていないのが現状である。

また、ゲームの難易度について、シューティングゲームには、確固たる指標がないのも問題である。AI研究で扱われている将棋やチェスなどのゲームにおいては、段位等の一般的な強さの指標があるが[2]、シューティングゲームの難易度を測る手法として考えうるものは、敵機の破壊数や、あるプレイヤーの得点、もしくは、自機の生存時間などに限られており、それらは、どれも設定の異なる個々のゲーム内でのみ通用する指標であり、一般的な観点から、そのシューティングゲームの難易度を計ることが出来るものではない[3]。

本研究では、ゲームの難易度の指標を人工知能研究における探索の視点から定義する。そして、敵機の行動決定のうち、大きくシューティングゲームの難易度を左右すると考えられる敵機攻撃弾の発射について、定義した難易度の観点から、シューティングゲームの難易度を向上させる手法を提案する。

2. シューティングゲームの定義

シューティングゲームが１人ゲームであるか、２人ゲームであるかについて考えてみる。まず、シューティングゲームを計算機（敵機側）とプレイヤー（自機側）の２人ゲームとして考えた場合、シューティングゲームにおいては、プレイヤーの勝ちについての明確な定義がない。なぜならば、シューティングゲームにおいては、例え自機が敵機を破壊しても、新たな敵機が耐えることなく出現するものが多く、ゲームは終了しないからである。ゲーム画面に出現する全敵機数が有限か無限かについても、個々のシューティングゲームによって異なることが多い（例：自機が最後のステージを超えるまで生き残る。敵機の親玉を破壊する。無限にゲームが続く等。。。）。逆に、負けについては、「自機が破壊されたときが、プレイヤーの負けであり、同時にそれがゲームの終了である」と言う形で明確に定義されていることが多い。すなわち、シューティングゲームを２人ゲームとして考えた場合、零和であるかないか、有限であるかないかの一律な定義をするのは難しい。逆に、１人ゲームとして考えた場合、プレイヤーが自機が破壊されないように、敵機の破壊数を最大化するよう最善を尽くすという観点で扱いが容易である[1]。

以上を考慮し、本研究では、シューティングゲームを以下のルールに従い１人ゲームとして定義する。

[ルール]

(1) ゲーム画面は、上方向を y 座標、右方向を x 座標として、x, y 座標値の最大値の絶対値を x_{max}, y_{max} とした場合、不等式

$-x_{max} < x < -x_{max}$,

$-y_{max} < y < -y_{max}$,

で表現される矩形領域である。

(2) ゲーム画面には、プレイヤーがその移動方向と攻撃を指示する「自機」が存在する。プレイヤーは、一定時間間隔 T 秒（1 時刻ステップ）おきに、「自機」に指示を出すことが出来る。「自機」は、ゲーム画面内において有限の大きさをもった領域で表現される。本研究では単純化のため「自機」が中心位置(x_s, y_s)にある半径 r_s の円領域であるとする。「自機」の移動方向は W 通り選べるとする。通常は、W=9 であり、静止を含む上下左右斜めの９通りの移動方向をプレイヤーは指示することが出来る。「自機」速さは、x,y 方向ともに一定値 V とする。「自機」の攻撃は、「自機攻撃弾」の発射によって行われる。プレイヤーが攻撃を指示したとき、「自機攻撃弾」は、自機位置(x_s, y_s)より上方向に速さ V_{sb} で移動する。「自機攻撃弾」は、単純化のため中心位置(x_{sb}, y_{sb})にある半径 r_{sb} の円領域であるとする。

(3) ゲーム画面には、計算機上で動く敵機行動制御アルゴリズムがその移動方向と攻撃を指示する「敵機」が複数存在する。敵機行動制御アルゴリズムは、一定時間間隔 t 秒おきに、各「敵機」に指示を出す(t=T とは限らない)。敵機行動制御アルゴリズムは、各敵機、敵機攻撃弾と自機との相対位置を元に敵機の移動方向と攻撃を指示する。各「敵機」は、ゲーム画面内において有限の大きさをもった領域で表現される。本研究では単純化のため i 番目の「敵機」が中心位置(x_{ei}, y_{ei})にある半径 r_{ei} の円領域であるとする。i 番目の「敵機」の移動速度を(Vx_{ei}, Vy_{ei})とする。「敵機」の攻撃は、「敵機攻撃弾」の発射と、「自機」への体当たりによって行われる。i 番目の「敵機」が j 番目に発射した「敵機攻撃弾」は、一定移動速度(Vx_{bij}, Vy_{bij})で移動する。「敵機攻撃弾」は、単純化のため中心位置(x_{bij}, y_{bij})にある半径 r_b の円領域であるとする。

(4) 各時刻において、「自機」領域と「敵機」領域が重なりあう部分があった場合は、「自機」は破壊されてゲーム終了となり、プレイヤーの負けとなる。各時刻において、「自機」領域と「敵機攻撃弾」領域が重なりあう部分があった場合も同様に「自機」は破壊されてゲーム終了となり、プレイヤーの負けとなる。各時刻において、「自機攻撃弾」領域と「敵機」領域が重なりあう部分があった場合、「敵機」は破壊されてゲーム画面から消滅する。自機は、ゲーム画面の外に出ることは出来ない。ゲーム画面外に出た「敵機」、「敵機攻撃弾」及び「自機攻撃弾」は、消滅する。ただし、通常は、V_{sb} は十分大きく、「自機」は、常に「自機攻撃弾」を発射していると仮定しても差し支えないので、本研究では、各時刻において、「自機攻撃弾」位置の x 座標と「敵機」位置の x 座標の差の絶対値が、$(r_{ei}+r_{sb})$の値の半分の値以下の場合に、「敵機」は破壊されてゲーム画面から消滅することとする。

(5)任意の時刻にて、敵機行動制御アルゴリズムは、新たな「敵機」をゲーム画面内に登場させることが出来る。

3. 難易度の定義と計算法

前節で述べた定義により、シューティングゲームは、自機が破壊されない限り、無限に続くゲームであると言える。ゲームの終わる時刻が不明であるため、シューティングゲームの難易度を定義する際には、ゲームが行われる全時間を短い時間の期間に区切り、各期間にてゲーム画面内に存在している敵機、敵機攻撃弾と自機との位置関係、速度、数等に応じて、ゲームの難易度を各期間毎に決めるのが適切である。そして、各期間のゲーム難易度を、ゲームが行われる全時間で総和もしくは平均したものをシューティングゲームの難易度と呼ぶことにする。

各期間に含まれる時刻ステップ数を n とすると、各ステップにて自機がとりうる移動方向は W 通りであるから、その期間の間に自機がとりうる移動経路の数は W^n 通りである。W^n 通りの各移動経路に対して、敵機の移動経路も W^n 通り存在し、自機の各移動経路と敵機の各移動経路は一対一で対応するものである。W^n 通りの自機および敵機移動経路の中で、自機が破壊される経路の総数を N_d とした場合、その期間中に自機が破壊される確率は、

$$P_d = \frac{N_d}{W^n} \qquad (1)$$

である。本研究では、自機撃墜率 P_d の値を各期間のシューティングゲームの難易度と定義する。このように定義された指標は、いわば、各期間において、プレイヤーが自機を生き残らせることが出来る指示の与え方の数がどれほどかということを示すものであり、P_d の値が、1 に近くなるほど、その数が少ないと言うことを意味する。この指標を用いることにより、異なるシューティングゲーム間の難易度比較を行うことが可能である。

4．敵機攻撃弾発射方法

これまでのシューティングゲームでは、各敵機が発射する攻撃弾の移動方向は、攻撃弾発射時点での、自機と各敵機を結ぶ線分に沿った自機に向かう方向に決められることが多かった。すなわち、敵機攻撃弾の速さを一律 V_b とした場合、敵機攻撃弾速度は、

$$r_{sei} = \sqrt{(x_s - x_{ei})^2 + (y_s - y_{ei})^2}$$
$$V_{xbij} = \frac{V_b(x_s - x_{ei})}{r_{sei}} \qquad (2)$$
$$V_{ybij} = \frac{V_b(y_s - y_{ei})}{r_{sei}}$$

で与えられる。この手法の問題点は、自機と敵機間の距離 r_{sei} が大きくなるにしたがって、自機が攻撃弾を回避するのが急激に容易になることと、攻撃弾速度の設定ルールが単純であるため、ゲーム画面中の敵機攻撃弾の数が多かったとしても、プレイヤーが敵機攻撃弾を回避するための技術を取得しやすいことである。この問題を緩和するための手法として、攻撃弾発射時点での自機位置から、0 度から 360 度の値をとる実数乱数で決定される角度の向きに、0 からある一定の値の範囲の乱数で決定される距離だけ離れた点(x_{srand}, y_{srand})を(2)式の(x_s, y_s)と置き換えた式

$$r_{sei} = \sqrt{(x_{srand} - x_{ei})^2 + (y_{srand} - y_{ei})^2}$$
$$V_{xbij} = \frac{V_b(x_{srand} - x_{ei})}{r_{sei}} \qquad (3)$$
$$V_{ybij} = \frac{V_b(y_{srand} - y_{ei})}{r_{sei}}$$

で、敵機攻撃弾移動速度を決定する手法も利用されてきた。この手法では、自機周りに敵機攻撃弾が適度に分散して飛来し、その分散の仕方も不確定なので、(2)式を使う手法に比べて、プレイヤーは、敵機攻撃弾回避の技術を取得しにくくなると言われているが、実際には、自機の攻撃に効果のない位置に攻撃弾を発射してしまうことも多く、その効果は安定して得られるものではない。また、このような効果に関する議論もこれまでは、明確な指標のない経験則で行われている。

本研究では、前節で定義した難易度の観点から、敵機攻撃弾の発射がより自機に対して効果的な攻撃となるような攻撃弾発射法について考えてみる。すなわち、前節で述べた、各期間における難易度 P_d を高く出来る攻撃弾発射法を提案することが本研究の目標である。以下にその手法を述べる。

まず、攻撃弾発射時点を各期間の始まりの時刻 0 とし、ゲーム画面内に既存の敵機攻撃弾がその後 n_e 時間ステップ（$n_e t$ 秒）移動を続けたとして、それらの既存攻撃弾によって自機が破壊される経路の総数を N_d 通りとした場合、新たに一つの攻撃弾が発射されることにより、自機が破壊される経路数の追加数をより大きく出来る攻撃弾発射方法が適切であると言える。今、各時刻ステップ k(k=0,,,,n_e)にて、ゲーム画面内の各位置に自機が存在する確率について考えて見る。時刻 0 での自機位置を(x_{s0}, y_{s0})とした場合、自機が存在しうる位置(x, y)は、s_x,s_y を整数($-n_e < s_x, s_y < n_e$)として、

$$x = x_{s0} + Vt\, s_x$$
$$y = y_{s0} + Vt\, s_y \qquad (4)$$

で与えられる。ゲーム画面を(4)式で与えられる各位置を中心位置とする格子 s で区切ると、各格子の大きさは、Vt である。各格子 s の座標は、(s_x,s_y)で表現される。時刻ステップ k にて、自機が格子 s に存在する確率を $P_s(k, s)$とする。時刻 0 で敵機により新規に発射された敵機攻撃弾の各時刻ステップでの位置に一番近い格子を s_{bk} とするとき、

$$\sum_{k=1}^{n} P_s(k, s_{bk}) \qquad (5)$$

を最大化する移動速度で攻撃弾を発射するのが適切であると言える。例えば、敵機攻撃弾速さが、一定値 V_b に決められているならば、攻撃弾移動方向角度θとして、$0 < \theta < 2\pi$ の範囲で適切な値を探索し、(5)式を最大化する値を選択すればよい。次節では、$P_s(k, s)$の計算方法について述べる。

5. 自機存在確率計算法

各時刻ステップでの自機存在確率 $P_s(k, s)$を計算するためには、時刻ステップ数 k を深さとし、各時刻 k で自機が選択する W 種類の移動方向を枝とする自機移動経路についての決定木探索を行い、各時刻ステップ k 上の各節点のうち、自機位置が格子位置 s にあるものの数を $C(k, s)$としたとき、以下の式で計算することが出来る。

$$P_s(k, s) = \frac{C_s(k, s)}{\sum_{s'} C_s(k, s')} \qquad (6)$$

$C(k, s)$の値は、時刻ステップ k において、自機が格子位置 s に至るための道筋の数を指す。ところが、実際には、(6)式内の $C(k, s)$の値を得るための決定木探索計算にかかる時間は、一般に深さ k が増すごとに W^kのオーダーで増加することが知られている。そのため、シューティングゲームのような実時間での計算が求められる場合には、通常の決定木探索手法を採用するのは適切ではない。そこで、本研究では、自機の移動が隣り合う格子間移動に限定されていることを利用した以下の方法で $C(k, s)$の値を計算する。

1. k=0 とし、(s_x, s_y)=(0,0)の格子を s_0 とし、

$$C_s(0, s) = \begin{cases} 1 & \text{if } s = s_0 \\ 0 & \text{if } s \neq s_0 \end{cases} \qquad (7)$$

とする。

2. k をインクリメントし、以下の式で、$C(k, s)$の値を計算する。

$$C_s(k, s) = \sum_{s' \in s_{neibor}} C_s(k-1, s') \qquad (8)$$

(s_{neibor}は、格子 s を含む格子 s 周りの W つの格子の集合。通常 W=9)

3. 各格子 s に位置する自機領域と、その時刻での既存敵攻撃弾位置(x_{bij}, y_{bij})に位置する敵攻撃弾領域が重なりあう部分がある場合、$C(k, s)$の値を 0 とする。

4.$k=n_e$になるまで、2.～3.を繰り返す。

本手法によれば、$C(k, s)$の値を求めるための計算量は、深さ k に比例した量のオーダーで済むので、通常の決定木探索法に比べて計算量を大幅に減らすことが出来、シューティングゲームが要求する実時間性を満たすことが可能となる。これは、この計算手法が、決定木の枝の情報を扱わずに、各時刻ステップでの各位置に至る道筋数のみに着目したことによる効果である。

6.実験による検証

本研究で提案した手法の効果を、著者が独自に開発したシューティングゲームプログラムを用いて検証した。本検証では、以下の二つの CASE を想定した。

CASE 1：自機直上方の異なる距離から自機に向けて一定の速度で発射された攻撃弾により、自機が攻撃された時.
CASE 2: 本研究で提案した攻撃弾発射方法を用いて、自機の初期位置から上方の一定距離に静止する複数の敵機が攻撃弾を発射した時。

上記、両 CASE にて、ゲーム画面大きさを x_{max}=200、y_{max}=400 ピクセル、自機大きさ r_s を 32 ピクセル、攻撃弾大きさ r_b を 4 ピクセル、自機速さ V を 60 [pixel/sec]、敵機攻撃弾速さ V_b を 60 [pixel/sec]、とした。T の値を 0.5[sec]とした。CASE 2 の攻撃弾発射方法における t を 0.2[sec]とした。敵機は、0.6[sec]秒おきに攻撃弾を発射するものとした。1 度刻みで 360 通りの攻撃弾発射方向角の中から(5)式を最大化する角度を攻撃弾発射角度とした。CASE 2 においては、敵機の自機からの距離を 100[pixel]から、400[pixel]の間で、敵機数を 1 機、2 機、および 4 機で変化させた結果を比較した。また、CASE 2 では、期間の開始時刻を、敵機が発射した最初の攻撃弾が、自機位置から 100[pixel]の距離に至る時刻とした。難易度判定での自機移動ステップ数 n を 5, 攻撃弾発射方法での敵機攻撃弾移動ステップ数 n_e を 15 とした。攻撃弾発射時間間隔を 0.66[sec]とした。

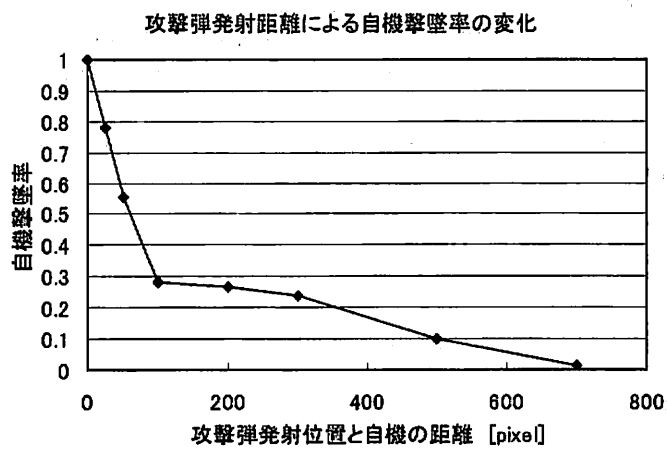

図 1　CASE1:攻撃弾発射距離の違いによる自機撃墜率の比較

図 1 に CASE 1 での、(1)式の定義によるゲーム難易度すなわち、自機撃墜率を示す。横軸は、攻撃弾発射位置と自機間の距離を pixel 数で表し、縦軸は自機撃墜確率を表す。図 1 によれば、距離 0 において、撃墜確率が 1 であるが、これは、いわゆる零距離射撃のためである。距離が 50pixel のところで、自機撃墜率は、大きく落ち込むものの、全体ではほぼ指数関数的な減少を見せている。これは、従来法による攻撃弾発射にて、自機と攻撃弾発射位置の距離が離れるとともに、急激に自機に攻撃弾があたりにくくなるという直感にも合っている。

続いて図 2 に CASE 2 にて、本研究で提案する攻撃弾発射方法の式(7),(8)により、各時刻ステップの各位置での自機の存在確率 $P_s(k, s)$を計算した結果を示す。図 2 には、計算時の予測既存攻撃弾位置も合わせて図示されている。図中の各格子の明るさが、自機の存在確率 $P_s(k, s)$の高さを表している。明るさは、各時刻ステップでの $P_s(k, s)$の最大値が白色となるように調整してある。実際には時間が進むとともに $P_s(k, s)$の値は急激に減少するので、もし確率 1 を白色としたならば、10 時刻ステップ

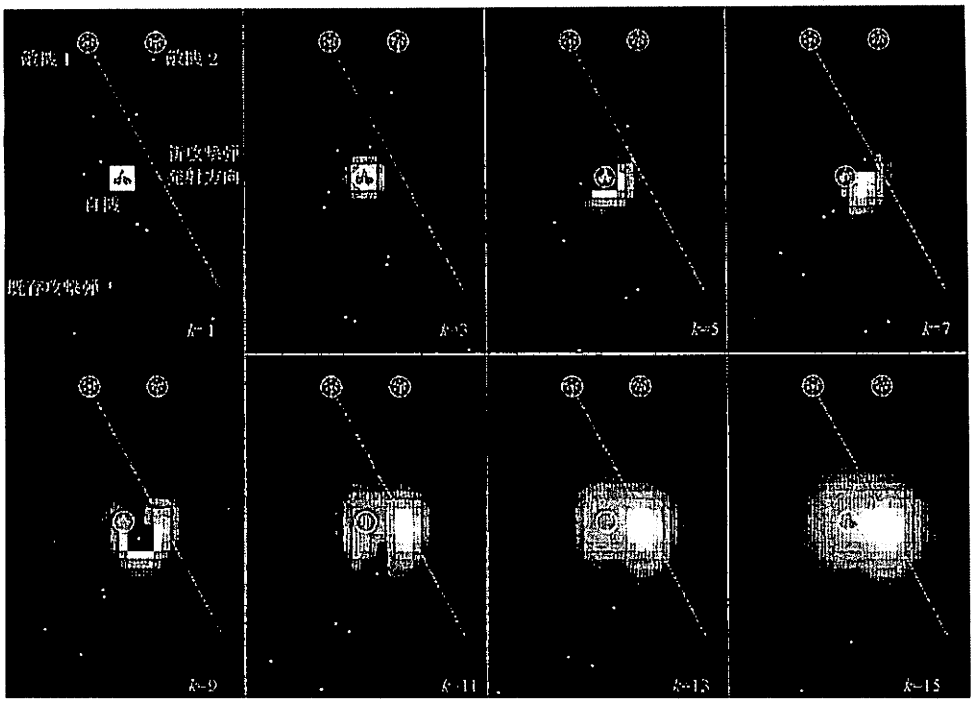

図 2　CASE2: 提案敵機攻撃弾発射方法による自機存在確率の計算結果

以上の未来では、図のほぼ全格子の色が黒色に近くなってしまうため、便宜的にそのようにしてある。図 2 は、敵機が 2 機左右に並び、距離が 200pixel の場合であり、左側に位置する敵機が攻撃弾発射をしようとしている場合を示している。図 2 によれば、時刻ステップ k=3 までは、自機位置中心に自機存在確率 $P_s(k, s)$の高い部分が広がっているが、時刻ステップ k=5 から 11 にかけての主に自機位置付近との自機の左側を通過する既存攻撃弾影響により、$P_s(k, s)$の高い部分が二つに分けられ、かつ右側の方が $P_s(k, s)$の値が高くなっている。時刻ステップ k=13 以降は、全ての既存攻撃弾が自機付近を通過し、結果として、自機から右側の 100pixel 離れたあたりに $P_s(k, s)$の極大値が現れている。(5)式を最大化する新攻撃弾発射方向は、射線が $P_s(k, s)$の高い部分を通るようになっているのがわかる。続いて、図 3 に提案攻撃弾発射方法と、既存の(2)(3)式による攻撃弾発射方法とで、同様に(1)で定義される自機撃墜率を測定した結果を示す。(3)式で使用する乱数の最大値は、50,100,150[pixel]の 3 種類の値で実験した。図 3 によれば、提案手法は、ほとんどの場合で、(2)(3)式を使用して攻撃弾を発射したよりも高い自機撃墜率を示している。唯一敵機が 1 機の場合で、距離 300pixel 付近の場合に、(2)式した場合と同等の自機撃墜率確率である。これは、敵機機数があまりに少なく、ゲーム画面中の攻撃弾数が少ないと、提案手法の効果が出にくいためとも考えられるが、その他の距離では、変わらず提案手法を使用した場合が最も自機撃墜率が高いので断言は出来ない。(3)式を使用した場合の自機撃墜率が、提案手法を使用した場合の自機撃墜率を上回るケースもごく少数存在しているが、全体的に(3)式を使用した場合の自機撃墜率の値は自機敵機間距離によって不規則に変化しており、性能は安定していない。(3)式は、(2)式を改良したものとされているが、場合によっては、(2)式を使用した場合よりも(3)式を使用した場合の方が、自機撃墜確率が低いこともある。(2)式を使用した場合は、敵機と自機間の距離が大きくなるにつれて、自機撃墜率が低下する傾向があるが、提案手法ではそのような傾向は緩和されている。このことは、提案手法により、遠く離れた位置からでも、効果的に敵機が自機を追い込むことが可能になったことを示す。提案手法を使用することで、既存手法に比べて約 2 割自機撃墜率が上昇したが、これは、自機生存率の視点で見れば、例えば敵機機数 4 の場合、生存率が、3 割から 1 割に減っている

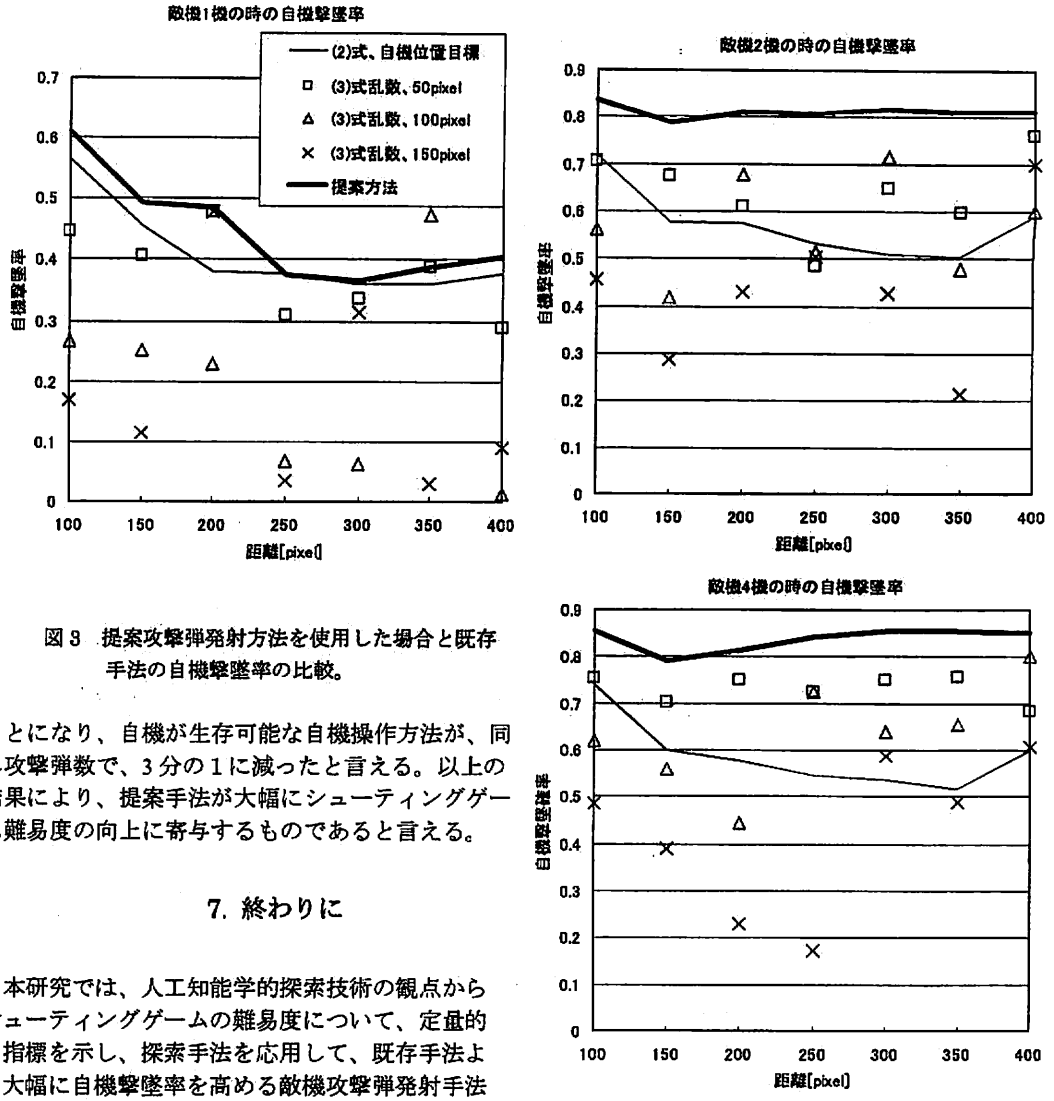

図3　提案攻撃弾発射方法を使用した場合と既存手法の自機撃墜率の比較。

ことになり、自機が生存可能な自機操作方法が、同じ攻撃弾数で、3分の1に減ったと言える。以上の結果により、提案手法が大幅にシューティングゲーム難易度の向上に寄与するものであると言える。

7. 終わりに

本研究では、人工知能学的探索技術の観点からシューティングゲームの難易度について、定量的な指標を示し、探索手法を応用して、既存手法より大幅に自機撃墜率を高める敵機攻撃弾発射手法を提案した。提案手法は、シューティングゲームの求める実時間性の要求を満たす効率的な探索計算法を核とするもので、未来における各時刻での各自機の生存確率を評価して、攻撃弾速度を決めるものである。提案手法の有効性は、シューティングゲームを利用実験で実証された。今後は、心理学的な観点も含めて、本研究を発展させていく予定である。

参考文献

[1] 人工知能学会編、「人工知能学事典」、pp.882-898、共立出版、2005年11月.

[2] J., Schaeffer, H., J., Herik, "Games, computers, and artificial intelligence", Artificial Intelligence, Vol. 134, p1-7, 2002.

[3] 小川　陽二郎他、公開番号：特許公開2005-95447、公開日2005年4月14日、「ゲーム制御プログラムおよび記録媒体」