

2007年度 卒業論文

ボリュームレンダリングを用いた
エネルギー波のリアルタイム形状変形

指導教員：渡辺 大地講師

メディア学部 ゲームサイエンス
学籍番号 M0103014
阿部 雅樹

2007年度 卒業論文概要

論文題目

ボリュームレンダリングを用いた
エネルギー波のリアルタイム形状変形

メディア学部

学籍番号：M0103014

氏名

阿部 雅樹

指導
教員

渡辺 大地講師

キーワード

ボリュームレンダリング、テクスチャベース法、FFD
CG エフェクト、リアルタイム3DCG

近年、アニメーションやビデオゲームといったコンテンツ上で3DCGグラフィックスを用いた様々な表現が開発されている。その中でもエネルギーの塊が強く発光、形状変化するといった現象はよく見かけ、映画やテレビ番組など、実写映像における特殊効果としても使用される機会が増え、一般的な表現となりつつある。そうした特殊効果をエネルギー波と名付け、「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。エネルギーとは空間中に存在するエネルギー波を構成する要素で、3次元空間上に分布し強い熱量を持っているものとする。3次元のビデオゲーム上ではこうした効果を2次元のテクスチャを用いて表現するのが一般的だが、視点に応じた光の強さを正確に表現できる手法ではない。新たな表現方法として、ボリュームレンダリングを用いる研究もなされてはいるが、まだ問題点が残っている。

そこで本稿では、ボリュームレンダリングを用いた手法で生成したエネルギー波を、リアルタイムで簡易的に形状変化させる手法を提案する。本手法ではボリュームレンダリングで生成した形状をリアルタイムに、擬似的に形状変形することが可能である。本論文ではいくつかのボリュームレンダリングを伴った画面構成を用意し、リアルタイム性や擬似的形状変形の有用性を検証した。

目次

第1章	はじめに	1
1.1	研究の背景と目的	1
1.2	本論文の構成	2
第2章	現状のエネルギー波の表現手法	3
2.1	エネルギー波	3
2.2	ビデオゲームにおけるエネルギー波の表現手法	3
第3章	ボリュームレンダリングを用いたエネルギー波の表現手法	6
3.1	ボリュームレンダリングの概略	6
3.2	テクスチャベース法によるエネルギー波生成の概要	7
3.3	エネルギー波関数の定義	8
3.4	ボリュームデータの生成	9
3.5	ボリュームデータの読み込み	9
3.6	スライスの配置	10
3.7	テクスチャの回転	10
3.8	テクスチャマッピング	11
3.9	加算ブレンディング	12
第4章	エネルギー波の形状変形	13
4.1	任意の視点に応じたスライスとボリュームテクスチャの再配置	14
4.2	制御軸の用意	15
4.3	局所座標系の算出	16
4.4	変形と補正	18
第5章	動作検証と考察	22
5.1	動作検証	22
5.2	考察	24
第6章	まとめ	27
	謝辞	29

第 1 章

はじめに

1.1 研究の背景と目的

アニメーションやビデオゲームなどの映像コンテンツにおいて、しばしばエネルギーの塊のようなものが強く発光、形状変化をする表現が用いられる。代表的な例として漫画「ドラゴンボール [1]」に出てくる気功波やビデオゲームにおける魔法のエフェクトなどが挙げられる。近年、こういった表現はアニメーションやビデオゲームだけでなく、映画やテレビ番組など、実写映像における特殊効果としても使用される機会が増え、一般的な表現となりつつある。本研究ではこの表現を総称してエネルギー波と呼称する。本稿ではエネルギー波を「空間中に存在するエネルギーの密度が高い部分が強い光で発光する」「形状変化を伴いながら目標に向かい移動する」現象と定義する。

現在の 3 次元のビデオゲーム上でエネルギー波を表現する手法で最も一般的なものが、2 次元のテクスチャを用いる手法である。視点に応じたポリゴンの回転やテクスチャの切り替えによってエネルギー波を表現している。この手法は 1 枚の絵としてエネルギー波を表現しているため、例えばゲームのユーザーが視点を自由に指定できる場合、視点に応じたエネルギー波の正確な光の強さは表現できない。

視点に応じたエネルギー波の正確な光の強さを表現する手法として、ボリュームレンダリングが挙げられる。坂井ら [2] はボリュームレンダリングのうち、レイ

キャストリング法とテクスチャベース法 [3][4][5][6] を用いてエネルギー波を生成し、どのような視点からでもエネルギー波の正確な光の強さを表現する事に成功した。しかしレイキャストリング法は高精度だがリアルタイム性に乏しい従来のボリュームレンダリングであり、テクスチャベース法はGPUを用いたボリュームレンダリングの代表的な処理で、レンダリングの処理の一部を任せる為処理は高速でも、精密さや処理できる容量に限界があるなど、一長一短な部分がある。ボリュームレンダリングに関する研究はGPUを用いての並列処理による高速化や [7][8]、GPUの処理系統での高速化が主流である [3]。そのため坂井らの手法で生成したエネルギーも、空間上に配置するだけは可能だが、形状変形や移動などエネルギー波の特性を表現する事は未解決のままである。

そこで本研究ではボリュームレンダリングを用いたエネルギー波のリアルタイム形状変形を目的とする。リアルタイム処理が他のボリュームレンダリングの手法より比較的容易であるテクスチャベース法に着目した。さらに処理に時間のかかるボリュームデータの改変ではなく、処理速度の速いテクスチャを改変する事で形状変化を表現する。変形手法には大域的形状変形 [9][10][11](以降「FFD」)の一種である軸変形に山野上ら [12] が改良を加えた手法を利用することで、複雑な形状変形も容易に実現可能となる。本稿では山野上が改良した軸変形を、「副軸を用いた軸変形」と呼称する。

1.2 本論文の構成

本論文は全6章で構成する。第2章で既存のエネルギー波の表現手法について述べ、次に第3章で本論文で利用するボリュームレンダリングを用いたエネルギー波について述べる。続く第4章で本論文で提案するボリュームレンダリングを用いたエネルギー波の変形手法を述べ、第5章において第4章を実際に適応した結果から検証、考察を行なう。最後に第6章において、本研究のまとめと展望を述べる。

第 2 章

現状のエネルギー波の表現手法

本稿で対象とするエネルギー波の概要とビデオゲームにおける一般的なエネルギー波の表現を述べる。

2.1 エネルギー波

本稿ではエネルギー波を「空間中に存在するエネルギーの密度が高い部分が強い光で発光する」「形状変化を伴いながら目標に向かい移動する」現象と定義する。エネルギーとは空間中に存在する、エネルギー波を構成する要素で、3次元空間上に分布し強い熱量を持っているものとする。

エネルギー波が世間一般に広がった切っ掛けとして、松本零士原作の「宇宙戦艦ヤマト [13]」や、鳥山明原作の「ドラゴンボール」などのアニメーション作品だと考えることが出来る。その中でも特に、「ドラゴンボール」は漫画やアニメーション、ビデオゲームなどのコンテンツとして広く世界にまで知れ渡っている作品である。作中に出てくる「かめはめ波」は特に、攻撃時のポージングや掛け声も含めて、最も多くの人に馴染みの深いエネルギー波だといえる。

2.2 ビデオゲームにおけるエネルギー波の表現手法

現在3次元のビデオゲームでエネルギー波を表現する手法として最も一般的なものが、2次元のテクスチャを用いる手法である。視点の変更に応じてテクスチャ

をマッピングしたポリゴンを回転させたり、テクスチャを切り替えたりすることでエネルギー波を表現する。図 2.1 は視点に応じてテクスチャを貼ったポリゴンが回転している様子を表す。

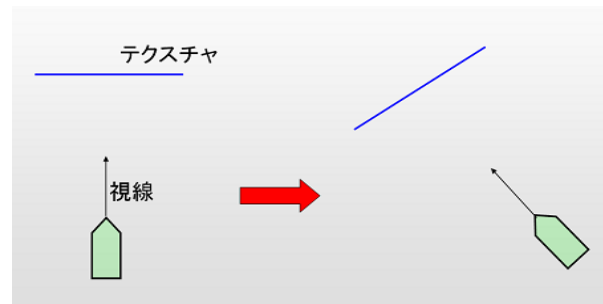


図 2.1: 視点に応じたポリゴンの回転

初めに数種類の視点から見たエネルギー波のテクスチャを画像を用意する。用意する枚数が多いほどエネルギー波の表現は豊かになる。

次にテクスチャを貼り付けたポリゴンを視点と垂直になるよう配置する。視点に応じて常に垂直になるよう保つ事によって、一枚の画像で表現していることを悟られづらくする。また、視点に応じて用意したテクスチャ画像の中から最適な画像をマッピングし直す。

このように 3 次元のビデオゲームではエネルギー波を 1 枚のポリゴンの回転と、複数のテクスチャの切り替えによって 1 枚の絵として表現している。しかしこの方法は用意したテクスチャ画像の位置でしかエネルギー波の正確な光の強さを表現できていない。正確な表現が出来る位置を増やしたいとなると、位置に応じた分だけテクスチャを用意することになり、非効率的である。

これに対してテクスチャとは別の表現手法も存在する。坂井らはエネルギー波の表現にボリュームレンダリングを用いることで、どの地点からエネルギー波を見ても、その光の強さを正確に表現する事を可能とした。この手法は視点に応じたエネルギー波の正確な光の強さは表現できるが、そのエネルギー波はただ空間

中に浮いているだけである。この状態はエネルギー波の「形状変化を伴いながら目標に向かい移動する」という特性は表現できていない。そこで本研究では坂井らのエネルギー波を元に、リアルタイムに形状が変化する事を目的とする。エネルギー波の形状変形の手法には副軸を用いた軸変形を利用する。

第 3 章

ボリウムレンダリングを用いたエネルギー波の表現手法

エネルギー波の表現は坂井らの提案したボリウムレンダリングを用いた手法を利用する。本章では本稿で扱うエネルギー波のレンダリング手順について述べる。

3.1 ボリウムレンダリングの概略

ボリウムレンダリングとは、ボリウムと呼ばれる 3 次元の半透明なオブジェクトを 2 次元平面状へ投影する手法の総称である。ボリウムは多くの場合、3 次元空間をサンプリングしたボクセルと呼ばれる単位立方格子の集合体として表す。ボリウムレンダリングは、はっきりとした境界を持たない炎や煙といった自然現象やエネルギー場、人体の内部などの可視化に利用する機会が多い [14]。ボリウムレンダリングは大別して、全てのサンプル点の寄与を計算して全体を表示する直接法と、前処理によって表示する情報を抽出してデータの一部を表示する間接法の 2 種類に分類できる。それぞれ複数手法があり、手法によって描画速度や得られる画像に大きな差異がある。またボリウムレンダリングを行うことに特化したツールやライブラリ、ハードウェアも多く存在する [15][16][17]。

通常ボリュームレンダリングとは直接法を指す。直接法は正確な画像が得られる反面、膨大な計算時間や記憶容量が必要になるため、大規模な計算機環境や特殊ハードウェアを使用できる場合に利用される [18]。一方で GPU の性能が向上している近年では、その機能を利用したテクスチャベース法と呼ばれる手法が提案されている。テクスチャベース法は処理の一部を GPU で行うことによって高速な処理を可能にする手法である。本稿ではリアルタイム性を重視しているため、このテクスチャベース法で生成したエネルギー波を利用する。

3.2 テクスチャベース法によるエネルギー波生成の概要

坂井らの研究では、テクスチャベース法によるエネルギー波の表現を、以下の手順で行った。

1. エネルギー波関数の定義
2. 3次元配列にエネルギー波の密度分布を格納
3. 3次元配列をテクスチャとして GPU に読み込ませる
4. 視点に垂直な矩形ポリゴンを複数枚配置
5. テクスチャの回転
6. 矩形ポリゴンにテクスチャの断面をマッピング
7. 視点から遠い順に α ブレンディング

テクスチャベース法では、エネルギー波の密度分布をテクスチャとして GPU に読み込ませ、テクスチャの断面をマッピングした複数枚のポリゴンを視点から遠い順に α ブレンディングすることで最終的なピクセルの色を得る。この時、エネルギー波の密度分布情報を保持したテクスチャをボリュームテクスチャと呼ぶ。視線に対して常に垂直な矩形ポリゴンをスライスと呼ぶ。以下、個々の処理について説明する。

3.3 エネルギー波関数の定義

エネルギー波の形状は各ボクセルに格納されている数値によって決定する。図 3.1 は各ボクセル毎に値を代入することでボリュームデータを生成している様子を表している。

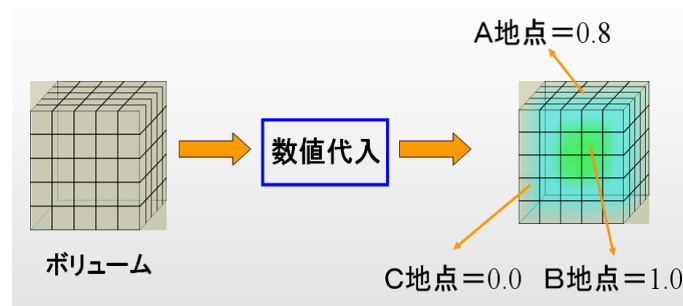


図 3.1: ボクセルに値を代入

一つ一つのボクセルに任意の値を設定してやればどのような形状でも表現可能だが、精密な表示ほどボクセル数は多くなり、その分エネルギー波の形状生成にかかる労力が大きくなる。そこで、最初にエネルギー波を作るためのエネルギー波関数を定義することで、生成にかかる労力を削減する。坂井らの研究では球体と円柱を組み合わせた形状を生成した。図 3.2 は最終的に得られるエネルギー波の例である。

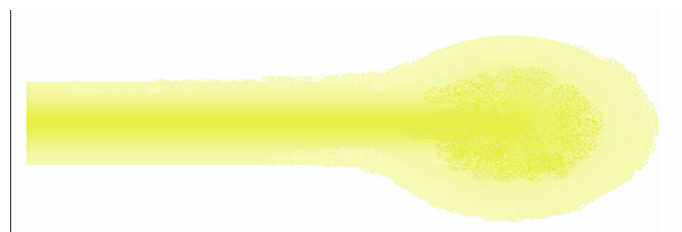


図 3.2: 得られるエネルギー波の形状

3.4 ボリュームデータの生成

関数を定義したら、エネルギー波をエネルギーの密度分布として3次元配列に格納する。空間をボクセルに分割し、各ボクセルの密度値と3次元配列の要素の対応をとる。配列の各要素は4つの値 $RGB\alpha$ をそれぞれ格納する。RGB 値が色情報、 α 値が透明度の情報である。この $RGB\alpha$ の値がボクセル値に相当し、3次元配列がボリュームデータに相当する。任意のボクセルにおいて、その地点の密度を変更したい場合には α 値を変更することで、その場所でのエネルギーの密度を変更するのと同じ効果を得ることができる。エネルギー波は中心に近いほど密度分布が濃くなっているため、中心から外側に向かって α 値を減少する。

3.5 ボリュームデータの読み込み

生成したボリュームデータはGPUにテクスチャとして読み込む。テクスチャは色情報、透明度情報といったデータの多次元配列である。テクスチャの配列内の個々のデータをテクセルと呼ぶ。GPUが3次元テクスチャをサポートしている場合、ボリュームを3次元テクスチャとして扱うことが可能である。今日ではパーソナルコンピュータに搭載できる比較的安価な汎用GPU[19][20][21]でも3次元テクスチャの機能をサポートしている。よって坂井らもボリュームテクスチャには3次元テクスチャを使用している。

ボリュームはGPUの処理によってCPUのメインメモリからGPUのビデオメモリにグラフィクスAPIを介して転送し、3次元テクスチャとして保持する。この際、GPUはテクスチャ座標を0~1の範囲に正規化する。3次元テクスチャの場合は x, y, z 座標のそれぞれが0~1の範囲に収まることに注意が必要である。この時、正規化した範囲をテクスチャ座標と呼称する。

3.6 スライスの配置

スライスが配置される3次元空間の座標系を、ワールド座標系と呼ぶ。実際に画像を描画するスクリーン上の座標系をスクリーン座標と呼ぶ。エネルギー波を画面中央に表示する為に、スクリーンの中心とワールド座標系の原点を一致させる。視点をZ軸上に、視線方向をワールド座標系の原点に設定する。全てのスライスの中心がZ軸上にあるようにスライスを等間隔に配置する。スライスにボリュームテクスチャをマッピングする事でエネルギーの密度分布を再構築する。図3.3はスライスをZ軸上に等間隔で配置した例である。

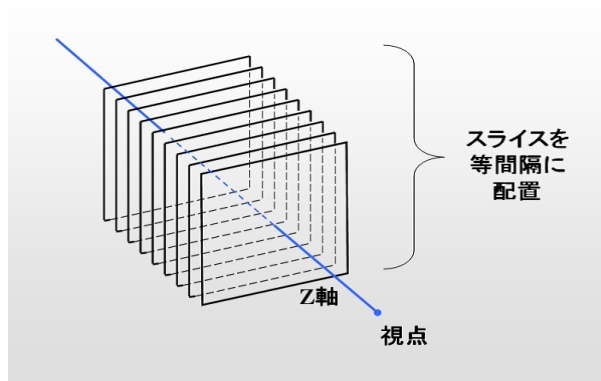


図 3.3: スライスを等間隔に配置

3.7 テクスチャの回転

観測地点の変更はボリュームを回転する事で表現する。これはスライスを常に視点に対して垂直に保つためである。視線とスライスの成す角度が減少すると、それに応じて投影面積が狭まりサンプリングするボクセルの数が減少する。その結果、正確なボリュームレンダリングを行うことができなくなる。坂井らはテクスチャ座標に回転行列を掛ける事でテクスチャを回転し、視点を変更している事と

同じ状況を設定した。つまり、視点変更の際に実際に変更を施すのはテクスチャの図柄であり、スライスの位置は常に一定である。

3.8 テクスチャマッピング

テクスチャマッピングとはポリゴンに画像を貼り付ける操作のことである。テクスチャマッピングでは、ポリゴンの各頂点に、読み出すテクセルのテクスチャ内での位置を表すテクスチャ座標との対応をとる。ポリゴンの頂点座標とテクスチャ座標を対応をとる事は、ワールド座標系とテクスチャ座標系が一致する事と同義である。図 3.4 どのテクスチャ座標がどの頂点座標と対応しているかを表している。

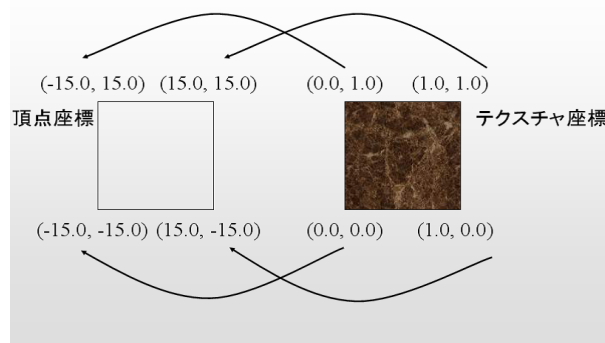


図 3.4: テクスチャマッピング

テクスチャベース法において、スライスの Z 座標にボリュームテクスチャの Z 座標を対応付ける事によって、スライスの位置にあわせてマッピングするボリュームテクスチャの断面が変化することになる。図 3.5 はスライスの位置に応じてボリュームを切り取る断面が変化している事を表している。

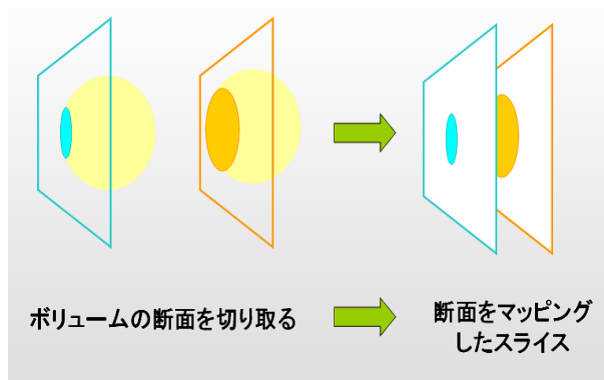


図 3.5: ポリュームを切り取る位置によって断面が変化

通常は対応した座標同士は一様に決定するが、あえて対応する座標同士を個別に変化させる事で規則性を乱すことは可能である。

3.9 加算ブレンディング

ポリュームテクスチャを張ったスライスを、視点の位置から遠い順に順次加算ブレンディングする。加算ブレンディングとは画像同士を重ね合わせるときに、透明度を表す α 値に基づいて色を足し合わせる処理である。加算ブレンディングを用いて画像を重ねる事で、ピクセルの透明度を考慮した画像を得る。以上がテクスチャベース法によるエネルギー波の表現手法である。図 3.6 は以上の手順で得た結果の画像である。

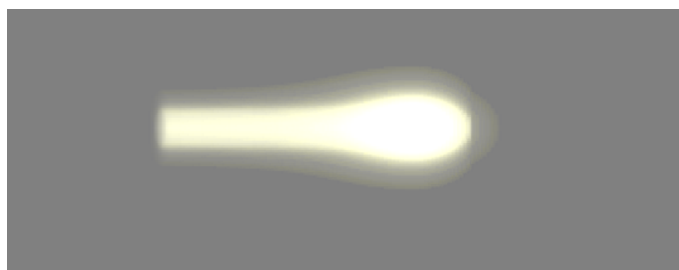


図 3.6: テクスチャベース法によるエネルギー波

第 4 章

エネルギー波の形状変形

本稿ではボリュームレンダリングを用いたエネルギー波を、副軸を用いた軸変形を利用してリアルタイムに形状変形する手法を提案する。ボリュームレンダリングを用いたエネルギー波には、坂井らが提案したエネルギー波関数と、リアルタイム性の高いテクスチャベース法を使用する。エネルギー波の形状変形には副軸を用いた軸変形を利用する。副軸を用いた軸変形は、変形したいモデルに対して変形媒体と呼ぶ媒体を新たに設定し、変形媒体とモデルの頂点を関連付ける。関連付けた変形媒体を操作する事によって、モデルの形状を変形していく手法である。また GPU の性能でテクスチャはキャッシュとして保存する事に着目し、本手法ではモデルの頂点の代わりに、ボリュームテクスチャの頂点を関連付け、変形の対象とする。

本研究では以下の手順でエネルギー波の形状変形を行った。

1. 任意の視点における、スライスとテクスチャの再配置
2. 制御軸の用意
3. 制御曲線における局所座標系の決定
4. 制御曲線の変形と補正

坂井らの研究ではテクスチャを回転することで視点の変更を表現しており、厳密な視点の変更は不可能である。そのためボリュームデータ以外のオブジェクトと同時に描画する事が出来ない。はじめに視点を任意に変更できるようにする為、視点に伴いスライスとテクスチャの再配置をする。

エネルギー波の形状変形は軸変形を用いる。通常変形媒体となる軸上の点と対応するものは変形対象であるモデルを構成する頂点であるが、本手法ではマッピングするテクスチャの座標を変形媒体と対応付ける。以上の手順で動的なエネルギー波の形状変形を可能にし、それに応じた視点の変更も可能にした。本章では以降、視点とエネルギー波の回転軸をワールド座標系の Y 軸とする。以下に個々の処理について説明する。

4.1 任意の視点に応じたスライスとボリュームテクスチャの再配置

通常テクスチャベース法は視線に対して常にスライスを垂直に保つという制約がある。視点の変更はボリュームを回転させる事で表現しているが、このままではボリュームデータとそれ以外のオブジェクトを同時に表示したときに視点の変更が出来ないことになる。視点が変わったらスライスの向きとボリュームテクスチャを再配置する必要がある。

まず、ワールド座標系における視点の初期位置ベクトルを V_1 、Y 軸を回転軸とし、回転後の視点の位置ベクトルを V_2 とする。 V_1 と V_2 の成す角度を θ とする。スライスは常に視点に対して垂直に配置するので、スライスも視点と同様に Y 軸を回転軸とし、 θ 分だけ回転する。この状態に 3 章で述べたテクスチャの回転を適応する。つまり、視点の回転に応じて同じ角度分、スライスとテクスチャを回転することで視点の変更を可能にした。なお視点はその位置に因らず、常に原点の方向を向いているものとする。

4.2 制御軸の用意

副軸を用いた軸変形では、変形媒体となる軸は制御曲線と副軸で構成する。モデルと制御曲線の間係を図 4.1 に示す。

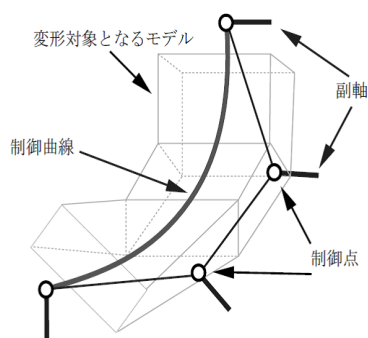


図 4.1: 制御軸の構成

初めに、変形媒体となる制御曲線を定義する。制御曲線は B-Spline 曲線 [22] で定義する。B-Spline 曲線とは、階数 s 、制御点 $R_i (0 \leq i < n)$ 、ノット列 x_i から成る曲線である。階数とは制御点の変形が影響する範囲を指し、ノット (knot) 列はその影響範囲を指定するための節目 (パラメータ区間の境界値) である。各制御点毎に B-Spline 基底関数と呼ぶ関数が存在する。B-Spline 基底関数とは、曲線の形状を変形した場合に、その変形に各制御点の位置がどの程度影響しているかということを示す関数で、 $k - 1$ 次の B-Spline 基底関数 $N_{i,k}(t)$ は以下のように定義する事が出来る。なお k を位数と呼ぶ。

$$\begin{aligned}
 N_{i,1}(t) &= x_i \leq t \leq x_{i+1} \quad \text{なら } 1 \text{ それ以外は } 0 \\
 N_{i,k}(t) &= \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \\
 \text{ただし } \frac{0}{0} &= 0
 \end{aligned} \tag{4.1}$$

本稿ではスライスを中心点を通る位置を制御曲線の初期位置として定義する。つまり視線、ワールド座標系のZ軸上に、直線の形状で配置する。

4.3 局所座標系の算出

次に変形媒体上の点Mにおける局所座標系を求める。正規化した3次元テキスト座標系から切り取ったボリュームテキストの頂点Pを、制御曲線に投影する。ここで投影とは、頂点Pから最も近い制御曲線状にある点Mを得ることである。

ここで頂点Pから制御曲線上に投影した点を M_t とする。 t は制御曲線におけるパラメータであり、 $0 \leq t \leq 1$ である。なお、 t がこの範囲に収まるように、視点から見て1番手前から1番奥までのスライス間の距離と制御曲線の長さが一致するように設定する。 M_t における局所座標系の算出方法で最も有名なのがフルネー機構 (Frenet frame) であるが、軸の姿勢を制御し辛く、局所座標系が簡単にねじれてしまうなどの問題がある。図4.2は局所座標系の一部である、制御軸の法線が反転している様子を示している。

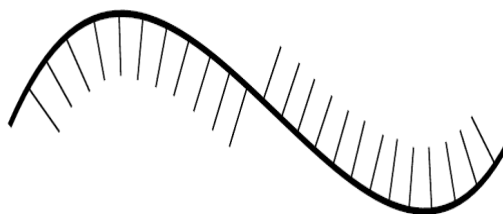


図 4.2: 法線の反転

この問題を解決するために副軸を生成する。副軸の方向と B-Spline 基底関数を利用すると、軸の姿勢や曲座標系が安定して求めることが可能になる。副軸は各

制御点から伸びる有向線分であり、本手法では初期方向をワールド座標系の Y 軸方向に正の向きをしているものと定める。以降副軸を用いた軸変形において、副軸を伴った制御曲線を制御軸と呼称する。図 4.3 は制御軸の初期状態を表す。また本稿では制御点 R を 5 個と設定している。

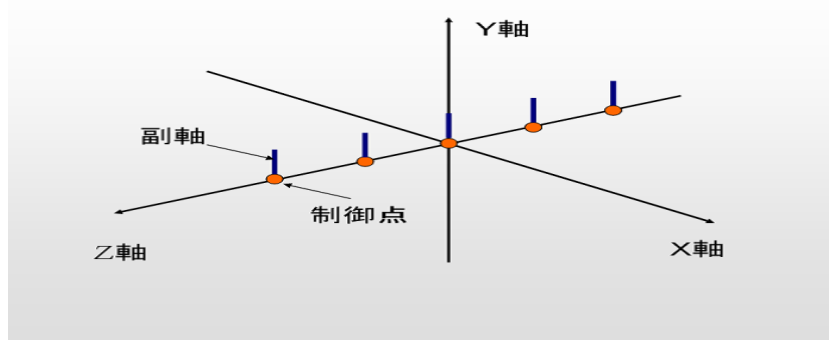


図 4.3: 制御軸の初期位置

M_t における局所座標系を $\sum(M_t; e_1, e_2, e_3)$ とする。なお、各座標軸を表すベクトル e_1, e_2, e_3 は単位ベクトルとする。 $\sum(M_t)$ は次のようにして求めることができる。

1. M_t における一階微分ベクトルを式 (4.2) を用いて求める。微分ベクトルは曲線の接ベクトルなので e_1 となる。

$$\mathbf{e}_1 = \sum_{i=0}^n N'_{i,n}(t) \mathbf{R}_i \quad (4.2)$$

$$N'_{i,1}(t) = 0$$

$$N'_{i,k}(t) = \frac{N_{i,k-1}(t) + (t - x_i)N'_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N'_{i+1,k-1}(t) - N_{iL1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

2. 副軸ベクトルを d_i として、点 M_t における e_2 を式 (4.3) を用いて求める。

$$\mathbf{e}_2 = \sum_{i=0}^n N_{i,n}(t) \mathbf{d}(i) \quad (4.3)$$

この際 B-Spline 曲線基底関数 $N_{i,k}(t)$ は式 (4.1) を用いる。

3. 点 $M(t)$ における e_3 を e_1 と e_2 の外積から求める。

$$e_3 = e_1 \times e_2 \quad (4.4)$$

以上が点 M_t における局所座標系 $\Sigma(M_t; e_1, e_2, e_3)$ である。図 4.4 は局所座標系 Σ を表している。

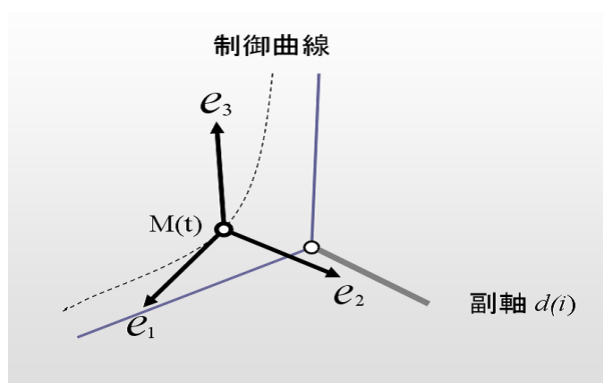


図 4.4: 局所座標系 Σ

4.4 変形と補正

まず変形の手法について述べる。制御軸の形状は制御点の位置によって決まるため、制御軸を変形するためには制御点を移動すればよい。本章では制御軸を X 軸方向へのみ変形するものとして説明する。通常テクスチャ座標は $(0.0, 0.0)$ 、 $(0.0, 1.0)$ 、 $(1.0, 1.0)$ 、 $(1.0, 0.0)$ の正規化した四頂点を指定するが、本手法ではテクスチャ座標と制御軸との対応をとり、それ以外の範囲を指定することによってポリウムデータの移動や形状変形を表現している。これはスライスに貼り付ける画像を加工しているという事になる。図 4.5 は正規化した四頂点以外を指定したときに得ることができる画像である。左の画像は本来中央にあった円が、テクスチャの頂

点座標を移動する事で、左にずれて表示している様子を表している。中央の画像はテクスチャの頂点座標を正規化した四頂点より拡大する事で円を縮小し、右の画像は逆に頂点座標の範囲を狭めることで円を拡大して表示している。

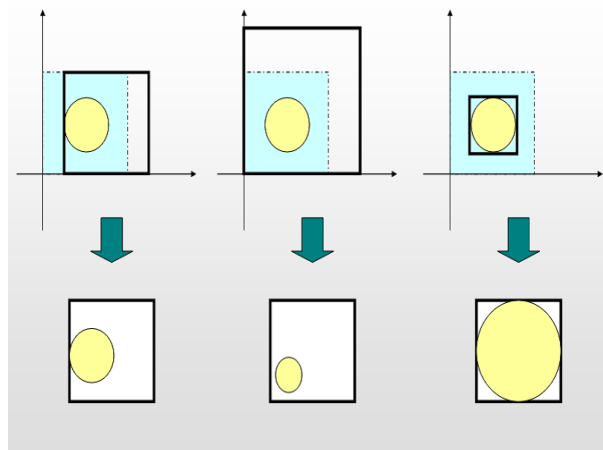


図 4.5: 正規化した範囲外を指定

図 4.6 は以上の変形手順によって得ることが出来るエネルギー波の変形結果を俯瞰した様子を表している。本手法ではテクスチャ座標の位置を変更しているため、スライス的位置は一定である様子が確認できる。

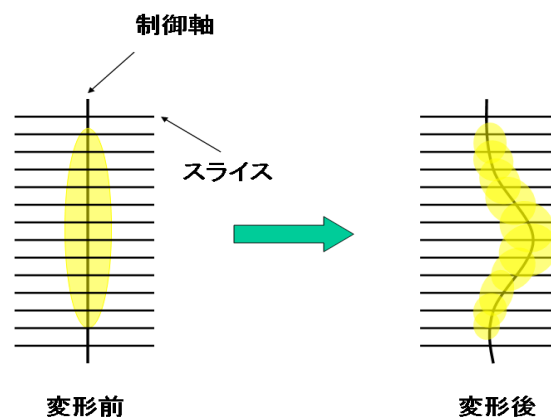


図 4.6: 制御軸変形によってエネルギー波の形状を変形する

本章の第1節では、視点を任意に変更できると述べた。これは視点と同時にテクスチャ座標を回転しており、制御軸が初期の直線形状から変化していない事を前提としている。また、本節の冒頭では制御軸を用いたエネルギー波の変形方法について述べた。こちらは制御軸がX軸方向にしか変形しないため、テクスチャ座標は同一平面上で移動することになり、視点が初期位置から変化しない事を前提としている。制御軸の形状変形を行った後に視点位置の変更をする状況はこれまでの手順では考慮していないため次の問題が発生する。図4.6で変形を施したエネルギー波の例を示しているが、これは初期視点位置から観察した結果である。この状態から視点を変更する場合、視点に応じてスライスも回転するため、それぞれのボリュームテクスチャが描く断面図を変更しなければ、視点変更前と同じエネルギー波の変形結果を得ることが出来ない。つまり視点に応じてテクスチャ座標を再設定する必要がある。図4.7は再設定を施した理想図であり、図4.6で変形を施した形状を、初期視点位置に対して視線が斜めになる位置から観察した場合、同一のスライスに貼り付けるボリュームテクスチャが異なることを表している。

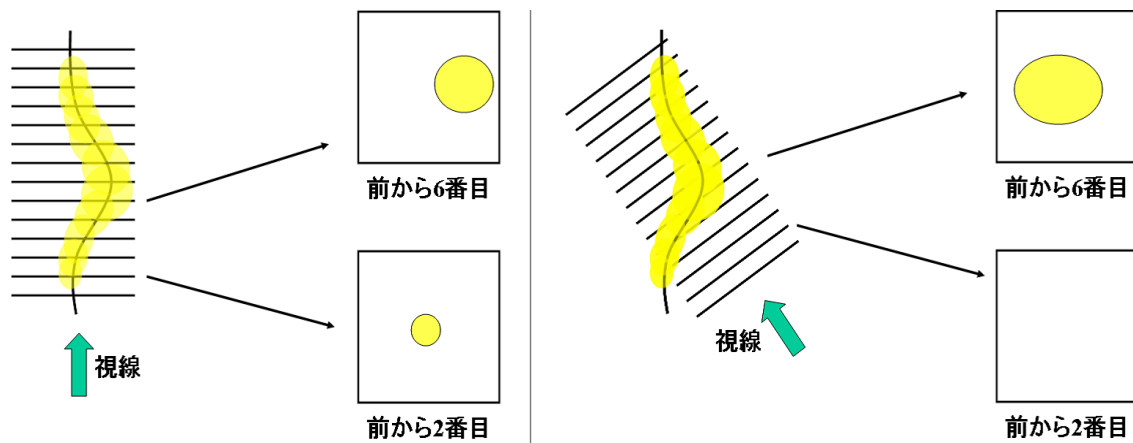


図 4.7: 同じ結果を得る為にはテクスチャ座標の再定義を必要とする

厳密には再設定をすることは可能だが、代数的に困難である。そこでこの問題を解決するために、本手法では制御軸の形状に対して視点に応じた補正を加える。

視点と制御軸の対応の取り方は次の通りである。視点の初期位置はワールド座標系の Z 軸上の正の位置とし、視線は原点を向く。この視点位置を V_1 とする。初めに制御軸を変形する。次に視点位置をワールド座標系の Y 軸を中心に、同座標系の X 軸上の正の位置に変更する。この位置を V_2 とする。視線は常に原点を向いているため、この視線も原点方向に向く。これは初期視点位置から Y 軸を中心に 90° 回転した位置に視点を移動したことになる。制御軸は X 軸方向にのみ変形している為、制御軸とボリュームデータの Y 軸方向や Z 軸方向の変位はない。したがって V_2 からの視点の場合、制御軸は初期状態と同じとみなせる。 n を制御点の個数とし、初めに変更した各制御点の位置を P_i とする。制御軸が直線状になる各制御点の位置、つまり各制御点の初期位置を Q_i とし、視点が V_1 から V_2 へ回転する間、 P_i と Q_i を線形補間することで、視点の回転途中における各制御点の位置 X_i を求める。使用する式 (4.5) は以下である。 t は V_1 から視点を回転した角度 θ とする。

$$X_i = \left(\frac{\pi}{2} - \theta\right)P_i + \theta Q_i \quad (4.5)$$

ただし $0 \leq \theta \leq \frac{\pi}{2} \quad i = 0, 1, \dots, 4$

次に視点を Z 軸上の負の方向へ回転する。この視点位置を V_3 とする。 V_3 から見た制御軸は V_1 から見たものと真逆になる。 V_2 から V_3 に回転する間も先ほどと同様に、線形補間を用いて各制御点の位置を算出する。X 軸上の負の方向にある視点位置を V_4 とし、 V_4 へ視点を移動する間も同様にして、制御点の位置を求める。以上のように制御軸に対して視点に応じた補正をかけることにより、どの視点から見ても同一の形状を参照できる状態が表現できる。また本手法では制御軸の形状を変化した後に視点の変更を行っているが、順序を逆にしても最終的な描画結果は変わらない。

以上がボリュームレンダリングを用いたエネルギー波の形状変形手法である。この手法を使えばリアルタイムでのボリュームレンダリングを用いたエネルギー波の形状変形が可能になる。

第 5 章

動作検証と考察

本章では、第4章で述べた手法によるボリュームレンダリングによるエネルギー波の形状変形の実行例を示すと同時に、本手法の有用性の検証を行う。次いで本手法に対しての考察を行う。

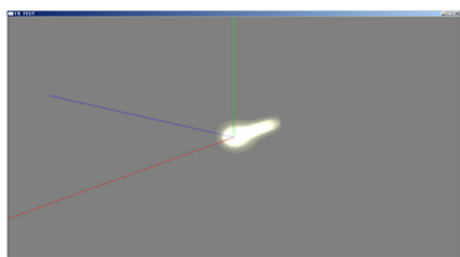
5.1 動作検証

制御軸を用いた形状変形をボリュームレンダリングによるエネルギー波に適応した結果を示す。実装にはグラフィックツールキットである FKSystem[23]、GPU用のグラフィックス言語に nVidia 社の Cg 言語を使用し、4章で説明した内容で実装を行った。形状変形手法を実行した環境は表 5.1 の通りである。

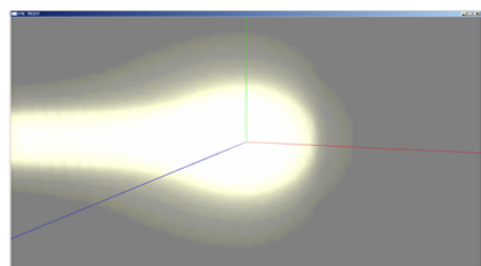
表 5.1: 実行環境

OS	WindowsXP SP2
CPU	AMD Athlon64 2.19GHz
GPU	NVIDIA Geforce 6600GT グラフィックスメモリ 128MB
メモリ	1024MB

エネルギー波の変形前と変形後の動作フレームレートを計り、更に実際に変形処理にかかった時間を計測する事でリアルタイム性の有無を確かめた。フレームレートとは1秒間に描画画面が更新する回数を表し、単位はfps(Frames Per Second)で表す。数値が大きければリアルタイム性も高くなる。変形前において、2つの異なったフレームレートの状況を用意し、どのような状況でも本手法が有用であることを確かめた。描画画面に対して、エネルギー波を描画する領域の割合が大きくなればなるほど、フレームレートは低くなる。図5.1、5.2はそれぞれ、エネルギー波の描画領域が異なる2つの状況において、変形前と変形後のエネルギー波の形状とそれぞれのfpsを表している。

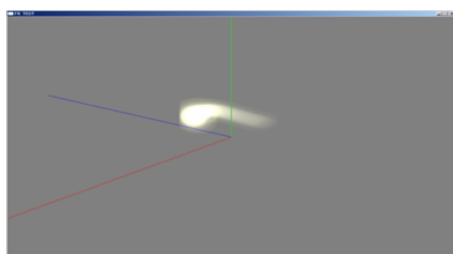


74fps

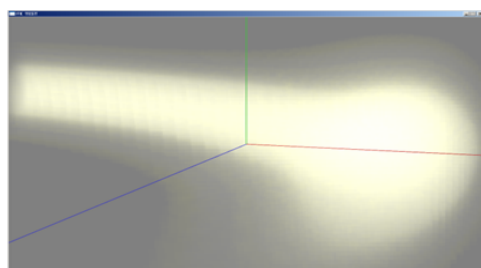


15fps

図 5.1: 変形前のそれぞれの fps



74fps



15fps

図 5.2: 変形後のそれぞれの fps

2つの異なった状況において、実際に変形処理にかかった時間を計測した表が5.2である。

表 5.2: 実際の処理時間

変形前後の fps	変形処理にかかった時間
74fps	0.003 ~ 0.005 秒
15fps	0.003 ~ 0.005 秒

以上により、本手法はリアルタイムな処理が可能だといえる。

次に制御軸をワールド座標系の X 軸方向へ変形操作を行った後に、視点の変更を行った処理結果を示す。原点からの距離を一定に保ち、視点をワールド座標系の Y 軸を中心に回転した。図 5.3 はそれぞれ、初期視点位置と初期視点位置から X 軸の正の方向へ 15° ずつ、 90° まで回転した位置から観察したエネルギー波である。

制御軸を変更した後に視点を変更しても、違和感無くエネルギー波を描画する事が出来たといえる。

5.2 考察

テクスチャベース法は処理の1部を GPU 側に任せる事により計算効率が大きく向上する利点があるが、ボリュームデータの再定義にはリアルタイム性を伴うことは出来ない。そこで本手法では、スライスにマッピングするボリュームテクスチャに焦点をあて、スライスに割り当てるボリュームテクスチャのテクスチャ座標を変更する事で、形状変化を擬似的に得ている。リアルタイム性の確認は出来たがいくつか問題点も残った。

ワールド座標系で X-Z 平面上の視点変更には対応しているが、Y 軸方向へ画角が上がる事に対処できていない。現状では自由に視点の変更は可能だが、本論文

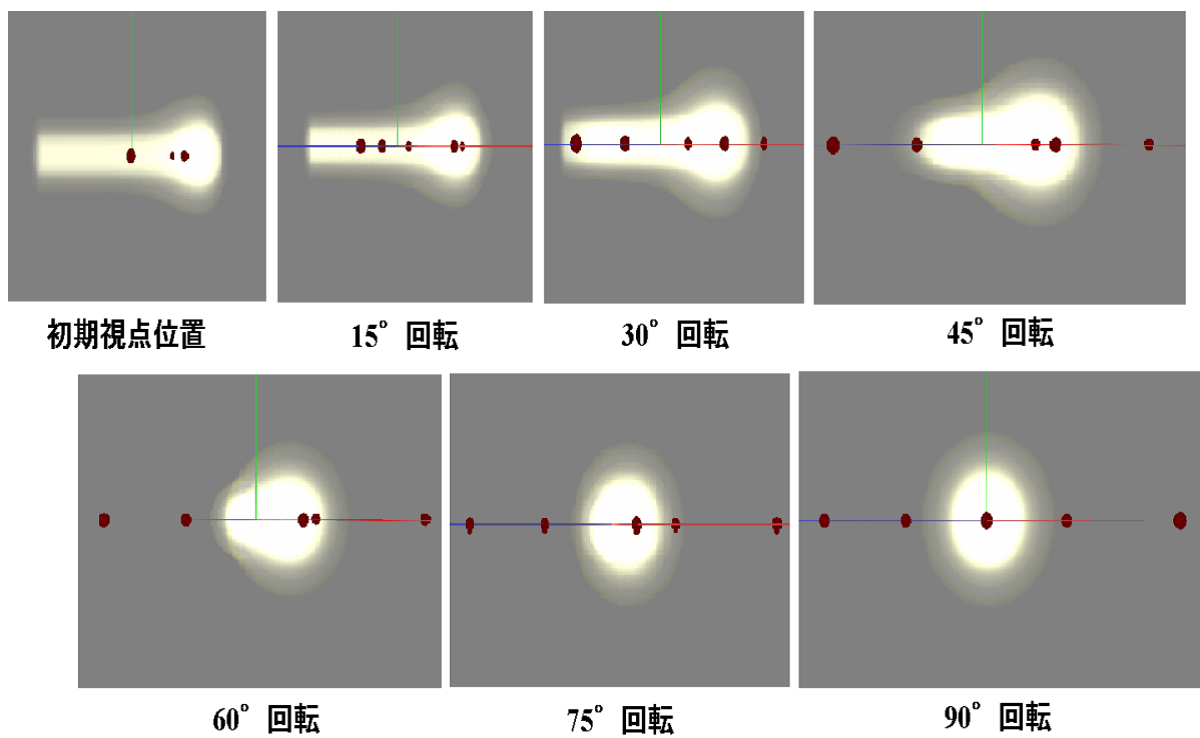


図 5.3: 視点に応じた補正の結果を示す

の形状変形手法が正確に反映している地点としない地点があるため、今後は上下の視点移動にも対応した補正処理又は変形処理を考案する必要がある。

第 6 章

まとめ

本論文の締めくくりとして、まとめと今後の展望に関して述べる。

本稿では坂井らのボリュームレンダリングによる新たなエネルギー波表現を拡張し、ボリュームレンダリングを用いたエネルギー波のリアルタイムな形状変形手法を提案した。坂井らの手法の中でも、リアルタイム処理に有利なテクスチャベース法と、その手法を用いて生成したエネルギー波を利用した。

形状変形の手法には山野上の FFD を利用した。FFD は変形媒体である制御軸とモデルを関連付けて変形を行うが、本稿ではモデルの変わりにテクスチャを対応させることでリアルタイム性を図った。制御軸は B-Spline 曲線による制御曲線と、制御点の局所座標である副軸で構成する。

正規化したテクスチャ座標値以上の値や、本来対応していないテクスチャ座標を指定する事により形状変形を表現し、リアルタイム性を伴うことに成功した。

しかし、現状ではいくつかの問題点も残っている。本手法では視点を自由に変更できる状況に拡張した結果、補正処理を加えないと視点によって観察しているエネルギー波の形状が変化してしまうという問題が挙げられる。また、補正処理は最終的な描画画面に映る段階で違和感を感じないようにするためのものであり、厳密にエネルギー波の形状を観察すると、視点に応じて形状が変化してしまう問題は解決していない。これはボリュームレンダリングを用いた表現の利点を活かせていない状況である。

この問題はテクスチャの構造を変えることで解決できるのではないかと考えられる。現在のテクスチャは四つの頂点で構成している、通常のテクスチャである。また、テクスチャを貼り付けるスライスは単なる矩形ポリゴンである。これらをメッシュテクスチャに切り替えたり、スライスを分割するなどして形状が表現できる幅を広げてやれば、これまでの3DCGモデルにあった様な革新的な進歩が望めるのではないだろうか。

今回はGPUの性能限界であるメモリ領域における処理の削減による高速化には一切触れていないため、高精度のボリュームデータや複数のボリュームデータを生成する事ができなかった。本研究の趣旨とは外れるが、GPUを用いたボリュームレンダリングの高精度、高速化において無視できない領域である。

現状ではまだまだ2次元のテクスチャを用いた表現の方が効率が良いが、近年のハードウェアの進歩をみればリアルタイムボリュームレンダリングが表現の主流になる日はそう遠くないと考える。本論文の手法が今後のエネルギー波表現の役に立つ事を願い、本論文の締めくくりとする。

謝辞

本論文を締めくくるにあたり、研究方針から手法、執筆に至るまでご指導頂きました本校メディア学部の渡辺大地講師及び、ゲームサイエンスプロジェクトのスタッフの三上浩司講師、小澤賢侍氏、中村太戯留氏に心より感謝いたします。また研究の手助けや研究者としての心得などをご指導して下さった本校大学院メディア学研究科博士課程の竹内亮太さん、メディア学の在り方についてのご指導や、心の清涼剤として数々の相談にのって下さった本校嘱託研究員の渡邊賢悟さん、ゲームサイエンスプロジェクトの先輩方に感謝の意を表します。加えて、昨年度の卒業生であり、多数の助言をくれた坂井悠基君、大場辰昌君、そして共に戦場を戦い抜いた同卒研究室の戦友や、生き方について多大な影響を与えて下さった宮内優太さん、そして私を支えて下さった全ての人に感謝します。

参考文献

- [1] 鳥山 明,「ドラゴンボール」,集英社,1985.
- [2] 坂井悠基,“ボリュームレンダリングによるエネルギー波の表現手法”,東京工科大学学士論文,2007.
- [3] 篠本雄基,“汎用 GPU を用いたボリュームレンダリングの高速化に関する研究”,京都大学大学院情報学研究科修士論文,2006.
- [4] 山崎俊太郎,加瀬究,池内克史,“PC グラフィクスハードウェアを利用した高精度・高速ボリュームレンダリング手法”,IPSJ CVIM,2001.
- [5] 千葉則茂,土井章男,「3次元CGの基礎と応用[新訂版]」,サイエンス社,2004.
- [6] Tom McReynolds,Silicon Graphics,“Programming with OpenGL: Advanced Techniques”,SIGGRAPH,pp.144-147,1997.
- [7] 丸山悠樹,中田智史,高山征大,津邑公暁,五島正裕,森眞一郎,中島康彦,富田眞治,“汎用グラフィクスハードウェアを用いた並列ボリュームレンダリングの実装”,IPSJ SIG Notes,2003.
- [8] 吉良祐司,“並列ボリュームレンダリングにおける画像重畳処理に関する研究”,京都大学工学部情報学科,2005.

- [9] Thomas W. Sederberg, Scott R. Parry, “Free-Form Deformation of solid geometric models”, Computer Graphics, Vol.20, No.4. Aug.1986.
- [10] 矢原弘樹, 水野一徳, 福井幸男, 西原清一, 持丸正明, 河内まき子, “FFD 法を用いた 3 次元足部モデルの解剖学的特徴点の位置推定における誤差要因の分析”, 芸術科学会論文誌 Vol.4 No.4 pp168-175. 2005.
- [11] 室伏絹子, “3DCG における FFD を用いた魚が泳ぐ際の体の変形に関する研究”, 東京工科大学学士論文, 2003.
- [12] 山野上寛, “制御軸を用いた大域的な形状変形手法に関する研究”, 慶応義塾大学大学院政策・メディア研究科, 1998.
- [13] 松本零士, 「宇宙戦艦ヤマト」, 秋田書店, 1974.
- [14] 「CGWORLD」, p36 ~ 41. WORKS CORPORATION. 2月号. 2004.
- [15] ボリュームレンダリングソフトウェア, 「VGStudio/VGStudioMAX」, 日本ビジュアルサイエンス株式会社,
<http://www.nvs.co.jp/archives/01_01_vgstudio/>.
- [16] 3D 画像処理エンジン, 「VolumePro」, TeraRecon.Inc, <<http://www.terarecon.co.jp/imaging/vol1.html>>.
- [17] 3次元テクスチャ・シェーダ, 株式会社ジェーエフピー,
<http://www.jfp.co.jp/three_d_t/>.
- [18] Lichtenbelt, B., Crane, R. and Naqvi, S, 「Introduction To Volume Rendering」, Hewlett-Packard, 1998.
- [19] NVIDIA, <<http://jp.nvidia.com/page/home.html>>.
- [20] ATI, <<http://www.amd.com/jp-ja/>>.

[21] Matrox, <<http://www.matrox.com/>>.

[22] 日本設計工学会, 「3次元CAD実践活用法」, コロナ社 p74-77,2006.

[23] 渡辺大地, FK Tool Kit System, <<http://www.media.teu.ac.jp/~earth/FK/>>.