2008年度　卒　業　論　文

逆遠近錯視立体の自動生成に関する研究

指導教員：渡辺 大地 講師

メディア学部 ゲームサイエンスプロジェクト
学籍番号　M0105260
高野 もも

Media Science

2008 年度　　卒　業　論　文　概　要

論文題目

逆遠近錯視立体の自動生成に関する研究

メディア学部 **学籍番号：M0105260**	**氏名**	高野 もも	**指導教員**	渡辺 大地 講師

キーワード	逆遠近錯視、トロンプルイユ、 アフィン変換、射影変換、テクスチャマッピング

逆遠近錯視とは、凹凸のある面に本来とはまったく逆の遠近感をもつ絵を描くことによって、凹凸が逆になって見える錯覚である。この絵の前を左右に移動すると、描かれている建物が自分についてくるかのように動いて見える。この錯覚を見るために特別な器具や技術は必要ないため、近年では公共道路や商業広告などに広く利用されている。しかし逆遠近錯視が起こる立体を制作する場合、透視図法の概念や、立体の凹凸を考慮した描画方法を考慮しなければならず、熟練の技術が必要である。そのため、逆遠近錯視の作品は一部の芸術家によって制作されている例がほとんどである。また逆遠近錯視について、錯視が起こるメカニズムや脳の働きなど、心理学からの研究は多くなされてきたが、逆遠近間錯視立体についての理論的な分析はなされていないのが現状である。そこで本研究では、物体が逆遠近錯視立体となる条件を分析し、3DCG 上で射影変換処理を用いてテクスチャ画像の変形を行うことで制作を容易にした。

目 次

第 1 章　はじめに **1**
1.1　研究背景 1
1.2　論文構成 5

第 2 章　逆遠近錯視の生成方法 **6**
2.1　立体の生成 7
2.2　テクスチャの変換 8
2.3　展開図の生成 10

第 3 章　実装結果と検証 **12**
3.1　実行結果 12
3.2　考察 15

第 4 章　まとめと展望 **19**

謝辞 **21**

参考文献 **22**

図 目 次

1.1 逆遠近錯視立体の作品 (上) と平面図 (下) 1
1.2 斜め方向から見たときの形状 2
1.3 逆遠近錯視作品の水平線と消失点 2
1.4 逆遠近錯視立体作品の一部分 3
1.5 道路上のイメージハンプ (1) 4
1.6 道路上のイメージハンプ (2) 4
1.7 90 度システム広告 4

2.1 透視法射影 7
2.2 透視法射影で描画した立方体 7
2.3 変形前の形状 (直方体) 7
2.4 変形後の形状 (錐台) 8
2.5 立体変形時の面の画像の歪み 9
2.6 変形前の形状 (直方体) 9
2.7 変形後の形状 (錐台) 10

3.1 使用画像 (1) 12
3.2 使用画像 (2) 13
3.3 使用画像 (3) 13
3.4 実行前 (1) 13
3.5 実行前 (2) 13
3.6 実行前 (3) 13
3.7 実行後 (1) 14
3.8 実行後 (2) 14
3.9 実行後 (3) 14
3.10 正面から見た直方体 15
3.11 テクスチャ分割数 1×1 の場合の錐台 16
3.12 テクスチャ分割数 2×2 の場合の錐台 16
3.13 テクスチャ分割数 10×10 の場合の錐台 16
3.14 テクスチャ分割数 20×20 の場合の錐台 16
3.15 展開図 (1) 17

3.16 展開図 (2) 17
3.17 3DCG から作成した立体物 (1) 17
3.18 3DCG から作成した立体物 (2) 17
3.19 3DCG から作成した立体物 (3) 17
3.20 3DCG から作成した立体物 (4) 17
3.21 3DCG から作成した立体物 (5) 18

第 1 章

はじめに

1.1 研究背景

芸術は人類の長い歴史の中で多様に進化を遂げると同時に、様々な形で人々の暮らしに彩りを与えてきた。その中でもトロンプルイユ (だまし絵)[1] は、芸術的な価値に加え、発見や驚きという魅力をもった手法である。この手法の一種に、逆遠近錯視というものがある。これはトロンプルイユと遠近法の両方の性質を持ち合わせた、立体的な芸術作品である。逆遠近錯視は、凹凸のある面に本来とは逆の遠近感をもつ絵を描くことによって凹凸が逆になって見え、さらに観察者がこの立体の前を左右に移動すると、描かれている絵が観察者に連動して移動しているように見える運動視差 [2][3][4] が発生する錯覚である。 逆遠近錯視の発見者は

図 1.1: 逆遠近錯視立体の作品 (上) と平面図 (下)

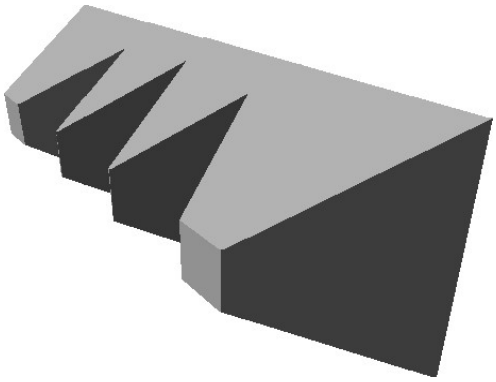

図 1.2: 斜め方向から見たときの形状

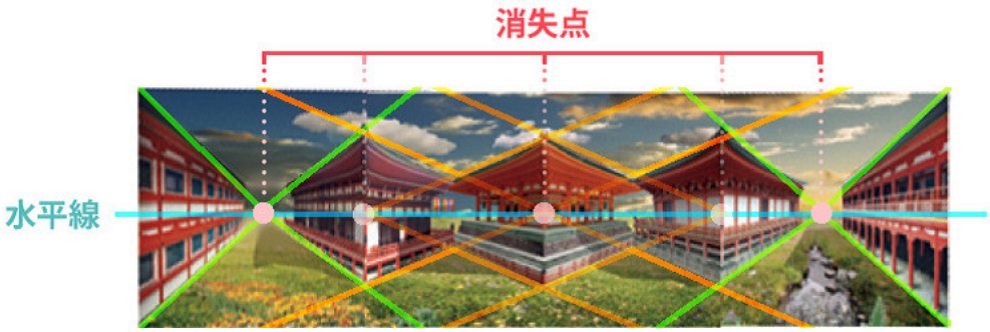

図 1.3: 逆遠近錯視作品の水平線と消失点

不明であるが、現在はイギリスの超現実主義の画家 Patrick Hughes[5][6][7] が、逆遠近錯視の専門家として多くの作品を制作している。彼の作品は世界中の美術館で見ることができる。

立体自体の特徴については、錐台を組み合わせた形でできていること、凹凸と絵の遠近感が逆になっていること、ひとつの錐台の投影面について、1 点透視図法を用いた絵になっていることが挙げられる。錐台とは、ピラミッドのような四角錐を底面と平行な面で切った形状のことである。Patrick Hughes の作品は錐台を単体あるいはいくつか並べて制作しているものが多いため、逆遠近錯視が発生しやすい形状であると思われる。図 1.1 は逆遠近錯視立体の正面図と平面図であり、図 1.2 は物体を斜め方向から見たときの形状を示した図である。立体が正面図を見て感じられる絵の遠近感とは逆の凹凸をもっていることが分かる。逆遠近錯視

立体に描く絵には、透視図法が用いられている。透視図法とは、消失点に近づく物ほど小さく表現することで奥行き感を出す表現手法のことである。逆遠近錯視の作品に透視図法が用いられていることを図 1.3 で示す。横に伸びた線が水平線、斜めの線が壁と壁との境界線、境界線が水平線に交わる点が消失点である。この図から、錐台の中心付近には必ず絵の消失点が存在していることが分かる。横に長い逆遠近錯視の作品は水平線上に複数の消失点があることが多いが、分解すると錐台と 1 点透視図法の絵が対になっていて、並んでいる構造になっている。図 1.3 の中央付近の 2 つの凹凸は錐台ではないが、錐台同士が重なっているために手前の面がなくなり、消失点もずれているものと思われる。図 1.4 はこの作品の錐台部分をひとつ切り取ったものである。実際には中心部分が一番手前にあるのだが、

図 1.4: 逆遠近錯視立体作品の一部分

立体に描かれた絵によって、あたかも自分が建物の間にいて、遠くの地平線を眺めているかのように見える。我々は毎日の経験から、遠いものは小さく、近いものは大きく見えることを理解し、さらに物体の凹凸を陰影によって把握している。そして物体を見る視点を変えると、見えないところが見えるようになり、見えていたところが見えなくなることも経験から理解している。逆遠近錯視は、これらの経験的な知識と体の動きにより脳が予測する情報と、目から得る情報の 2 つの視覚情報に矛盾が生じることによって錯覚を起こすものである。このため図 1.4 のように、逆遠近錯視立体を正面から見るとき、凹凸に関係なく透視図法を用いた絵となって見えるよう変形がなされている。

図 1.5: 道路上のイメージハンプ (1)

図 1.6: 道路上のイメージハンプ (2)

図 1.7: 90 度システム広告

逆遠近錯視はもともと芸術的な作品であったが、錯視を知覚するために特別な器具や技術が不要なため、近年その技法は芸術分野以外の多方面でも広く利用されている。例を図 1.5、図 1.6、図 1.7 に示す。図 1.5、図 1.6 の車線両脇の三角柱のように見えるものが錯視図形で、イメージハンプという。接近すると錯視によって山型のブロックが立ち上がって見え、ドライバーの注意を引くというものである。ハンプとは道路に凹凸を作り運転者の注意を促すものの意味であるが、イメージハンプは錯視を利用する平面図形 [8] であり、その効果も実証されている [9]。

図 1.7 の中央にある 2 つの平行四辺形が錯視図形で、90 度システム広告という。90 度システム広告は、特定の場所から見ると画像が立ち上がって見えるものである。画像を歪めて床に描くことで、広告スペースの節約の効果もある。主にサッ

カースタジアムなどに使われ、J リーグでは 2002 年から使われている。

逆遠近錯視について、心理学の分野での研究は Norman D. Cook ら [10][11][12][13] によって多くなされており、立体に描く絵の遠近線の有無と影の誇張によって、錯視の効果が高まるということや、錯視が起こる視野の角度や距離などが明らかになったほか、人は一般的に平面の画像を見るとき 3 個の物体の前後関係を手がかりに奥行きの情報を得ることから、逆遠近錯視を引き起こすためには、立体に線遠近法と一致した少なくとも 3 個の物の描写が不可欠であることが確認された。また林ら [14] は、錯視を認識するときの脳の働きについての調査を行った。しかしながら、逆遠近錯視立体の作成方法についての理論的な分析をした研究は存在していない。立体に逆遠近錯視に適した絵を描画する場合は、透視図法を用い、かつ凹凸を考慮しながら縦横の比率の歪んだ絵を描画しなければならない。そのため、本来の分野である芸術的方面においても、第一人者として挙げた Patrick Hughes 以外には制作できる人物がほとんどいないのが現状である。

そこで本研究では、このような現状を踏まえ、まず逆遠近錯視立体の数学的な特徴を分析し、立体の生成に必要な 3DCG 上での表現手法の方針を示した。次に、生成に必要な計算をプログラミングを用いて実装し、制作する際に熟練の技術が必要な部分を自動化することで、逆遠近錯視立体の制作を容易にすることを目的とした。

1.2 論文構成

本論文は 4 章で構成する。第 2 章で逆遠近錯視の生成手法について述べ、第 3 章では第 2 章で述べた手法をもとに開発したプログラムを用いて変換処理の実行結果を示す。第 4 章で今後の展望を述べる。

第 2 章

逆遠近錯視の生成方法

本章では、逆遠近錯視立体の生成方法について述べる。以下は簡単な生成手順である。

1. 直方体の生成
2. 領域の形状のみ錐台に変形
3. 各面のテクスチャを変形
4. 展開図の生成

直方体から錐台への形状の変形は、透視法射影 [15][16][17][18] に基づいて行う。透視法射影は、3D 空間上において投影中心を撮影点に設定し、そこから注視点に向かって広がる四角錐の間を切り取って表示させる方法である。 図 2.1 は高さが height、幅が width の、near から far までの範囲の物体を表示することを表している。透視法射影の投影中心は撮影点にあるため、投影線は放射状に広がる。これにより投影中心から遠い物体は小さく、近いものが大きく表示される。また、透視法射影で描画された空間では、投影面が一致すれば形状が変わっても見た目が変わりない。そこで本研究では透視法射影を利用し、投影面の形状を変えないまま直方体から錐台に形状と画像を変形することで逆遠近錯視立体を生成する。なお、本章では空間での点要素をイタリック体とし、その位置ベクトルを同じアルファベットのボールド体で表すものとする。

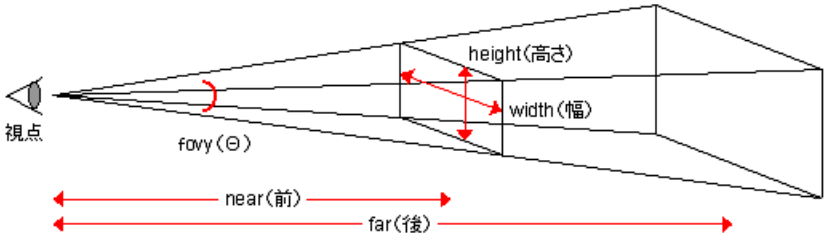

図 2.1: 透視法射影

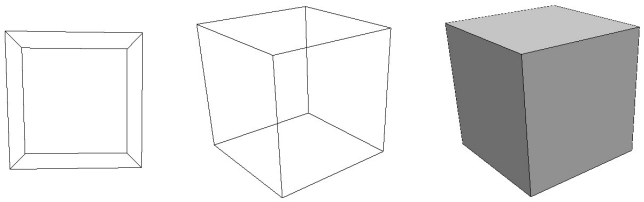

図 2.2: 透視法射影で描画した立方体

2.1 立体の生成

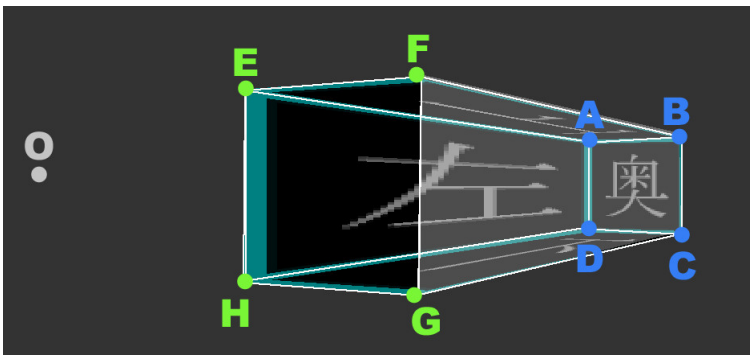

図 2.3: 変形前の形状 (直方体)

変形前の形状を図 2.3、変形後の形状を図 2.4 に示す。3D 空間上の水平方向を x 軸、垂直方向を y 軸、奥行きを z 軸とおき、カメラの座標を O とおく。カメラの注視点は $-z$ 方向に設定する。次に、xy 平面に対して底面が平行な直方体を設定する。カメラに対して奥側の面の頂点を $ABCD$、手前側の頂点を $EFGH$ とおく。

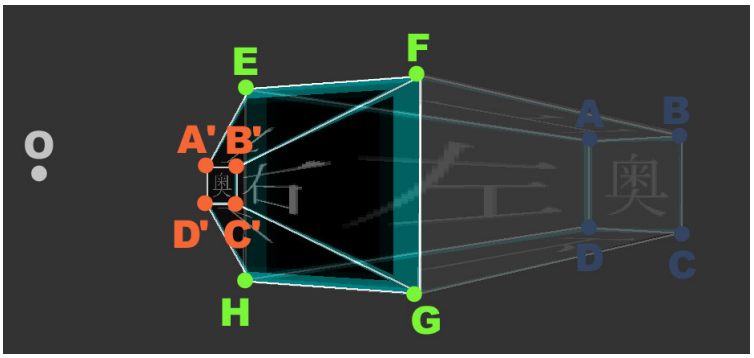

図 2.4: 変形後の形状 (錐台)

同様に、変形後の形状となる xy 平面に対して底面が平行な錐台を設定する。カメラに対して手前側の面の頂点を $A'B'C'D'$、手前側の頂点を $EFGH$ とおく。

正面図の形状を変えないまま頂点の移動をする。すなわち頂点の移動先は投影線上にあるので、A' を例に、移動先の座標を以下の式で表す。

$$\mathbf{A}' = (1-t)\mathbf{A} + t\mathbf{O} \tag{2.1}$$

B'、C'、D' についても同様に以下のように表す。

$$\mathbf{B}' = (1-t)\mathbf{B} + t\mathbf{O} \tag{2.2}$$

$$\mathbf{C}' = (1-t)\mathbf{C} + t\mathbf{O} \tag{2.3}$$

$$\mathbf{D}' = (1-t)\mathbf{D} + t\mathbf{O} \tag{2.4}$$

また z 軸について平行移動でもあるので、以下のようになる。

$$A'_z = B'_z = C'_z = D'_z \tag{2.5}$$

z 成分の厳密な値は定まっていないため、EFGH よりも手前にあればよいこととする。

2.2 テクスチャの変換

次に、テクスチャの変形を行う。立体を変形する際、底面を除く各側面は矩形から台形に変形をする。このときの画像内部の変形について、格子状の模様をつけ

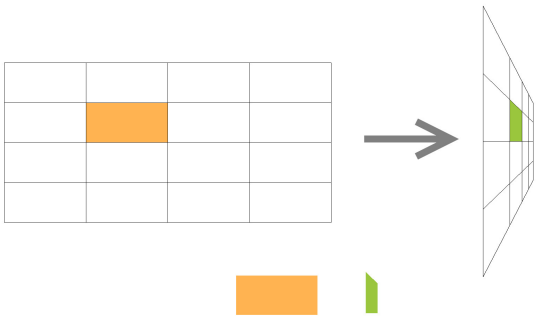

図 2.5: 立体変形時の面の画像の歪み

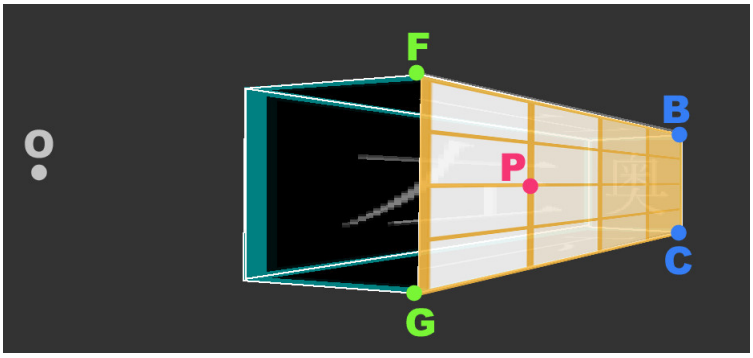

図 2.6: 変形前の形状 (直方体)

た矩形で説明する。これを図 2.5 に示す。左側は矩形を正面から見たもの、右側は矩形の形状は変えずに左側から見たものである。この面を正面から見ると格子はすべて同じ形をしているが、左側から見ると、遠近感により奥側にある格子ほど凝縮される。この遠近感を保ったまま画像の変形を行うため、矩形から台形へ形状だけ変形するのでは歪みが生じてしまい、正確な遠近感が表現できない。そのため先ほど立体に対して行った透視投影に基づいた変形処理を、立体の面のあらゆる点についても行う。変形前の形状を図 2.6、変形後の形状を図 2.7 に示す。面 $ABCD$ 上の点を P とする。また、面 $A'B'C'D'$ 上の形状変形後の P の移動先の点を P' とする。頂点のときと同様に、P' も投影線に沿って移動するため、以下のように表す。

$$\mathbf{P'} = (1 - s)\mathbf{P} + s\mathbf{O} \tag{2.6}$$

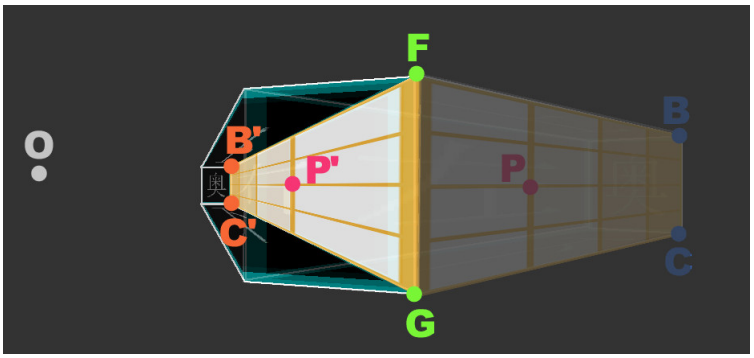

図 2.7: 変形後の形状 (錐台)

また P' は面 $A'B'C'D'$ 上にあるので、P' は線分 OP と面 $A'B'C'D'$ の交点である。面 $A'B'C'D'$ の式を以下のように設定する。平面上の三点 A', B', C' に対し、実数 u, v を用いて

$$\mathbf{A}' + u\mathbf{L} + v\mathbf{M}(\text{ただし、}\mathbf{L} = \mathbf{B} - \mathbf{A}; \mathbf{M} = \mathbf{C} - \mathbf{A}) \tag{2.7}$$

式 (2.6) と式 (2.7) から、P' の座標を以下の式で表す。

$$\mathbf{P}' = (1 - s)\mathbf{P} + s\mathbf{O} = \mathbf{A}' + u\mathbf{L} + v\mathbf{M} \tag{2.8}$$

(2.8) 式を解くことで、交点を求め P' の位置を決定する。この処理を立体の面のあらゆる点において同様に行えば、正確な変形の表現ができるが、実際にこの方法でテクスチャ画像を点で描画すると膨大な処理になってしまう。本研究では通常の線形変換で移動しても支障がなく、なおかつ処理速度にも支障が出ない程度に画像を細かく分割し、分割した各頂点を新平面の対応する座標に移動することで、疑似的に歪みを低減した。

2.3 展開図の生成

前節で生成した立体をもとに、展開図を生成する。展開図は立体のすべての面を一平面上に広げたものである。本研究では中央の面を基準面に設定し、中央の面と側面のなす角を求めた。その角度から、隣接する辺を軸に回転移動を行い展開図を生成した。

まず側面 $B'FGC'$ 上の平行でないベクトル $\mathbf{A}$、$\mathbf{B}$ を用いて法線ベクトル $\mathbf{M}$ を算出する。

$$\mathbf{F} - \mathbf{G} = \mathbf{A}, \mathbf{F} - \mathbf{C}' = \mathbf{B} \tag{2.9}$$

$$\mathbf{M} = \mathbf{A} \times \mathbf{B} \tag{2.10}$$

ただし、「$\mathbf{A} \times \mathbf{B}$」は $\mathbf{A}$ と $\mathbf{B}$ の外積を表す。同様に、奥の面 $A'B'C'D'$ 上の平行でないベクトル $\mathbf{C}$、$\mathbf{D}$ を用いて法線ベクトル $\mathbf{N}$ を算出する。

$$\mathbf{A}' - \mathbf{B}' = \mathbf{C}, \mathbf{A}' - \mathbf{D}' = \mathbf{D} \tag{2.11}$$

$$\mathbf{N} = \mathbf{C} \times \mathbf{D} \tag{2.12}$$

面同士の角度は、それぞれの法線ベクトルのなす角と等しいので、$\mathbf{M}$ と $\mathbf{N}$ のなす角 θ より回転する角度を算出する。

$$\theta = \arccos \frac{\mathbf{M} \cdot \mathbf{N}}{|\mathbf{M}| \cdot |\mathbf{N}|} \tag{2.13}$$

ただし、「$\mathbf{M} \cdot \mathbf{N}$」は $\mathbf{M}$ と $\mathbf{N}$ の内積を表す。奥の面に対する側面の角度はすべて等しいので、すべての面で θ 分の回転移動を行う。

第 3 章

実装結果と検証

本研究ではプログラムを 3DCG ツールキットである Fine Kernel Tool Kit[19] を用いて実装し、実際にモデリングと検証を行った。本章では、実装と検証の結果を述べる。

3.1 実行結果

図 3.1、図 3.2、図 3.3 はテクスチャに使用した画像で、左から順に奥の面、上面、下面、左面、右面である。図 3.4、図 3.5、図 3.6 は各画像を立体の内側の面にマッピングした直方体で、左から順に正面、右方向、さらに右方向から見たときの実行画面である。図 3.7、図 3.8、図 3.9 は直方体から変形を実行した後の錐台で、左から順に正面、右方向、さらに右方向から見たときの実行画面である。

図 3.1: 使用画像 (1)

次に、テクスチャ分割の実装結果を示す。各分割数のテクスチャを以下の図に示す。図 3.10 は直方体を正面から見たものである。また、図 3.11 が分割数 1×1、

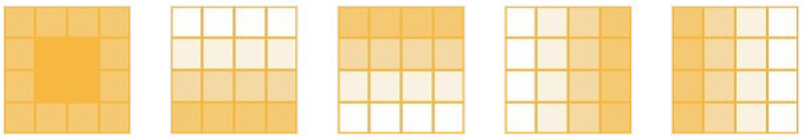

図 3.2: 使用画像 (2)

図 3.3: 使用画像 (3)

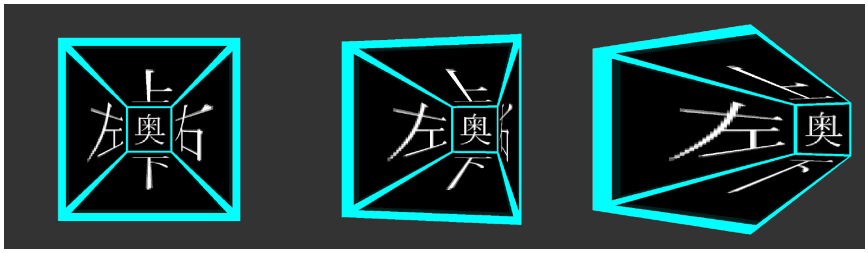

図 3.4: 実行前 (1)

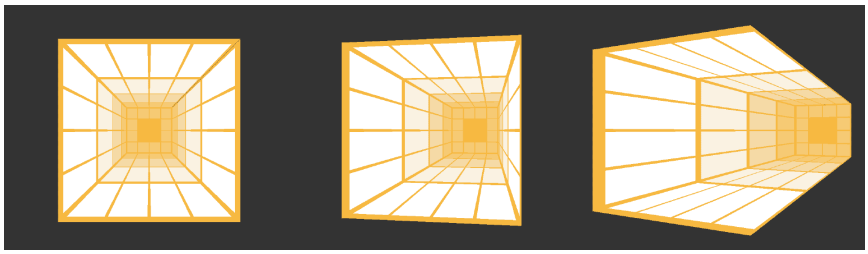

図 3.5: 実行前 (2)

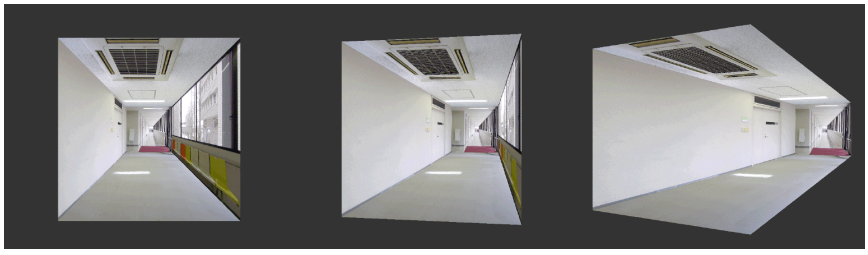

図 3.6: 実行前 (3)

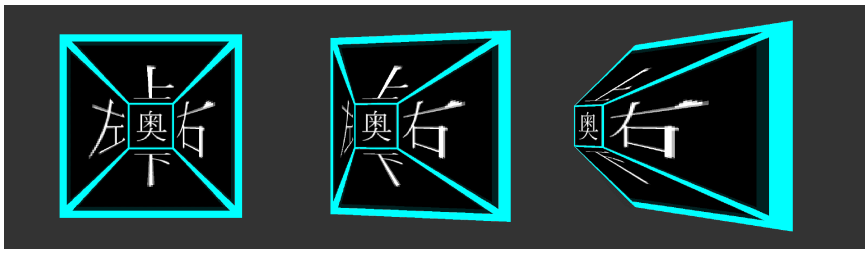

図 3.7: 実行後 (1)

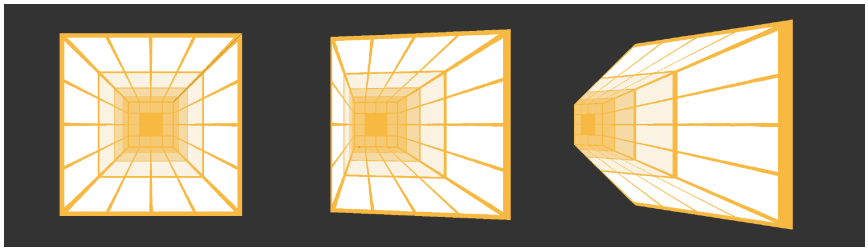

図 3.8: 実行後 (2)

図 3.9: 実行後 (3)

図 3.12 が分割数 2 × 2、図 3.13 が分割数 10 × 10、図 3.14 が分割数 20 × 20 の場合の各テクスチャの様子である。

また、プログラムで生成した展開図をプリントアウトして立体物を作った。生成した展開図の画像を図 3.15、図 3.16 に、実際に組み立てた様子を図 3.17、図 3.18、図 3.19、図 3.20、図 3.21 に示す。

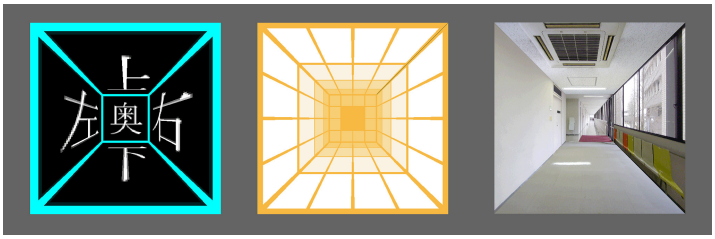

図 3.10: 正面から見た直方体

3.2 考察

形状については、実行前と実行後では正面すなわち投影面から見た画面にほとんど変化が見られないが、斜め方向から立体を見ると実行前とは形状が異なっていることから、透視法射影に基づいた変形が実現できたといえる。また実行後画像の中央の立体が、実行前の形状とほぼ同様に見えることから、凹凸が逆になって見えるという逆遠近錯視立体の特徴ももっているといえる。

テクスチャの分割については、分割数を多くするほど変形に伴う歪みが減っているのが分かる。このことから分割数を増やすことで、変換後の形を近似的に表現できることを確認した。

プログラムから生成した展開図から立体物を作り肉眼で観察したところ、3DCG 上で得られた逆遠近錯視立体の効果が、実際に作った立体物でも得られた。これらの結果により、本手法が逆遠近錯視立体の自動生成に有効であることを確認した。

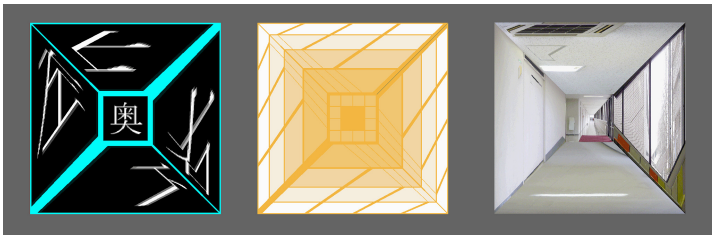

図 3.11: テクスチャ分割数 1 × 1 の場合の錐台

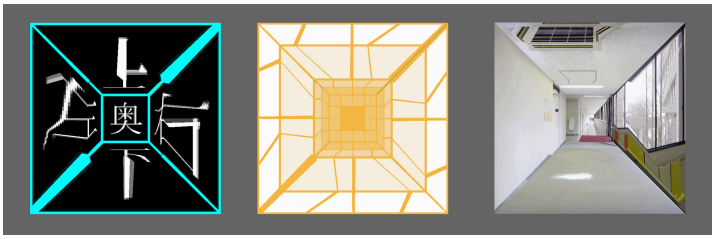

図 3.12: テクスチャ分割数 2 × 2 の場合の錐台

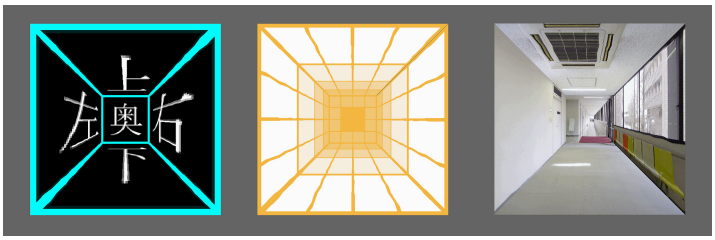

図 3.13: テクスチャ分割数 10 × 10 の場合の錐台

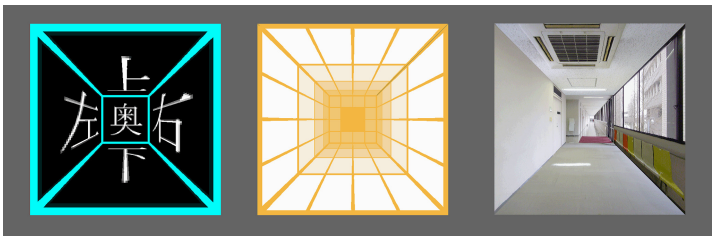

図 3.14: テクスチャ分割数 20 × 20 の場合の錐台

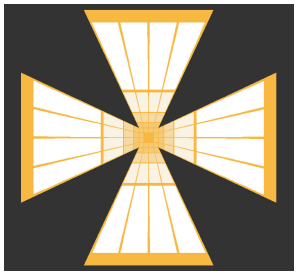

図 3.15: 展開図 (1)

図 3.16: 展開図 (2)

図 3.17: 3DCG から作成した立体物 (1)

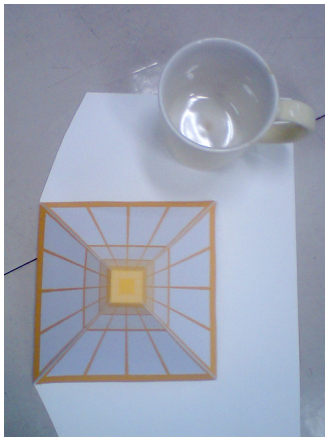

図 3.18: 3DCG から作成した立体物 (2)

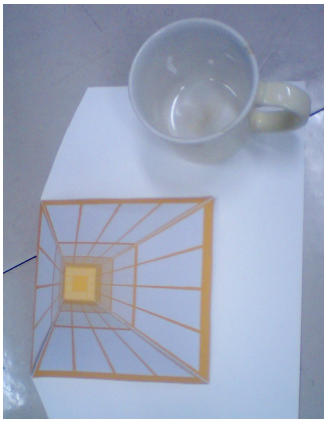

図 3.19: 3DCG から作成した立体物 (3)

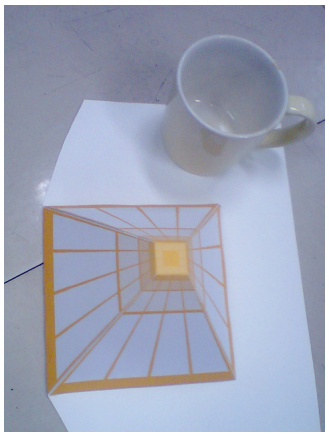

図 3.20: 3DCG から作成した立体物 (4)

図 3.21: 3DCG から作成した立体物 (5)

第 4 章

まとめと展望

本研究では、射影変換を用いて逆遠近錯視立体を 3DCG 上で自動生成する提案を行った。また提案に基づいてプログラミングによる実装を行い、提案が有効であることを確認できた。

本研究によって逆遠近錯視立体のシミュレーションから展開図の生成までが可能になったが、展開図をもとに実際の立体物を生成することについては未だ課題が残っている。まず展開図の画像解像度については、現状ではスクリーンショットによって展開図の画像を生成しているので、高解像度のテクスチャを貼り付けても印刷に反映することができない。また立体を展開図に変形したときも面は 3DCG 空間上の照明をそのまま反映するので、カメラや照明の位置によっては組み立てたときに画像が不適切な色合いや明度になる可能性があり、今後はこれを考慮または照明を反映しないようにする処理が必要と思われる。

次に、実際の逆遠近錯視立体の作品の形状は、錐台を複数組み合わせた形や屏風のような形など、複雑なものが多い。複数の錯視立体並べることで錯視を強める効果もあるため、この問題に対応できるようなシステムの開発が必要と考えられる。

最後に、実際の逆遠近錯視立体の作品は、錐台の面を超えて物体を描いたり、壁があることによってできる見せかけの陰影などを描くことで、錯視をより強めている。この問題点については、Teddy[20] のように投影面から直接画像を編集し、

錐台に射影することで対応できると考えられるが、この方法を用いる場合は透視図法の知識が必要になるため、ある程度絵が描けることが前提条件となる。

謝辞

本論文を執筆するにあたり、渡辺大地講師をはじめゲームサイエンスの先輩方、本当にお世話になり…まくりました。地球が吹っ飛んでも崩れなそうな私のマイペースに付き合わせてしまってごめんなさい。発表直前までアルバム制作やってごめんなさい。コンビニの梅干はおいしいですね。と、ごめんなさいだらけですが、ありがとうございました！

また、幾多の死線を共に切り抜けてきた同研究室の友人も、ありがとう。大変だったけど、みんなと過ごすことができてとっても楽しい 1 年間でした。

そして TUT-Winds のユカイな団員のみんな、帰りが遅くなっても、ちょっと文句を言いつつ暖かく迎えてくれた両親とフグに、この場をお借りして厚く厚くお礼申し上げます。

参考文献

[1] “錯視とトリックアート入門 トロンプルイユ壁画.” http://www.geocities.jp/sakushiart/trh.htm.

[2] 福田忠彦, “感覚の生理と心理.” http://gc.sfc.keio.ac.jp/cgi/class/class_top.cgi?2006_14454.

[3] “動きで形を見る.” http://www.pri.kyoto-u.ac.jp/brain/brain/23-1/index-23-1.html.

[4] 林部敬吉, “心理学における 3 次元視研究の動向-2003-,” **静岡大学情報学研究** *(9)* , 59–82 (2003).

[5] Nicholas J Wade, Patrick Hughes, “Fooling the eyes: trompe l’oeil and reverse perspective,” *Perception* (1999).

[6] Patrick Hughes, “Deeperspective,” (Flowers East, 1999).

[7] Patrick Hughes, “Reverspective Homepage.” http://www.msichicago.org/scrapbook/scrapbook_exhibits/reverspective/index.html.

[8] 野田和宏, “遠近法によるアナモルフォーシスの解析,” (2007).

[9] 千葉県市川市, “資料 1 イメージハンプ設置の対策効果評価.” http://www.city.ichikawa.lg.jp/common/000027661.pdf.

[10] Norman D. Cook, Asami Yutsudo, Naoki Fujimoto and Mayu Murata, “Factors contributing to depth perception: behavioral studies on the reverse perspective illusion,” *Department of Informatics* (Received 17 August 2006; accepted 10 March 2007).

[11] Norman D. Cook, Takefumi Hayashi, Toshihiko Amemiya, Kimihiro Suzuki, Lorenz Leumann, “Effects ofvisual-field inversions on the reverse-perspective illusion,” *Department of Informatics* (2002).

[12] Norman D. Cook, Asami Yutsudo, Naoki Fujimoto, Mayu Murata, “On the visual cues contributing to pictorial depth perception,” *Kansai University* (2008).

[13] Norman D. Cook, “逆遠近感の錯覚.” `http://www.res.kutc.kansai-u.ac.jp/~cook/JRPIndex.html`.

[14] Takefumi Hayashi, Chie Umeda, Norman D. Cook, “An fmri study of the reverse perspective illusion,” *Department of Informatics* (2007).

[15] 魏大名, 先田和弘, Roman Durikovic, 向井信彦, Carl Vilbrandt, [**コンピュータグラフィックス**], オーム社 (2003).

[16] 中前栄八郎, 西田友是, [*3***次元コンピュータグラフィックス**], オーム社, 昭晃堂 (1986).

[17] 平野拓一, “ゲームとアフィン (Afin) 変換.” `http://www-antenna.ee.titech.ac.jp/~hira/hobby/edu/afin_trans/math_html/index.html`.

[18] 島田静雄, “CAD・CG のための基礎数学.” `http://www.teu.ac.jp/clab/kondo/research/cadcgtext/CadCgMathematics.html`.

[19] 渡辺 大地, “Fine Kernel Tool Kit System.” `http://fktoolkit.sourceforge.jp/`.

[20] 五十嵐 俊夫, 松岡 聡, 田中 英彦, “手書きスケッチによるモデリングシステム Teddy,” **情報処理学会プログラミングシンポジウム** , 31–38 (2000.1).