芸術科学会論文誌 Vol. 13, No. 3, pp. 144-153

リアルタイム 3DCG における衝突を考慮したエネルギー波表現

仁藤将輝 [1)]（非会員）　**渡辺大地** [2)]（正会員）
柿本正憲 [2)]（正会員）　**三上浩司** [2)]（正会員）
1）東京工科大学大学院バイオ・情報メディア研究科メディアサイエンス専攻
2）東京工科大学メディア学部

Energy-wave Expression Considering a Collision in Real-time 3DCG

Masaki Nito[1)](Non-Member)　**Taichi Watanabe**[2)](Member)
Masanori Kakimoto[2)](Member)　**Koji Mikami**[2)](Member)
1) Graduate School of Bionics, Computer and Media Sciences, Tokyo University of Technology
2) Faculty of Media Science, Tokyo University of Technology
nito210ri@gmail.com

アブストラクト

漫画やアニメーションなどの創作コンテンツにおいてエネルギー波の表現は一般的になった。しかし 3DCG のビデオゲームにおいては、エネルギー波の衝突後の形状を制作者の意図どおりに表現することは難しい。本研究はリアルタイム 3DCG において、エネルギー波の多彩な衝突形状を容易に生成可能な新たな手法を提案する。衝突後のエネルギー波の形状を形状構成要素とし、8 つのパターンに分類した。エネルギー波をパーティクルの集合体とし、パーティクルは形状構成要素の 8 つのパターンのうちのいずれかに所属させた。所属したパターンによりパーティクルの運動方程式とレンダリング手法を決定した。パターンの所属にはパラメータや確率を利用した提案した手法を実装し、実際のコンテンツと比較検証することで本手法の有効性を確認した。これにより、多彩なエネルギー波の衝突形状を簡易なパラメータ設定のみで生成可能とした。

Abstract

Energy-wave expression has become common in creative content, such as animation and comics. But in a video game of 3DCG, it is difficult to represent the shape of Energy-wave intended by the author. We propose a new method that can easily generate a variety of shapes of the Energy-wave after the collision in real-time 3DCG. We classified Energy-wave into eight patterns as a component shape of Energy-wave after a collision. We considered Energy-wave as a collection of particles, and classified each particle into any one of the patterns of eight. Pattern of the component shape of Energy-wave determines the rendering method and the equation of motion of the particle. We used a parameter and probability for the classification of particle. We have confirmed the validity of the study by implementing the proposed method and comparing the actual content. As a result, we made it possible to express various shapes of Energy-wave after a collision by simple parameters.

1. はじめに

アニメーションや漫画といった創作コンテンツの中で、エネルギーの塊が強く発光し移動していく表現は一般的になった。実世界では起こりえないこの創作表現をアニメ愛好家などの間では「エネルギー波」と呼称されている。本研究もこれに習い「エネルギー波」と呼ぶこととする。エネルギー波は創作コンテンツ内において主に攻撃方法として表現される。エネルギー波の定義については阿部ら[1]と同様に「空間中のエネルギーの密度が高い場所が強く発光する」「形状変化を伴いながらある地点に向かって移動する」現象と定義する。定義中のエネルギーとは、創作コンテンツ内において空間中に存在するエネルギー波を構成する要素である。エネルギーは3次元空間上に分布し、密度が高い部分が強く発光するものとする。

エネルギー波の外観は、実世界での光の道筋が観測できる現象である薄明光線やレーザーと類似している。しかしエネルギー波は光とは異なり、光速で移動しない。またエネルギー波は他の物体と衝突し、衝突により形状が大きく変化する。そして衝突後のエネルギー波は間もなく消えていく特徴を持つ。これらの点が実世界における薄明光線やレーザーと異なる。

エネルギー波の衝突後の形状は創作コンテンツの作品ごとにさまざまに表現される。図1は創作コンテンツでの衝突後のエネルギー波の形状表現の一例である。

出典:"ドラゴンボールZ 危険なふたり!超戦士はねむれない"
©東映

図1. 創作コンテンツでの衝突後のエネルギー波形状

3DCGのビデオゲームにおいてエネルギー波を表現する手法として、少数の矩形ポリゴンにテクスチャアニメーションを利用する手法がある。この手法は処理負荷が軽く、リアルタイム性に優れているため多くのビデオゲームで用いる。しかしながら、この手法はエネルギー波の形状を描いたテクスチャ画像をあらかじめ用意する必要がある。そのため、衝突する物体の形状によってエネルギー波の形状を柔軟に変形することはできない。近年ではエネルギー波などのゲームエフェクトの表現に3Dエフェクト用ミドルウェアであるBISHAMON[2]や、高機能ゲームエンジンであるUnity[3]を利用するゲームが増えている。これらのミドルウェアやゲームエンジンを利用しゲームエフェクトを表現する際は、従来の手法に加え、パーティクルシステム[4][5]を利用することが多い。パーティクルシステムは大量のパーティクルをビルボードテクスチャなどで描画することにより、炎、煙、雪、雲といった不定形なものを表現することを得意とする。パーティクルシステムを利用し衝突を考慮したエネルギー波を表現するためには、衝突後のエネルギー波の特性を考慮した新たな運動方程式を定める必要がある。

そこで本研究では阿部らの定義に加え、「エネルギー波は他の物体との衝突が発生し、衝突によりエネルギー波の形状は変化する」「衝突後のエネルギー波は間もなく消えていく」という定義を追加する。そして、リアルタイム 3DCG において簡易なパラメータの設定のみでエネルギー波の多彩な衝突表現を可能にする新たな手法を提案する。まず衝突後のエネルギー波の形状を形状構成要素として 8 パターンに分類する[6]。パーティクルシステムと同様にエネルギー波をパーティクルの集合体とし、パーティクルは形状構成要素の 8 パターンのうちいずれかに所属する。所属したパターンによりパーティクルの運動方程式とレンダリング手法を決定する。パターンの所属にはパラメータや確率を利用する。パーティクルの所属する割合により最終的なエネルギー波の形状を決める。提案手法を実装し、実際のコンテンツとの比較検証をすることにより本研究の有効性を確認した。これにより、多彩なエネルギー波の衝突形状を簡易なパラメータの設定のみで可能とした。

本論文は全 6 章で構成する。第 2 章で関連研究について述べる。第 3 章でエネルギー波の衝突表現の多様性とその分析について述べる。第 4 章で提案手法を述べ、第 5 章にて提案した手法を実装し、手法の有効性を確認する。第 6 章でまとめと今後の課題を述べる。

2. 関連研究

エネルギー波の表現に関する研究として、阿部らはエネルギー波を構成するエネルギーの分布を解析的線積分可能な関数で定めた。これにより 3 次元空間中のエネルギーの分布に対し正確なエネルギー波表現を任意視点において実現可能とした。しかしエネルギー波と他の物体との衝突に関しては考慮していない。Nowrouzezahrai ら[7]はフォトンマッピングのアルゴリズムを利用し、アーティストが容易にボリュームライトを生成できるシステムを開発したが、このシステムはリアルタイム 3DCG には不向きである。仁藤ら[8]はエネルギー波の発光により発生する光源効果を、擬似的に点光源を配置することにより実現した。しかしながらエネルギー波が他の物体へ与える影響のうち、光源効果のみを実現しており、衝突に関しては考慮していない。

流体シミュレーションの自然な動きを保ちつつ、流体の動きをユーザが制御しさまざまな流体形状を生成する手法を提案した研究がいくつかある。Treuille ら[9]や Fattal ら[10]は、煙がユーザの指定した形状に形状変化するアニメーション生成手法を提案した。Dobashi ら[11]はユーザが指定した形状に一致するように動く積乱雲のシミュレーションを提案した。佐藤ら[12]は複雑なパラメータを必要とせず、ユーザが指定したステップ数で指定した形状に形状変化する爆発シミュレーションを提案した。これらの研究は実世界における煙、雲、爆発などの現象をもとにシミュレーションを行っているため、実世界には存在しないエネルギー波に対しこれらの手法を直接適用することはできない。

3. エネルギー波の衝突表現

3.1 表現の多様性

エネルギー波の衝突表現は創作コンテンツの世界観や制作者によりさまざまである。同じ作品内であってもシーンごとにエネルギー波の衝突表現が異なることも多い。しかし、エネルギー波の性質を考慮することにより、エネルギー波の形状をある程度分類することができる。エネルギー波が実世界の光と似た性質を持つ場合、エネルギー波の動きは直線的となり、グレア効果を表すような放射状へ鋭く広がる形状や、熱により火花が散っているような形状となる。エネルギー波が流体と似た性質を持つ場合、物体との衝突時には多数の飛沫が飛散したような形状や、流体の質量により衝撃波が発生しているような形状となる。エネルギー波は創作の現象であるが、実世界における複数の現象を模倣しているため表現に多様性が生じていると考えることができる。

3.2 形状構成要素の抽出

本研究ではエネルギー波のさまざまな衝突表現について、制作年代や作品の傾向を問わず、漫画、2Dアニメーション、3Dアニメーション、特殊撮影、ビデオゲームにおいて、約50作品、約150シーンを基に調査した。調査の結果、本研究ではエネルギー波の形状は4つの形状構成要素により成り立つと分析した。これら4つの形状構成要素はそれぞれ、2パターンずつ典型的な形状を持つと考えた。

(1) 大域的形状

第1の形状構成要素は大域的形状である。大域的形状とは、エネルギー波全体の大域的な形状を示すもので、大域的にみてエネルギー波が規則的に円状に広がっているか(円状)、規則性はなく不規則状に広がっているか(不規則状)の違いを持つ。図2は大域的形状の2つの要素の違いを表す模式図である。

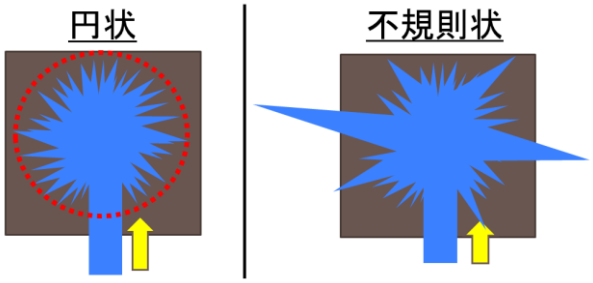

図2. 大域的形状の2つの要素

(2) 連続性

第2の形状構成要素は連続性である。連続性とは、エネルギー波の衝突後、衝突の勢いで飛び散ったエネルギーが衝突した位置から分離せずつながっており、連続しているかを表す。連続しているならば連続型、分離しているならば分離型とする。図3は連続性の2つの要素の違いを示す模式図である。

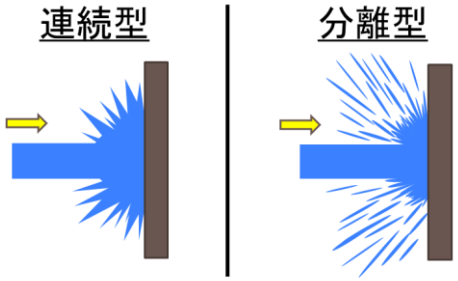

図3. 連続性の2つの要素

(3) 飛沫形状

第3の形状構成要素は飛沫形状である。飛沫形状とは、連続性が「分離型」であった場合のエネルギーひと塊の形状である。飛沫形状は、粒状か棒状かの違いを持つ。図4は飛沫形状の2つの要素の違いを表す模式図である。

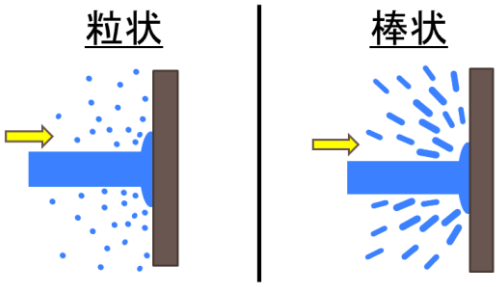

図4. 飛沫形状の2つの要素

(4) 飛沫先端形状

第4の形状構成要素は飛沫先端形状である。飛沫形状が「棒状」であった場合のエネルギーひと塊の先端が丸型か鋭利型かの違いを持つ。図5は飛沫先端形状の2つの要素の違いを表す模式図である。

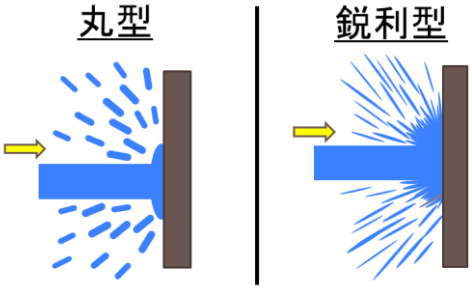

図5. 飛沫先端形状の2つの要素

形状構成要素の関係をまとめると8パターンに分類できる。分類した8パターンをそれぞれA、B、C、D、E、F、G、Hとし、表1に形状構成要素の関係を示す。ただし、実際のコンテンツ中でのエネルギー波は、これら8パターンのうち、いくつかのパターンが複合する場合もある。また飛沫形状や飛沫先端形状は定性的な要素であるため、どちらに近いかで分類する。

表1. 形状構成要素の関係表

			円状	不規則状
連続型			A	B
分離型	粒状		C	D
	棒状	丸型	E	F
		鋭利型	G	H

4. 提案手法

本章では衝突後の多彩なエネルギー波形状を生成するための提案手法を述べる。また本章以降はユーザ入力値を「パラメータ」と呼称する。パラメータはユーザが変更しない限り値が変わることはない。

4.1 パーティクルによる形状生成

パーティクルシステムと同様にエネルギー波をパーティクルの集合体として扱う。基本的な方針としてパーティクルをエネルギー波の形状に分布させ、パーティクルの位置にテクスチャを加算合成し、常にテクスチャを視点方向へ向けることでエネルギー波の形状をレンダリングする。テクスチャは円状にエネルギーが分布しているものを利用することで、いずれの視点から見ても 3 次元空間上のエネルギーの分布にあわせエネルギー波の形状をレンダリングすることができる。図 6 は利用するテクスチャの例、図 7 はパーティクル分布によるレンダリングの模式図である。

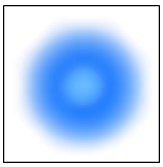

図 6. 利用するテクスチャの例

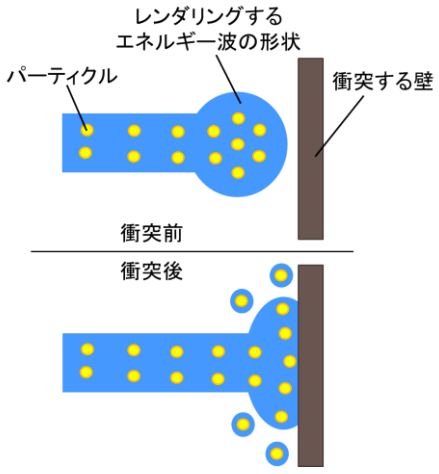

図 7. パーティクル分布によるレンダリングの模式図

4.2 パーティクルごとのパターン所属

すべてのパーティクルは 3.2 節で抽出した形状構成要素の 8 パターンのうち、いずれか 1 つのパターンに必ず所属する。パーティクルは所属するパターンごとに運動方程式とレンダリング手法が決まる。大域的形状の円状か不規則どちらかの所属によりパーティクルの運動方程式が決定する。連続性、飛沫形状、飛沫先端形状のいずれかの所属によりパーティクルのレンダリング手法が決定する。図 8 はパーティクルが所属パターンごとに運動方程式とレンダリング手法を決定することを表した図である。

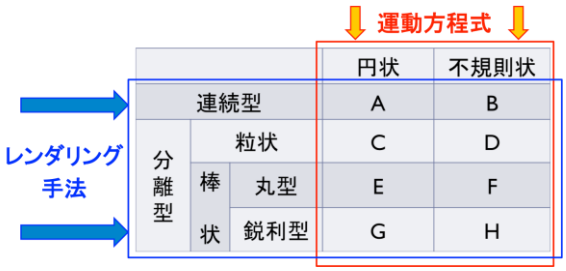

図 8. 所属パターンごとの運動方程式とレンダリング手法

エネルギー波を構成するパーティクルがどのパターンに所属するかの割合により、最終的なエネルギー波の形状が決まる。

パーティクルの所属パターンの決定にはユーザが定めるパラメータと確率を利用する。詳細な決定方法は4.4節にて述べる。

4.3 運動方程式

パーティクルの運動方程式は所属パターンが円状であるか、または不規則状であるかで大きく異なる。所属パターンの決定にはパラメータ円状分布確率$\alpha(0 \leqq \alpha \leqq 1)$を用いる。$\alpha$の値により式(1)のように確率的に決まる。

$$\begin{cases} \alpha & \cdots \text{円状に所属する確率} \\ 1-\alpha & \cdots \text{不規則状に所属する確率.} \end{cases} \quad (1)$$

式(1)で決定した所属パターンに従い、パーティクルの運動方程式を求める。

4.3.1 円状

所属パターンが円状のパーティクルは、大域的にみてパーティクルが円状に分布するように運動する。本手法では物体からの抗力**D**と周囲のパーティクルからの反発力**R**を導入する。この 2 つの力により物体と衝突時のパーティクルの加速度**A**の向きを決める。図 9 が加速度**A**の向きを求める模式図である。

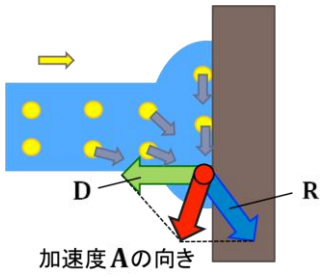

図 9. 加速度**A**の向きを求める模式図

以下に**D**の求め方を述べる。物体からの抗力にはペナルティ法を用いる。ペナルティ法は物体間の多少の侵入を許し、侵入した量に応じた力を抗力とする。パラメータであるバネ入力値をk、ダンパ入力値をh、パーティクルが衝突する物体に侵入した距離をz、衝突する物体の面の法線方向を**n**、パーティクルの速度を**w**とすると抗力**D**は式(2)で求める。

$$\mathbf{D}=kz\mathbf{n}+h(\mathbf{w}\cdot\mathbf{n})\mathbf{n}. \tag{2}$$

次に反発力**R**の求め方を求める。先述通り、反発力は周囲のパーティクルから受けると考える。本手法は粒子法の 1 つである SPH 法[13][14]を参考とした。各パーティクルは一定の有効半径を持ち、各パーティクルが持つ有効半径内の他のパーティクルから力を受けることとする。ただし他のパーティクルが同位置に存在した場合は力を考えない。反発力を受けるパーティクルの有効半径内の他のパーティクル数をn、反発力を受けるパーティクルの位置を**P**、有効半径内の他のパーティクルの位置を$\mathbf{Q}_i$、距離 1 のときの反発力を反発力入力値rとしたとき、周囲のパーティクルから受ける力の合計**R**は式(3)で求める。

$$\mathbf{R}=r\sum_{i=1}^{n}\frac{\mathbf{Q}_i-\mathbf{P}}{|\mathbf{Q}_i-\mathbf{P}|^2}. \tag{3}$$

図 10 がパーティクルと**R**の関係を表す模式図である。

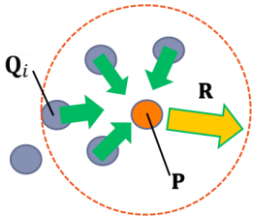

図 10. 反発力**R**を求める模式図

最後に**D**と**R**、パラメータcにより**A**を求める。**A**は式(4) により求める。

$$\mathbf{A}=\frac{\mathbf{D}+\mathbf{R}}{|\mathbf{D}+\mathbf{R}|}c. \tag{4}$$

式(4)によりパーティクルごとに**A**の向きは異なるが、**A**の大きさは同じとなる。すべてのパーティクルの**A**の大きさはcにより決まる。これにより、パーティクルは大域的にみて円状に分布する。なおパラメータであるパーティクルの寿命の値により、衝突後パーティクルが消滅するまでの時間は決まる。本節では円状の運動方程式について述べているが、cやパーティクルの寿命に対し、ランダム幅を持たせ、値にばらつきを与えることで不規則状のパーティクル分布を生成することも可能である。

4.3.2 不規則状

不規則状に分布しようとするパーティクルに対しては、制御点を用いる。本研究は見た目の制御を重視するため、よりユーザが直接的に制御しやすい制御点を用いることとした。パーティクルは物体と衝突後、制御点へ向かって移動し、制御点の位置で寿命が尽き消滅する。制御点は 1 つのエネルギー波に対しいくつか事前に設定する。制御点は 3 次元空間上の絶対的な位置ではなく、エネルギー波ごとの相対的な方向と位置を表す。制御点が複数存在する場合は制御点ごとに確率を設定し、各パーティクルがどの制御点に移動していくかを確率で決める。図 11 は制御点を 3 つ設定し、パーティクルが 3 方向へ枝分かれするレンダリング結果の例である。

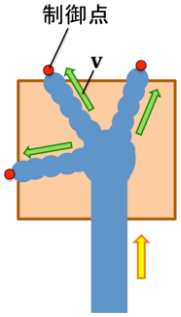

図 11. 制御点を 5 つ設定した典型例

パーティクルの衝突位置を**H**、制御点の位置を**E**、パーティクルの寿命をパラメータτとしたとき、パーティクルの衝突後の速度**v**は式(5)で求める。

$$\mathbf{v}=\frac{\mathbf{E}-\mathbf{H}}{\tau}. \tag{5}$$

制御点の位置を変えない場合、エネルギー波は終始同じ形状のまま動かないが、任意の時間間隔でいくつかの制御点を移動、または新たに配置することで、キーフレームアニメーションのようにエネルギー波の形状を変形することも可能である。また Perlin noise[15]を利用し毎フレーム制御点をランダムに移動することでより自然な動きのアニメーションとなりやすい。

制御点の利用によりユーザは容易に不規則状のパーティクルの分布が可能になるが、制御点の配置位置によっては円状のパーティクル分布を生成することも可能である。

4.4 レンダリング

前節ではパーティクルの運動方程式について述べたが、本節ではパーティクルの所属パターンごとのレンダリング手法について述べる。レンダリング手法の所属パターンは、はじめに連続型か分離型のどちらかに決まり、次いで粒状か棒状かが決まり、最後に丸型か鋭利型かが順番に決まる。

なお衝突後のパーティクルは残りの寿命に応じ透明度を下げて描画することでエネルギー波が自然に消滅するようにみえる。

4.4.1 連続型と分離型

連続型か分離型の決定には連続確率$\beta(0 \leqq \beta \leqq 1)$を用いる。4.3節の円状分布確率と同様に次の式(6)により確率的に決まる。

$$\begin{cases} \beta & \cdots \text{連続型に所属する確率} \\ 1-\beta & \cdots \text{分離型に所属する確率.} \end{cases} \tag{6}$$

分離型の場合は次節の粒状か棒状のレンダリング手法に従う。連続型の場合は本節のレンダリング手法に従う。連続型のレン

ダリングでは、各パーティクルは衝突した位置を各々記憶しておく。衝突した位置から現在のパーティクル位置に向けテクスチャを追加して配置し描画することで連続型を表現する。

1つのパーティクルに対しM個のテクスチャを利用するとし、パーティクルの衝突位置を$\mathbf{P}_0$、パーティクルの現在位置を$\mathbf{P}_1$、をとると$i(i = 1, 2, 3 \ \ldots \ M)$番目に表示するテクスチャ位置$\mathbf{T}_i$は式(7)で求める。

$$\mathbf{T}_i = \mathbf{P}_0 + \frac{\mathbf{P}_1 - \mathbf{P}_0}{M}(i-1). \qquad (7)$$

また連続型ではパーティクルが現在位置に近づくほどテクスチャサイズの倍率を下げて描画する。これにより連続型の形状を表現する。図12がテクスチャ配置を利用した連続型の表現の模式図である。

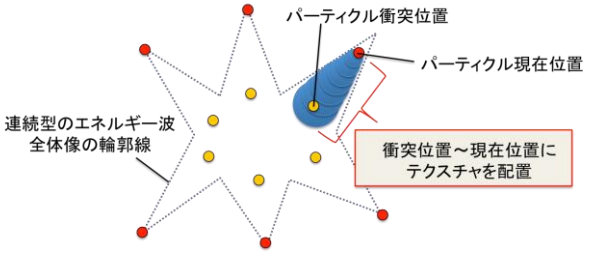

図12. 連続型のレンダリング

4.4.2 粒状と棒状

本節では、分離型の場合に生じる各飛沫形状のレンダリングについて述べる。飛沫のサンプリング数という整数のパラメータNを設定する。$N = 1$ならば粒状、$1 < N$ならば棒状となる。粒状の場合は単純にパーティクルの位置に1つのテクスチャを表示する。棒状の場合は粒状の表示も含み、パーティクル進行方向両側にN個のテクスチャを配置し描画する。パーティクルの位置を$\mathbf{Q}$、パーティクルの進行方向単位ベクトルを$\mathbf{d}$、テクスチャ表示間隔をパラメータuで表したとき、$j(j = 1, 2, 3, \ \ldots \ N)$番目に表示するテクスチャ位置$\mathbf{U}_j$は式(8)で求める。

$$\mathbf{U}_j = \mathbf{Q} + (-1)^j u(j-1)\mathbf{d}. \qquad (8)$$

これにより棒状の表現が可能である。図13がテクスチャの追加配置を利用した棒状の表現の模式図である。

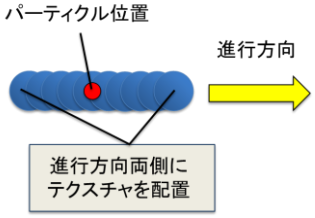

図13. 棒状のレンダリング

4.4.3 丸型と鋭利型

レンダリング手法が棒状の場合、飛沫の丸みというパラメータ$t(0 < t \leq 1)$により飛沫先端形状を表現する。tの値を描画するテクスチャサイズに乗算し、乗算時は先端のテクスチャほどtの影響力を強める。これにより、tが1に近いほど丸型となり、tが0に近いほど鋭利型となる。

j番目に表示するテクスチャサイズS_jは式(9)で求める。ただし、Sはパーティクル位置におけるテクスチャサイズである。

$$S_j = \frac{N - (j-1)(1-t)}{N} S. \qquad (9)$$

これにより鋭利型を表現可能である。図14がパラメータtを利用し、鋭利型にテクスチャサイズを変更する模式図である。

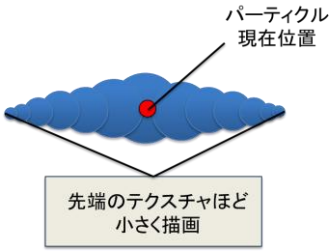

図14. テクスチャ配置による鋭利型の表現

5. 動作検証

本章では 4 章で述べた手法を実装する。実装した結果から、本手法の有用性を検証する。3D グラフィクス API は OpenGL をもとにした 3DCG ツールキットである Fine Kernel Toolkit System[16]を利用した。検証に使用した実行環境を表 2 に示す。

表2. 実行環境

OS	Windows 7 Enterprise 64bit
CPU	Intel Core i7 3.4GHz
メモリ	8.00GB
GPU	NVIDIA GeForce GTX 560 Ti

表 2 の実行環境でパーティクルの放出を続け、10 秒間の平均フレームレートを計測した。パーティクル数とテクスチャ数を 4 つのケースで計測した結果を表 3 に示す。連続型や棒状のレンダリングでは 1 つのパーティクルに対し複数のテクスチャが必要であるため、ケース 3、ケース 4 ではパーティクル 1 に対しテクスチャ数を 5 として計測した。

表3. 処理速度の計測

	ケース 1	ケース 2	ケース 3	ケース 4
パーティクル数	1221	2410	1221	2410
テクスチャ数	1221	2410	6105	12050
平均 フレームレート	85.1 FPS	31.3 FPS	26.5 FPS	10.4 FPS

パラメータの基準値を表 4 のように定め、比較検証を行う。表 4 の上段 4 つのパラメータが形状構成要素の 8 パターンを生成する本手法における重要なパラメータである。図 15 が基準となるパラメータ設定によるレンダリング結果である。テクスチャは 4.1 節の図 6 を利用した。

表 4. 基準のパラメータ設定

円状分布確率	α	1.0
連続確率	β	0.3
飛沫のサンプリング数	N	12
飛沫の丸み	t	0.5
パーティクルの寿命	τ	40
衝突時の加速度の大きさ (円状分布-連続型)	c	1.5 ランダム幅 0.5
衝突時の加速度の大きさ (円状分布-分離型)	c	2.5 ランダム幅 1.0
反発力入力値	r	4.0
バネ入力値	k	1.0
ダンパ入力値	h	0.8
テクスチャ表示間隔	u	0.4

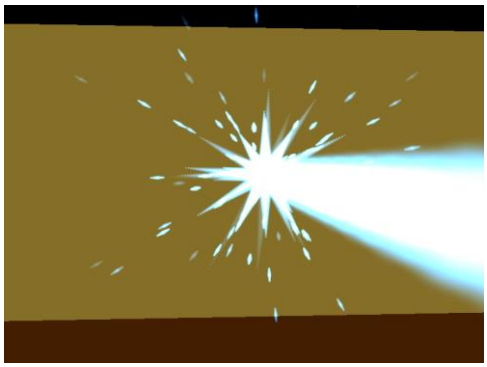

図 15. 基準のパラメータ設定でのレンダリング結果

図 16 では円状分布確率を変更し、大域的形状における円状と不規則の比較を行った。不規則状では制御点を 4 点利用した。

図 17 は連続確率を変更し、連続性における連続型と分離型の比較を行った図である。

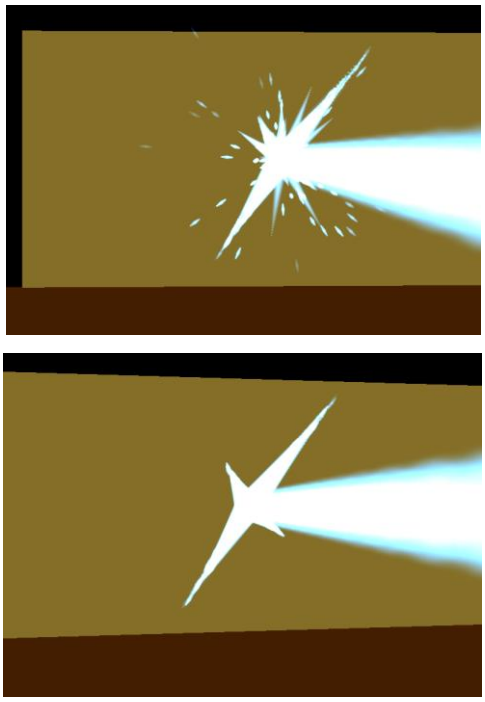

図 16. [上図] 円状+不規則状($\alpha = 0.5$)
[下図] 不規則状($\alpha = 0.0$)

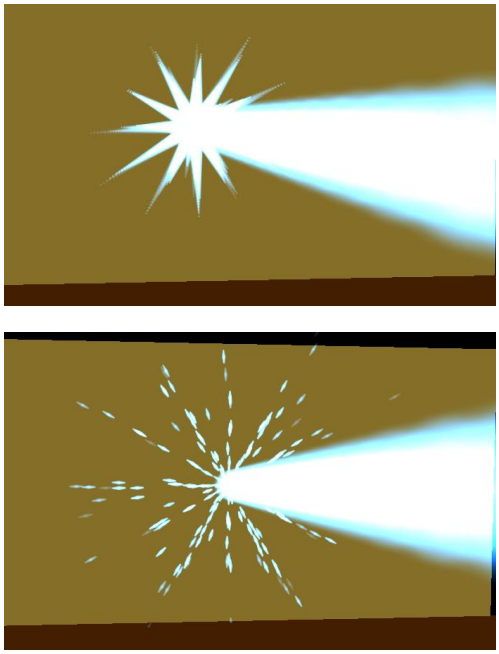

図 17. [上図] 連続型($\beta = 1.0$)
[下図] 分離型($\beta = 0.0$)

図 18 は飛沫の長さを変更し、飛沫形状における粒状と棒状の比較を行った図である。

図 19 は飛沫の丸みを変更し、飛沫先端形状における丸型と鋭利型の比較を行った図である。

図 16、図 17、図 18、図 19 の比較から、本手法の 4 つのパラメータ変更のみで 3.2 節で述べた形状を簡易に生成できることが確認できる。

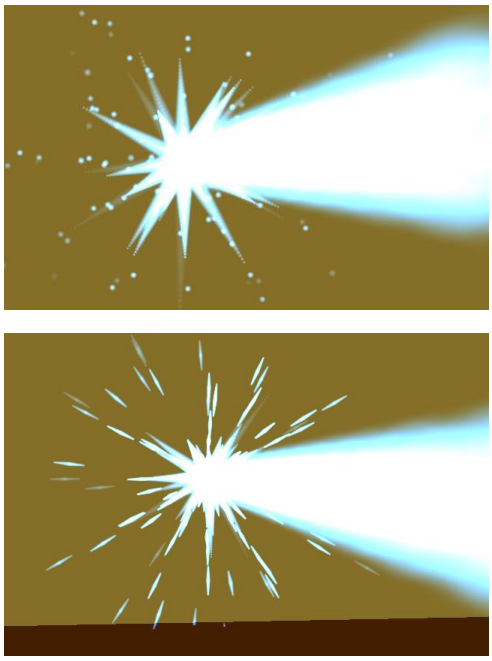

図 18.[上図] 粒状($N = 1$)
[下図] 棒状($N = 24$)

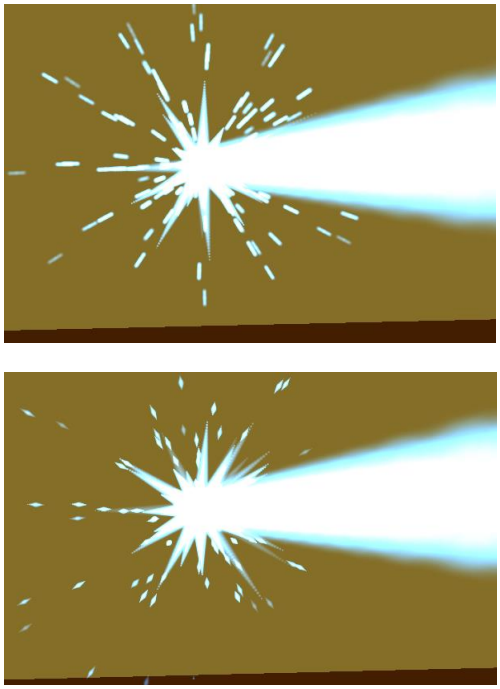

図 19. [上図] 丸型($t = 1.0$)
[下図] 鋭利型($t = 0.1$)

図 20 では基準のパラメータ設定で他の視点位置からの様子である。任意視点においても問題なくレンダリングが可能であることが確認できる。

図 21 は他の物体に 2 度衝突している様子である。多様な場面においても本手法が利用可能であることが確認できる。

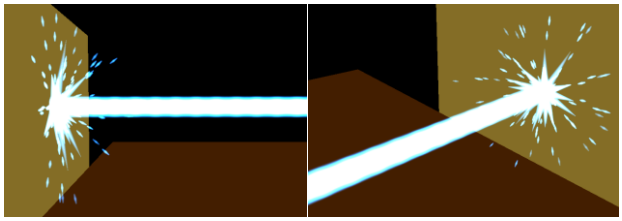

図 20. 他の視点位置からみた様子

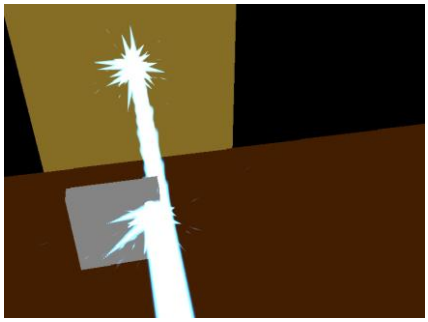

図 21. 物体に 2 度衝突している様子

図 22 ではパーティクルの分布を不規則状とし、制御点の位置へのランダム幅の利用や、制御点の移動によるアニメーション効果を利用した応用例を確認できる。

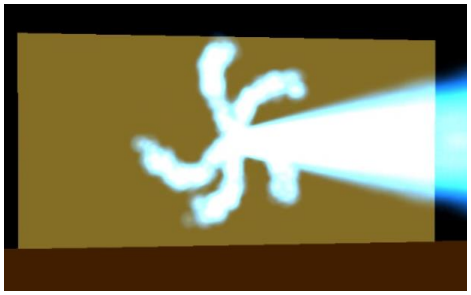

図 22. 制御点を利用した応用例

図 23 は実際のコンテンツと本手法の比較結果である。実際のコンテンツに似せるには 30 分程の時間を要したが、円状分布確率、連続確率、飛沫サンプリング数、飛沫の丸みの 4 つのパラメータを設定し概観を似せるだけならば数分で可能であった。

出典: "機動戦士ガンダムエクストリームバーサス"
©バンダイナムコゲームス、創通、サンライズ
[実際のコンテンツ]

[本手法]

図 23. 実際のコンテンツとの比較

6. おわりに

本研究ではリアルタイム3DCGにおいて、多彩なエネルギー波の衝突表現を簡易なパラメータ変更のみで生成可能な新たな手法を提案した。

本手法では、創作コンテンツにおける多数のエネルギー波の衝突シーンをもとに形状の分析を行った。分析の結果、形状構成要素を8パターン抽出した。パーティクルシステムと同様にエネルギー波をパーティクルの集合体として扱い、パーティクルを形状構成要素の8パターンのいずれかに所属させた。パーティクルの運動方程式やレンダリング手法をパターンごとに設定した。パーティクルのパターン所属にはパラメータや確率を利用

した。本手法により、多彩なエネルギー波の衝突形状を簡易なパラメータの変更のみで可能とした。調査した150シーンのうち70～80%程度の形状を本手法で再現が可能であると考える。

本研究では直方体への衝突のみを実装したが、凸形状であればより複雑な形状への衝突にも適用が可能である。しかし、非凸形状や複数の障害物への衝突時にはパーティクルが物体に複数回衝突することも考えられるが、本手法は複数回の衝突に関しては想定していない。現状の手法で複数回の衝突を行うと、パーティクルの所属パターンが円状の場合は、2度目以降の衝突時の大半のパーティクルは周囲に他のパーティクルが存在しないため、周囲のパーティクルからの反発力**R**は零ベクトルとなる。すなわち、大半のパーティクルは衝突前の速度に関わらず、衝突時の加速度**A**の向きは抗力**D**の向きと同じとなり、衝突後は衝突面に対し垂直に一定の速さで跳ね返る単調な動きとなる。パーティクルの所属パターンが不規則状の場合は、制御点の位置はエネルギー波ごとの相対的な方向と位置で決定している点と、寿命は何度目の衝突においてもユーザが定めたパラメータτとしている点により、パーティクルの衝突後の速度**v**は1度目の衝突時と変わらず、物体へめり込んでいく表現となる。このように、複数回の衝突表現については課題が残る。

また現状の手法では表現できないエネルギー波形状への対応も課題である。エネルギー波が物体と衝突後、流体のように衝突する物体に沿って動いていく形状や、分離したエネルギー波の形状が自由な曲線を描いている場合など、対応が難しい表現もある。さらに他の課題として、処理の高速化が必要である。パーティクルを大量に利用する場合や、品質の高いレンダリング結果を得るには多くのテクスチャの表示が必要であり、大きな処理負荷がかかる。この課題の解決策の1つとしてGPGPUやシェーダの利用を考えることができる。ジオメトリシェーダなどを利用することでより高速にテクスチャを表示し、CPUの処理負荷の軽減が期待できる。これらの課題を解決し、多種多様なエネルギー波の衝突形状を高速に表現することができれば、創作コンテンツにおける表現の幅をさらに広げることができるだろう。

参考文献

[1] 阿部雅樹, 渡辺大地, “エネルギー波表現のリアルタイムレンダリング”, 芸術科学会論文誌, Vol. 9, No. 3, pp. 93-101, 2010.

[2] マッチロック株式会社, http://www.matchlock.co.jp/.

[3] Unity Technologies, http://unity3.com/.

[4] W. T. Reeves, “Particle Systems. A Technique for Modeling a Class of Fuzzy Objects”, ACM Transactions on Graphics, Vol. 2, No. 2, pp. 91-108, 1983.

[5] Tomas Akenine-Möller, Eric Haines, “リアルタイムレンダリング 第2版 REAL-TIME RENDERING Second Edition”, ボーンデジタル, pp. 284-285, 2006.

[6] 仁藤将輝, 渡辺大地, “リアルタイム3DCGにおける衝突を考慮したエネルギー波表現”, NICOGRAPH2013 論文集, pp. 97-104, 2013.

[7] D. Nowrouzezahrai, J. M. Johnson, A. Selle, D.Lacewell, M. Kaschalk, W. Jarosz, “A programmable system for artistic volumetric lighting”, ACM Transactions on Graphics, Vol.30, Issue 4, Article No. 29, 2011.

[8] 仁藤将輝, 渡辺大地, “リアルタイム3DCGにおけるエネルギー波の光源効果の表現”, CEDEC 2012, http://cedil.cesa.or.jp/session/detail/980/.

[9] A. Treuille, A. McNamara, Z. Popović, J. Stam, “Keyframe control of smoke simulations”, ACM Transactions on Graphics, Vol. 22, No. 3, pp. 716-723, 2003.

[10] R. Fattal, L. Dani, “Target-driven smoke animation”, ACM Transactions on Graphics, Vol. 23, No. 3, pp. 441-448, 2004.

[11] Y. Dobashi, K. Kusumoto, T. Nishita, T. Yamamoto, “Feedback control of cumuliform cloud formation based on computational fluid dynamics”, ACM Transactions on Graphics, Vol. 27, Issue 3, Article No. 94, 2008.

[12] 佐藤周平, 土橋宜典, 山本強, 安生健一, “最適な初期強度分布の推定および予測制御による爆発シミュレーションの制御”, 映像情報メディア学会誌, Vol. 65, No. 10, pp. 1446-1451, 2011.

[13] L. B. Lusy, “A numerical approach to the testing of the fission hypothesis”, The astronomical journal, Vol. 82, No. 12, pp. 1013-1024, 1977.

[14] M. Müller, D. Charypar, M. Gross, “Particle-based fluid simulation for interactive applications”, In Proceedings of the 2003 SIGGRAPH/Eurographics symposium on Computer animation, pp. 154-159, 2003.

[15] K.Perlin, “Improving noise”, ACM Transactions on Graphics,Vol. 21, No. 3, pp. 681-682, 2002.

[16] Fine Kernel Project, Fine Kernel ToolKit System, http://fktoolkit.sorceforge.jp/.

仁藤 将輝

2012年東京工科大学メディア学部卒業. 2014年東京工科大学大学院バイオ・情報メディア研究科メディアサイエンス専攻修士課程修了.

渡辺 大地

1994年慶応義塾大学環境情報学部卒業. 1996年慶応義塾大学大学政策・メディア研究科修士課程修了. 修士(政策・メディア). 1999 年より東京工科大学メディア学部講師. 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. 情報処理学会, 芸術科学会, 画像電子学会会員.

柿本 正憲

1982 年, 東京大学工学部電子工学科卒業. 同年, (株) 富士通研究所入社. 以来, コンピュータグラフィックスの研究開発に従事. 1989～1990 年, 米国ブリガムヤング大学訪問研究員. 1993 年, 富士通研究所退職. CG 機器メーカー, 映像制作会社, 1995 年, 日本シリコングラフィックス (株) (現, 日本 SGI), 2011 年, シリコンスタジオ (株) を経て, 2012 年より東京工科大学教授. 2005 年, 在職のまま東京大学大学院情報理工学系研究科博士課程修了. 博士 (情報理工学). 芸術科学会, 画像電子学会会員. 情報処理学会グラフィクスと CAD 研究会主査.

三上 浩司

1995 年慶應義塾大学環境情報学部卒業, 博士 (政策・メディア : 2008 年慶応義塾大学). 日商岩井, 株式会社エムケイなどでメディアコンテンツのプロデュースに従事. 1999 年より東京工科大学片柳研究所クリエイティブ・ラボに従事し, 現在はメディア学部准教授. 主に 3DCG を利用したアニメ, ゲームの制作技術と管理手法に関する研究開発に従事. ACM SIGGRAPH, 芸術科学会, 情報処理学会, 日本デジタルゲーム学会ほか所属.