様式 C-19

科学研究費補助金研究成果報告書

平成２１年　５月　２６日現在

研究種目：基盤研究(C)
研究期間：2007～2008
課題番号：19500093
研究課題名（和文）　大規模群れ動作の表現と高速化に関する研究
－群れ動作へのフラクタル性の導入－
研究課題名（英文）　Representation of a large school and its efficient simulation
- Introducing the fractality -

研究代表者
吉田　典正（YOSHIDA NORIMASA）
日本大学・生産工学部・准教授
研究者番号：70277846

研究成果の概要：
本研究では，大規模な群れの動作生成および表示を効率化するために，群れを再帰的に定義し動作させる階層的 boid アルゴリズムを考案した．通常の boid アルゴリズムは，個体数を n としたときに O(n^2)の計算量を必要とするが，階層的 boid アルゴリズムは，階層数を深くすると O(n)の計算量に近づく．視点に依存して動作と表示の両方を簡略化する手法，および階層的 boid アルゴリズムを用いることにより，群れの動作生成および表示の大幅な効率化を実現できることを確認した．

交付額

（金額単位：円）

	直接経費	間接経費	合　計
２００７年度	1,800,000	540,000	2,340,000
２００８年度	1,100,000	330,000	1,430,000
年度			
年度			
年度			
総　計	2,900,000	870,000	3,770,000

研究分野：　総合領域
科研費の分科・細目：　情報学・メディア情報学・データベース
キーワード：　グラフィクス，群れの動作生成，boid アルゴリズム

１．研究開始当初の背景

技術の進歩によって，現在のコンピュータの処理能力は飛躍的に向上している．また，コンピュータは，コンピュータグラフィックスの技術によって，豊かな表現が可能になった．今や，映画やコンピュータゲームなどの分野では，コンピュータグラフィックスの技術は必要不可欠である．コンピュータグラフィックスは，雨や雷などの自然現象や人の群れや魚群の表現，破壊などの物理現象などのリアリスティックな表現とともに発展してきた．本研究では，大規模な魚群の群れの動作生成と表示の高速化を研究対象とする．

鳥や魚の群れの動作を生成するアルゴリズムとして boid アルゴリズムが知られている．しかしながら，boid アルゴリズムを用いて，現在のコンピュータで対話的な速度でシミュレーションできる個体数は，我々の実装結果では数千体程度である．一方，自然界において，海で見られる魚の群れは，時に数十万体以上の個体からなる大規模なものになる場合がある．群れのシミュレーションに関する研究は多く存在するが，数十万体以上のもの群れをシミュレーションする研究は他

に見られない．大規模な群れの動作を効率的に生成し，表示することのできる手法が望まれる．

２．研究の目的

本研究は，大規模な魚群の生成と表示に着目し，従来手法より高速に群れのシミュレーションを行うことを目的とする．boid アルゴリズムを用いて，リアルタイムで大規模な魚群のシミュレーションを行うためには，群れの動作生成の効率化と群れの描画処理の効率化という 2 つの問題を解決する必要がある．本研究では，これらの問題点を改善する，大規模な魚群シミュレーションのための高速化手法について述べる．本研究のポイントは，階層化による探索処理の効率化と詳細度制御による描画処理の効率化である．

３．研究の方法

大規模な魚群をリアルタイムでシミュレーションするためには，動作生成処理と描画処理の高速化を行わなければならない．本研究では動作生成処理の効率化を階層的 boid アルゴリズムによって図り，描画処理の効率化のために遮蔽カリングを用いる．さらに，動作生成および描画処理の両方において，詳細度の制御を行い，視点からの距離および描画しようとする物体が見えるかどうかを考慮して詳細度を決定する．決定した詳細度によって，視点から遠く見えない部分は描画処理を大幅に効率化させ，視点に近い部分は処理を詳細に行う．

(1) 階層的 boid アルゴリズム

boid アルゴリズムにおいて，もっとも計算時間を必要とする処理は，すべての個体の中からもっとも近い個体を探索する処理である．従来の boid アルゴリズムは，図 1(a)に示すように，一番近い仲間の個体を探索する際に，群れ内の全個体を対象として（一般に）線形探索を用いて一番近い個体を見つける．階層的boidアルゴリズムは，図1(b)に示すように，群れを階層的に小さな群れに分割し，1 つの群れを”ユニット”と呼ぶ．階層的 boid アルゴリズムの基本的なアイディアは，探索対象を，ユニット内の個体に狭めることによって効率化を図ろうとするものである．階層 boid アルゴリズムでは，階層的に（多段階に）ユニットを作成する．

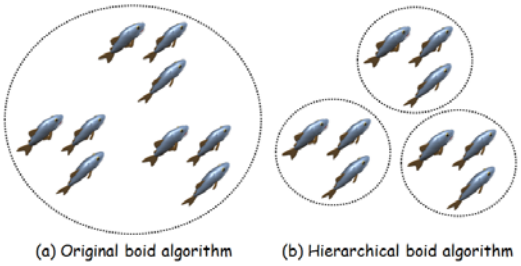


図 1　boid アルゴリズム(a)と階層的ボイドアルゴリズム(b)

階層を作成するためには，階層数 m と各階層における分割数 d を指定する． $m=4$，$d=2$ の例を図 3 に示す．まず，階層 1 のユニット H_1N_1 を作成し，次に階層 2 のユニット H_2N_1，ユニット H_2N_2 を作成していくという操作を指定した階層に達するまで行う．階層的 boid アルゴリズムにおいて，個体数を n，階層数を m とし，各階層の分割数および，各ユニット内の個体数を $\sqrt[m]{n}$ とすると，計算コストは $O\left(n^{(m+1)/m}\right)$ となる．また $m\to\infty$ の極限では計算コストは $O(n)$ に収束する．階層的 boid アルゴリズムでは，階層数を深くすれば動作生成時間は短縮されるが，その分メモリ消費量が増える．またユニットごとに boid アルゴリズムに用いるパラメータを設定する手間を必要とする．

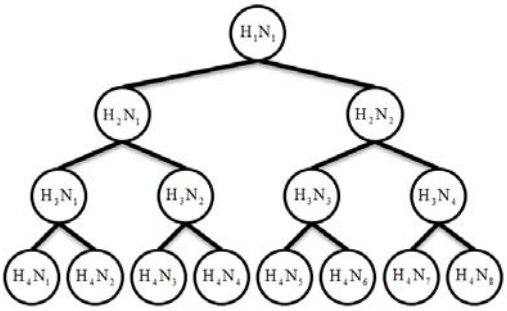


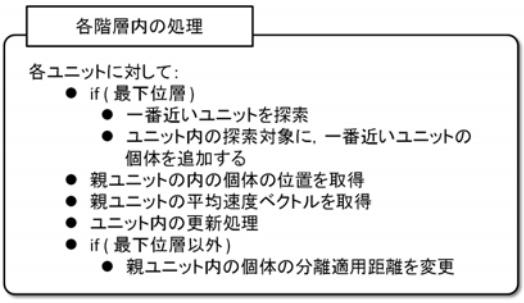
図２　階層の例

階層的boidアルゴリズムの個体位置の更新処理の概要を図 3 に示す．階層的 boid アルゴリズムでは位置の更新処理は最下位層から開始し，最上位層に向かってボトムアップ的に行う．図 2 の例について述べると，まず最下位のユニットから，各ユニットごとに個別に boid アルゴリズムを実行させる．次に，上位層(階層 3)へパラメータを渡し，ユニット H_3N_1 から H_3N_4 で boid アルゴリズムを適用する．その後，上位に移り同様の処理を最上位階層に達するまで繰り返し行う．

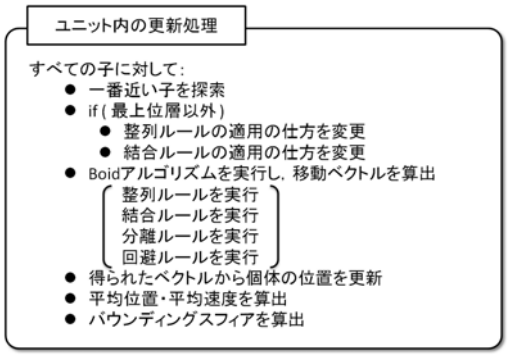
(2) 描画処理の効率化

本研究では，描画処理の効率化を図るために視錐台カリング，遮蔽カリングおよびポリゴンの詳細度制御を用いる．視錐台カリングは，視錐台内に入らない物体を計算によってカリングする手法である．遮蔽カリングは，物体を描画するとどのくらいのピクセルが可視であるのかをグラフィックスハードウェアに問い合わせを行い，遮蔽問い合わせの結果，ピクセル数が指定した閾値以下の場合は，物体が遮蔽されているので，物体をカリングする．本研究ではカリングを効率的に行うため

に，階層的 boid アルゴリズムで作成した階層構造を利用し，主として，前のフレームにて遮蔽されていた物体に対してのみ問い合わせを実行する．



(a)　各階層内の処理



(b)　ユニット内の更新処理

図３　更新処理の概要

(3) 詳細度制御アルゴリズム

本研究では，階層的 boid アルゴリズムをより効率的に実行するために，見える個体に対しては，見える個体のみに階層的 boid アルゴリズムを実行し，見えない個体に対しては図 4 のように親ユニットの平均位置，または boid アルゴリズムによって得られたベクトルを用いることによって位置の更新処理を行う．また，本研究では描画するポリゴンに対しても詳細度制御を行う．本研究では，階層的 boid アルゴリズムや遮蔽カリングによる処理負荷を考慮し，描画に使用するポリゴンメッシュは事前に用意しておき，ユーザーが指定した

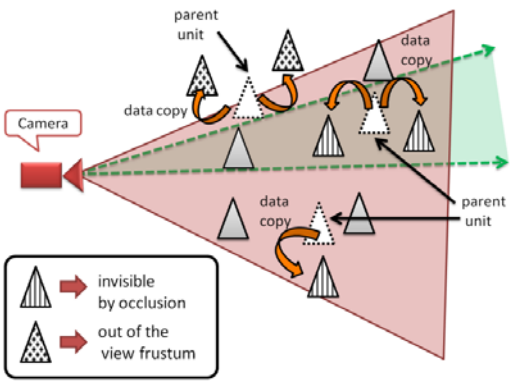


図４　視点依存による簡略化

値と視点からの距離に応じてポリゴンメッシュの詳細度を選択する．なお，ポリゴンメッシュの詳細度のレベル数は 3 とした．

４．研究成果

図 5 に，通常の boid アルゴリズム，および階層的boidアルゴリズムにおいて階層数が2，3，4，5 の場合の 10 ステップの平均動作生成処理時間を示す．横軸は個体数を示し，縦軸は，1 ステップあたりの動作生成処理時間(s)を示している．なお，CPU には Intel Core2 Duo 2.13 GHz を，GPU には NVIDIA GeForce 9800GTX を用いた．グラフから分かるように，階層的 boid アルゴリズムでは処理時間が大幅に改善され，階層数が増えるほど個体数に対する動作生成時間のグラフが直線に近づく様子が分かる．

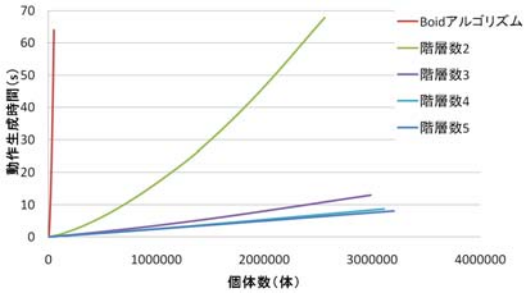


図５　階層的 boid アルゴリズムの計算時間

表 1 に，階層的 boid アルゴリズム，描画処理の効率化，および詳細度制御アルゴリズムのすべての効率化処理を利用して，階層数 5 の場合における計算時間（１０フレームの平均）を示す．我々の実装結果では，階層的アルゴリズムのみを利用した場合と比較して，すべての効率化処理を利用したほうが 3 割程度の効率化を実現できることが分かった．

表１　提案手法の計算時間

No. of individuals	Hierarchy	Division	Boid algorithm(s)	Rendering(s)	Culling(s)	Total(s)
32	5	2	0.00018	0.001824	0.008708	0.010712
3125	5	5	0.010919	0.104638	0.026919	0.142476
100000	5	10	0.856752	1.465396	0.314522	2.636671
759375	5	15	22.27327	4.841698	1.54056	28.65553

図６，図７に，階層数が 3，個体数が 27,000 の群れを示す．図６は被食者が捕食者から逃げる様子であり，図７は群れが形成される様子を示す．

本研究の成果を次に示す．

(a) 階層性的 boid アルゴリズムによる効率的な魚の群れの動作生成

boid アルゴリズムを再帰的に利用する階層的 boid アルゴリズムにより，動作生成が大幅に効率化できることを示した．また，群れないでの個体の衝突回避や，障害物の回避なども考慮したアルゴリズムを考案した．

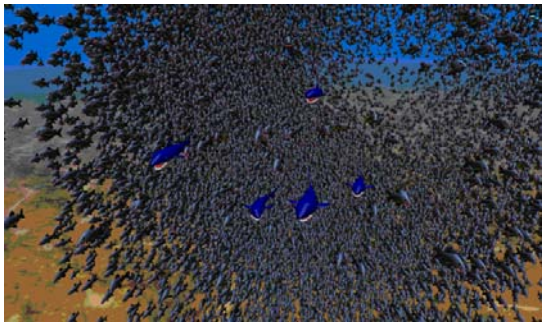


図６　被食者が捕食者から逃げようとする様子

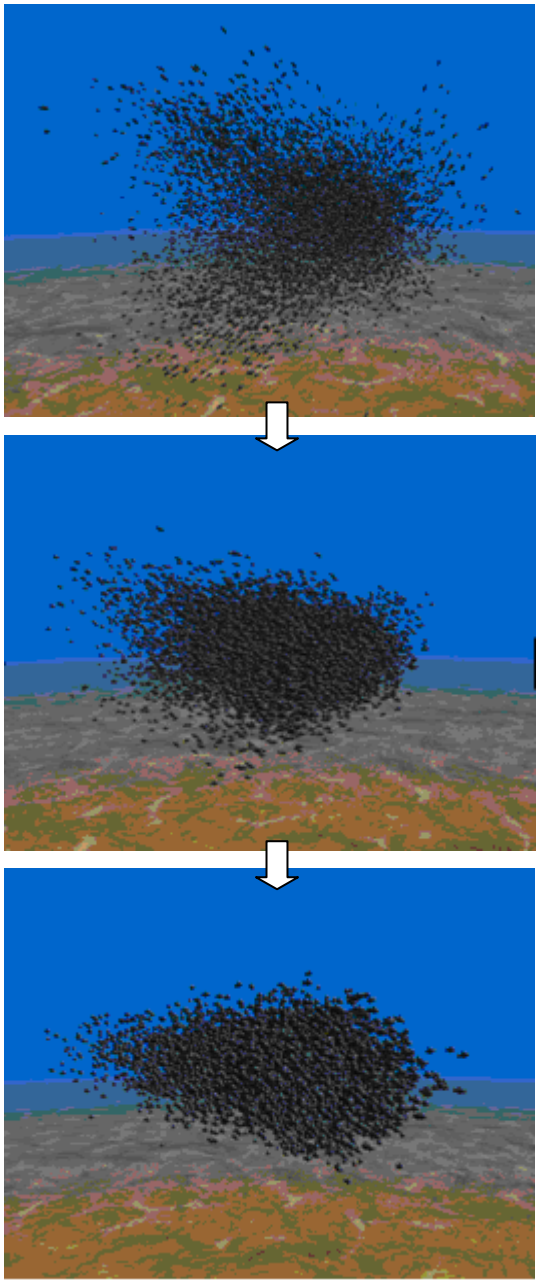


図７　群れを形成する様子

(2) 動作および表示の詳細度制御アルゴリズムの考案

詳細度制御とは，物体の重要度（視点からの距離など）に応じて，物体の複雑さを制御する手法である．例えば，遠くの物体は少ない数の三角形で表示し，視点に近い物体は多くの三角形を用いる．本研究では，この考えを表示だけでなく，視点から他の「部分群れ」に隠される「部分群れ」の動作を簡略化させることによって動作生成を効率化するアルゴリズムを提案した．

階層的boidアルゴリズムおよび詳細度制御アルゴリズムによって，動作生成の速度を大幅に改善できるが，階層性を持たせることによって，衝突回避時などにおいて，不自然な振る舞いをする場合がある．今後の課題としては，衝突回避などにおける自然な振る舞いをするアルゴリズムの構築，さらなる高速化などがあげられる．

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計　３　件）

① 吉田典正，佐藤大輔，山下安雄，映像を参考にした魚の群れの動作生成とその高速化，日本大学生産工学研究報告 A，第 42 巻，第 1 号，2009，掲載決定，査読有．

② 石橋 佳明，吉田 典正，大規模な魚群シミュレーションのための階層的boidアルゴリズム，情報処理学会グラフィクスと CAD 研究会，Vol.2008，No.109，p.37-42，2008，査読無．

③ Y. Ishibashi and N. Yoshida, View-dependent animation of a large school of fish, Image Electronics and Visual Computing Workshop, 1P-11, 2007, 査読有．

６．研究組織

(1)研究代表者

吉田　典正(YOSHIDA NORIMASA)

日本大学・生産工学部・准教授

研究者番号：70277846