グラフィクスとＣＡＤ　96－8
（１９９９．８．２３）

3 次元ボクセル空間における w 変換電子透かし法の CG モデリング

上原慎矢　　青木由直
北海道大学大学院工学研究科

2 次元平面でのフラクタル図形を処理する w 変換理論を 3 次元に拡張して，3 次元空間でのフラクタル点系列を処理する w 変換理論を用いて，ボクセル空間のデータに簡易な 3 次元の樹木状モデルを透かし情報として埋め込む方式を提案する。実際に 3 次元ディジタル画像に対して埋めこまれた点系列を取り出し，もとの 3 次元樹木状モデルで透かし情報が表現される様子を CG 技法で可視化した実験結果を示す。

CG Modeling for Watermarking by Using w Transform in a 3D Voxel Space

Shin-ya Uehara　　Yoshinao Aoki
Graduate school of Engineering,Hokkaido University

We propose a watermarking technique where we embed signature information into a 3D digital image and we extract signature such as 3D tree models by a fractal figure processing. We analyze the proposed technique using a w transform theory, which was originally proposed to analyze fractal figures in 2D space, by extending this theory into 3D space. We visualize the 3D watermarking processing using a volume rendering technique, where 3D tree models generated with spatial series of points in a 3D space, are displayed.

1 はじめに

3次元モデルを利用した電子透かし法としては, 文献 [1] で提案された方法等があるが, 3次元ディジタル画像 [2] に対する電子透かし法として, 文献 [3] で提案された w 変換の考え方を利用した透かし法を提案する。

空間の点に対する w 変換は, 点に作用して点の位置を変える変換を考えるとき, ある初期点にこの変換を繰り返し適用して最終的に別のある点を得ることができ, この得られた点を初期点の w 変換とするというものである。w 変換は用いる変換の選び方により変換途中の点系列が非等間隔に生成できるところに特徴がある。

1次元での w 変換の例として, 具体的に次のような数列を考える。

$$
\begin{aligned}
x_{n+1} &= x_n + \frac{1}{2^n} \qquad (n \geq 1) \\
x_1 &= 0
\end{aligned}
$$

ここで,x は点の位置を表すとすると, これは変換として x 軸上での点の移動を行い, 点系列が非等間隔に生成されていると解釈することができる。(図 1) この変換は, 極限として $x = 1$ に収束するので, 変換回数を増やしていくと,w 変換後の点の位置は $x = 1$ に限りなく近づくことになる。

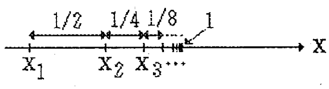

図 1: 1次元での w 変換の軌跡

次に 2 次元での例として, 次のような変換を考える。図 2 参照。l(点間の初期距離),θ は変換に用いるパラメタとして既知, 今 n 番目の点について $\vec{r_n}$,θ_n であるとき,n+1 番目の 2 つの点の位置をそれぞれ $\vec{r} + \vec{r_n} + \overrightarrow{r^+_{n+1}}$, $\vec{r} + \vec{r_n} + \overrightarrow{r^-_{n+1}}$ とする。ここで,

$$
\vec{r} = \sum_{N=1}^{n-1} \overrightarrow{r_N}
$$

$$
\begin{aligned}
\overrightarrow{r^+_{n+1}} &= (\frac{l}{n+1}\cos(\theta_n + \theta), \frac{l}{n+1}\sin(\theta_n + \theta)) \\
\overrightarrow{r^-_{n+1}} &= (\frac{l}{n+1}\cos(\theta_n - \theta), \frac{l}{n+1}\sin(\theta_n - \theta))
\end{aligned}
$$

である。

この変換を原点からはじめ, 各点を直線で結ぶと木状の図形を得ることができる。

このように w 変換の軌跡である点系列は, 生成される順番があるために順に結ぶということができ, これにより簡単な図形を表すことができる。もし, 単に複数の点を考えるだけでは各点の結び方を別に決めなければならなくなる。

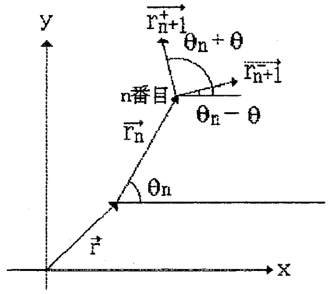

図 2: 点の位置ベクトル

2 埋めこみと復元

w 変換はその変換途中の点が空間に非等間隔に分布する点系列になるので, これを埋めこみ先の画像にある輝度で埋めこむ。抽出を試みるときは, 先ず埋め込み後の画像の輝度から, 原画像の輝度を差し引いたものに対して, 初期点を指定する。次に, 埋め込むときに使用した変換を適用し, 得られた点の輝度に対する閾値処理を行い, 埋め込み時の輝度以上の輝度をもつ点を埋めこんだ点として抽出する。

即ち, 鍵として,w 変換を適用する初期点と変換方法を持つことになる。

3 実験

用いるボクセル空間は, サイズが 16×16×16, 階調がモノクロ 256 色とする。

埋め込みを行う原画像として, ボクセル空間の中心が輝度 255 で, 中心から 8 ボクセルの距離のボクセルの輝度を 0, 間は線形に補間した画像を用いる。(中心からの距離が 8 ボクセル以上の部分は輝度 0 とする。) これをワイヤーフレームで表示したものと, レンダリングしたものをそれぞれ図 3, 図 4 に示す。

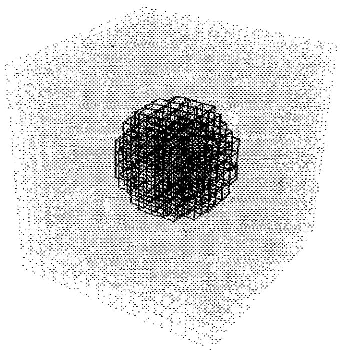

図 3: 原画像 (輝度 127 以上を黒で表示)

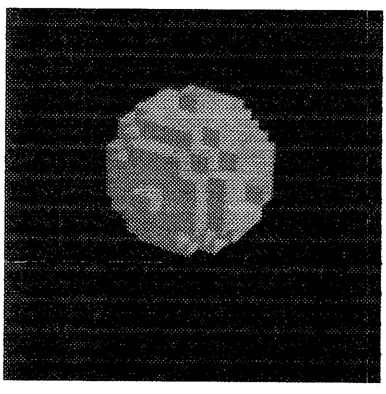

図 4: 原画像 (輝度 127 以上をレンダリング)

この原画像に 0 から 9 の輝度をランダムにとる雑音を付加する。これをレンダリングしたものを図 5 に示す。

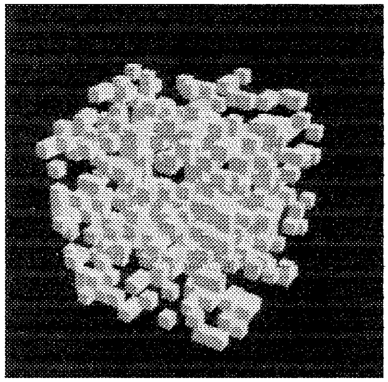

図 5: 雑音 (輝度 5 の部分を見やすく輝度を変更してレンダリング)

次にある変換により点系列を発生させて埋めこむ, 埋め込む点の輝度は 9 としている。

具体的に実験に使用する変換としては,3 次元アフィン変換である平行移動を用いる。平行移動量を示すベクトルを $\vec{t}$ とすると, その要素は右手系での極座標を用いて $\vec{t} = (|\vec{t}|\sin\theta\sin\phi, |\vec{t}|\cos\theta, |\vec{t}|\sin\theta\cos\phi)$ となり,θ,ϕ はランダムに決めることにし,$|\vec{t}|$ は変換を繰り返す度に減少させることにする。この変換をある点に対して 2 回行い, 変換後の点を埋め込むという作業を 3 回繰り返す。1 点に対して変換後の 2 点が埋め込まれるので, 最初の点を含めて計 7 個の点を埋めこむことになる。ここで, 平行移動量がボクセルの分解能以下にならないように $|\vec{t}|$ の初期値を選ぶ必要がある。この点系列を直線で結んだもの, さらに点に対応するボクセルをワイヤーフレームで表示したもの, レンダリングしたものをそれぞれ図 6, 図 7, そして図 8 に示す。

同じものを視点を変えて表示したものを図 9, 図 10, 図 11 に示す。

埋めこみ後の画像から原画像を差し引くと, 雑音に点系列を加えたものが得られる。これに対して抽出を試みるが, 埋め込み時にランダムに角度を決めたことから,1 回の変換で 100%抽出できるわけではなく, 変換を繰り返して輝度 9 以上の点を取り出すことになる。抽出回数を変化させながら抽出した結果,4 つ

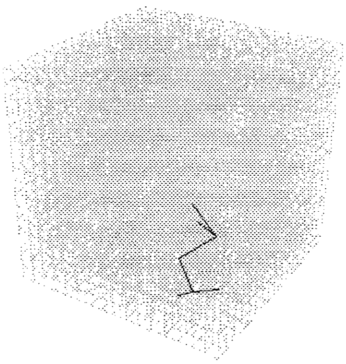

図 6: 点系列 (直線で結んだもの)

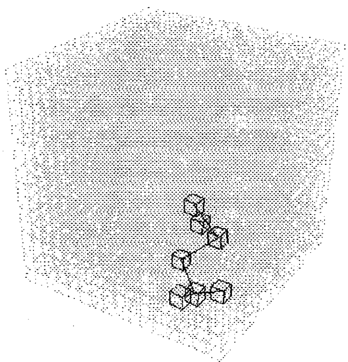

図 7: 点系列 (ボクセルも表示したもの)

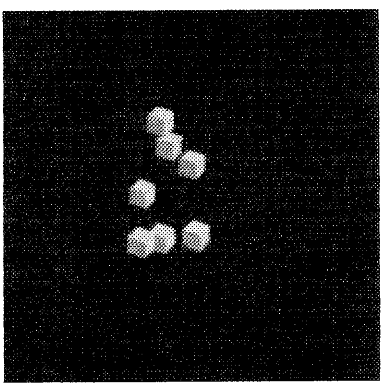

図 8: 点系列 (レンダリングしたもの)

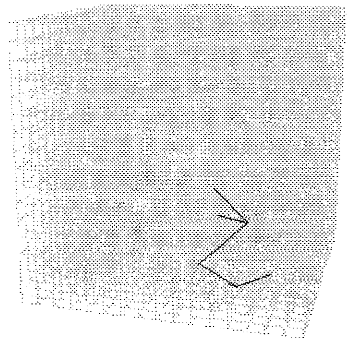

図 9: 点系列 (直線で結んだもの)

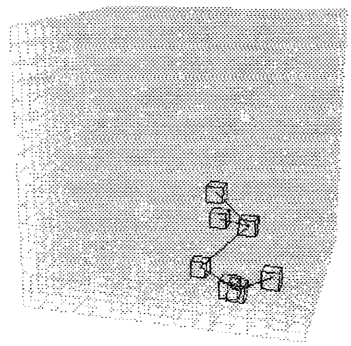

図 10: 点系列 (ボクセルも表示したもの)

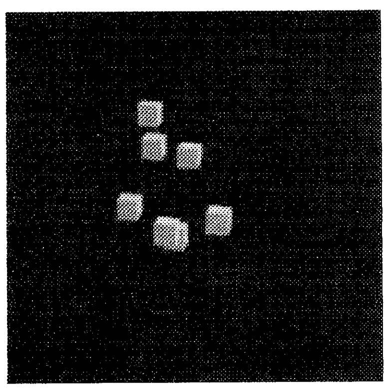

図 11: 点系列 (レンダリングしたもの)

の埋めこんだ点 (約 57%) を取り出すのに 14 回の変換,6 つの埋めこんだ点 (約 86%) を取り出すのに 32 回の変換を要した。この結果から変換を繰り返す回数を増やすほど抽出される割合が高くなることが確認できた。

4 つの点を抽出した場合の結果を図 12〜図 14 に示す。

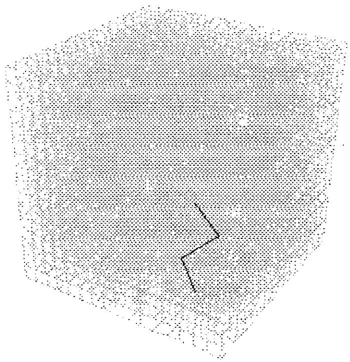

図 12: 抽出点系列 (直線で結んだもの)

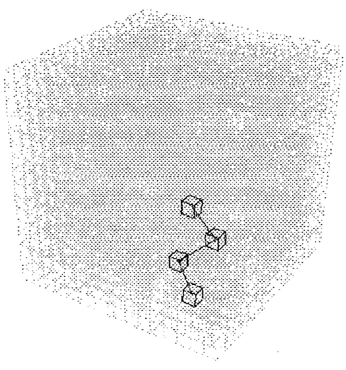

図 13: 抽出点系列 (ボクセルも表示したもの)

次に,6 つの点を抽出した場合の結果を図 15〜図 17 に示す。

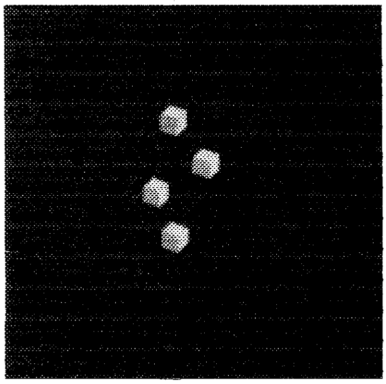

図 14: 抽出点系列 (レンダリングしたもの)

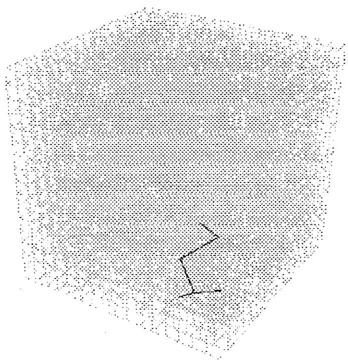

図 15: 抽出点系列 (直線で結んだもの)

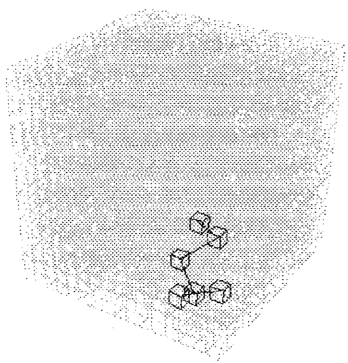

図 16: 抽出点系列 (ボクセルも表示したもの)

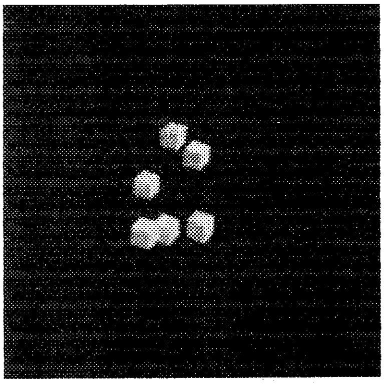

図 17: 抽出点系列 (レンダリングしたもの)

4 クリッピングへの対応

この方式では, 点系列の生成規則があれば途中の点からたどることや, 最後の点から逆にたどることができる。先の 1 次元での例では点間の距離を 2 倍にしていけばよく,2 次元での例では, 着目点が何番目の点かがわかっている必要があるが, 点間の距離を r とするとき, 半径 r の円の交点の輝度を調べ, 輝度に対する閾値処理をすればよいことになる。同様に 3 次元では球の交わりの円について閾値処理をする。

このことを利用すると, 初期点だけでなく終点の位置も保持しておき, 例えば初期点がクリッピングされた場合に終点からたどることでクリッピングされていない領域の点系列を抽出することや, 点系列の間がクリッピングされたときに, 初期点と終点の双方からたどって抽出する方法などが考えられる。また, 埋め込む点が多い場合は, 点系列そのものをいくつかの点系列にわけて考え, 各々について上記の考え方を適用するなどの方策も考えられるが, 当然ながらそれぞれの初期点, 終点, 場合によってはその点が何番目であるかという情報をもっていなければならない。

5 おわりに

透かしの主な目的から透かしは幾何変換に対して耐性がなければならない [4] が, この埋め込み法は点系列を埋め込み, その点系列を発生する変換と初期値を鍵としてもつという性質上幾何変換に対して弱く, 今後この点に対する改善が課題である。また,2 次元の画像に対して計算ホログラム (CGH) を埋め込む方法も文献 [5] で提案されている。ホログラフィは 2 次元データから位相情報を用いることで 3 次元情報を再生するので,CGH を用いた 3 次元画像への透かし情報埋め込みを本論文の手法と組み合わせた方法も 3 次元での電子透かし法の一手法と考えられ, 将来の検討課題である。

参考文献

[1] 赤羽和彦, 市川周一"三次元ポリゴンモデルの位相データを用いた電子透かしの一手法"電子情報通信学会 1999 年総合大会講演論文集, 情報・システム 2,SMD-9,pp.387-388(1999-03)

[2] 電子情報通信学会"電子情報通信ハンドブック第 1 分冊"(オーム社,1988)

[3] 青木由直"フラクタル図形生成におけるオペレータ法と葉脈曲線解析への応用" 電子情報通信学会論文誌 D-II,vol.J82-D-II,No.4(April,1999)

[4] 松井甲子雄"電子透かしの基礎 マルチメディアのニュープロテクト技術"(森北出版,1998)

[5] 青木由直"計算ホログラムを用いた電子透かし法"電子情報通信学会論文誌 A,Vol.J82-A,No.7(1999-07)