東京大学大学院新領域創成科学研究科
人間環境学専攻

修士論文

自己相似図形を利用した次元変換による多次元データの多解像度可視化手法

2011 年 2 月 14 日提出

指導教員　岡本　孝司　教授　　印

学生証番号　９６７２５

藤原　孝紀

目次

図目次

表目次

第 1 章

序論

1.1 研究背景

1.1.1 多次元化・大規模化するデータ

近年，測定技術の向上や計算機の処理速度の向上，アルゴリズムの発展，研究対象の多様化などにより，測定やコンピュータシミュレーションなどで得られるデータが多次元化してきている．例えば，MRI や CT などの医療機器が利用されるようになり，それらの機器では，身体・物体の 2 次元断面を多数得ることができ，その断面の集まりから 3 次元画像を得ることができるようになった．また，その他，3 次元計測技術等も発達してきている．さらに，コンピュータによる数値計算の分野でも，ボリュームデータ（3 次元構造を持つ中身の詰まったデータ）やボリュームデータの時系列変化（時系列ボリュームデータ）などといった高次元のデータが得られるようになった．さらには，物理学や数学の分野では，数式上でより高次元の現象を扱うようなことも頻繁に行われる．こうした傾向から，3 次元データや時系列 3 次元データ（4 次元データ），さらに大きな次元を持つデータといった高次元のデータを扱うことが多くなってきている．

扱うデータの次元が大きくなってきた以外にも，計測機器の高精度化 (解像度やフレームレートの上昇)，計算機の処理速度向上，記憶媒体の大容量化・低価格化などにより，取得，保存されるデータが大規模化してきている．

1.1.2 多次元・大規模データの可視化の必要性

多次元なデータ・大規模なデータはその形状的な複雑さやデータ数の多さのために低次元・小規模なデータよりもデータが示す形状や傾向の理解が困難となっており，理解を補助するような方法が必要である．その補助の 1 つの手段として，可視化が考えられる．実際に，3 次元医療データ，実験データ，シミュレーションデータ，時系列 3 次元データなどを対象とする Scientific Visualization という研究分野が存在する．この分野では，3 次元の形状をいかに表示するか，3 次元形状の中からいかに特異的な点を見つけるか，3 次元形状をいかに速いスピードで表示を行うかといったような研究が多く行われている．また，Information Visualization と呼ばれる分野では，いかにして大規模のデータを限られたディスプレイで表示するか，大規模データの中から必要なデータを表示するか，大規模データ間の関連性を表示するか

といったようなことが研究されてきている．

このように多次元や大規模なデータの可視化の必要性を感じとり，可視化研究が熱心に行われてきている．次節からは，こうした可視化研究の例をいくつか取り上げながら，既存の可視化手法について簡単に説明を行う．

1.2　多様な次元データの可視化

1.2.1　ボリュームデータの可視化

ボリュームデータの 3 次元可視化手法の 1 つに，ボリュームレンダリングがある．この手法は，ボリュームデータを可視化する際に最も利用される手法であり，研究もさかんに行われている．ボリュームレンダリングは，1988 年ごろにその基礎となる技術が開発された ([20], [3], [25], [28]) ．ボリュームレンダリングでは，ボリュームデータを構成するボクセルから視点まで運ばれる色情報を遮蔽度・透明度を考慮した形で表示することで，3 次元データの表面だけでなく，ボリューム内部の表示も可能にした．3 次元データの内部が表示可能であるという性質から，医療画像やシミュレーションデータなど内部を見る必要がある様々なデータを可視化するのに用いられている．基礎手法が開発されてから，ボリュームレンダリングの高速化や時系列ボリュームデータへの対応 [21]，イレギュラーグリッドへの対応などアルゴリズムの改善が多く研究されてきている [13]．近年では，扱うボリュームデータのサイズが大規模化し，データの値等の把握が困難となってきているため，データの値の把握を容易にするため Illustrative Volume Rendering などの手法が開発されている．この手法では，可視化結果における重要な箇所や特徴的な箇所を強調して表示することができ，必要な情報が把握しやすくすることができる．例えば，Hsu らは，時系列ボリュームデータの把握を補助するために，複数の時間におけるボリュームデータを Illustrative に可視化することを行った [9](Fig.1.1)．Hsu らの手法により，3 次元の構造の把握がしやすくなるだけでなく，複数時間のデータが確認できるため，ある程度時間的な遮蔽が解消される．

ボリュームレンダリングには，内部のデータ情報を可視化することが可能であるといった長所を有していたが，3 次元可視化手法であるため，視点との位置関係やデータの位置によっては，データの値が分かりにくい，もしくは完全に分からないような場合がある (視点依存性，空間的遮蔽)．例えば，Fig.1.2 左図では矢印で示した他の領域と孤立して存在する領域が確認できるが，右図ではその領域を確認することは不可能である (矢印で示した付近に存在するはずである)．このような視点依存性・空間的遮蔽は，ボリュームレンダリングの持つ大きな問題の 1 つである．また，時系列ボリュームデータをボリュームレンダリングで可視化した際には，時間変化を表すために通常アニメーションが用いられる．しかし，アニメーションを用いると表示されている時間以外のデータが観察できないといった時間的遮蔽が生じるなどの問題がある．

1.2.2　ボリュームデータの 2 次元可視化

Scientific Visualization の分野では，ボリュームデータを上述のような 3 次元ベースの可視化手法で可視化を行うことが多い．その一方で，3 次元ベースの可視化手法の場合，すでに前小節で述べたように視点依存性や空間的遮蔽が生じるなどの問題がある．そのため，可視化対象によっては，3 次元空間データ

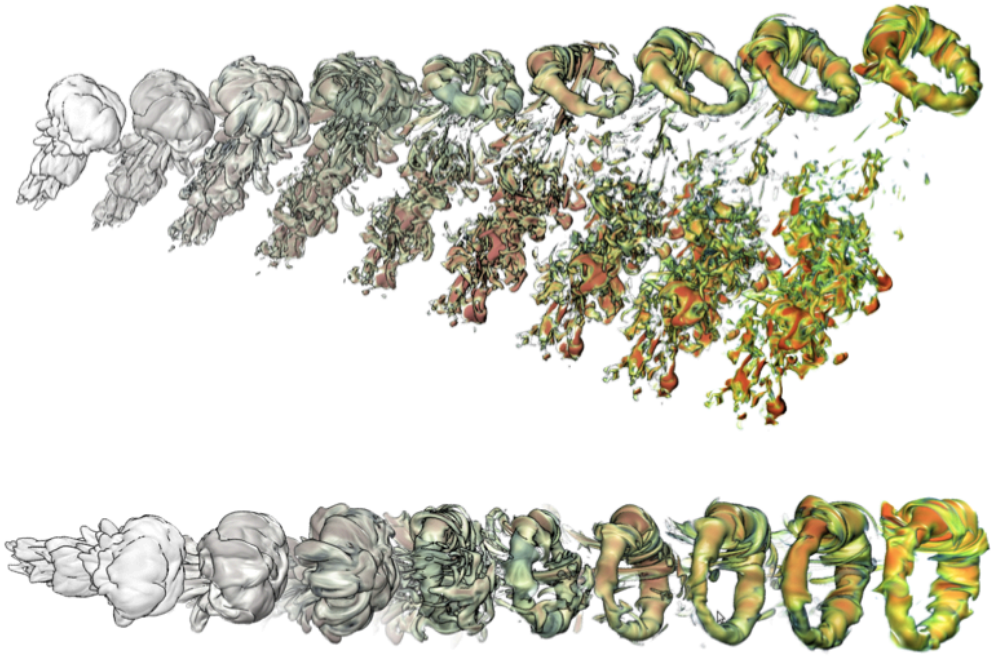


Fig. 1.1: Visualization example of time-varying volume data with illustrative volume rendering [9].

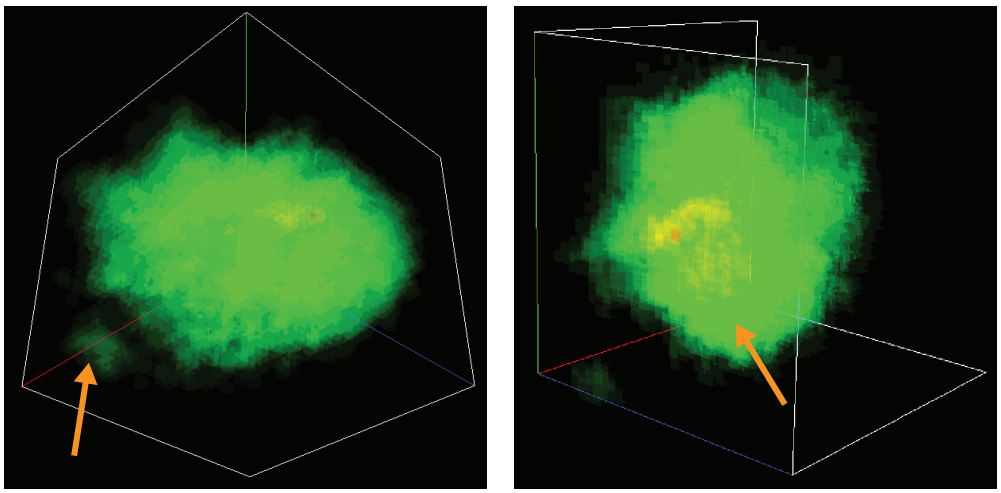


Fig. 1.2: An example of spatial occlusion.

を 2 次元平面化して表示するようなことも行われている．例えば，医療分野では，血管などの管構造の形状に着目したい場合などがあるが，この際に 3 次元の体の画像から取得したいのは血管にそった面だけである．3 次元画像から管形状にそった面を表示する手法として，Medical Axis Reformation(MAR)[8] や Curved Planar Reformation(CPR)[12] がある．これら手法では，Fig.1.3 のように，管の主軸に沿った平面を構成することを行い，その平面の描画を行うことで，3 次元の管形状を平面で表示することを可能としている．Fig.1.4 に CT 画像を従来型の冠動脈造影，ボリュームレンダリング，MAR の可視化結果を載せ

ている．MAR による可視化結果は，ボリュームレンダリングに比べ，管形状の太さなどが分かりやすく表示されていることが分かる．

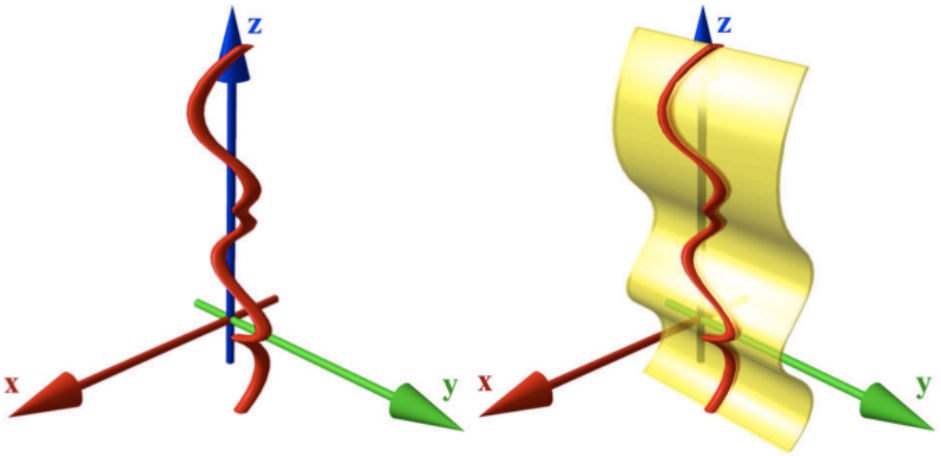


Fig. 1.3: Principle of the CPR visualiaztion[12]

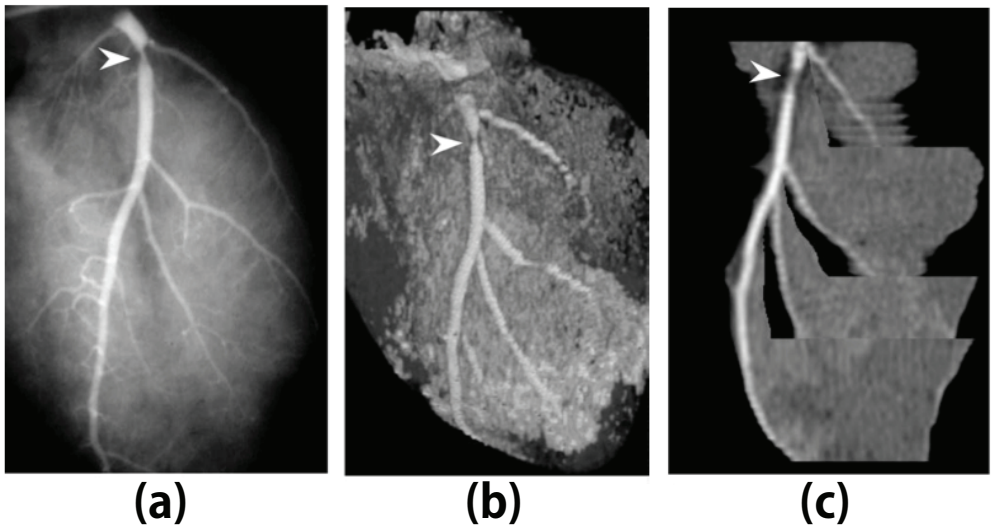


Fig. 1.4: Images of a postmortem pig heart demonstrate the various visualization methods used with coronary electron-beam CT angiography in comparison with conventional coronary angiography. (a) Conventional coronary angiogram. Electron-beam CT angiograms reconstructed with (b) MAR, (c) volume rendering [8].

その他にも，チューブ形状を 2 次元化する手法として，Radial Ray-casting がある [19]．この手法は，Fig.1.5(右図では管の主軸が直線に成るように左図の形状を変え，再構成を行っている．) のような管形状のものを管の主軸に対して垂直な面方向を 1 次元化し，それを主軸方向に並べ，2 次元化する．面の 1 次元化の際には，主軸と面の交点を中心として動径方向に Ray-casting を行い，それらを 1 次元状に並べる．Fig.1.6 の下から 2 段目の図が Radial Ray-casting による可視化結果である．Radial Ray-casting は，今ま

でボリュームデータを外部から内部方向を見た様子 (outside-looking-in) を可視化するといった手法と違い，物体内部から外部に向けた方向 (inside-looking-out) の可視化手法となっているといった特徴を持つ．

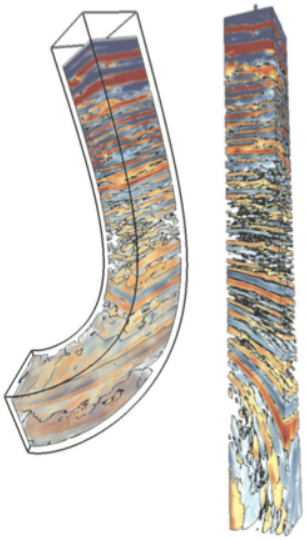


Fig. 1.5: This figure shows the seismic volume in the close vicinity to the well, before and after reformation [19].

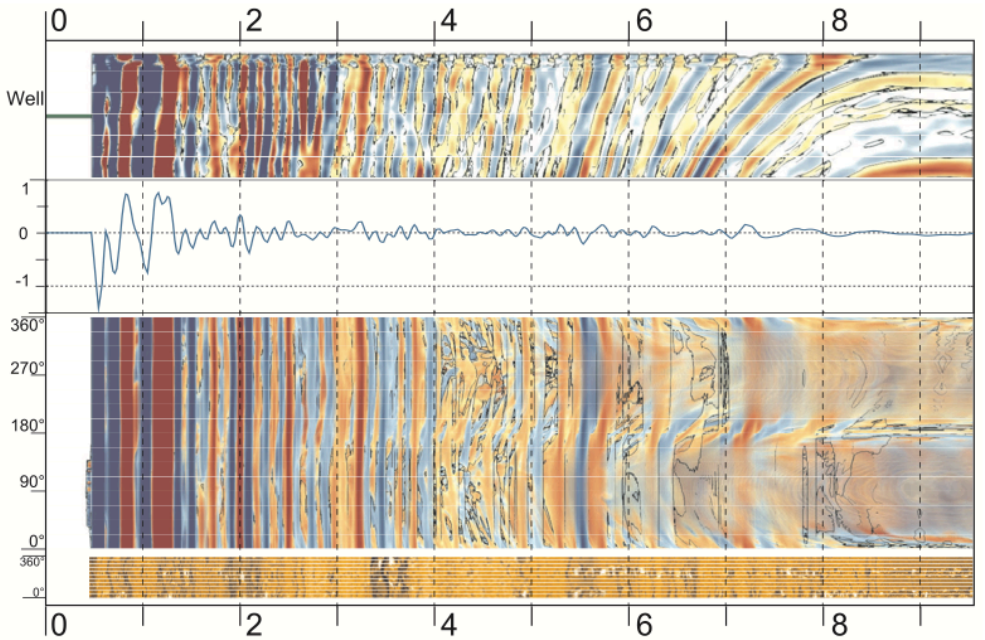


Fig. 1.6: Visualization result with radial ray-casting [19].

これらの 2 次元化手法の問題点としては，2 次元化の対象形状が管形状に限られていること，CPR や MAR は主軸に沿った 2 次元断面を取っているだけで 3 次元の空間情報が多く失われていること，Radial

Ray-casting は，主軸と垂直な断面を 1 次元化する際に Ray-casting を用いているので，通常の Ray-casting によるボリュームレンダリングと同様にボリュームデータを構成する各値を一意に表現できないこと，といったことが挙げられる．

1.2.3　ボリュームデータの理解を補助する可視化

ボリュームデータは形状が 3 次元，かつ大規模なデータになりやすいといった性質から通常の 2 次元データや小規模データより理解が困難である．そのため，上述のボリュームデータの形状や 3 次元空間内での値の分布を表示する手法以外にも，その形状を決める情報 (例えば，ボリュームレンダリングの際に利用する表示詳細度情報) の表示方法やデータの持つ値や傾向を多方面から見る方法なども開発されてきている．

LOD Map は [29]，ボリュームレンダリングの際に利用する表示詳細度 (level-of-detail, LOD) 情報の可視化手法として開発された．表示詳細度情報は，ボリュームレンダリングによる多解像度可視化を実現するために利用される．LOD Map は，情報的化学分野で多用される treemap[11] の概念を利用している手法である．また，LOD Map を表示詳細度情報を変更するインターフェースとしても利用することが可能である．そのため，LOD Map を用いることで，表示詳細度情報の効果的な可視化と同時に詳細度を調整し，可視化パラメータの操作を行うことができる．Fig.1.7 は，ボリュームデータをボリュームレンダリングしたものとその LOD Map である．

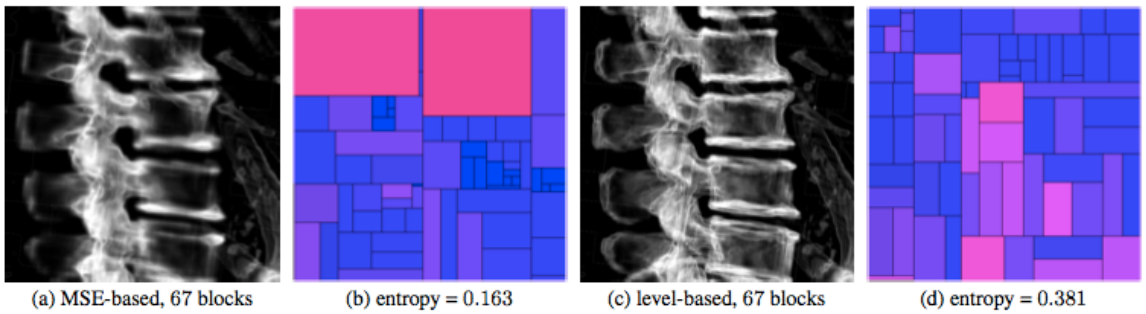


Fig. 1.7: LOD map results for a part of the spine of the VisWoman data set. (a) and (c) show the rendering of two different LODs with the same block budget. (b) and (d) are the LOD maps for (a) and (c) respectively [29].

Voxel Bars[23] は，ボリュームデータを構成するボクセルの持つ様々な属性値を同時に 5 つまで表示できる可視化手法である．この手法は，Pixel Bar Charts[16] を利用した手法となっている．Voxel Bars を利用してボリュームレンダリングの伝達関数を操作することが可能であり，効果的なボリュームレンダリングの可視化結果を得ることができる．Fig.1.8 は，通常の散布図と Voxel Bars の比較と，Voxel Bars を利用して，伝達関数を変化させた結果である．

Akiba らは，時系列多変数ボリュームデータのための可視化手法を開発した [1]．Akiba らの手法では，３つの可視化結果を同時に表示する． 1 つは変数間の相関関係を示すパラレルコーディネート，2 つ目は時系列変化を表示するヒストグラム，3 つ目はある時刻の各変数の空間的な形状を 1 画面に表示したもの

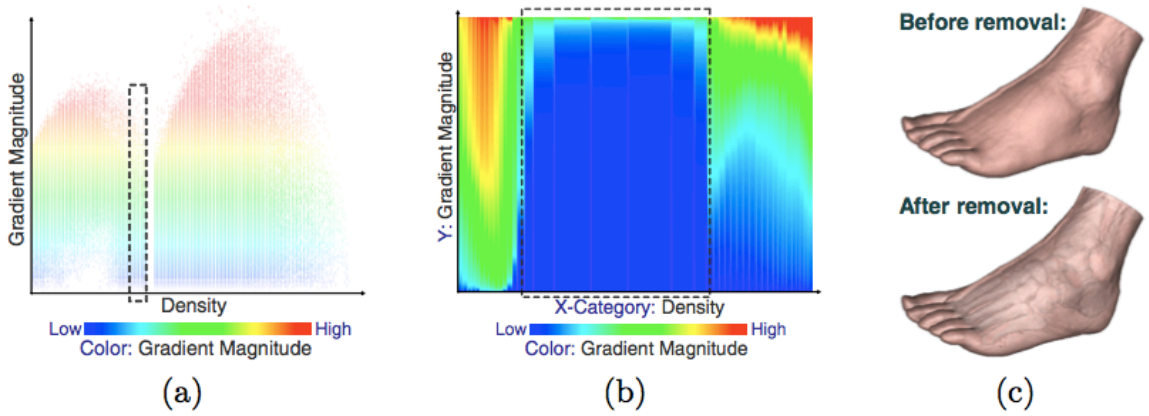


Fig. 1.8: Comparison between (a) scatterplot and (b) VoxelBars. The regions enclosed by dashed lines in (a) and (b) correspond to the same group of voxels. After removing the blue cluster in that region, the inner bone structures become visible with the outer layer of the skin preserved as shown in (c) [23].

である．これら 3 つの可視化結果により，同時に時間変化・変数間の関係・空間形状を把握することができ，時系列多変数ボリュームデータといった複雑な情報を持つデータの理解を用意にすることができる．Fig.1.9 に Akiba らの手法による可視化結果例を載せる．

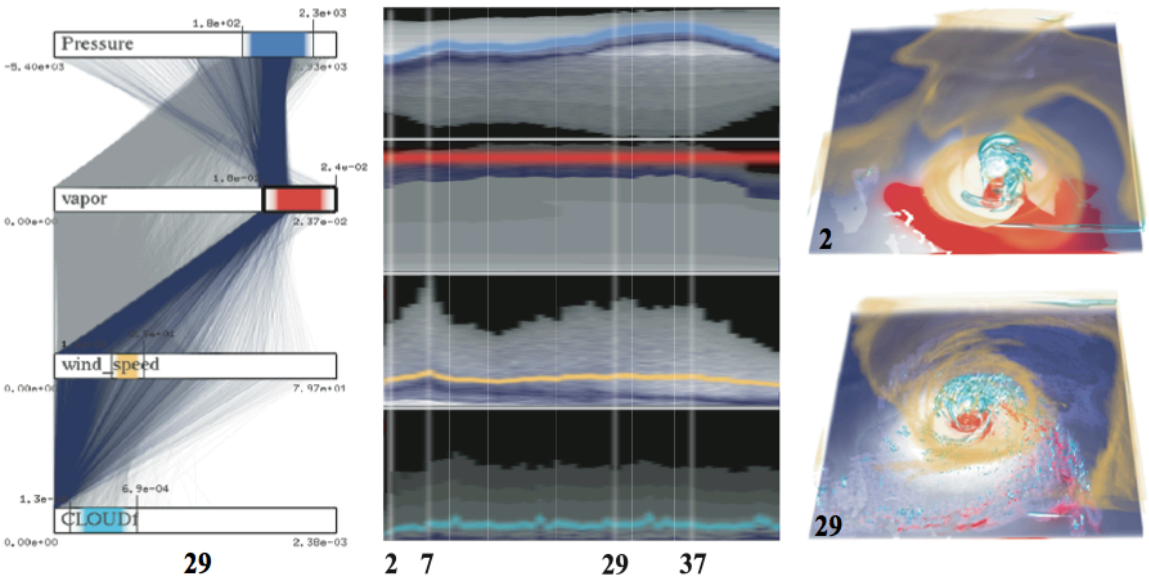


Fig. 1.9: Exploration of the hurricane dataset to find correlation between cloud, wind speed, vapor and pressure. Three components in our UI are shown including the multivariate view for timestep 29 (a), the temporal view for the four variables(b), and the spatial views for timestep 2 and 29. [1]

これらの手法は，ボリュームデータを多角的に理解するために利用されるため，ボリュームレンダリングなどの空間表示手法と同時に利用されることが多い．これらの手法により，効果的なボリュームレンダ

リングの変数の決定やボリュームデータの理解が得られるが，ボリュームレンダリングの持つ空間的遮蔽・視点依存性・時間的遮蔽を解消することで，ボリュームデータの理解の補助を行うような方法は存在しない．

1.2.4　高次元空間構造を持つデータの可視化

4 次元以上の空間構造を持つ高次元のデータは，そのまま空間構造を維持したままの様子を表示することが不可能である．そのため，空間構造を持つ高次元データの可視化方法としては，投影像，断面像，写像を描画するといった方法が取られる．例えば，Fig.1.10 は方程式 $(z_1)^n + (z_2)^n = 1\,(z_1, z_2 \in \mathbb{C})$ によって作られる 4 次元空間図形から 4 次元空間形状を持つ 2 次元断面を取り出し，それを 3 次元空間に投影した結果の例である [6]．断面や投影による手法であると，全データを反映した可視化ができないため，我々の提案手法では写像を用いて高次元データの可視化を行う．

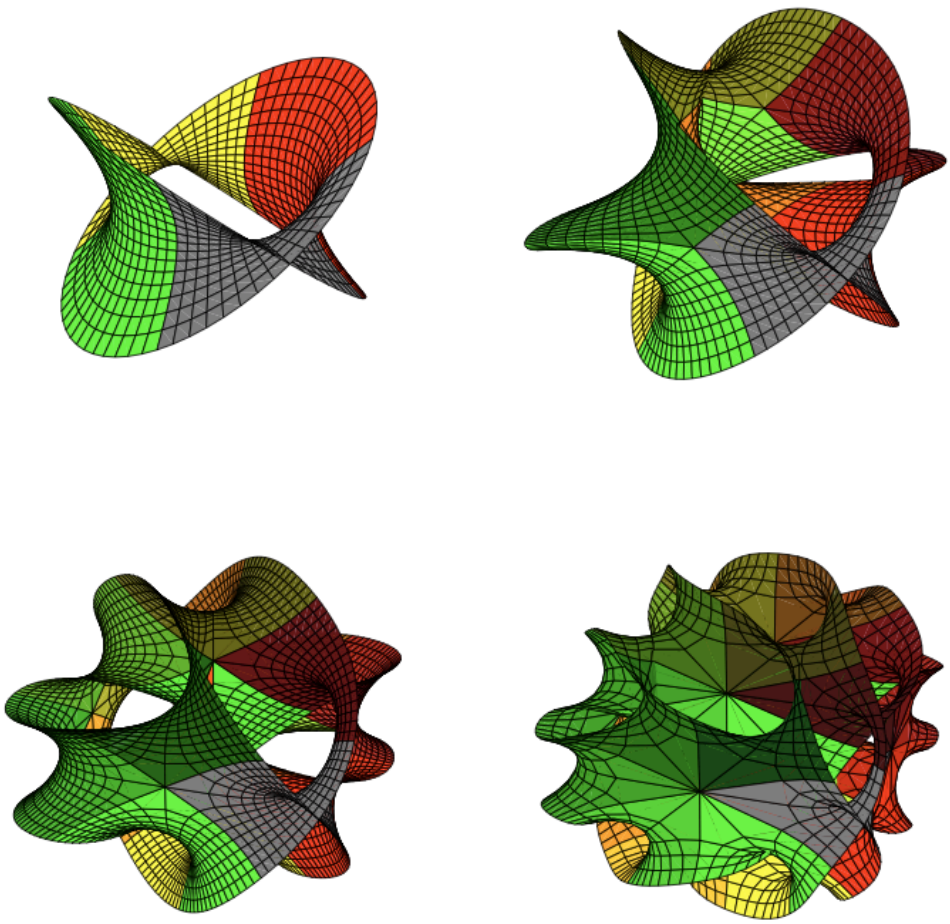


Fig. 1.10: Examples of 2D cross sections in 4D projected to 3D space (n=2, 3, 4, 5) [6].

1.2.5　情報的可視化分野の高次元データの可視化

データベースのような構造を持つデータはしばしば高次元のデータとなることがあり，情報的可視化分野の可視化対象とされてきた．Keim らは，画面上の各ピクセルが 1 つの値を色を用いて表現するといった pixel-oriented visualization を行っており，その 1 つの事例として VisDB[17] が挙げられる．VisDB では，大規模なデータベースへのクエリの複数次元の返答情報に対して，1 結果を 1 ピクセルで表示する手法で可視化を行い，ユーザーがデータベースへのクエリを決定するプロセスを補助するような手法となっている．Fig.1.11 は，VisDB を使用して，クエリの返答情報を可視化した例である．

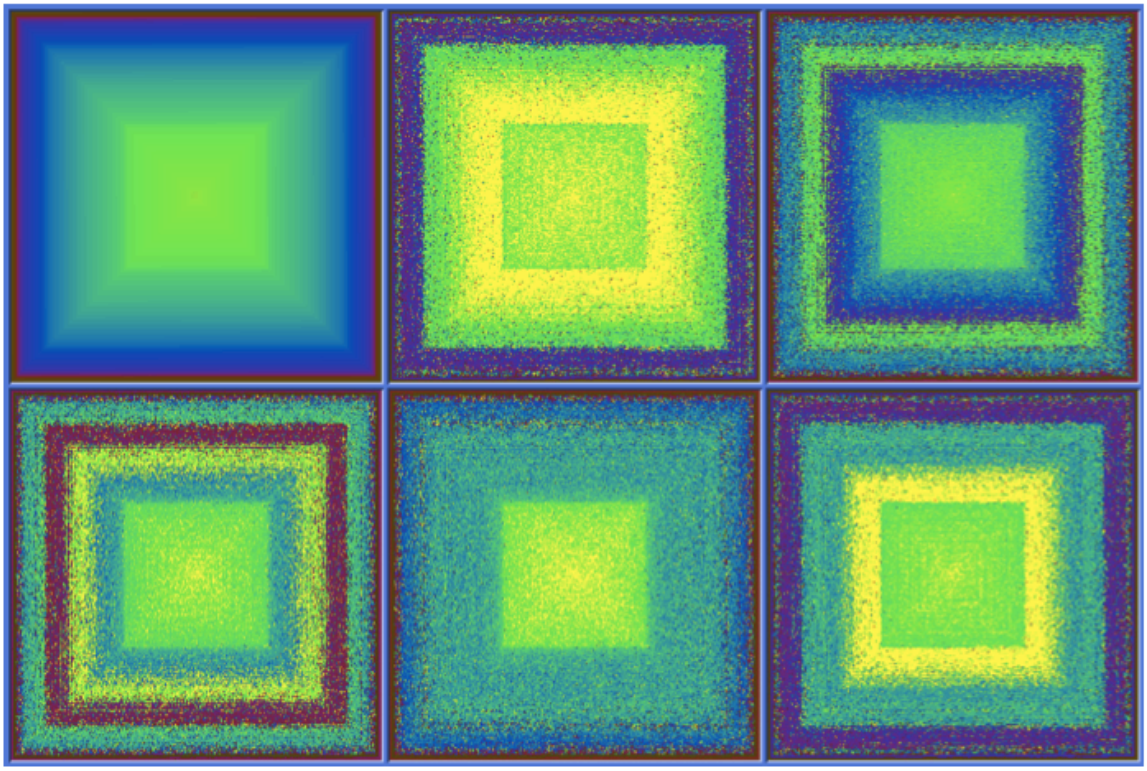


Fig. 1.11: Visualization result with VisDB [17].

VaR(Value and Relation) display[33] では，数百次元におよぶ大規模なデータセットを 2 次元上に表示する手法である．この手法では，各次元のデータの持つ値を表示するだけでなく，各次元データの関連性も同時に表すことが可能である．VaR display では，各次元の値を pixel-oriented な方法で並べ，その並べられた結果を 1 枚の glyph として扱う．そうしてできた，複数の glyph の配置の仕方によって各次元データの関連性を表現する．その際には，多変量解析で用いられる多次元尺度構成法 (MDS)[2] や Jigsaw map[30] というデータの並べ方に関する手法に基づき，配置を行う．Fig.1.12 に MDS および Jigsaw map に基づいて各 glyph を配置した結果例を載せる．

また，これらのような pixel-oriented な手法では，各次元のデータ（1 次元のデータ）を限られた 2 次元スペース上にできるだけ多くの情報を詰めるために，矩形や円形などの 2 次元平面化して並べて表示し

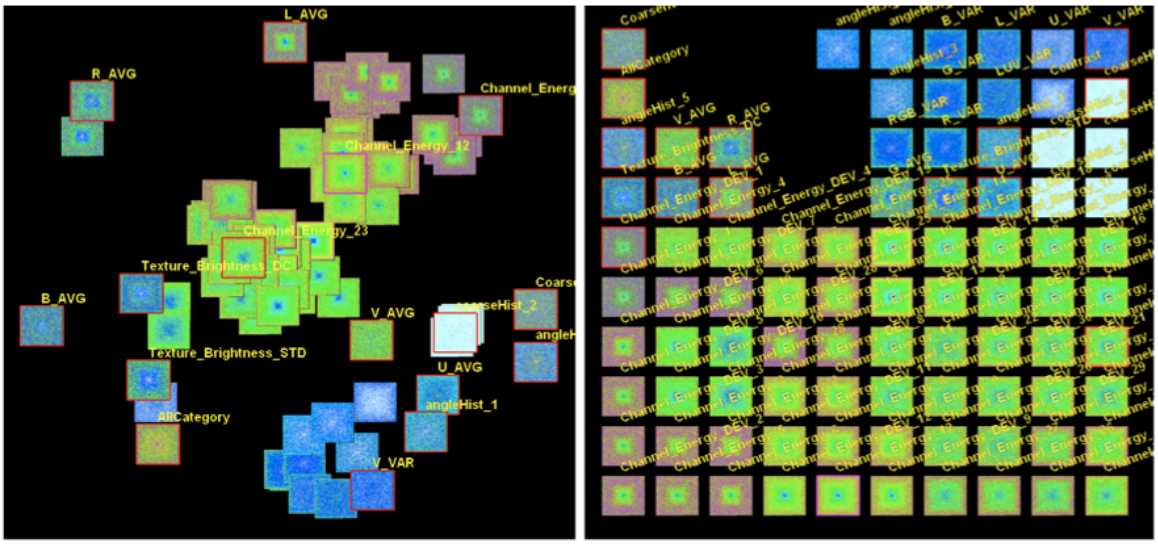


Fig. 1.12: Visualization results with VaR display. Left: based on MDS algorithm, Right: based on Jigsaw map algorithm [33].

ており，各次元データの高次元化を行っているとみなすこともできる．さらに，データの値等を表現するピクセルの 2 次元上での並べ方についても，詳細に議論されている [14], [15].

情報的高次元データの可視化手法では，空間的な構造を持つデータを主対象とした可視化手法ではないため，空間的な階層構造や位置情報などを反映した可視化を行わないといった点で，空間的な高次元データの可視化には適さない．

1.2.6　多変量解析における低次元化

アンケートなどで取得されるデータは，複数の変量を持つ多変量のデータとなることが多い．こうした複数の変量を持つデータは，各変数が 1 つの次元方向の値を取っているとみなすと，その変数の数だけの次元を持つ高次元データである．多変量データを解析するための手法は多数開発されているが，ここでは本研究と関係のある低次元化による多変量データ解析の方法を取り上げる [43].

主成分分析

多変量解析における低次元化の手法として主成分分析が挙げられる．主成分分析では，多変数の相関構造を考慮して，多変数を少数の次元の合成変数 (通常 2, 3 次元) に低次元化変換し，多変数データが持つ情報を低次元な結果から解釈できるようになる．

多次元尺度構成法

他の低次元化手法として，多次元尺度構成法 (MDS) が挙げられる．MDS では，多変量を持つ複数データを，そのデータの持つ多変量の値の類似度に基づいて，データを 2 次元上などに配置する手法である．この手法では，類似度が高いデータは低次元化後の図形上で近くに配置されるようになる．多次元尺度

構成法のほかにも類似度が高いデータを近くに配置させる手法として，自己組織化マップなども挙げられる．

1.3　フラクタルや多解像度化を利用した情報要約可視化

1.3.1　フラクタルの概念を用いた可視化

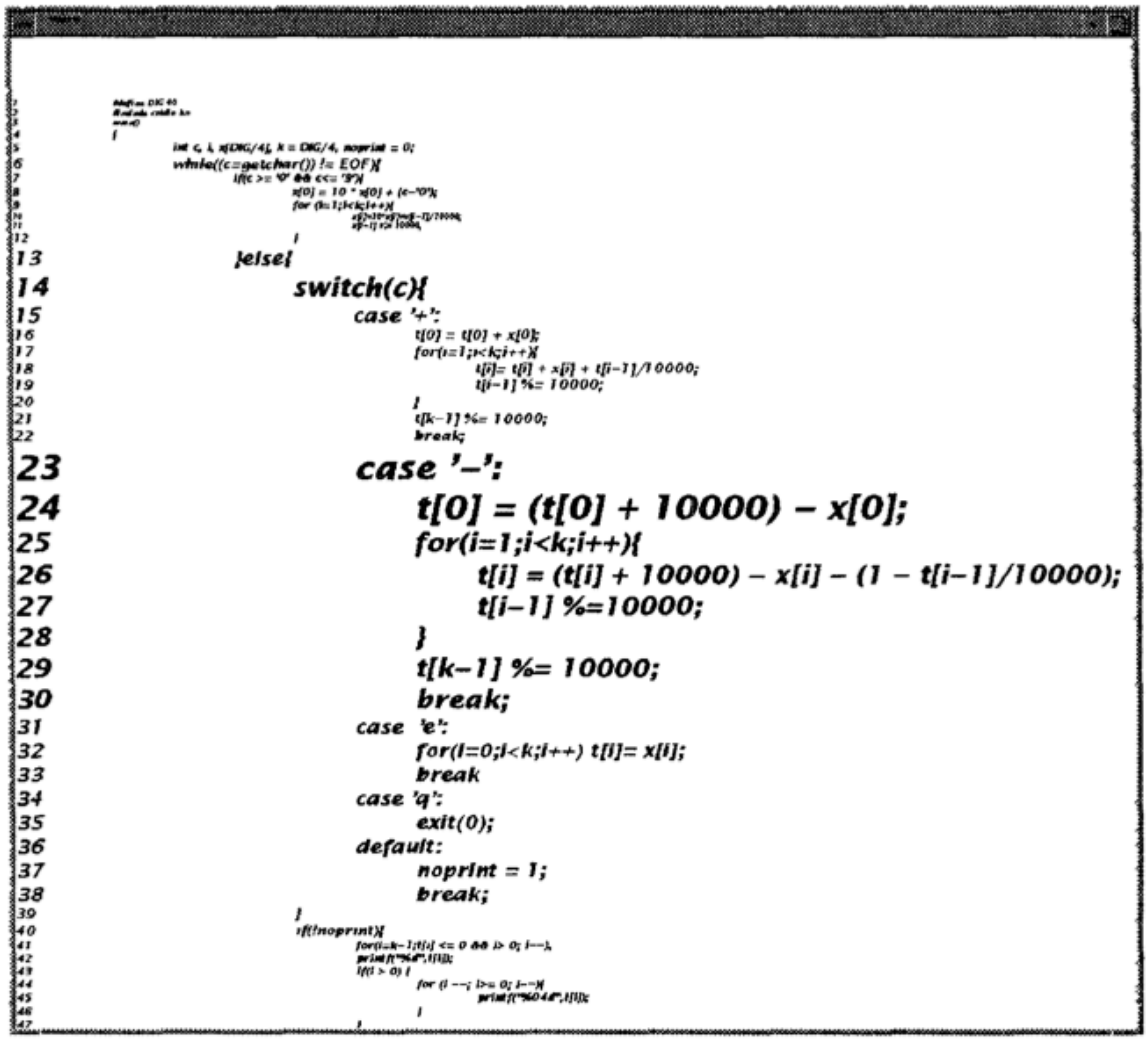
我々の手法では，フラクタルを用いた多解像度可視化を行うことで，データの要約やデータの階層性の保持を実現しているが，同様にフラクタルや多解像度可視化を利用して，着目すべきデータの箇所の強調や必要性の少ないデータの非表示および簡略化などを行う可視化手法がある．

Fractal Views[18] では，フラクタル理論的な重要度の概念を導入して，可視化されるオブジェクトの一部のカリングを行うことで，表示情報量を一定量に保った可視化を実現している．Fractal Views では，可視化の前に情報量の制御を行うため，様々な情報を対象にでき，また可視化の方法も様々な手法を利用できる．Fig.1.13 では，プログラムコードを Fractal Views を用いて可視化を行っている．

Yang ら [31] は，同様のフラクタル的重要度の考えを Web ドキュメントの構造に対して用いることで，インタラクティブな文書の要約可視化を実現し，表示可能領域や計算資源が限られた環境 (例えば，携帯電話) でも文書を容易に表示できるようにしている (Fig.1.14).

1.3.2　多解像度化を利用した可視化

多解像度化を利用して，表示すべき情報を効率的に可視化した例として，Hao らの手法 [7] がある．Hao らは大規模時系列データに対し，関心度 (the degree of interest, DOI) の概念と多解像度化を用いることで，表示領域を有効に活用し，全データの表示，およびユーザーの認知負荷を低減した可視化を実現している．Fig.1.15 はその可視化結果例である．



```
switch(c){
    case '+':
        t[0] = t[0] + x[0];
        for(i=1;i<k;i++){
            t[i]= t[i] + x[i] + t[i-1]/10000;
            t[i-1] %= 10000;
        }
        t[k-1] %= 10000;
        break;
    case '-':
        t[0] = (t[0] + 10000) - x[0];
        for(i=1;i<k;i++){
            t[i] = (t[i] + 10000) - x[i] - (1 - t[i-1]/10000);
            t[i-1] %=10000;
        }
        t[k-1] %= 10000;
        break;
    case 'e':
        for(i=0;i<k;i++) t[i]= x[i];
        break
    case 'q':
        exit(0);
    default:
        noprint = 1;
        break;
}
```

Fig. 1.13: Fractal Views Application example: program code with multiscalable font mode[18].

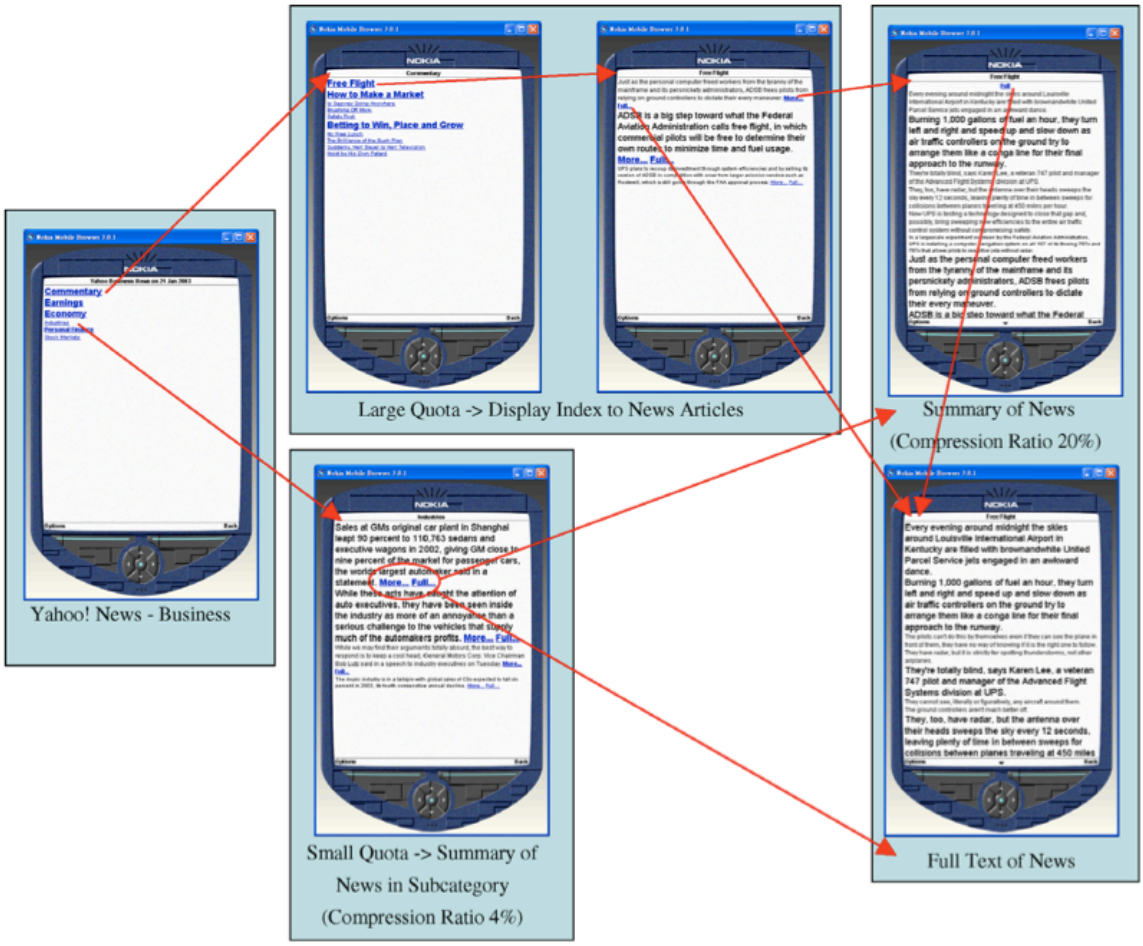


Fig. 1.14: Financial news delivery system on mobile devices [32].

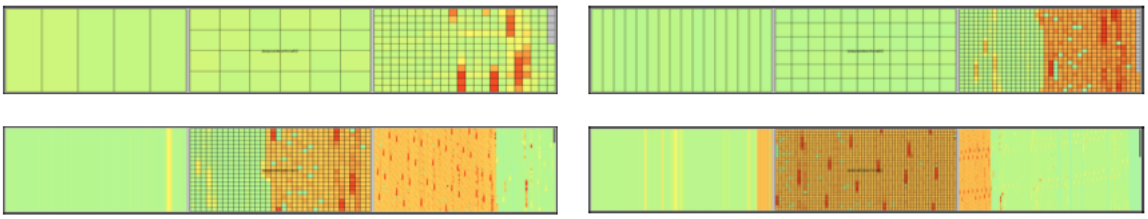


Fig. 1.15: CPU utilization values for 288, 864, 7.776, and 25.920 values (top-left to bottom-right) taken from one target host on a network. The display allows visually analyzing large time intervals, keeping the most recent data (leftmost partition in each image) at highest resolution. The cell sizes for the older data are decreased by increasing data size [7].

1.4　フラクタル図形を用いたボリュームデータの 2 次元展開可視化手法

本研究の基礎となる既存研究として，岩丸らのフラクタル図形を用いたボリュームデータの 2 次元展開可視化手法がある [34], [35], [36], [37], [38] ．詳細は，論文及び特許公開の [34], [35], [36], [37], [38] に譲るが，本研究の理解補助のため，この節で概略を示す．この手法では，ボリュームデータの持つ空間的な八分木構造 (Octree) をシェルピンスキーのカーペットと呼ばれるフラクタル図形上に写像展開することでボリュームデータの 2 次元化表示を実現する．2 次元化することで 3 次元データの持つ空間的遮蔽性や視点依存性を解消し，3 次元データの理解を補助する可視化手法となっている．

1.4.1　ボリュームデータとシェルピンスキーのカーペットの解像度と要素数

ボリュームデータは解像度 R のとき，8^R 個の要素 (ボクセル) を持つ．ここで，ボリュームデータの解像度が R であるとは，ボリュームデータが各次元方向に 2^R 個の分割数を持つものとする．Fig.1.16 に解像度 1,2,3 のときのボリュームデータを構成するボクセルの様子と要素数を載せる．

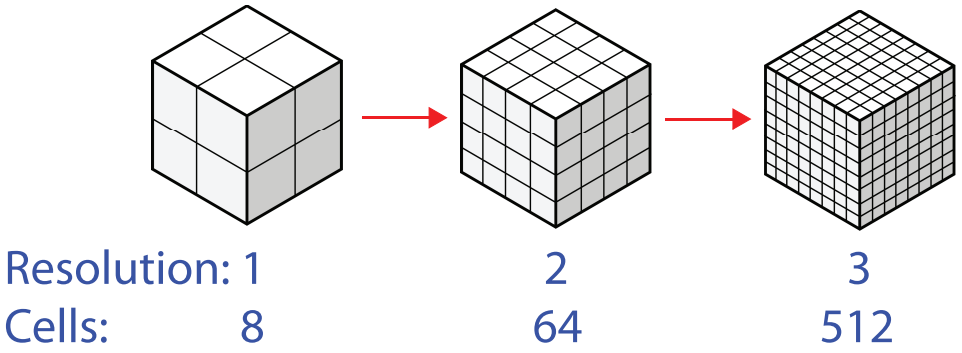


Fig. 1.16: Volume data with resolution 1, 2 and 3.

一方で，シェルピンスキーのカーペットと呼ばれるフラクタル図形も解像度 R のとき，8^R の要素 (矩形領域) を持つ．ここで，フラクタル図形の解像度が R とは，フラクタル図形を作成する再帰的なルールの適用回数が R 回である状態を指すこととする．シェルピンスキーのカーペットの場合，各矩形を 9 領域に分割し，9 領域の中心領域を空の領域に置き換えるといったルールを再帰的に適用するため，解像度 1,2,3 のときの様子は Fig.1.17 のようになる．

ボリュームデータをシェルピンスキーのカーペット上に展開するときには，ボリュームデータとシェルピンスキーのカーペットが解像度 R のとき，ともに要素数 8^R を持つことを利用する．

1.4.2　ボリュームデータの 2 次元展開

ボリュームデータの要素数と一致する 2 次元図形が前小節で得られたので，ボリュームデータとシェルピンスキーのカーペットの各要素間に 1 対 1 のマッピングルールを定めれば，ボリュームデータの 2 次元展開が実現する．マッピングルールを決める際には，解像度 1 のときのマッピングルールを定め，ボ

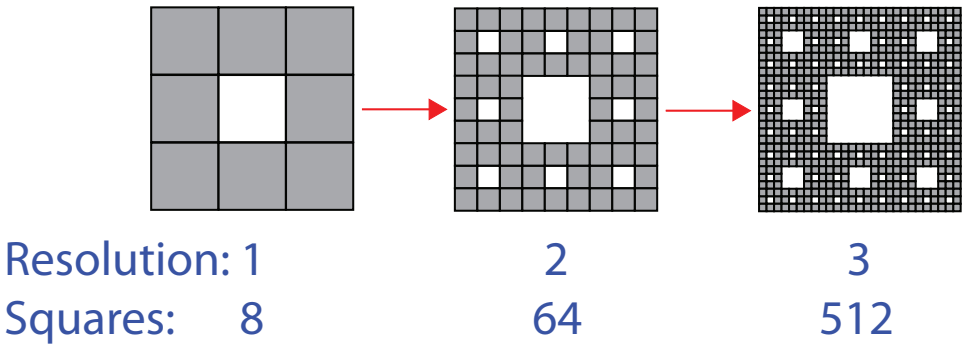


Fig. 1.17: Sierpinski carpets with resolution 1, 2 and 3.

リュームデータがより高解像度なときには，解像度 1 のときのマッピングルールを再帰的に適用する．これは，ボリュームデータ，シェルピンスキーのカーペットはともに再帰的な分割によって解像度が変化するために可能なことである．解像度 1 のときのマッピングルール例を Fig.1.18 に載せる．また，解像度が 2 の場合に，再帰的にマッピングルールを適用した例を Fig.1.19 に載せる．この Fig.1.19 からも分かるように，再帰的なルールの適用は局所的に行うことが可能であり，ボリュームデータが多解像度を持つときにも多解像度を持つ 2 次元可視化結果として表示することが可能である．そのため，ボリュームデータの持つ Octree 構造をそのまま 2 次元上に表示することが可能である．

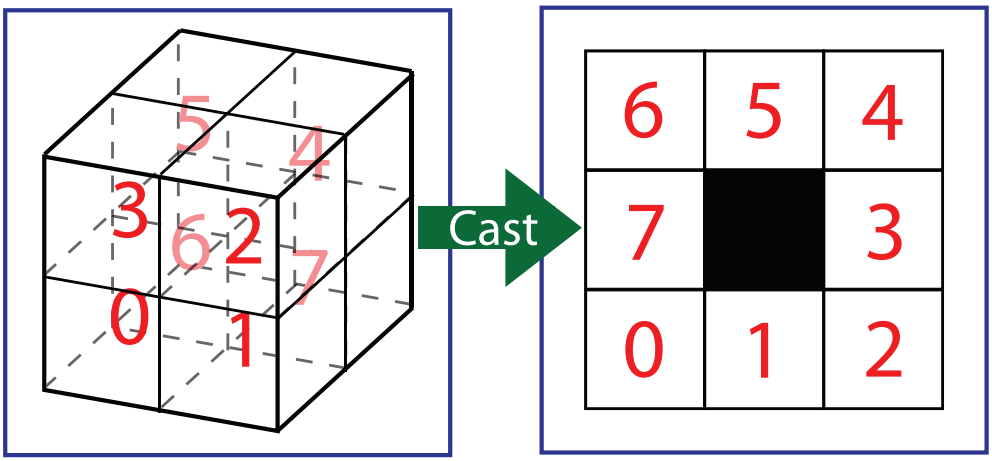


Fig. 1.18: A mapping rule example for resolution 1.

こうして実現したボリュームデータの 2 次元展開は空間的遮蔽や視点依存性を解決するだけでなく，他にも複数の利点を有する．1 つは多解像度化可能なため，注目したい箇所だけ高解像度化し，他の部分は低解像度で表示できるため，ボリュームデータを要約した 2 次元結果を得ることが可能である．また，マッピングルールを再帰的に適用して展開を行うため，解像度 1 のときのマッピングルールが展開結果の様子を左右する．そのため，対象データに応じてルールを決めることによって，効果的な可視化が実現する．マッピングルールとその可視化結果の関係に関しては，岩丸らの研究では詳細に検討していなかったため，第 6 章で詳しく説明する．さらに，ボリュームデータと 2 次元展開結果の各要素間には 1 対 1 の

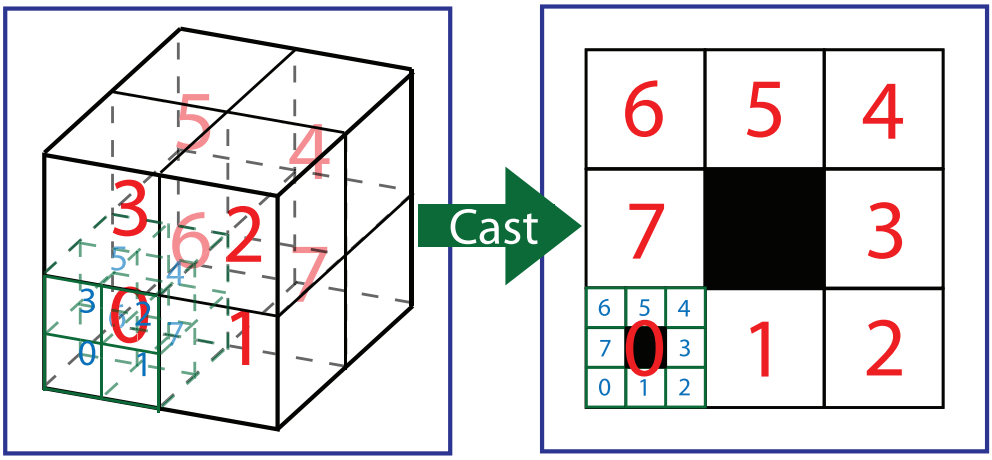


Fig. 1.19: A multi-resolution expansion example.

写像関係があるため，2 次元展開結果をインターフェースとして利用して，ボリュームデータの要素の指定や，ボリュームレンダリングの結果を変化させることができる．これらの特性を岩丸らの手法は持つため，ボリュームレンダリングによる可視化結果と連携して利用することで，ボリュームデータのより効果的な可視化，操作，理解を実現することができる．Fig.1.20 にボリュームレンダリングと連携利用した適用事例を載せる．Fig.1.20 では，シェルピンスキーのカーペットの空の領域にも値が格納されているが，空の領域を利用して空の領域と同階層に含まれる要素の値の平均値を表示している．

1.5　研究目的

本章では，多次元なデータ・大規模なデータはその形状的な複雑さやデータ数の多さによって低次元・小規模なデータよりもデータが示す形状や傾向の理解が困難となっており，理解を補助するような可視化手法が必要であること，およびその可視化手法に関する既存研究を述べてきた．既存研究では，3 次元空間データを 2 次元化して表示することや情報的な高次元データを低次元化して表示することを行った事例はあるが，より高次元な空間データなど多様な次元を可視化対象にする手法は研究されていない．

そこで，本研究の目的として，岩丸らの既存手法を発展・一般化させ，多様な次元のデータを効果的に可視化することを可能にするための手法を開発すること，また，実際に可視化事例を示すことを挙げる [40], [41], [42], [5]．具体的には，高次元なデータを低次元化して表示可能にすること，低次元なデータの高次元化により表示領域を有効に使う方法を示すこと，次元種類別に扱いたいデータ (例えば，時系列ボリュームデータは時間と空間の次元からなるデータであり，それぞれを次元種類別に扱って可視化処理を行うことが望まれる) を可視化する方法を示すこと，さらにこれらの方法とともに認知負荷や描画負荷を低減させる多解像度化処理などの方法を示すこと，時系列ボリュームデータや 4 次元以上のデータの可視化例を示すこと，を行う．提案手法により，高次元データの空間的・時間的遮蔽を含まない可視化，大規模データの効率的な画面配置，さらにデータの要約可視化を可能にし，多次元・大規模なデータの理解の補助を行うことができるようになる．

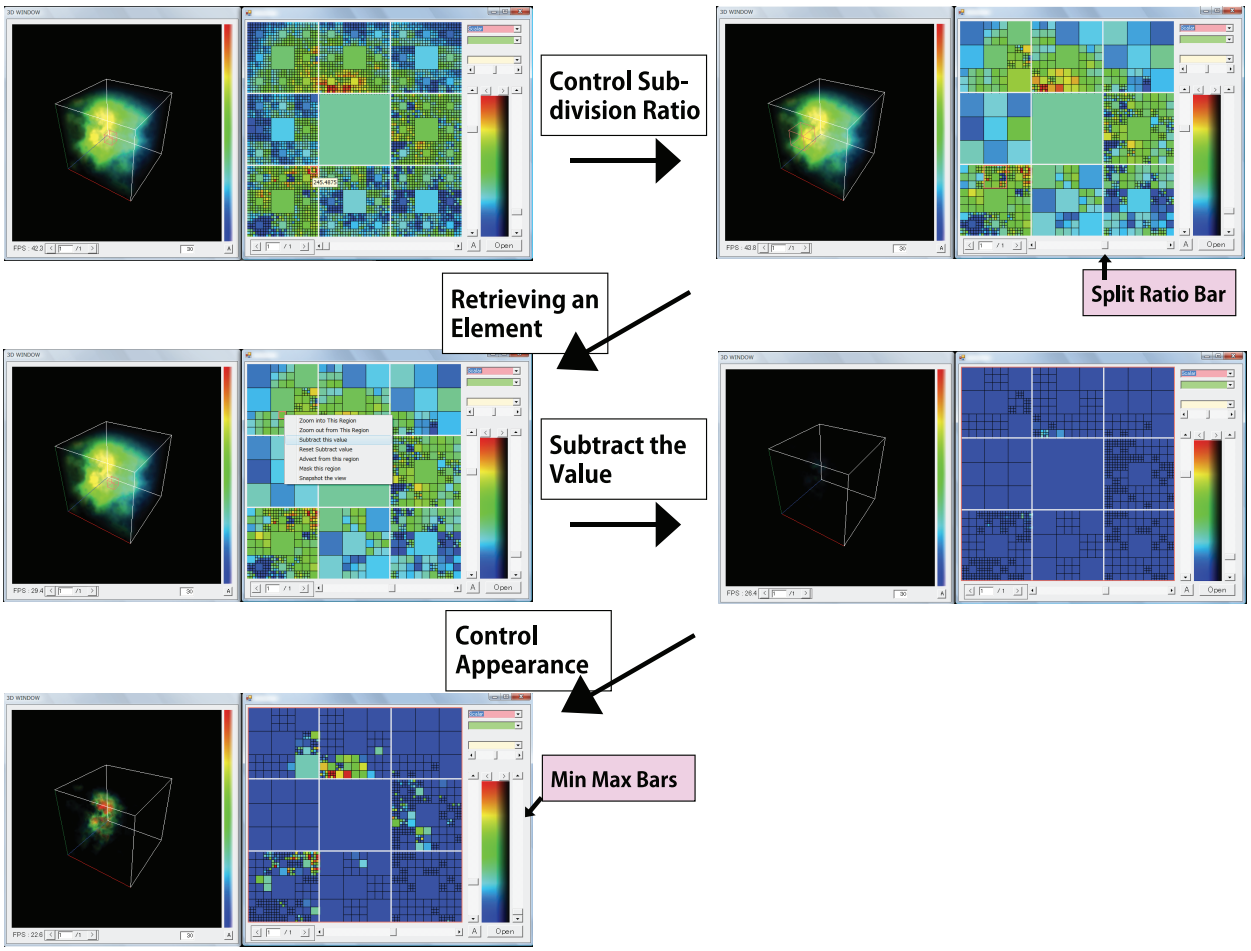


Fig. 1.20: An example pf visualization application for 3D data and its workflow. We subtracted the value of the selected region and highlight the regions of higher values with “Min Max Bars”(color appearance control equipment)

1.6　論文の構成

ここで，本論文の構成を述べる．本章では，本研究の研究背景や関連する既存研究・手法を述べ，それらの状況を踏まえた研究目的を述べた．次からの 2 から 6 章で，提案手法の理論や性質について述べる．2 章では．提案手法で多解像度化可能な次元変換を可能にするために利用する自己相似図形について詳しく述べる．3, 4 章では，自己相似図形を利用しながら再帰的なマッピングを行うことで任意の次元データを低次元化・高次元化して多解像度可視化する方法について述べる．5 章では，さらに次元変換を次元種類別に行う方法や可視化結果の多解像度化の方法を述べる．6 章では，本手法のマッピングの際に利用するマッピングルールの有用な決定方法例を挙げる．7 章では，2 章から 6 章で述べた提案手法を用いた可視化事例を挙げ，その可視化結果について議論を行う．さらに次の 8 章では，すべての章の内容を踏まえて，本手法の有する特徴について考察する．最後に 9 章で，本研究の結論を述べる．

第 2 章

自己相似図形

2.1　自己相似図形と領域数

本節では，自己相似図形の持つ領域数について述べる．はじめに自己相似図形の 1 つであるシェルピンスキーのカーペットを例として取り上げて説明する．シェルピンスキーのカーペットは再帰的なルールを 1 回適用するとその領域数は 8 倍になっていく (Fig.2.2)．一般に自己相似図形の初期状態の領域数を 1 とし，一回のルールの適用で領域数が a 倍になるとすると，R 回のルールの適用で領域数は a^R となる．今後，この再帰的なルールを R 回繰り返して作られる自己相似図形を解像度 R の自己相似図形と呼ぶこととする．再帰的なルールの作成方法によって，様々な領域数を持つ自己相似図形が作成できる．例として Fig.2.1 から Fig.2.8 を載せている．Fig.2.1 は正方形を 4 分割するといったルールで作成され，解像度 R のとき領域数 4^R を持つ．Fig.2.2 はシェルピンスキーのカーペットで，正方形を正方形 9 領域に分けて 9 領域の中心部を空の領域にするといったルールで作成され，解像度 R のとき領域数 8^R を持つ．Fig.2.3 はシェルピンスキーのギャスケットで，正三角形を正三角形 4 領域に分けて 4 領域の中心部を空の領域にするというルールで作成され，解像度 R のとき領域数 3^R を持つ．Fig.2.4 は，正六角形をその正六角形の 3 分の 1 の一辺の長さの各辺を持つ正六角形 6 つと置き換えるというルールで作成され，解像度 R のとき要素数 6^R を持つ．自己相似図形は 2 次元だけでなく，1 次元や 3 次元などその他次元のものも考えられる．Fig.2.5 は，カントール集合と呼ばれる 1 次元自己相似図形の例で，線分を 3 等分して中央の線分を空の領域にするといったルールで作成され，解像度 R のとき領域数は 2^R となる．Fig.2.6 は，3 次元自己相似図形の例で，立方体を x, y, z 方向に 2 分割ずつしてできたもので，解像度 R のとき領域数は 8^R である．Fig.2.7 は，立方体を x, y, z 方向に 4 分割し 64 領域の立方体に分けたあと Fig.2.7 のように立方体の一部を取り除き 16 領域だけを残すルールで作成され，解像度 R のとき領域数は 16^R となる．Fig.2.8 は，立方体を x, y, z 方向に 4 分割し 64 領域の立方体に分けたあと Fig.2.8 のように中心部分から 32 領域を空の領域にするといったルールで作成され，解像度 R のとき領域数は 32^R となる．このように様々な領域数を持つ自己相似図形を作成することが可能である．

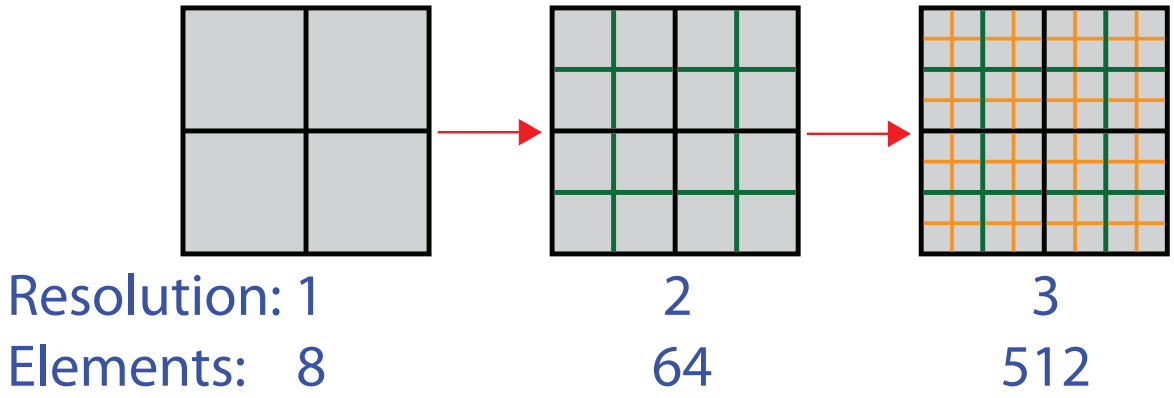


Fig. 2.1: Self-similar object of 4 elements for resolution 1 with resolution 1, 2 and 3.

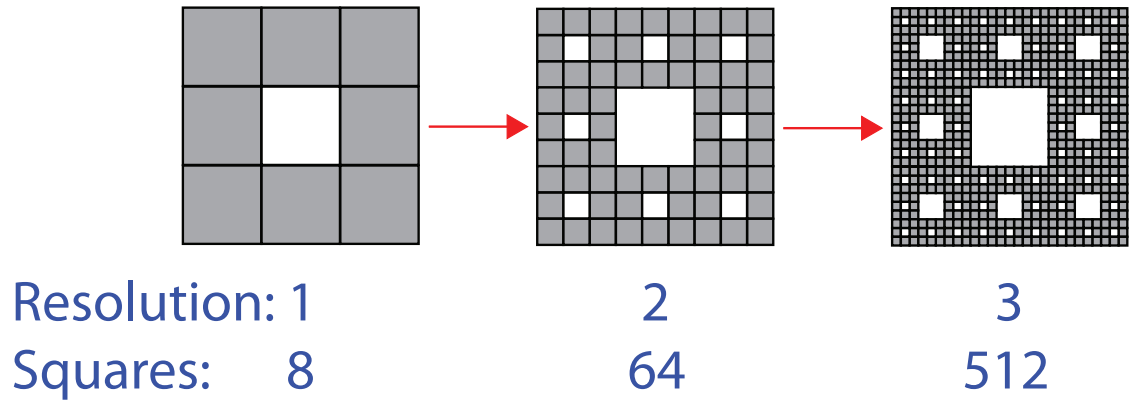


Fig. 2.2: Sierpinski carpet with resolution 1, 2 and 3.

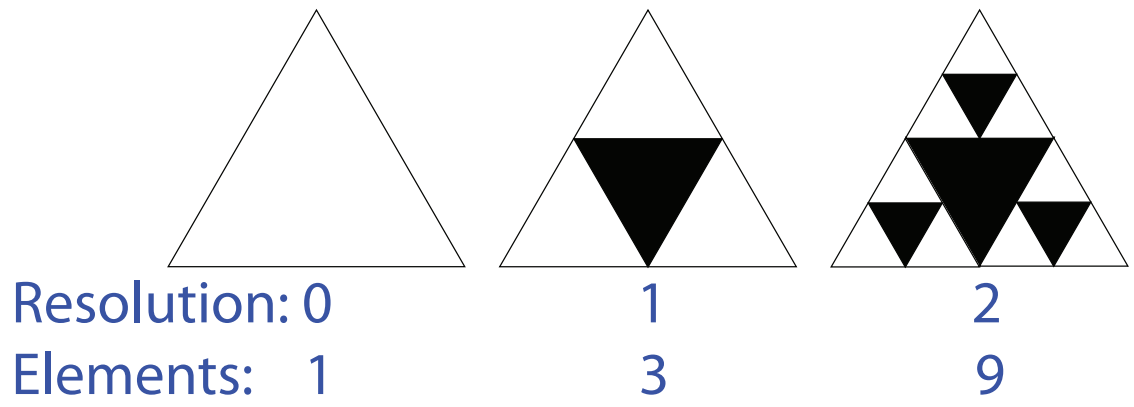


Fig. 2.3: Sierpinski gasket with resolution 0, 1 and 2.

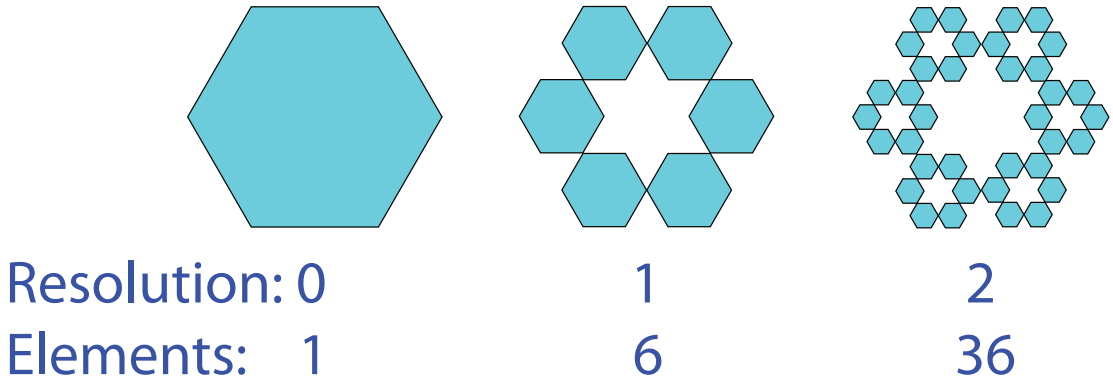


Fig. 2.4: Hexagon fractal with resolution 0, 1 and 2.

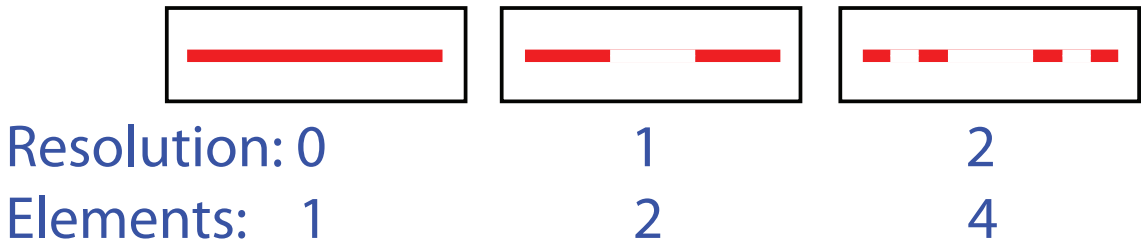


Fig. 2.5: An example of 1D fractal with resolution 0, 1 and 2 (Cantor set).

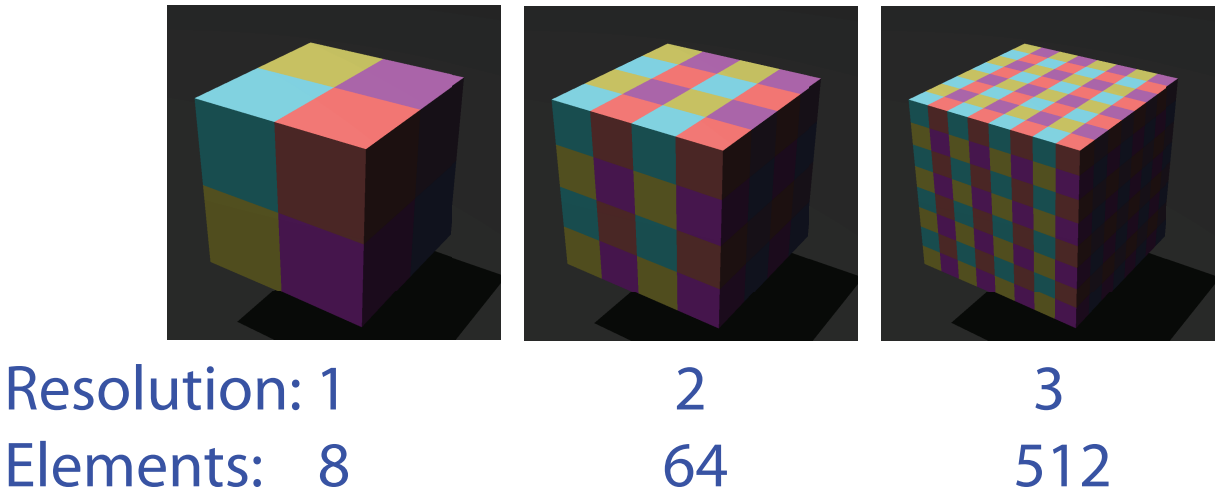


Fig. 2.6: An example of 8 elements 3D fractal for resolution 1.

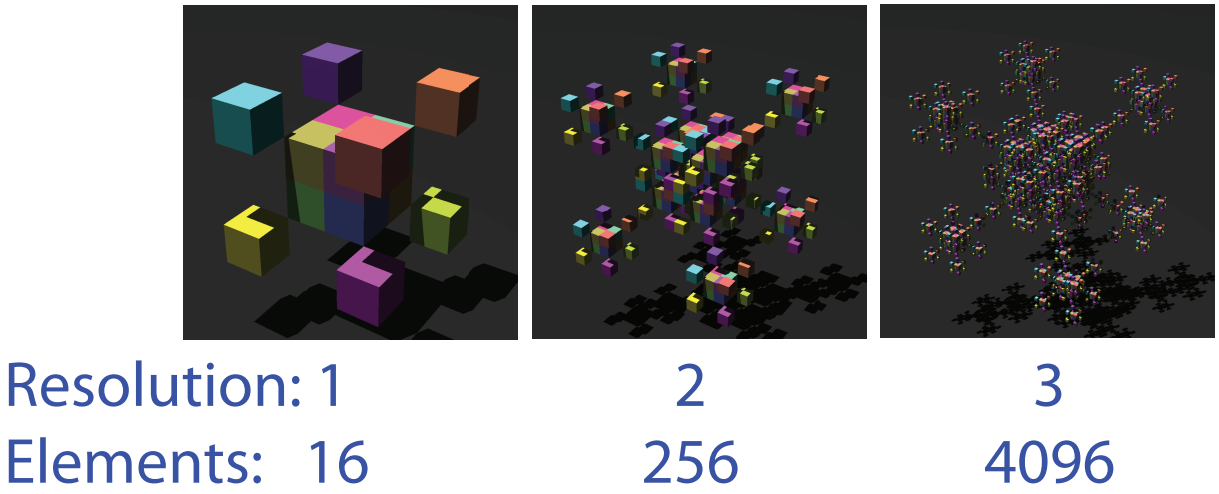


Fig. 2.7: An example of 16 elements 3D fractal for resolution 1.

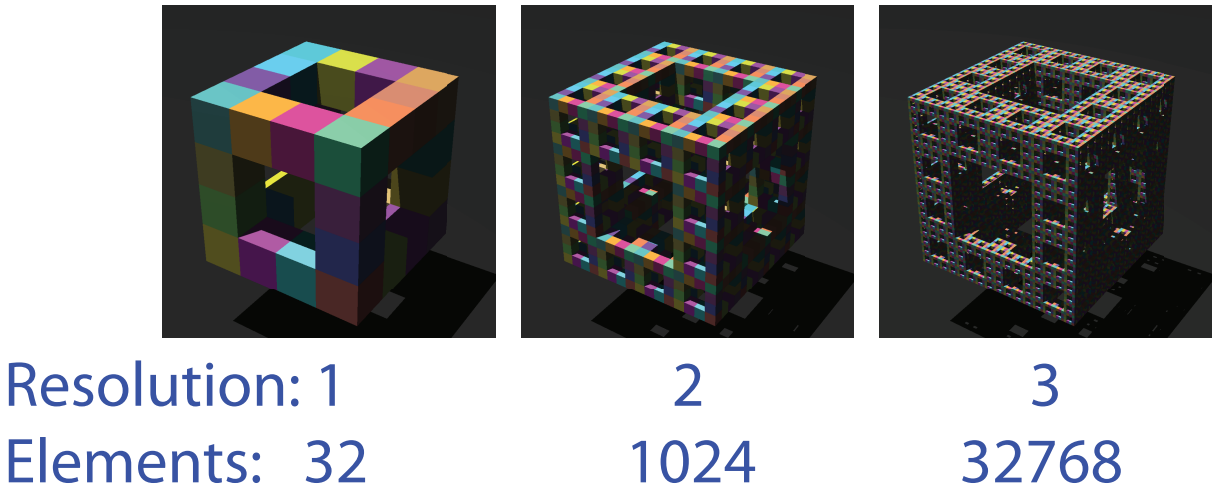


Fig. 2.8: An example of 32 elements 3D fractal for resolution 1.

2.2　組み合わせ自己相似図形

本論文では，「組み合わせ自己相似図形」という図形を新たに定義し，利用する．「組み合わせ自己相似図形」とは，複数の自己相似図形を組み合わせることで新たにできる自己相似図形のことであることとする．複数の自己相似図形を組み合わせて，自己相似図形を作成するため，様々な領域数を持つ自己相似図形が容易に作成することができる．

次に組み合わせ自己相似図形の例を示す．解像度 1 のときのシェルピンスキーのカーペットの各矩形領域を解像度 1 の 4 領域自己相似図形に置き換える (Fig.2.9)．そうしてできた新しい図形を解像度 1 の自己相似図形として扱うと，解像度 1 のとき領域数 32 を持ち，解像度 R のとき領域数 32^R を持つ自己相似図形が得られる．2 次元組み合わせ自己相似図形の例として，解像度 1 のとき要素数 16, 32, 64 の自己相

似図形の例をそれぞれ Fig.2.10, 2.11, 2.12 に載せる．Fig.2.10 は，解像度 1 のとき 4 領域の自己相似図形の各領域を同じく解像度 1 のとき 4 領域の自己相似図形で置き換えたもの (あるいは，解像度 1 のとき 4 領域を持つ自己相似図形の解像度 2 のときの図形を解像度 1 の状態として扱ったものとも考えられる) である．Fig.2.11 の左図は解像度 1 のシェルピンスキーのカーペットの各矩形領域を解像度 1 のとき 4 領域を持つ自己相似図形に置き換えたもの，右図は解像度 1 のとき 4 領域を持つ自己相似図形の各矩形領域を解像度 1 のシェルピンスキーのカーペットで置き換えたものである．Fig.2.12 の左図は，Fig.2.10 の各矩形領域を解像度 1 のとき 4 領域を持つ自己相似図形で置き換えたもの (あるいは，解像度 1 のとき 4 領域を持つ自己相似図形の解像度 3 のときの図形を解像度 1 の状態として扱ったもの)，右図は解像度 1 のシェルピンスキーのカーペットの各矩形領域を同じく解像度 1 のシェルピンスキーのカーペットで置き換えたもの (あるいは，解像度 2 のシェルピンスキーのカーペットを解像度 1 の状態とみなして扱ったもの) である．

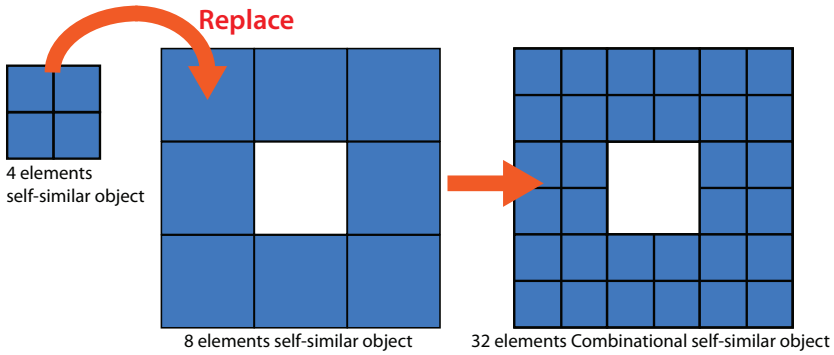


Fig. 2.9: Process of making cominational self-similar object.

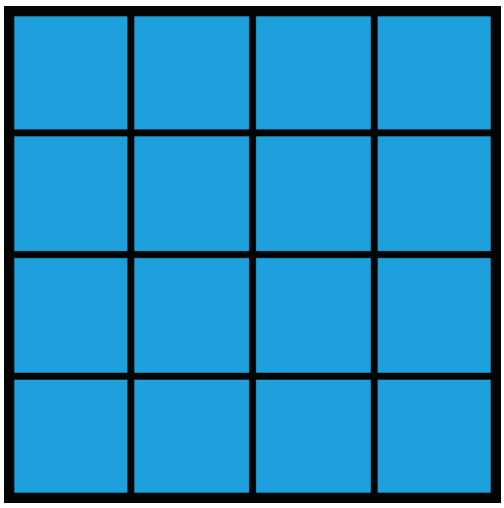


Fig. 2.10: 16 elements self-similar object for resolution 1.

また，2 次元以外の他の次元でも組み合わせ自己相似図形は作成できる．3 次元の組み合わせ自己相似図形として，解像度 1 のとき要素数 64, 128, 256 のものをそれぞれ Fig.2.13, 2.14, 2.15 載せる．Fig.2.13 は x,y,z 方向に 2 分割され，解像度 1 のとき領域数 8 を持つ自己相似図形の各立方体領域を同じく解像度

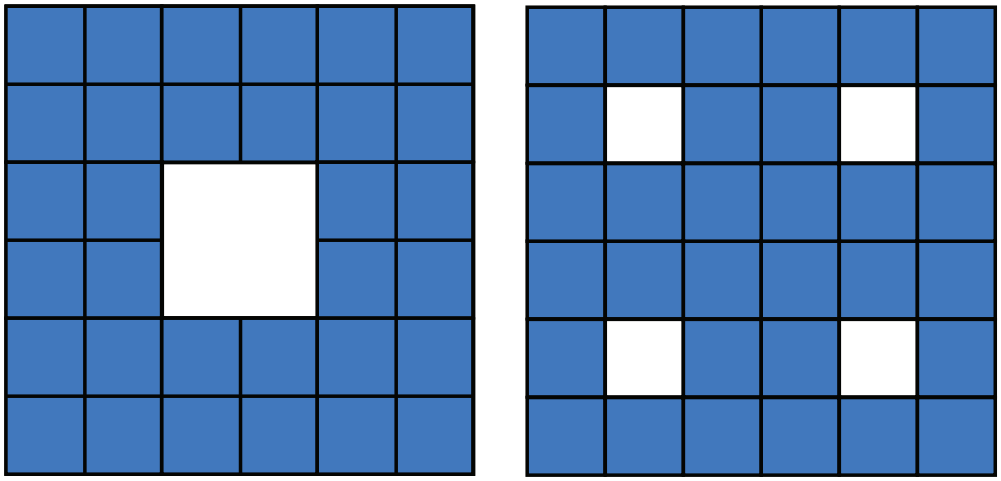


Fig. 2.11: 32 elements self-similar objects for resolution 1.

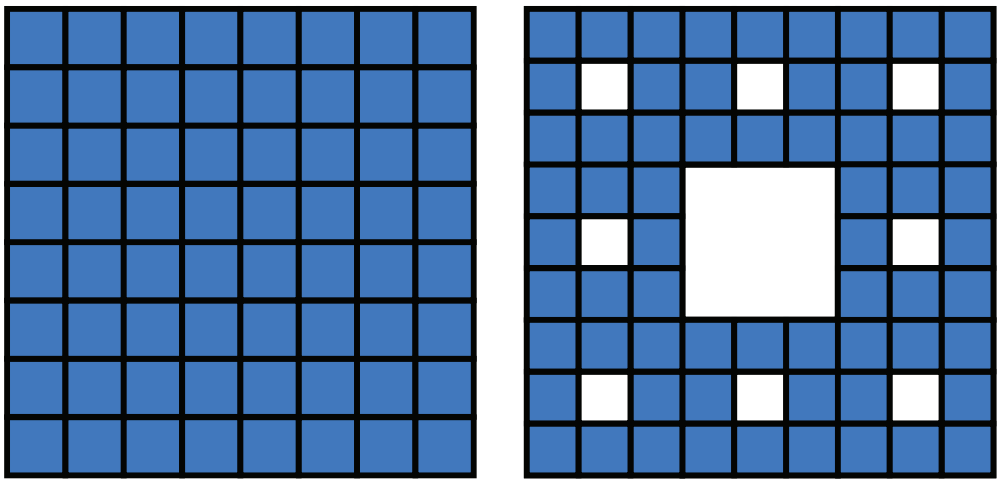


Fig. 2.12: 64 elements self-similar objects for resolution 1.

1 で領域数 8 を持つ自己相似図形に置き換えたものである (あるいは，解像度 1 のとき領域数 8 を持つ自己相似図形の解像度 2 のときの図形を解像度 1 の状態としてみなしたものとも考えられる)．Fig.2.14 左図は解像度 1 のとき領域数 16 を持つ 3 次元自己相似図形の各立方体領域を解像度 1 で領域数 8 を持つ自己相似図形に置き換えたもので，右図は解像度 1 のとき領域数 8 を持つ 3 次元自己相似図形の各立方体領域を解像度 1 で領域数 16 を持つ自己相似図形で置き換えたものである．Fig.2.15 左図は解像度 1 のとき領域数 32 を持つ 3 次元自己相似図形の各立方体領域を解像度 1 で領域数 8 を持つ自己相似図形に置き換えたもので，右図は解像度 1 のとき領域数 8 を持つ 3 次元自己相似図形の各立方体領域を解像度 1 で領域数 32 を持つ自己相似図形で置き換えたものである．

組み合わせ自己相似図形の利点の 1 つとして，組み合わせ自己相似図形の作成時に用いる自己相似図形の解像度 1 のときの領域数を素因数として持つ領域数の組み合わせ自己相似図形ならどんな領域数のものでも作成可能であるということが挙げられる．例えば，2 次元組み合わせ自己相似図形に用いる自己相似図形の例として解像度 1 のとき 4, 8 領域を持つものを挙げたが，これらの図形を用いると解像度 1 のと

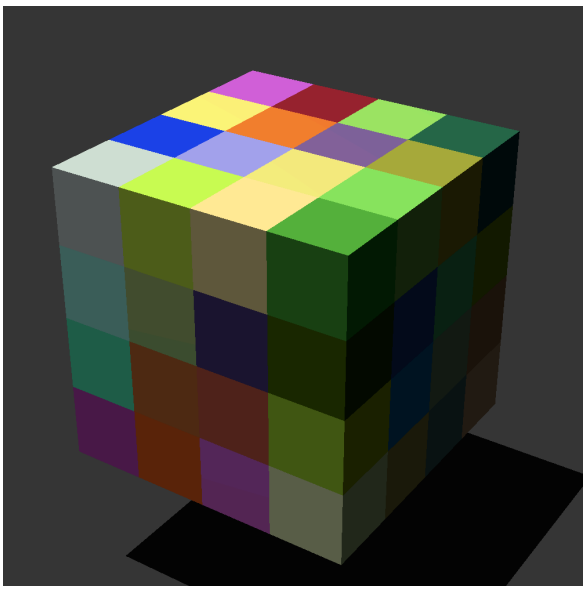


Fig. 2.13: 64 elements 3D self-similar objects for resolution 1.

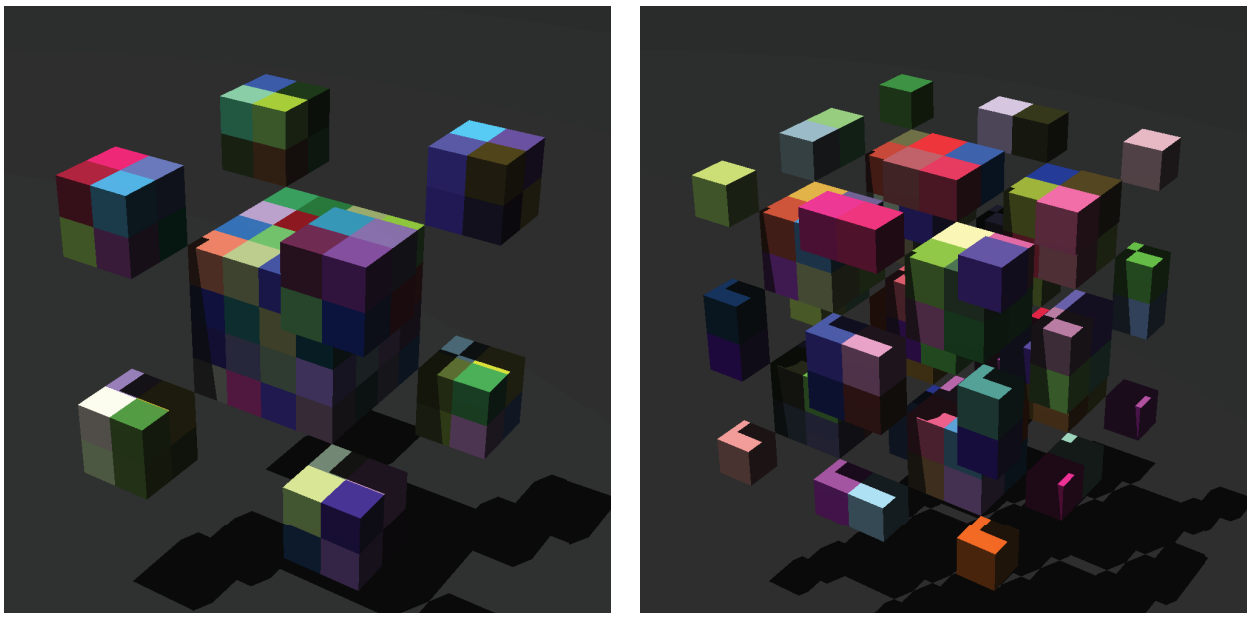


Fig. 2.14: 128 elements 3D self-similar objects for resolution 1.

き領域数 $4^m \cdot 8^n (m,n$ は 0 以上の整数) の自己相似図形 (領域数 $4,8,16,32,64,128$ の自己相似図形など) を作成することができる.

このように組み合わせ自己相似図形を用いると様々な要素数の自己相似図形が得られるため，組み合わせ自己相似図形は多様な要素数を持つデータを自己相似図形上にマッピングしたい場合などにおおいに役立つ.

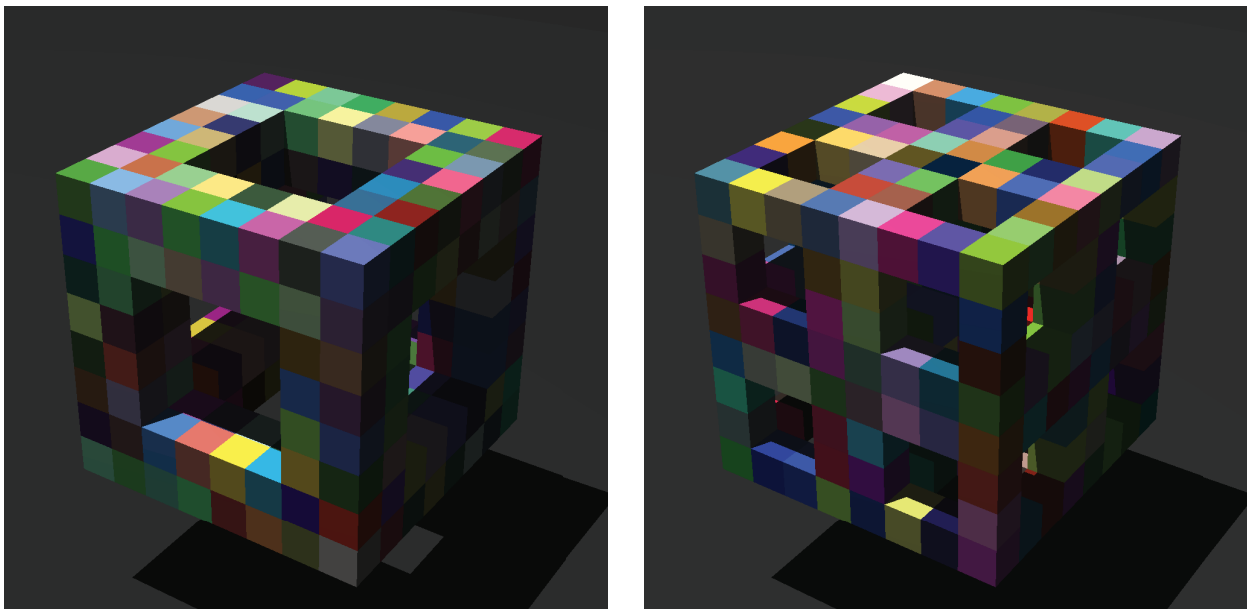


Fig. 2.15: 256 elements 3D self-similar objects for resolution 1.

2.3　多解像度を持つ自己相似図形

自己相似図形は，その自己相似図形を作るために再帰的なルールを一部分のみに適用することが可能である．そのため，高解像度な部分と低解像度な部分を持つ多解像度な自己相似図形を作成することができる．また，組み合わせ自己相似図形も自己相似図形であるため，多解像度を持つ自己相似図形も作成可能である．Fig.2.16 は，解像度 1 のとき 32 要素を持つ組み合わせ自己相似図形が多解像度を持つ場合の例である．

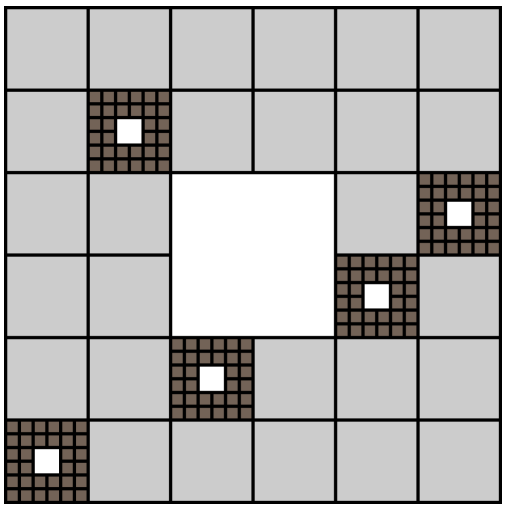


Fig. 2.16: An example of multi-resolution self-similar object.

2.4　正方形や立方体で構成される自己相似図形の分類

本節では，正方形や立方体で構成される自己相似図形をその自己相似図形の持つ正方形や立方体の位置の中心点からの対称性や空白領域に応じて分類分けを行えるようにする．ここで扱う正方形や立方体で構成される自己相似図形とは，$n \times n$ の 2 次元正方形領域，またはその一部を空の領域にして作成される自己相似図形 (例えば，シェルピンスキーのカーペット) や，3 次元立方体領域，またはその一部を空の領域にして作成される自己相似図形であるとする．対称性を持った自己相似図形を用いて低次元化や高次元化を行った場合，その対称性を活かしたマッピングルールを定めることができるなどのメリットがある．

また，以下に述べない点対称性や線対称性などの対称性も考えられるが，それらは一般的な定義のまま正方形や立方体で構成される自己相似図形に利用できるため説明しない．

2.4.1　放射対称性

ある自己相似図形の中心点から等距離にある箇所はすべて正方形，立方体がある領域，または空の領域となっている場合に「放射対称性」を持つということとする．Fig.2.17 に 2 次元の場合の同じ正方形領域数からなる放射対称・非放射対称の自己相似図形を載せる．

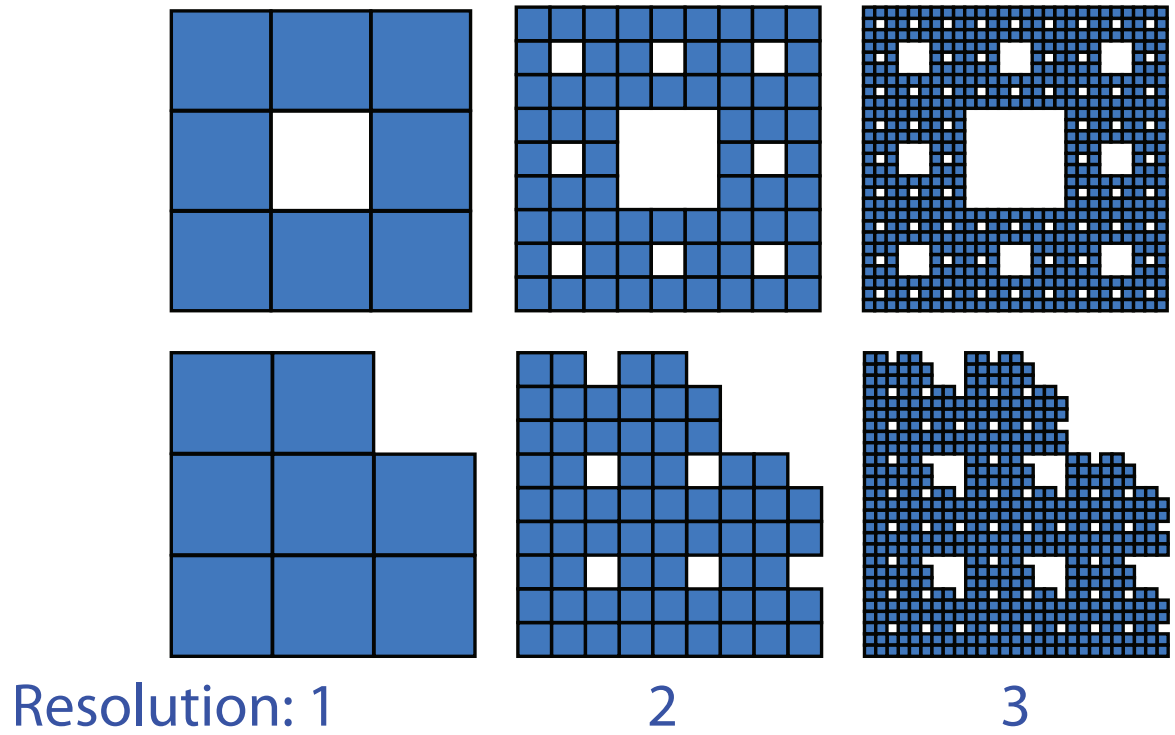


Fig. 2.17: Examples of 2D radial symmetry and asymmetry self-similar objects.

2.4.2　空白領域の分布

ある自己相似図形が空白領域を持ち，解像度 1 のすべての空白領域がそれぞれ他の 1 つ以上の空白領域と辺または面で接しているとき，その自己相似図形は「解像度 1 で空白領域が集まっている」ということとする．逆にその条件を満たしていない自己相似図形の場合には，「解像度 1 で空白領域が散らばっている」ということとする．Fig.2.18 に同正方形領域数を持つ 2 次元自己相似図形の解像度 1 で空白領域が集まっている場合のものと散らばっている場合のものを載せる．

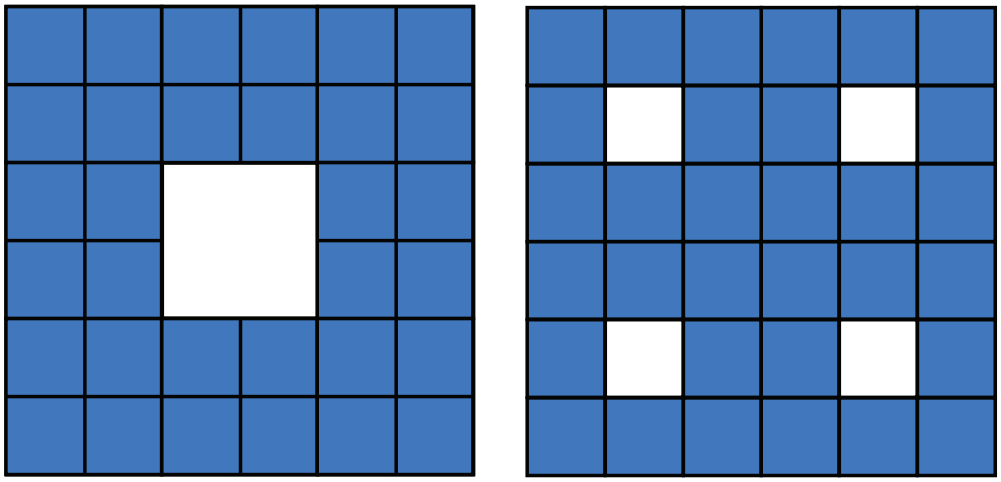


Fig. 2.18: Examples of blank distribution of 2D self-similar objects.

2.5　自己相似図形のまとめ

本章では，自己相似図形の解像度と領域数について説明し，さらに多様な領域数を作るために組み合わせ自己相似図形という考えを新たに作成した．自己相似図形の領域数は対象データを次元変換後の図形へ一対一のマッピングを行う際に重要となるものである．また，自己相似図形の多解像度化可能性についても説明した．自己相似図形の多解像度化は，データのマッピング後の結果を要約化したり，描画負荷を低減する際に利用される．自己相似図形の分類方法についても説明を行った．これらの分類方法は，次章以降において提案手法等を説明する際に利用する．自己相似図形の持つ再帰性や多解像度化が可能であるという性質を利用することが次章以降で説明する提案手法の大きな特徴の 1 つである．

第 3 章

高次元データの低次元化手法

3.1　低次元化の目的

可視化の対象となるデータは多様な次元を持ち，ときにデータが高次元であるときがある．例えば，ボリュームデータは 3 次元データ，時系列ボリュームデータは 3 次元の空間と 1 次元の時間方向を持つ 4 次元時空データであるが，これらのデータは通常ボリュームレンダリングなどの 3 次元可視化手法とアニメーションを併用して表示する．しかし，そうした手法で可視化を行った場合には，3 次元，または 4 次元のデータをそのままの次元として扱って表示するため，空間的な遮蔽と時間的な遮蔽が生じるといった欠点が存在する．ボリュームデータや時系列ボリュームデータを低次元化して表示することにより，空間的遮蔽・時間的遮蔽のどちらも解消することができる．また，4 次元以上の高次元データの場合は，低次元化せずに可視化を行うことは不可能である．低次元化によってあらゆる高次元のデータを表示することが可能となる．そのため，高次元空間での様々な現象の理解を助ける可視化を行うことができる．

3.2　低次元化手法の概要

本低次元化手法では，高次元のデータを低次元化し，1, 2, 3 次元上，および 0 次元的に表示する．その際には，高次元データの解像度・要素数と展開先の図形の解像度・要素数を一致させ，1 対 1 の写像を行う．さらに，低次元化の際に自己相似図形を利用することにより，対象データの位置的な階層構造を保持した可視化，さらに多解像度化を可能としている．Fig.3.1 に低次元化手法の概要図を載せる．

次節からは，具体的に低次元化の手法を述べるが，一般的に利用されるディスプレイが 2 次元であるために最も利用頻度が高いと思われる高次元データの 2 次元化表示について初めに詳しく述べた後に，高次元データの 2 次元以外の低次元化表示について述べる．低次元化の際に利用する高次元データと同要素数を持つ自己相似図形については，その形状が無限に考えられるが，説明を簡単にするために次に示す条件を満たす自己相似図形だけを取り上げて説明する．これらの条件を満たす自己相似図形は，放射対称性を持つため点対称などの位置関係が維持できる，できるだけ密に詰まったピクセル数やボクセル数で表示できるため，表示領域を有効に利用できるなどの長所を持つ．

「説明に用いる自己相似図形を決定する条件」

1. 放射対象性を持つもの．

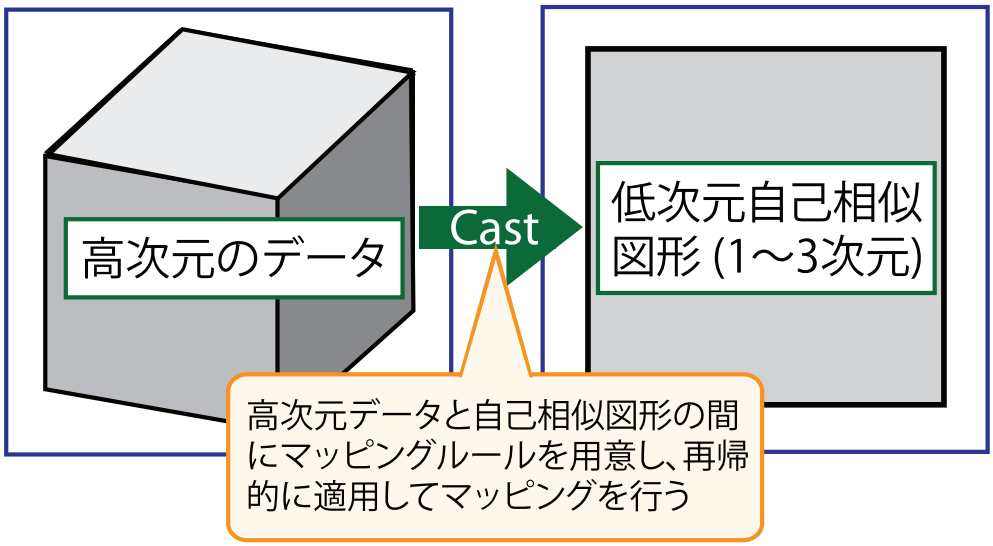


Fig. 3.1: Summary of the dimensional reduction method

2. できるだけ 1 辺の長さ，ピクセル数，ボクセル数が少ないもの．ただし，空白領域もカウントに含めることとする．
3. 空白領域が集まっているもの．
4. 上記 1 から 3 の条件をすべて満たすような自己相似図形が存在しない場合は，1, 2, 3 の順番に条件を優先して自己相似図形を決める．

3.3　n 次元次元データと自己相似図形の解像度と要素数

低次元化を行うために，はじめに低次元化の対象とするデータの次元数とそのデータの解像度と要素数，またマッピングを行う先として利用する自己相似図形の解像度と要素数について述べる．

3.3.1　n 次元データの解像度と要素数

次元数 n のデータが解像度 R を持つとき，データを構成する要素数は 2^{nR} 個となる．ここでは，解像度 R とは各 n 個の次元方向にデータが 2^R 個分割されていることを示すこととする．$n = 1,2,3$ のデータの例を Fig.3.2 に示す．

3.3.2　再帰的分割によってできる自己相似図形の解像度と要素数

シェルピンスキーのカーペット等に代表される再帰的な分割ルールにより作成される自己相似な図形は，再帰的なルールの適用回数により図形の詳細さが変わる．ここで，再帰的な分割ルールを R 回適用した自己相似な図形の解像度を R と定めることとする．例えば，シェルピンスキーのカーペットは解像度 1 の要素数 8 を持ち，解像度 R のとき要素数は 8^R となる (Fig.1.17)．一般に解像度 1 のとき要素数 a の自

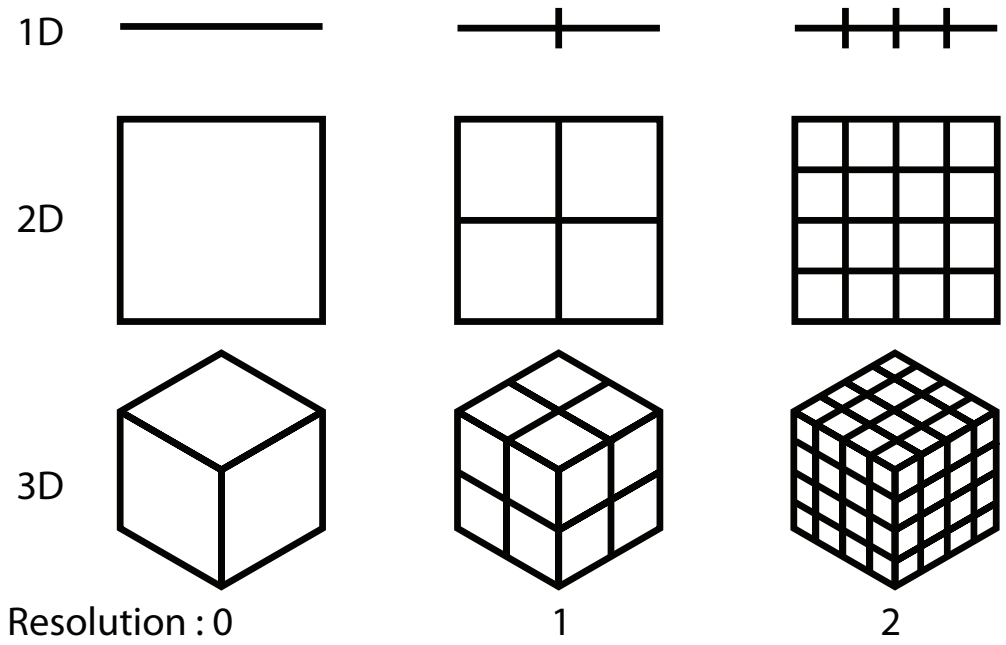


Fig. 3.2: Dimensions and elements with resolutions 1, 2 and 3

己相似図形は，解像度 R のとき要素数 a^R となる. また，前節 3.3.1 で挙げた各次元軸方向に 2^R 分割された図形も，2^{R-1} 分割された図形の各要素を各次元軸方向に 2 分割することで得られることから，再帰的な分割によってできる自己相似な図形であると考えられる. さらに，自己相似図形は局所的に再帰的な分割ルールを適用することが可能であり，Fig.3.3 や Fig.3.7 のように一部分だけ高解像度化することや低解像度のまま残して置くことが可能である. そのため，自己相似図形は多解像度表示が可能な図形である.

3.4　高次元データの 2 次元化

3 次元データの 2 次元化可視化の既存手法について 1.4 節で述べた．本節では，より高次元なデータ (4D, 5D, 6D, ...) も 2 次元化できるように既存手法を拡張した可視化方法について述べる．拡張の際には，高次元データの要素数に着目し，2.2 節で述べた組み合わせ自己相似図形を利用する．

解像度 R，次元数 $n_2 (n_2 \geq 2)$ の高次元データは要素数 $2^{n_2 R}$ であるため，解像度 1 のとき要素数が 2^{n_2} であるような 2 次元自己相似図形を作成し，再帰的に分割・マッピングを行えばよい ($n_2 = 3$ のときが既存手法に対応する)．単純に展開先となる 2 次元正方形領域を縦横 2 分割にしていくと，解像度があがるたびに要素数が 4 倍になっていくため，正方形領域を持つ自己相似図形のうち解像度 1 のときの最小要素数を持つものは，解像度 1 のとき要素数 4 のものである．解像度 1 のとき要素数 4 のものを用いて組み合わせ自己相似図形を作成すれば，空の領域が存在せず使用領域が最も高くなるため，大規模なデータとなる可能性が高い高次元データのマッピングに最も利用しやすい．しかし，解像度 1 のとき要素数 4 のものを用いて作成される組み合わせ自己相似図形は素因数に 4 を持つ要素数 4^N(N は 0 以上の整数) のもののみである．いま低次元化の対象としている高次元データは要素数 2^{n_2}(素因数 2) のため，このような組み合わせ自己相似図形だけでは展開が不可能である．そこで，解像度 1 のとき要素数 8 の自己相似図形も

利用して組み合わせ自己相似図形を作成する．そうして作られる自己相似図形は，$4^N \cdot 8^M$(M,N は 0 以上の整数) である．高次元データの要素数 $2^{n_2 R}$ は，次元数が奇数・偶数の場合で，式 3.4.1 のように変形できる．式 3.4.1 より，解像度 1 のとき要素数 4, 8 の自己相似図形を利用して組み合わせ自己相似図形を作成すれば，高次元データを展開することが可能なことが分かる．偶数次元の場合は，正方形を縦横方向に繰り返し 2 分割を行ってできる自己相似図形の解像度 k のもの，奇数次元の場合は，解像度 1 のとき 8 要素を持つ 2 次元自己相似図形の各要素を縦横ともに $(k-1)$ 回分割したものを利用すればよい．8 要素を持つ自己相似図形として，解像度 1 のシェルピンスキーのカーペットがあげられる．Fig.3.3 に奇数・偶数のときに利用する 2 次元図形を示し，Fig.3.4, 3.5 にボリュームデータと時系列ボリュームデータのマッピングの例を載せる．

$$2^{n_2 R} = \begin{cases} 4^{kR} & (n_2 = 2k) \\ (8 \cdot 4^{k-1})^R & (n_2 = 2k+1) \end{cases} \tag{3.4.1}$$

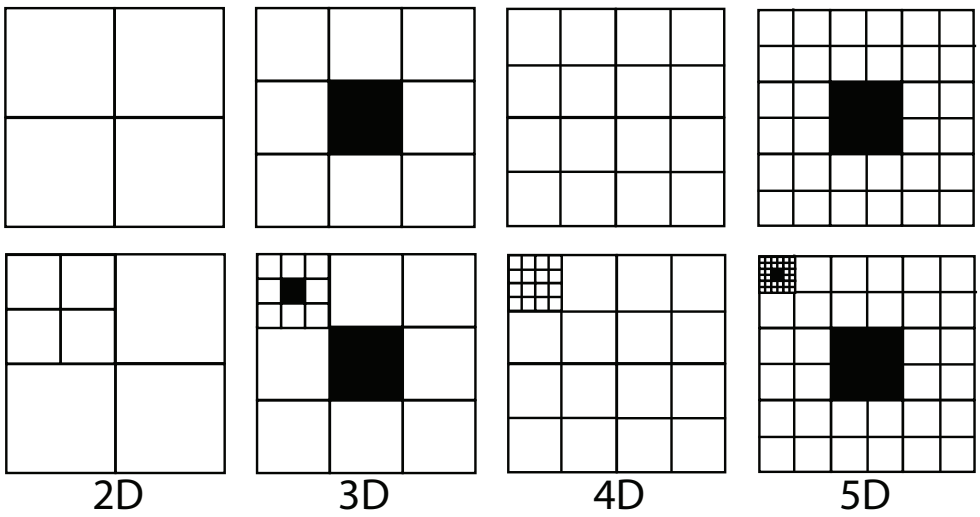


Fig. 3.3: Self-similar objects for 2D, 3D, 4D, 5D. Each object has 4, 8, 16, 32 elements for resolution 1.

3.5　高次元データの 1 次元化

解像度が R のとき，次元数が $n_1 (n \geq 1)$ の高次元データは要素数 $2^{n_1 R}$ であるため，解像度が 1 のとき要素数が 2^{n_1} であるような 1 次元自己相似図形を作成し，再帰的な分割ルールとマッピングルールによって高次元データを 1 次元図形上にマッピングすればよい．1 次元図形は，単純に 2 分割を再帰的に行うと，解像度 n_1 を持つときに要素数が 2^{n_1} となる．そこで，この要素数 2^{n_1} の 1 次元図形と次元数 n_1 のデータの間に解像度 1 のマッピングルールを決めて再帰的に分割することによって，高次データの 1 次元化を行うことができる．Fig.3.6 に 3 次元データを 1 次元化する際のマッピングルールの例を挙げる．

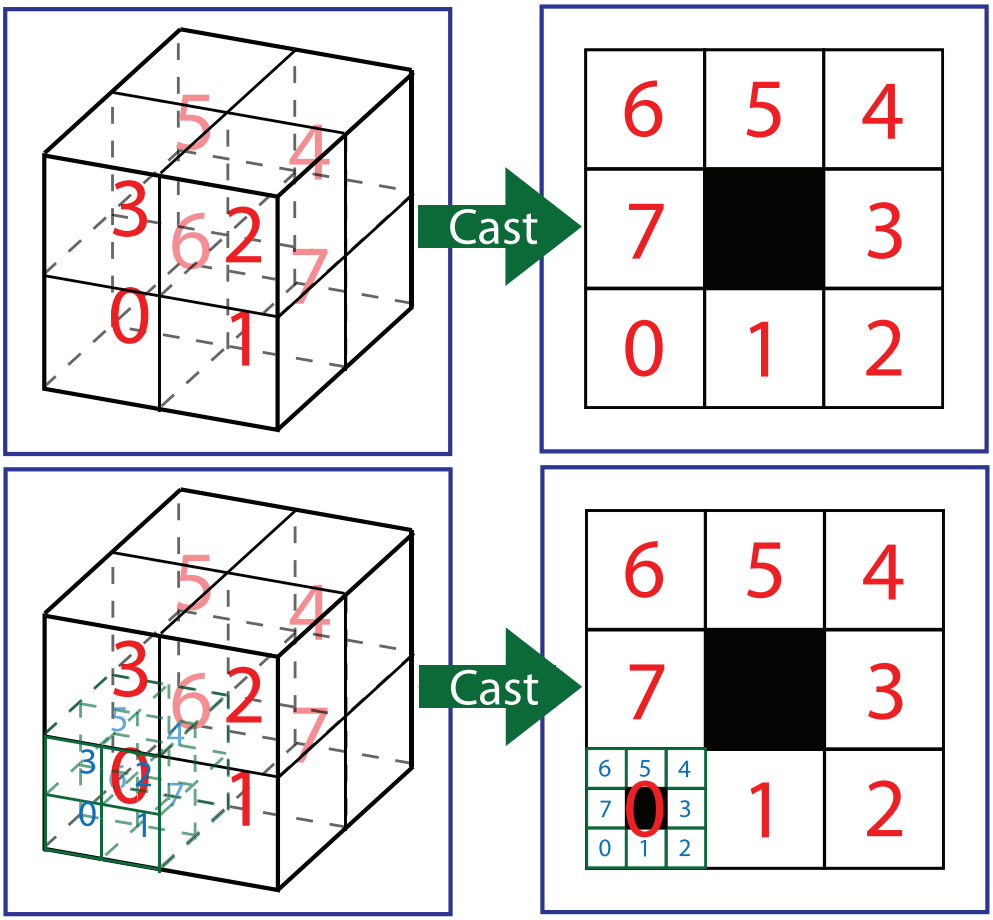


Fig. 3.4: A mapping rule example of dimensional reduction for high-dimensional data (3D to 2D). And a multi-resolutional mapping example.

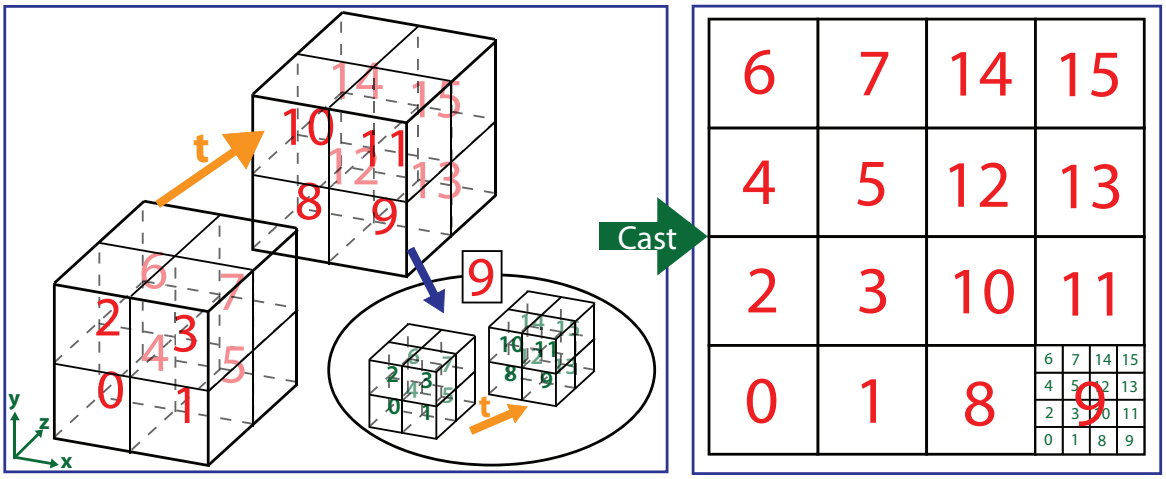


Fig. 3.5: A mapping rule example of dimensional reduction for high-dimensional data (time-varying 3D to 2D). And a multi-resolutional mapping example.

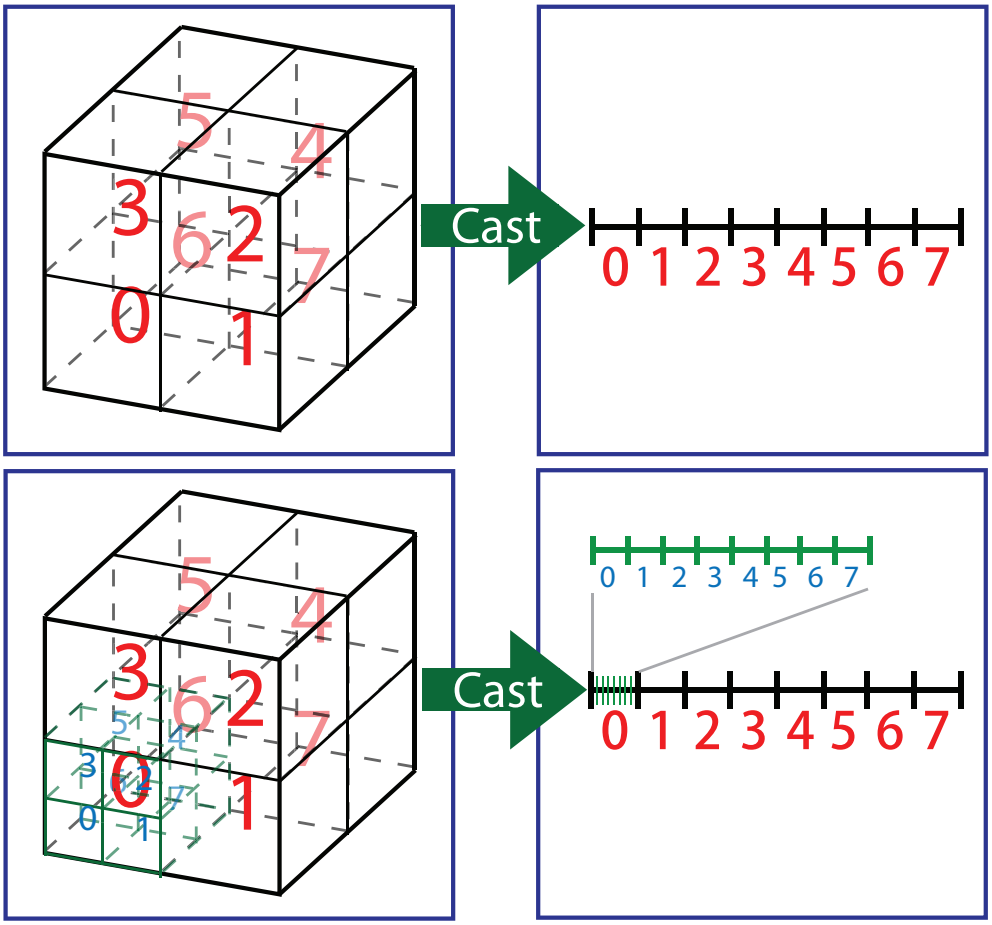


Fig. 3.6: A mapping rule example of dimensional reduction for high-dimensional data (3D to 1D). And a multi-resolutional mapping example.

3.6　高次元データの 3 次元化

解像度 R，次元数 $n_3(n_3 \geq 3)$ の高次元データの要素数は $2^{n_3 R}$ であり，解像度 1 のとき要素数が 2^{n_3} であるような 3 次元図形を作成し，再帰的に分割・マッピングを行えばよいが，2 次元化のときと同様に場合分けが必要となってくる．$n_3 = 3k, 3k+1, 3k+2(k$ は 1 以上の整数) のそれぞれの場合によって，式 3.6.1 のように表すことができる．式 3.6.1 より，$n = 3k$ のときは，解像度 k の単純に縦横高さに 2 分割を繰り返した立方体，$n = 3k+1$ のときは，解像度 1 のとき 16 要素を持つ 3 次元自己相似図形の各要素を縦横高さ共に $(k-1)$ 回分割した 3 次元図形，$n = 3k+2$ のときは解像度 1 のとき 32 要素を持つ 3 次元自己相似図形の各要素を縦横高さ共に $(k-1)$ 回分割した 3 次元図形を利用すればよい．解像度 1 のとき 8 要素，16 要素，32 要素を持つ 3 次元自己相似図形と一部分だけ解像度を挙げた例を Fig.3.7 に示す．要素数の一致する図形が得られたので，解像度 1 のときの高次元データから 3 次元図形へのマッピングルールを定めれば，3 次元化が可能となる．

$$2^{n_3 R} = \begin{cases} 8^{kR} & (n_3 = 3k) \\ (16 \cdot 8^{k-1})^R & (n_3 = 3k+1) \\ (32 \cdot 8^{k-1})^R & (n_3 = 3k+2) \end{cases} \tag{3.6.1}$$

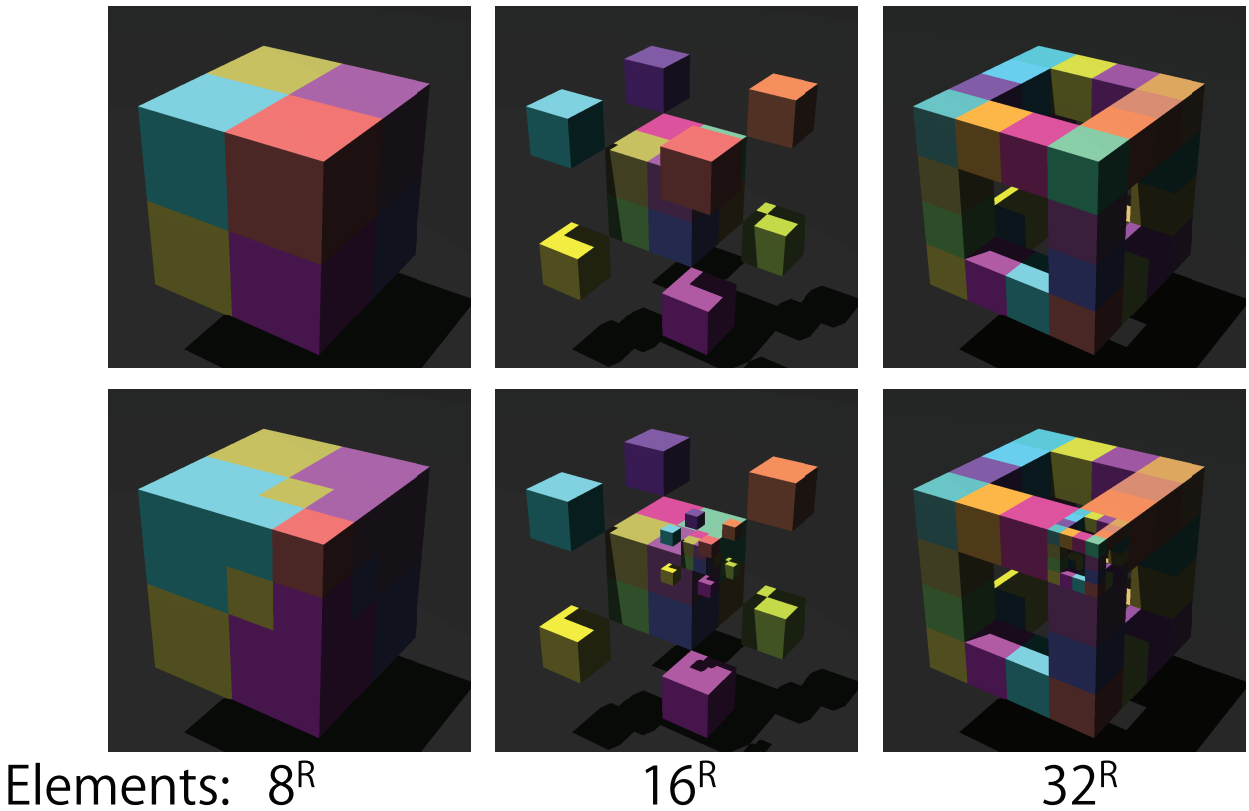


Fig. 3.7: 3D self-similar objects

3.7　高次元データの 0 次元的扱い

高次元のデータを低次元化すると 1, 2, 3 次元の結果像を得ることができる．この結果像を空間上に配置できる glyph[33] として扱うことによって，あたかも高次元データを点 (0 次元図形) と同様に扱うことができる．その概略図を Fig.3.8 に載せる．このように 0 次元的に扱うことで，ある値の大小に応じて多数の高次元データの glyph を並べることや，VaR display[33] のように多次元尺度構成法などを用いて値の分布似ているデータは近くに集まるように並べることができるようになる，しかし，仮想的に点として扱っているだけであり，点として表示する数が増えると，1 glyph あたりに使えるスペースが減り，重なりが増える，glyph のサイズが小さくなるなどといったことが起こる．また，通常の点表示と違い，隣接する点の値の比較が容易でないなどの問題がある．

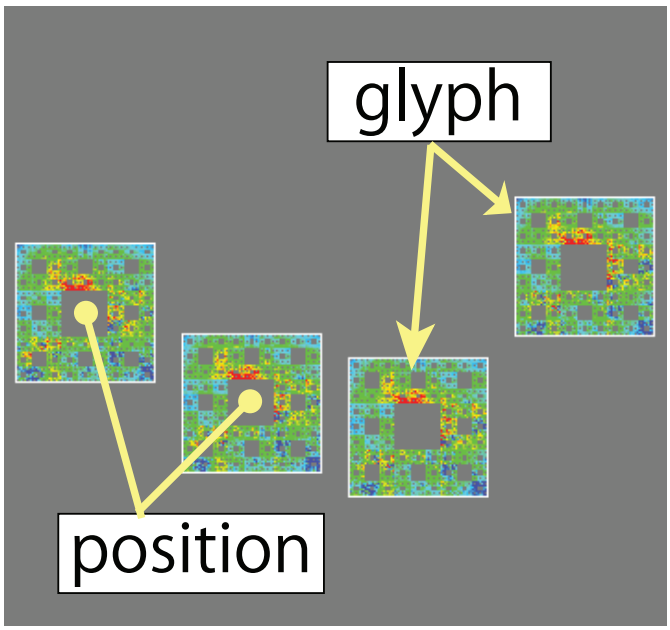


Fig. 3.8: Summary of glyph concept visualization

3.8　値の表示方法

今までの節で高次元データの各値が 2 次元上で表示される位置を決めることができるようになった．しかし，可視化を行うには各値の表現方法を決めなければならない．その表示方法としては，文字を利用する，色を利用する，イラストを利用するなど様々な方法が考えられるが，色を利用して表示する方法の一例を述べる．値を色で表現するには，データの値から色を表す値へ変換を行う．一般にこのデータの値から色へ変換する際に利用する関数は伝達関数と呼ばれ，ボリュームレンダリングやその他可視化分野で広く研究されている [13]．まず，データの値を 0 から 1 の値 h に次の式で変換する．ただし，データの値を v とし，また 2 つの閾値 $v_{max}, v_{min}(v_{max} \geq v_{min})$ を定めている．

$$h = \begin{cases} 1 & (v \geq v_{max}) \\ \frac{v - v_{min}}{v_{max} - v_{min}} & (v_{min} \leq v < v_{max}) \\ 0 & (v < v_{min}) \end{cases} \tag{3.8.1}$$

さらに，Fig.3.9 のように h の値と色相を対応させ，データの値 v を色情報 RGB 値に変換する．

Fig. 3.9: h and Color. When $h = 0$, color is blue. When $h = 1$, color is red.

2 次元上で色を利用して値を表現するには，RGB 値を持つ色情報で十分であるが，3 次元上では，さら

に透明度 $\alpha(0 \leq \alpha \leq 1)$ を定め，遮蔽を低減させ，ある程度内部の方も透視させて表示できるようにするのが望ましい．α 値の高いところほど明瞭に表示されるため，α 値が高いところが強調された可視化結果になる．そのため，この α 値を決定する関数を定めるのも重要なことである．ここでは，簡単な変換例として，以下に h から α を求める関数の一例を示す．$a(>0)$ は α 値を左右する定数である．

$$\alpha = \begin{cases} 1 & (h \geq 1/a) \\ ah & (h \leq 1/a) \end{cases} \tag{3.8.2}$$

この関数に基づいて決められる α 値はデータの値の大きいところ (赤く表示されるところ) ほど大きくなるので，データの値の大きいところが明瞭に表示されるようになる．逆に値が 0 のところは，まったく表示されない．

ここでは，簡易な伝達関数の例を挙げたが，伝達関数により可視化結果が大きく変わるため，可視化対象に応じて適切に伝達関数を決定することが大切である．しかし，この論文では，伝達関数の決定方法は他の論文に譲り，これ以上詳細には述べない．

3.9　多解像度化

高次元データを低次元化することで 1, 2, 3 次元結果像を得ることができるようになった．この結果は自己相似図形上で表され，Fig.3.4, 3.5, 3.6, 3.7 からも分かるように，多解像度化して表示することが可能である．多解像度化により，描画命令数減少による描画負荷の低減，重要箇所だけ高解像度化してできる要約的な可視化結果による認知負荷の低減といったことがもたらされる．多解像度化をする際には，様々なルールを選択可能である．１つ目は，何を基準として多解像度化するかである．例えば，同階層領域に存在する値の分散値・最大値などである．２つ目は，多解像度化する際に利用するアルゴリズムである．例えば，ボリュームデータを多解像度化する際には Octree 構造を作成することが多いが，Octree 構造の作り方は多数存在する．ここでは，値の分散に基づくトップダウン型の木構造構築・走査による多解像度化の方法を一つ紹介する．

まず，高次元データのある領域における値の変動係数を以下のように定める．今，ある領域 S には，N 個の要素が含まれているとする．またある領域 S に含まれる i 番目のデータポイントの値は v_i と書かれることとする．ここでデータポイントとは，高次元データを構成する最小の領域群の 1 つを指すために用いている．ある領域 S に含まれる N 個の要素の平均値 m, 標準偏差 s, 変動係数 c は以下の式で計算できる．

$$m = \frac{\sum_{i \in S} v_i}{N} \tag{3.9.1}$$

$$s = \sqrt{\frac{\sum_{i \in S} {v_i}^2}{N} - m^2} \tag{3.9.2}$$

$$c = s/m \tag{3.9.3}$$

この平均値 m や変動係数 c を保持するノードから成る木構造を作成する．最も木の上部に存在するノード (ルートノード) は，高次元データの全要素を含む領域の平均値と変動係数を保持し，その子ノー

ドは n 次元データを各次元方向に 2 分割してできた 2^n 領域のうちの 1 領域の平均値と変動係数を保持する．さらにその子ノードは同様に現在のノードを各次元方向に 2 分割してできた領域のうち 1 つ領域の平均値と変動係数を保持させる，といったルールを再帰的に適用する．ただし，領域内に 1 要素しか含まれないようになった場合には，再帰的なルールの適用を止める．こうしてできた木構造は Fig.3.10 のようになる．ここでは図を簡単にするため，2 次元データの場合の木構造 (Quad tree) を載せている．

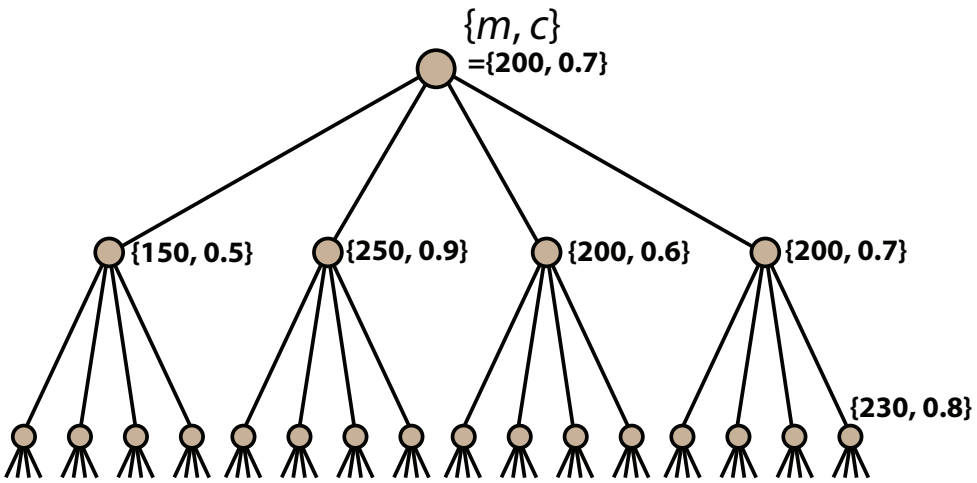


Fig. 3.10: An example of quad tree structure. Each node has average value and coefficient of variation. m value means average value and c value means coefficient of variation.

次に多解像度化する際に用いる閾値を定め，この閾値をもとに木構造をトップダウン型で走査する．その際には，以下の再帰的なルールを利用する．

1. ルートノードがら走査を始める．
2. 現在いるノードが閾値以上の変動係数を持つ場合は，子ノードの方へ移動する．そうでない場合は現在いるノードで止まる．
3. 2 のルールを再帰的に適用する．

例えば，閾値を 0.65 にした場合，Fig.3.11 のように走査される．このルールに基づき走査し，走査が止まったところの各ノードをその階層に応じた大きさで描画することで，多解像度可視化を実現する．こうしてできた可視化像は，値の散らばりが大きいところは高解像度で，散らばりが小さいところは低解像度で表示された要約的な可視化が行われている．例として，多解像度化を行わない場合の結果像と，多解像度化を行った場合の結果像を Fig.3.12 に載せる．また，Fig.3.12(a) から (d) における閾値と表示されている領域数を Table.3.1 に載せる．領域数がおよそ 4 分の 1 になるように多解像度化されていても，データのおおよその様子は変わらず表示されており，要約的に可視化されていることが確認できる．

3.10　可視化結果のインターフェースとしての利用

高次元データと低次元化可視化結果像の間には 1 対 1 の写像関係があるため，低次元化された結果像上で位置を指定することによって，もとの高次元データの位置を指定することができる．また，低次元可

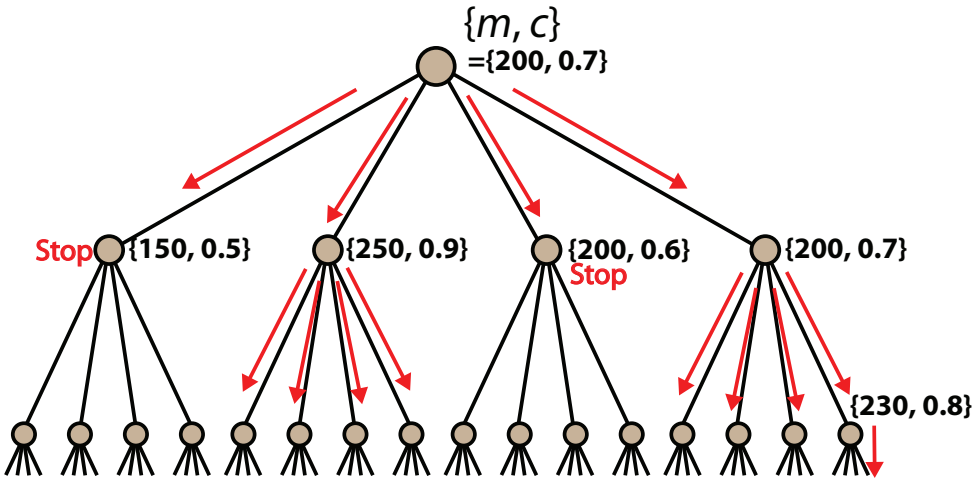


Fig. 3.11: An example of tree traverse

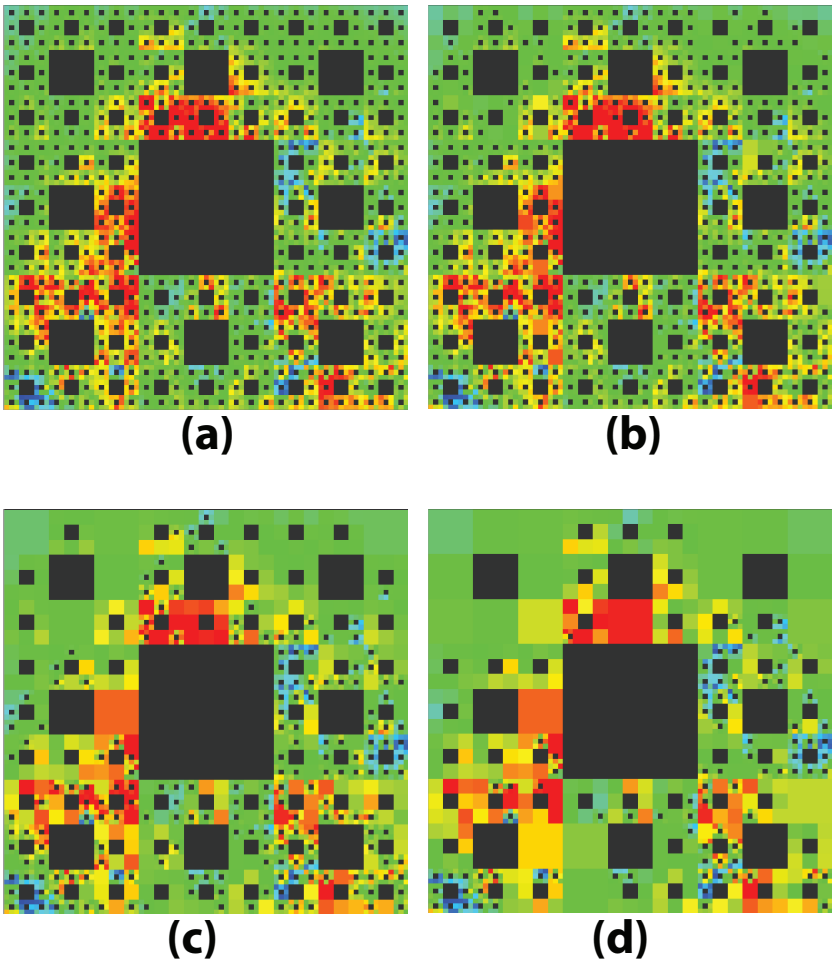


Fig. 3.12: Comparation of Multi-resolutional result.

Table. 3.1: Threshold and Number of Regions of Fig.3.12

	(a)	(b)	(c)	(d)
Threshold	0.00	0.05	0.10	0.15
Regions	4096	3634	1821	974

視化像にはもとデータの値が表現されており，データの値をもとにした関数操作など (例えば，指定した値を全体から減算するなど) も行うことができる．さらに，これら位置指定や値に基づく関数操作は他の領域の値も参考にしながら行うことができる．例えば，岩丸らの既存研究では，2 次元上に表示された値を指定し，その値の大きさをすべての領域から減算することで，3 次元結果・2 次元結果ともに指定領域より値の高い部分が分かりやすく可視化できた (1 章 Fig.1.20 参照)．このように低次元化結果像を高次元データを操作するインターフェースとして利用することが可能である．

3.11　低次元化手法まとめ

本章では，任意の高次元データを低次元化し，1, 2, 3 次元画像として表示する方法について説明した．また低次元化の結果像を多解像度化する方法や，可視化インターフェースとしての利用可能性についても説明した．低次元化により，4 次元以上のデータを可視化すること，空間的遮蔽や時間的遮蔽を遮蔽のない形で可視化することが可能になった．

第 4 章

低次元データの高次元化手法

4.1　高次元化の目的

低次元データを高次元化することによって，要素数の多い次元データを効率よくディスプレイ内に敷き詰めることができるようになる．例えば，大規模データを２次元状に並べて表示するような研究は情報的可視化方法の 1 分野として数多く研究されている [14], [15], [17], [33].

4.2　高次元化手法の概要

本高次元化手法では，1, 2 次元のデータを高次元化し，2, 3 次元可視化像を得る多解像度化可能な高次元可視化手法について述べる．その際には，低解像度を持つ低次元データとより高次元（2D, 3D）の自己相似図形の要素数を一致させ，低次元データの各要素の自己相似図形上へのマッピングルールを作成し，再帰的にマッピングルールを適用することで高次元化を実現する．Fig.4.1 に概要図を載せる．また高次元化された結果像を低次元化結果と同様に glyph として扱ったり，低次元データを操作するためのインターフェースとしても利用できるといったことも付記しておく．

4.3　1 次元データの 2 次元化

解像度が R のとき，1 次元データの要素数は 2^R である．一方，2 次元図形のうち解像度 1 のときの要素数が最小な自己相似図形は要素数 4 のものである．そのため，1 次元データの 4 要素を最小単位として，2 次元化する必要がある．Fig.4.2 のような Morton の並びに 4 要素を並べれば，Morton の並び自身が自己相似な形となっているため，1 次元データの多解像度化可能な 2 次元化可視化が実現する．Fig.4.3 に，1 次元から 2 次元画像へのマッピングルールの例を載せている．ただし，ここでは Morton の並びを例としてあげたが，Morton 並び以外のマッピングルールであっても高次元化は可能である．

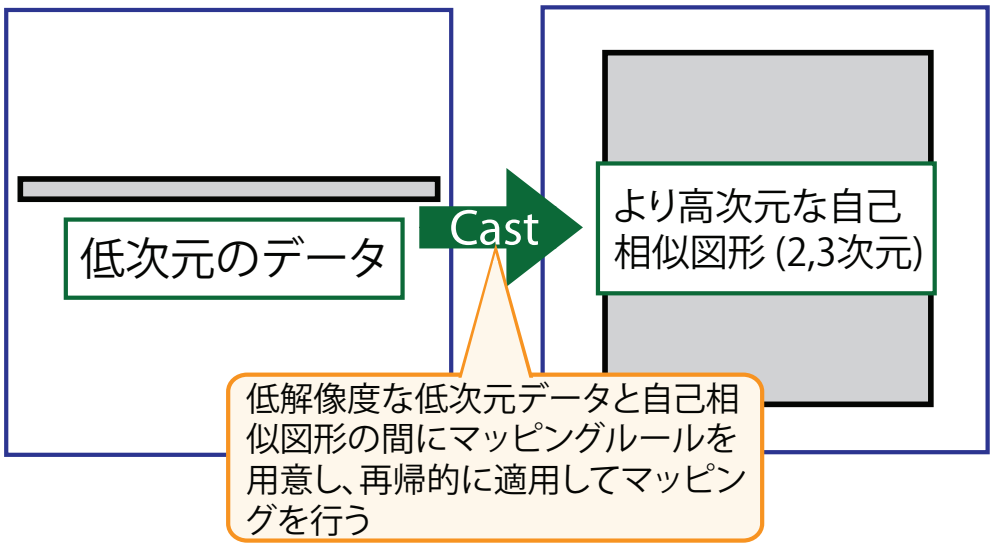


Fig. 4.1: Summary of the dimensional increase method

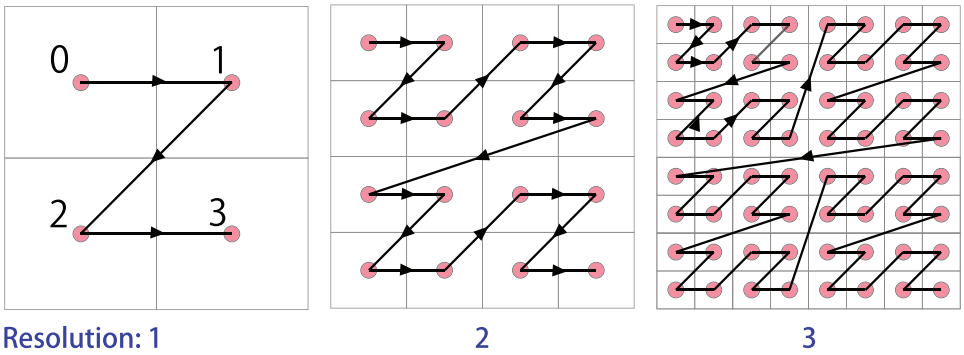


Fig. 4.2: Morton orders with resolution 1, 2 and 3.

4.4　1 次元データの 3 次元化

3 次元図形のうち立方体を各要素の形として持ち，解像度 1 のときの要素数が最小な自己相似図形は要素数 8 であるため，1 次元データの 8 要素を最小単位として，3 次元化すればよい．3 次元での Morton の並び（Fig. 4.4）上に 8 要素を並べ，このマッピングを再帰的に適用することで，1 次元データの多解像度化可能な 3 次元可視化が実現する．ここでも 2 次元化の場合と同様に，Morton の並び以外のマッピング方法も考えられる．

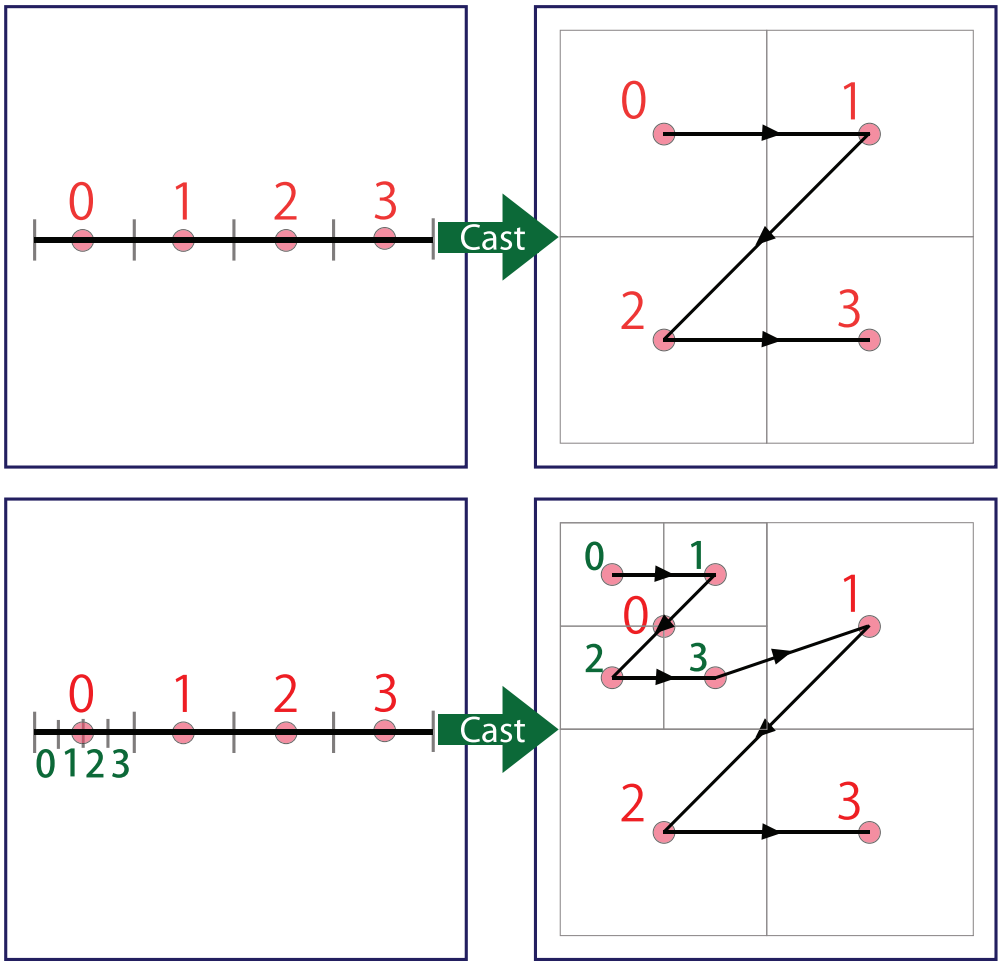


Fig. 4.3: A mapping rule example of dimensional increase(1D to 2D). And a multi-resolutional mapping example.

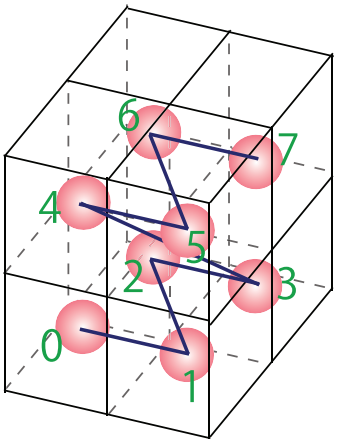


Fig. 4.4: 3D Morton order.

4.5　2 次元データの 3 次元化

解像度が R のとき，2 次元データは要素数 4^R である．そのため，2 次元データの要素数 64 を最小単位として，解像度 2 の単純分割立方体上に写像する (Fig.4.5 上) か，2 次元データの要素数 16 を最小単位として，Fig.3.7 に挙げた 16 要素を基本要素数とする 3 次元フラクタル図形上に写像する (Fig.4.5 下) ことで 3 次元化が行える．

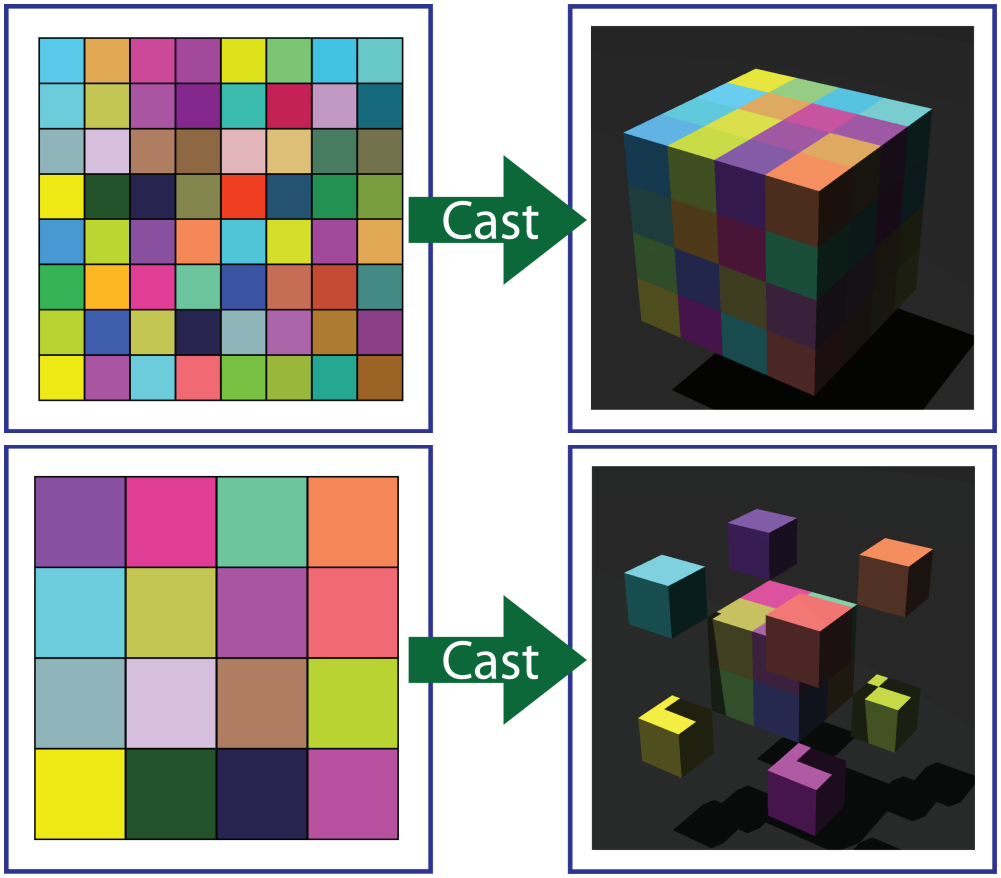


Fig. 4.5: A mapping rule example (2D to 3D). Top: 2D data with resolution 3 (64 elements) to 3D self-similar object (64 elements). Bottom: 2D data with resolution 2 (16 elements) to 3D self-similar object (16 elements). The color areas in 2D are mapped onto the same color areas in 3D

4.6　多解像度化

高次元化の場合も低次元化のときと同様に可視化像を多解像度化することができる．多解像度化の際には，低次元化のときと同様にはじめに平均値と変動係数を持つノードから構成される木構造を作成する．ただし，低次元化の場合と異なることとして，n 次元の低次元データの解像度を上げる際に分割する単位数が 2^n でないことが挙げられる．そのため，再帰的に領域を分割して変動係数を求める際に，各次元方

向に 2 分割ずつするのではなく，1 次元データを 2 次元化する際には 4 分割，1 次元データを 3 次元化する際には 8 分割，2 次元データを 3 次元化する際には，4 分割または 8 分割することになる．多次元化した場合の結果は，Fig.4.3 の下図のようになる．

4.7　高次元化手法のまとめ

本章では，1, 2 次元データを高次元化し，2, 3 次元画像として表示する方法，およびその表示画像の多解像度化の方法について説明した．高次元化により大規模データなどを画面使用率の高い状態での表示や，多解像度と組み合わせることで要約的な表示が可能になった．前章の低次元化と本章の高次元化により任意次元のデータが 1, 2, 3 次元画像として表示できるようになった．

第 5 章

次元別の次元変換手法

5.1　次元別の次元変換の目的

3, 4 章では，任意次元データのすべての次元を同様に扱って次元の変換を行ったが，ある複数の次元を持つデータを次元の種類別に低次元化・高次元化したいような場合がある．例えば，時系列ボリュームデータは，空間の次元 (3 次元) と時間の次元 (1 次元) を合わせ持つ 4 次元時空間データであり，4 次元すべてを同様に扱い次元変換を行うと，可視化結果から時間変化や空間変化を別々に把握することが困難となる，空間と時間が同等の解像度を持つ必要があるなどといった短所が存在する (7 章の 7.1.3 節に可視化例を挙げている)．そのため，時系列ボリュームデータの場合は空間次元を低次元化し，時間次元をそのままの 1 次元，もしくは高次元化して空間と時間を別の軸で表現したような可視化が適切であり，そのような可視化が可能な手法が必要である．このように次元の種類別に低次元化・高次元化することが有効であるデータが存在し，そうしたデータに対して次元別の次元変換を行うことで，データの理解を促すことができるようになる．

5.2　次元別の次元変換の概要

ある次元を持つデータの次元変換を行った結果を可視化するには，次元変換後の次元数が 3 以下でなければならない．そのため，次元別に次元変換を行うためには，次元別に低次元か高次元化を行い，次元別に次元変換を行った合計の次元数が 3 以下になるようにして，各次元方向にデータ値を並べることで次元別の次元変換が行える．Fig.5.1 に次元別の次元変換の概要図を載せる．Fig.5.1 では，3 次元を持つ次元群 1 と 2 次元を持つ次元群 2 の合計 5 次元から構成されるデータを次元群 1 を 2 次元化，次元群 2 を 1 次元化して，3 次元の次元別次元変換の結果像を得る様子を載せている．

5.3　次元別の次元変換の方法

本説では，一般的な次元の場合について述べる前に，4 次元データの場合の例を挙げて，次元別の次元変換について述べる．次元別に次元変換を行うことが多いと考えられる 4 次元データ例としては，時系列ボリュームデータなどが考えられる．

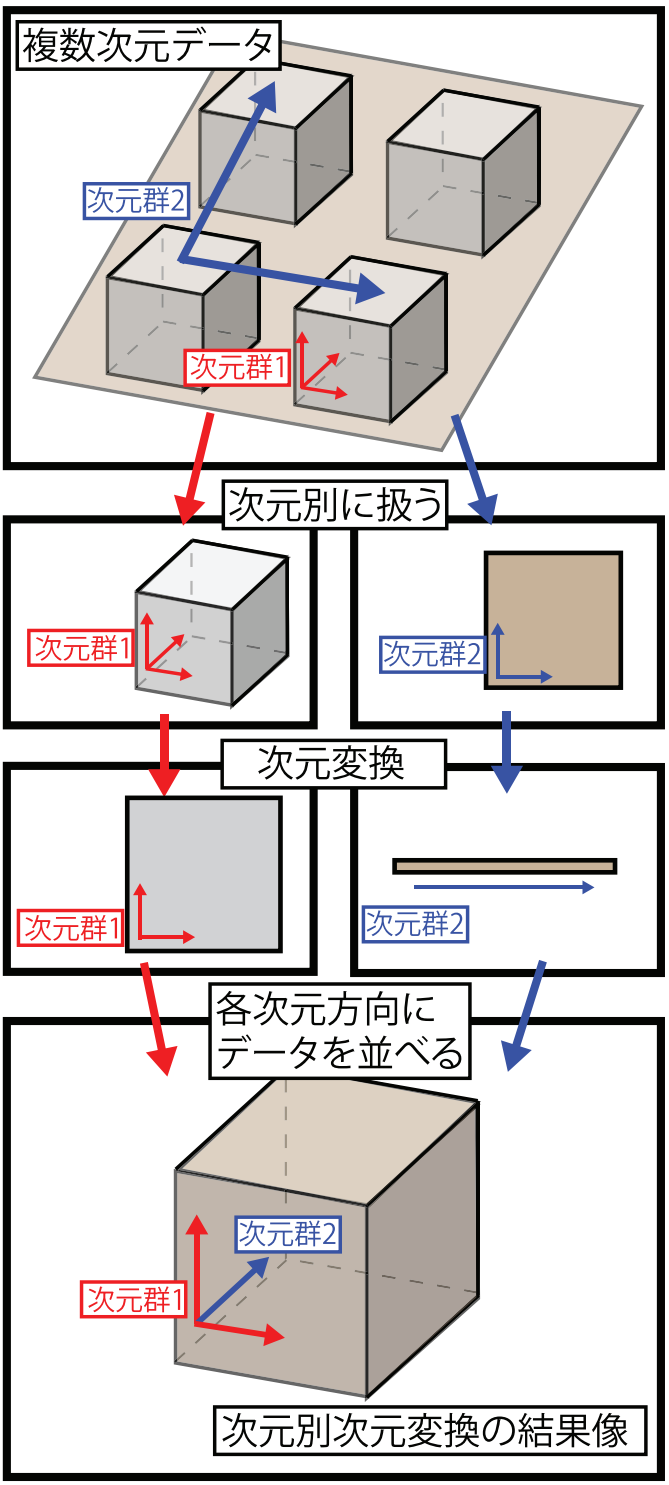


Fig. 5.1: Summary of dimension transfer of the multi-type dimensional data.

はじめに扱う次元の種類とその種類に所属する次元数を表す方法として，次元種類が A と表され，その次元種類が持つ次元数が n のとき，$A(n)$ という記号を用いることとする．さらに，ある複数次元データ M が $A(a),B(b),C(c),\cdots$ で構成されている場合，$M=A(a)+B(b)+C(c)+\cdots$ と書いて良いこととする．例えば，時系列ボリュームデータを空間次元 S，時間次元 T からなるデータとして扱う場合には，時系列ボリュームデータ V は $V=S(3)+T(1)$ と表すことができる．一般的な 4 次元データ D_4 は $D_4=A(4),A(3)+B(1),A(2)+B(2),A(2)+B(1)+C(1),A(1)+B(1)+C(1)+D(1)$ といった 5 種類の異なる次元群の組み合わせが考えられる (さら次元種類に含まれる次元数の選び方も複数通りかある．例えば，$A(3)+B(1)$ の場合は，${}_4C_1=4$ より 4 通りの次元種類を作るときの次元の組み合わせ方が考えられる)．これら 5 つの場合について，各次元群の低次元化・高次元化を行い，3 次元以下の可視化像を得なければならない．例えば，$A(3)+B(1)$ の場合には，1. A 次元群を 2 次元化し，B 次元群方向に並べ 3 次元化，2. A 次元群を 1 次元化し，B 次元群方向に並べ 2 次元化，3. A 次元群を 1 次元化し，B 次元群を 2 次元化し 3 次元状に可視化するといったことが考えられる (Fig.5.2)．さらに，glyph の考えを利用すれば，次元群を 0 次元的に扱うことができるようなるため，A 次元群を glyph 化 (0 次元化) し，B 次元群方向に並べて 1 次元表示，B 次元群を 2 次元化，3 次元化して 2 次元表示，3 次元表示などといったことができるようになる

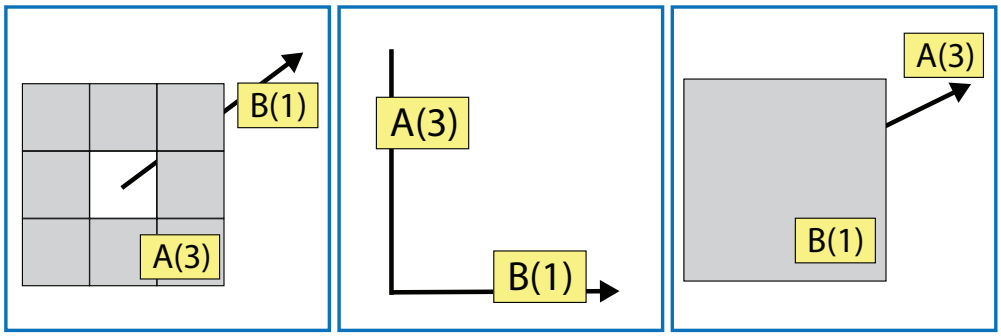


Fig. 5.2: Image of dimensional reduction and increase visualization for the data with A(3) and B(1).

一般的な n 次元データ D_n の場合についても，4 次元データの場合と同様に各次元群を次元変換し，3 次元以下の次元を持つ可視化像として表示すればよい．glyph の考えを利用しない場合について考えると，各次元群はいくら低次元化しても 1 次元以下にはならないため，$D_n=A(n),A(a)+B(n-a),A(a)+B(b)+C(n-a-b)$ の 3 種類の次元群までであれば，3 次元以下の可視化像をえることができる．つまり，glyph を用いない場合は，n 次元を 3 種類の次元にまでしか別にして扱うことはできないということである．glyph を利用すれば更なる次元種類を扱えるようになるが，実際には 0 次元でないものを 0 次元的に扱っているため，glyph を含めた次元別次元変換の多解像度化などが複雑化するため，以後では glyph を利用しない場合のみを考える．

5.4　次元別次元変換における多解像度化

次元別に次元変換をした場合の多解像度化は単純にすべての次元を同等に低次元化や高次元化を行った場合の多解像度化と比べ複雑になる．単純に次元別に 3.9 節で述べた方法で多解像度化を行い，多解像度可視化像を得ようとすると同一領域でも次元種類によって扱うべき解像度が異なる場合があるためうまくいかないことがある．Fig.5.3 は次元別に次元変換をしたあとに次元別に多解像度化しようとした場合の例である．ここでは，2 種類の次元群をそれぞれ 1 次元化したもので，各次元ともに解像度 R のとき領域数 2^R を持ち，各次元共に多解像度化をほどこさない場合解像度 2 を持つ場合の図を考えている．(a) 図はすべての横方向のデータポイントで縦方向の次元に基づき多解像度化を行った場合 (縦方向だけを考慮して多解像度化を行った場合)，(b) 図はすべての縦方向のデータポイントで横方向の次元に基づき多解像度化を行った場合 (横方向だけを考慮して多解像度化を行った場合) である．(a), (b) 図のように各次元別にその次元方向の変化だけを考慮に入れた場合は 3.9 節で述べた方法でも多解像度化することができる．しかし，各領域の解像度を (a), (b) で多解像度化されているどちらの状態も満たすように可視化像を作ることはできない．(a), (b) どちらの状態も満たすことができる箇所は (c) 図で色付けされた箇所だけである．このように 3.9 で述べた方法のままでは次元別の次元変換における多解像度化が実現できないため，次元別の次元変換のための多解像度化手法が必要である．

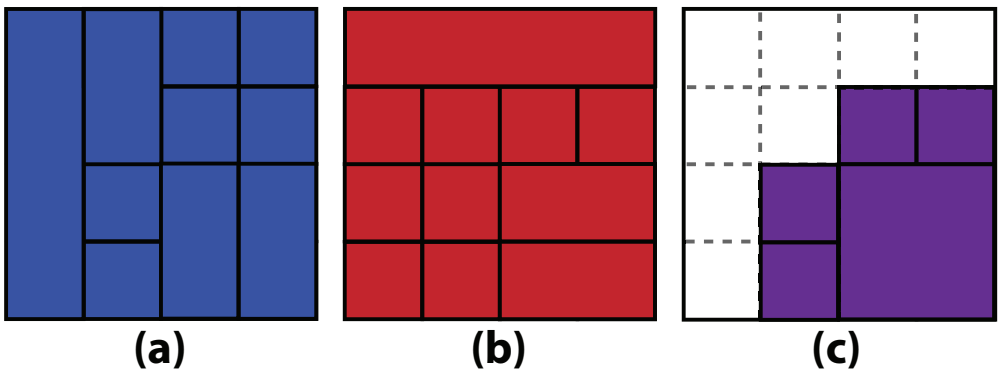


Fig. 5.3: A problem of multi-resolution in multi-type dimensional data

次元別の次元変換をした場合の多解像度化の既存研究としては，時系列ボリュームデータにおけるレンダリングのための木構造構築が挙げられる [26], [4]．これらの木構造は大規模なデータである時系列ボリュームデータのレンダリング速度をあげるために用いられるため，レンダリングの際にレンダリング領域数ができるだけ少なくなるように，前時刻の結果をできるだけ再利用できるような木構造を構築できるように研究されている．しかし，これら手法では，空間を表す木構造の各ノードがその空間領域における時間変化を表す木構造の親ノードへのポインターになっているものや (Time-Space Partitioning(TSP) 木 [26])，またはその逆になっている場合が多く (Space-Partitioning Time(SPT) 木 [4])，走査の際に先に移動する木構造が決まっている．例えば，TSP 木であると先に空間構造を表す木から走査をはじめ，空間を表す木の各ノードにおいて時間構造を表す木を走査する．また，SPT 木の場合はその逆である．このような木構造を構築して走査を行うと，同じ多解像度化のための閾値を用いていても，TSP 木を用いるか

SPT 木を用いるか (空間・時間どちらの構造を表す木から走査するか) によって多解像度化された結果が変わってくるため，次元によって多解像度化の際の扱いに非平等性が生じる．多解像度化における非平等性が存在すると，多解像度化に使う閾値の他に次元別に多解像度化の優先度を決定しなければならない．さらに，次元別に優先度が決められていると，優先度が高いものが低解像度化された場合，閾値を変化させても優先度が低いものは高解像度化されないといった問題が出てくる．そのため，多解像度化した可視化結果がすべての次元種類を考慮した要約的なものであるとは言い難い．また，これらの手法は次元種類が 2 つまでの場合までしか扱っていない．そこで，本節では，すべての次元種類に対し平等性があり，3 種類以上の次元がある場合でも扱うことのできる多解像度可視化開発手法を説明する．開発手法では，木構造をデータの値や閾値に応じて構成する．また，glyph を利用しない場合のみについて述べる．

5.4.1　2 種類の次元群を扱う場合の多解像度化

はじめに 2 種類の次元群 D_1, D_2 を扱う場合について述べる．D_1, D_2 におけるデータポイント d_1, d_2 がそれぞれ D_1, D_2 内の領域 $\mathscr{D}_1$, $\mathscr{D}_2$ に含まれている範囲における D_1, D_2 のエラー Err_{D_1}, Err_{D_2} は次の式 5.4.1 から 5.4.8 を用いて計算できる．D_1, D_2 のエラーとはそれぞれ D_1, D_2 方向の値の散らばり度合いを表す数値で 0 から 1 の間の実数で表される．

$$m_{D_1}(d_2) = \frac{\sum_{d_1 \in \mathscr{D}_1} v_{d_1,d_2}}{n(\mathscr{D}_1)} \tag{5.4.1}$$

$$s_{D_1}(d_2) = \sqrt{\frac{\sum_{d_1 \in \mathscr{D}_1} v_{d_1,d_2}^2}{n(\mathscr{D}_1)} - (m_{D_1}(d_2))^2} \tag{5.4.2}$$

$$c_{D_1}(d_2) = \frac{s_{D_1}(d_2)}{m_{D_1}(d_2)} \tag{5.4.3}$$

$$Err_{D_1} = \frac{\sum_{d_2 \in \mathscr{D}_2} c_{D_1}(d_2)}{n(\mathscr{D}_2)} \tag{5.4.4}$$

$$m_{D_2}(d_1) = \frac{\sum_{d_2 \in \mathscr{D}_2} v_{d_1,d_2}}{n(\mathscr{D}_2)} \tag{5.4.5}$$

$$s_{D_2}(d_1) = \sqrt{\frac{\sum_{d_2 \in \mathscr{D}_2} v_{d_1,d_2}^2}{n(\mathscr{D}_2)} - (m_{D_2}(d_1))^2} \tag{5.4.6}$$

$$c_{D_2}(d_1) = \frac{s_{D_2}(d_1)}{m_{D_2}(d_1)} \tag{5.4.7}$$

$$Err_{D_2} = \frac{\sum_{d_1 \in \mathscr{D}_1} c_{D_2}(d_1)}{n(\mathscr{D}_1)} \tag{5.4.8}$$

v_{d_1,d_2} は d_1,d_2 で決まるデータポイントにおけるデータの値，$n(\mathscr{D}_1),n(\mathscr{D}_2)$ は領域 $\mathscr{D}_1,\mathscr{D}_2$ におけるデータポイントの総数，$m_{D_1}(d_2),m_{D_2}(d_1),s_{D_1}(d_2),s_{D_2}(d_1),c_{D_1}(d_2),c_{D_2}(d_1)$ は上記の式で計算される平均値，標準偏差，変動係数である．

また，領域 $\mathscr{D}_1,\mathscr{D}_2$ に含まれるデータの平均値 m は次の式で計算される．

$$m=\frac{\sum_{d_1\in\mathscr{D}_1}\sum_{d_2\in\mathscr{D}_2}v_{d_1,d_2}}{n(\mathscr{D}_1)n(\mathscr{D}_2)} \tag{5.4.9}$$

次にエラー Err_{D_1},Err_{D_2} に対する閾値 T_{D_1},T_{D_2} を決め，その閾値とエラーの大小に基づき木構造を作成する．すべてのノードは平均値 m とエラー Err_{D_1},Err_{D_2} とそのノードの平均値やエラーを計算したときの領域を表す $\mathscr{D}_1,\mathscr{D}_2$ を持つ．木構造作成には，以下に示す再帰的なルールを用いる．ある領域 $\mathscr{D}$ がデータポイント a から b までの範囲を示すとき，$\mathscr{D}[a,b]$ と書くこととする．今次元群 D_1,D_2 はそれぞれ $\mathscr{D}_1[0,a^{R_1}-1],\mathscr{D}_2[0,b^{R_2}-1]$ の範囲に含まれるデータを持つこととする (総データ数は，$a^{R_1}b^{R_2}$ 個)．ここで，a,b は解像度 1 のときのデータの要素数，R_1,R_2 はデータの持つ最大の解像度である．

「2 種類の次元群を多解像度化するための木構造作成ルール」

1. ルートノードは，$\mathscr{D}_1[0,a^{R_1}-1],\mathscr{D}_2[0,b^{R_2}-1]$ で表される領域の平均値 m とエラー Err_{D_1},Err_{D_2} を持ち，このノードからスタートし木構造を構築する．
2. 現在いるノードの持つエラー Err_{D_1},Err_{D_2} の値と閾値 T_{D_1},T_{D_2} を比較した結果 (a) から (d) に応じた処理を行う．現在のノードは領域 $\mathscr{D}_1[p,p+a^{r_1}-1],\mathscr{D}_2[q,q+b^{r_2}-1]$ を持つものとする．ここで p,q は 0 以上の整数，r_1,r_2 は現在のノードが持つ領域の D_1,D_2 における最大の解像度を表す．
   (a) $Err_{D_1}\geq T_{D_1}$ かつ $Err_{D_2}\geq T_{D_2}$ のとき，現在のノードの下に ab 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．ab 個の子ノードは．領域 $\mathscr{D}_1[p,p+a^{r_1}-1],\mathscr{D}_2[q,q+b^{r_2}-1]$ を D_1 方向に均等に a 分割と D_2 方向に均等に b 分割してできた各領域を持つ (それぞれ領域 $\mathscr{D}_1[p,p+a^{r_1-1}-1]\mathscr{D}_2[q,q+b^{r_2-1}-1],\mathscr{D}_1[p+a^{r_1-1},p+2a^{r_1-1}-1]\mathscr{D}_2[q,q+b^{r_2-1}-1],\mathscr{D}_1[p+2a^{r_1-1},p+3a^{r_1-1}-1]\mathscr{D}_2[q,q+b^{r_2-1}-1],\cdots,\mathscr{D}_1[p+(a-1)a^{r_1-1},p+a^{r_1}-1]\mathscr{D}_2[q,q+b^{r_2-1}-1],\mathscr{D}_1[p,p+a^{r_1-1}-1]\mathscr{D}_2[q+b^{r_2-1},q+2b^{r_2-1}-1],\mathscr{D}_1[p+a^{r_1-1}-1],p+2a^{r_1-1}-1]\mathscr{D}_2[q+b^{r_2-1},q+2b^{r_2-1}-1],\cdots,\mathscr{D}_1[p+(a-1)a^{r_1-1},p+a^{r_1}-1]\mathscr{D}_2[q+(b-1)b^{r_2-1},q+b^{r_2}-1]$ と表される)．また各ノードは領域の情報以外にその領域の平均値 m とエラー Err_{D_1},Err_{D_2} を持つ．
   (b) $Err_{D_1}\geq T_{D_1}$ かつ $Err_{D_2}<T_{D_2}$ のとき，現在のノードの下に a 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．a 個の子ノードは．領域 $\mathscr{D}_1[p,p+a^{r_1}-1],\mathscr{D}_2[q,q+b^{r_2}-1]$ を D_1 方向に均等に a 分割した各領域を持つ (それぞれ領域 $\mathscr{D}_1[p,p+a^{r_1-1}-1]\mathscr{D}_2[q,q+b^{r_2}-1],\mathscr{D}_1[p+a^{r_1-1},p+2a^{r_1-1}-1]\mathscr{D}_2[q,q+b^{r_2}-1],\mathscr{D}_1[p+2a^{r_1-1},p+3a^{r_1-1}-1]\mathscr{D}_2[q,q+b^{r_2}-1],\cdots,\mathscr{D}_1[p+(a-1)a^{r_1-1},p+a^{r_1}-1]\mathscr{D}_2[q,q+b^{r_2}-1]$ で表されれる)．また各ノードはその領域の平均値 m とエラー Err_{D_1},Err_{D_2} も持つ．
   (c) $Err_{D_1}<T_{D_1}$ かつ $Err_{D_2}\geq T_{D_2}$ のとき，現在のノードの下に b 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．b 個の子ノードは．領域 $\mathscr{D}_1[p,p+a^{r_1}-1],\mathscr{D}_2[q,q+b^{r_2}-1]$ を D_2 方向に均等に b 分割した各領域を持つ (それぞれ領域 $\mathscr{D}_1[p,p+a^{r_1}-$

$1]\mathscr{D}_2[q,q+b^{r_2-1}-1],\mathscr{D}_1[p,p+a^{r_1}-1]\mathscr{D}_2[q+b^{r_2-1}-1,q+2b^{r_2-1}-1],\mathscr{D}_1[p,p+a^{r_1}-1]\mathscr{D}_2[q+2b^{r_2-1}-1,q+3b^{r_2-1}-1],\cdots\mathscr{D}_1[p,p+a^{r_1}-1]\mathscr{D}_2[q+(b-1)b^{r_2-1}-1,q+b^{r_2}-1]$ と表される). また各ノードはその領域の平均値 m とエラー Err_{D_1},Err_{D_2} も持つ.

(d) $Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ のとき，子ノード作成は行わず，再帰的なルール 2 も適用しない.

3. 現在のノードがすべて子ノード作成が行えなくなると処理を終了する．また，できあがった木構造 $Tree_{T_{D_1},T_{D_2}}$ と子ノード作成が行えなくなったノードの情報 I_{foot} を保持しておく.

例として，時系列ボリュームデータの空間次元を 1 次元化し横方向に並べ，時間次元はそのまま 1 次元で縦方向に並べたものの場合の多解像度化の処理の様子を Fig.5.4 に載せる．図のアルファベットは空間次元を D_1，時間次元を D_2 として扱ったときに，その領域が上の (a) から (d) の処理が適用されることと対応している．なお，空間次元は解像度が 1 あがると領域数が 8 倍に，時間次元は 2 倍になることを注記しておく．また，Fig.5.4 に対応する木構造構築の様子を Fig.5.5 に載せる．図中の [,] 内に書かれた数値は，上のものがノードが持つ空間次元におけるデータポイントの範囲，下のものが時間次元におけるデータポイントの範囲である.

描画の際には，I_{foot} に書かれた情報を利用する．I_{foot} には，再帰的な再分割が止まったノードが書かれてあるので，この情報の持つそのノードが示す領域 $\mathscr{D}_1,\mathscr{D}_2$ に対応する次元変換後の領域にノードの所持する平均値に対応する色をつけることで，多解像度化された結果の描画が完了する．多解像度化された結果例として，自動車周りの圧力分布のシミュレーションから得られた時系列ボリュームデータに対し，縦軸方向を時間軸，横軸方向を空間軸にとった場合を Fig.5.6 に載せる．ここでは，最大の空間解像度を 3，時間解像度を 5 とした場合のデータを扱っている (計 $(2^3)^3\cdot 2^5 = 512\cdot 32 = 16384$ データポイント)．ただし，Fig.5.6 では，横軸方向 (空間次元方向) は一部分のみを取り出して表示している．各図の下に書かれた数値 T_s,T_t はそれぞれ空間次元，時間次元に対する閾値を表しており，またその閾値での領域数も載せている．図 (a) は多解像度化を行わない場合に相当している.

I_{foot} 以外に保存している木構造 $Tree_{T_{D_1},T_{D_2}}$ は後の小節で述べる指定が閾値変化した際の木構造の再構築に利用する.

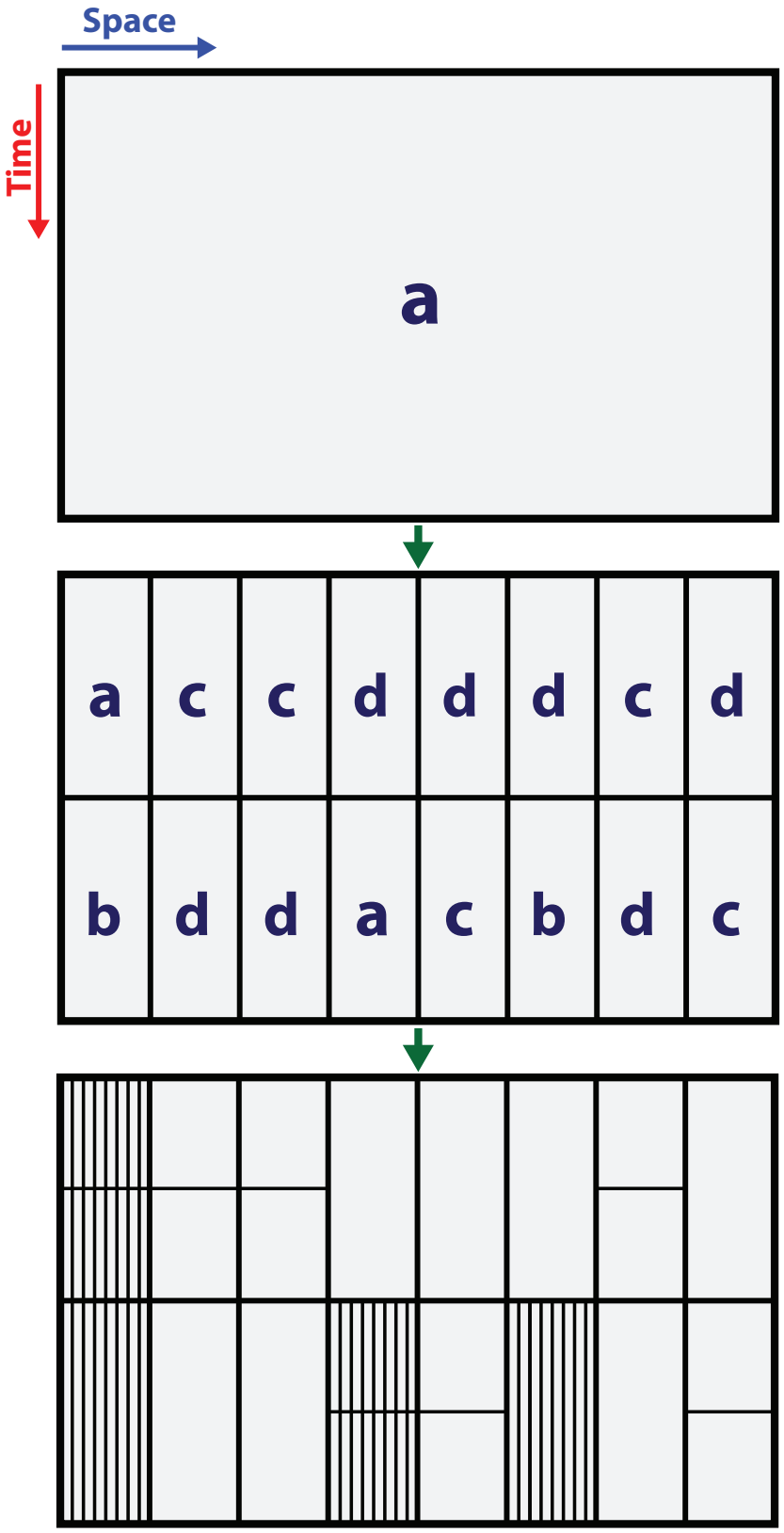


Fig. 5.4: An example of multi-resolution process for time-varying volume data

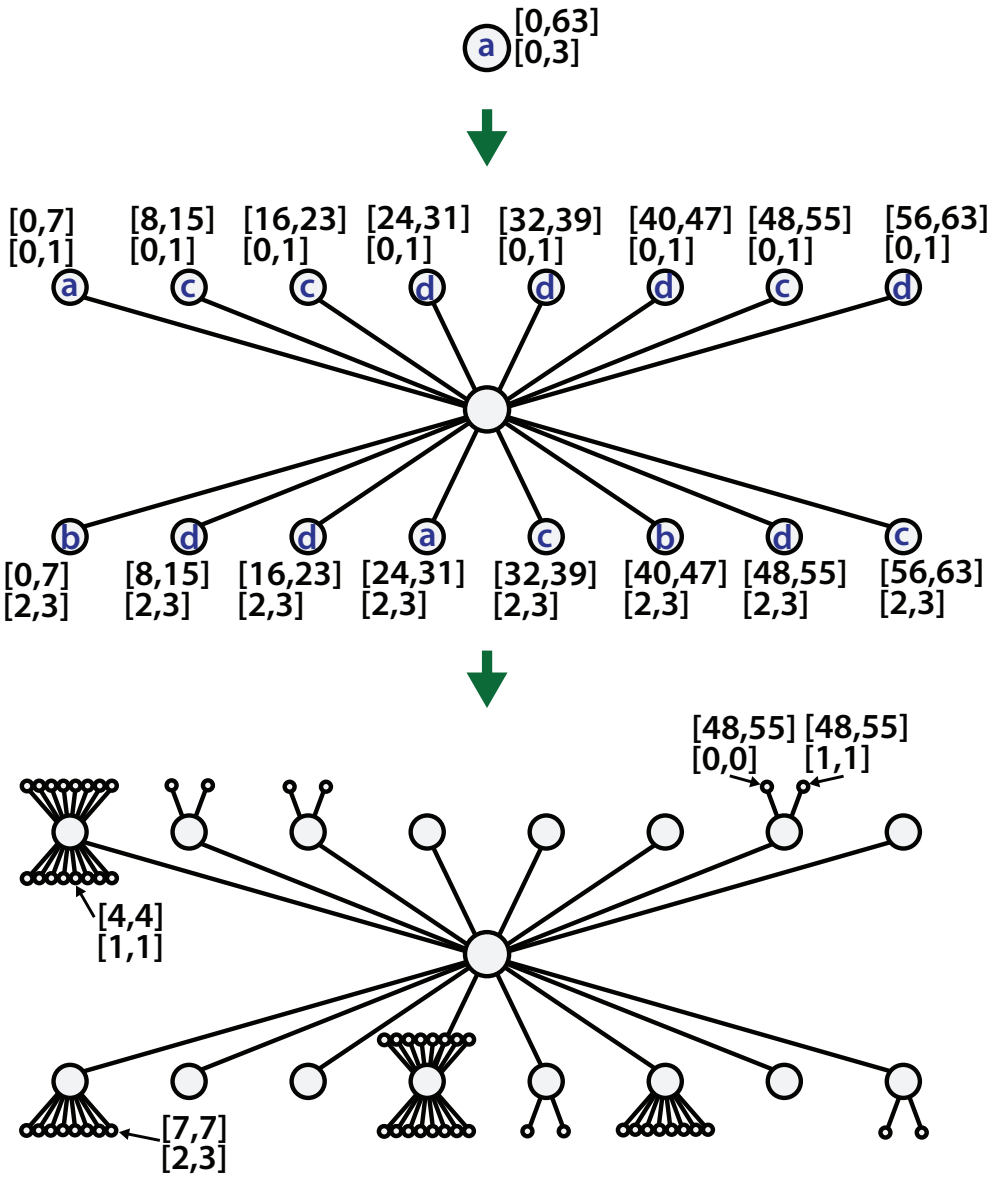


Fig. 5.5: Tree structure corresponding with Fig.5.4

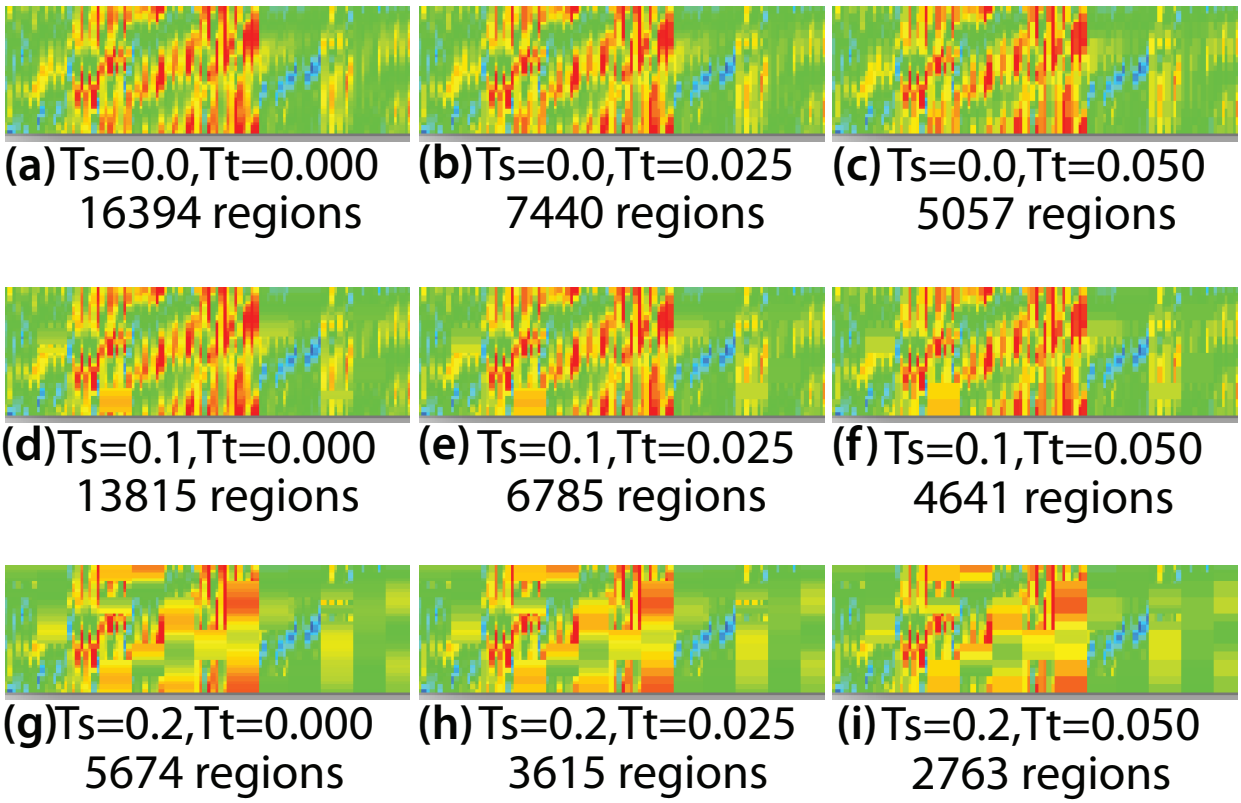


Fig. 5.6: Multi-resolution results of time-varying volume data from numerical simulation.

5.4.2　3 種類の次元群を扱う場合の多解像度化

3 種類の次元群 D_1, D_2, D_3 を扱う場合の多解像度化も基本的な手法の方針は 2 種類の次元群を扱う場合と同じである．異なる部分は，エラーを求める式や再帰的な分割における分岐数が増加するところである．

はじめにエラーを求める式を示す．D_1, D_2, D_3 におけるデータポイント d_1, d_2, d_3 がそれぞれ D_1, D_2, D_3 内の領域 $\mathscr{D}_1, \mathscr{D}_2, \mathscr{D}_3$ に含まれている範囲における D_1, D_2, D_3 のエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ は次のように計算できる．

$$m_{D_i}(d_j, d_k) = \frac{\sum_{d_i \in \mathscr{D}_i} v_{d_i,d_j,d_k}}{n(\mathscr{D}_i)} \tag{5.4.10}$$

$$s_{D_i}(d_j, d_k) = \sqrt{\frac{\sum_{d_i \in \mathscr{D}_i} v^2_{d_i,d_j,d_k}}{n(\mathscr{D}_i)} - (m_{D_i}(d_j, d_k))^2} \tag{5.4.11}$$

$$c_{D_i}(d_j, d_k) = \frac{s_{D_i}(d_j, d_k)}{m_{D_i}(d_j, d_k)} \tag{5.4.12}$$

$$Err_{D_i} = \frac{\sum_{d_j \in \mathscr{D}_j} \sum_{d_k \in \mathscr{D}_k} c_{D_i}(d_j, d_k)}{n(\mathscr{D}_j) n(\mathscr{D}_k)} \tag{5.4.13}$$

i, j, k は次元種類の添字であり，$(i, j, k) = (1,2,3), (2,3,1), (3,1,2)$ のそれぞれの場合が $Err_{D_1}, Err_{D_2}, Err_{D_3}$ を求める場合に対応する．v_{d_i,d_j,d_k} は d_i, d_j, d_k で決まるデータポイントにおけるデータの値，$n(\mathscr{D}_i), n(\mathscr{D}_j), n(\mathscr{D}_k)$ は領域 $\mathscr{D}_i, \mathscr{D}_j, \mathscr{D}_k$ におけるデータポイントの総数，$m_{D_i}(d_j, d_k)$，$s_{D_i}(d_j, d_k), c_{D_i}(d_j, d_k)$ は上記の式で計算される平均値，標準偏差，変動係数である．

また，領域 $\mathscr{D}_1, \mathscr{D}_2, \mathscr{D}_3$ に含まれるデータの平均値 m は次の式で計算される．

$$m = \frac{\sum_{d_1 \in \mathscr{D}_1} \sum_{d_2 \in \mathscr{D}_2} \sum_{d_3 \in \mathscr{D}_3} v_{d_1,d_2,d_3}}{n(\mathscr{D}_1) n(\mathscr{D}_2) n(\mathscr{D}_3)} \tag{5.4.14}$$

次に木構造構築の方法について述べる．エラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ に対する閾値 $T_{D_1}, T_{D_2}, T_{D_3}$ を決め，その閾値とエラーの大小に基づき木構造を作成する．すべてのノードは平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ とそのノードの平均値やエラーを計算したときの領域を表す $\mathscr{D}_1, \mathscr{D}_2, \mathscr{D}_3$ を持つ．木構造作成には，以下に示す再帰的なルールを用いる．今次元群 D_1, D_2, D_3 はそれぞれ $\mathscr{D}_1[0, a^{R_1} - 1], \mathscr{D}_2[0, b^{R_2} - 1], \mathscr{D}_3[0, c^{R_3} - 1]$ の範囲に含まれるデータを持つこととする (総データ数は，$a^{R_1} b^{R_2} c^{R_3}$ 個)．ここで，a, b, c は解像度 1 のときのデータの要素数，R_1, R_2, R_3 はデータの持つ最大の解像度である．

「3 種類の次元群を多解像度化するための木構造作成ルール」

1. 木構造の最上層のノードは，$\mathscr{D}_1[0, a^{R_1} - 1], \mathscr{D}_2[0, b^{R_2} - 1], \mathscr{D}_3[0, c^{R_3} - 1]$ で表される領域の平均値 m とエラー Err_{D_1}, Err_{D_2} を持ち，このノードからスタートして木構造を構築する．

2. 現在いるノードの持つエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ の値と閾値 $T_{D_1}, T_{D_2}, T_{D_3}$ を比較した結果 (a) から (h) に応じた処理を行う．現在のノードは領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1], \mathscr{D}_2[p_2, p_2+b^{r_2}-1], \mathscr{D}_3[p_3, p_3+c^{r_3}-1]$ を持つものとする．ここで p_1, p_2, p_3 は 0 以上の整数，r_1, r_2, r_3 は現在のノードが持つ領域の D_1, D_2, D_3 における最大の解像度を表す．
   (a) $Err_{D_1} \geq T_{D_1}$ かつ $Err_{D_2} \geq T_{D_2}$ かつ $Err_{D_3} \geq T_{D_3}$ のとき，現在のノードの下に abc 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．abc 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_1 方向に均等に a 分割と D_2 方向に均等に b 分割と D_3 方向に均等に c 分割をしてできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (b) $Err_{D_1} \geq T_{D_1}$ かつ $Err_{D_2} \geq T_{D_2}$ かつ $Err_{D_3} < T_{D_3}$ のとき，現在のノードの下に ab 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．ab 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_1 方向に均等に a 分割と D_2 方向に均等に b 分割してできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (c) $Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} \geq T_{D_2}$ かつ $Err_{D_3} \geq T_{D_3}$ のとき，現在のノードの下に bc 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．bc 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_2 方向に均等に b 分割と D_3 方向に均等に c 分割してできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (d) $Err_{D_1} \geq T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ かつ $Err_{D_3} \geq T_{D_3}$ のとき，現在のノードの下に ac 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．ac 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_1 方向に均等に a 分割と D_3 方向に均等に c 分割してできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (e) $Err_{D_1} \geq T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ かつ $Err_{D_3} < T_{D_3}$ のとき，現在のノードの下に a 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．a 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_1 方向に均等に a 分割してできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (f) $Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} \geq T_{D_2}$ かつ $Err_{D_3} < T_{D_3}$ のとき，現在のノードの下に b 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．b 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_2 方向に均等に b 分割してできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (g) $Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ かつ $Err_{D_3} \geq T_{D_3}$ のとき，現在のノードの下に c 個の子ノードを作成し，それぞれの子ノードに再帰的なルール 2 を適用する．c 個の子ノードは．領域 $\mathscr{D}_1[p_1, p_1+a^{r_1}-1]\mathscr{D}_2[p_2, p_2+b^{r_2}-1]\mathscr{D}_3[p_3, p_3+c^{r_2}-1]$ を D_3 方向に均等に c 分割してできる各領域を持ち，その領域の平均値 m とエラー $Err_{D_1}, Err_{D_2}, Err_{D_3}$ も持つ．
   (h) $Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ かつ $Err_{D_3} < T_{D_3}$ のとき，子ノード作成は行わず，再帰的なルール 2 も適用しない．

3. 現在のノードがすべて子ノード作成が行えなくなると処理を終了する．また，できあがった木構造 $Tree_{T_{D_1},T_{D_2},T_{D_3}}$ と子ノード作成が行えなくなったノードの情報 I_{foot} を保持しておく．

例として，D_1, D_2, D_3 の解像度 1 のときの要素数 a, b, c がそれぞれ 2, 4, 8 のときに，1 領域が上の (a) から (h) の分割ルールが適用された場合の様子を Fig.5.7 に載せる．

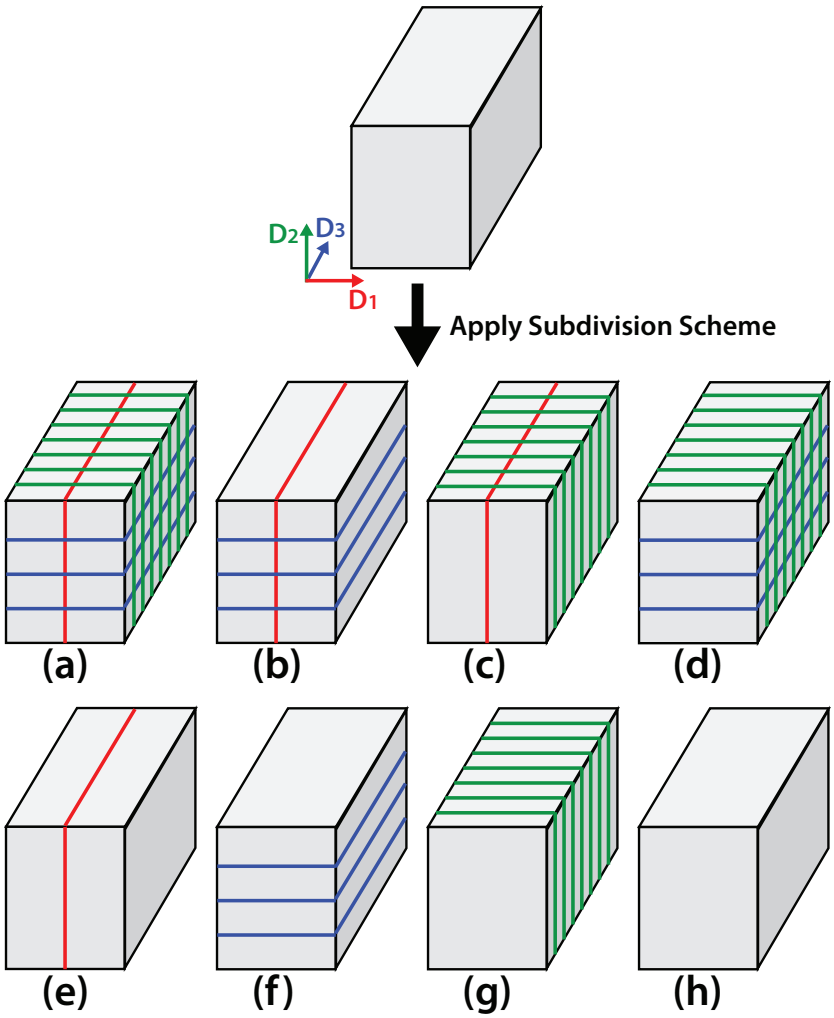


Fig. 5.7: Application of subdivision scheme for 3 type dimensions.

描画の際には，2 種類の次元種類の場合と同様に I_{foot} に書かれた情報を利用する．I_{foot} には，再帰的な再分割が止まったノードが書かれてあるので，この情報の持つそのノードが示す領域 $\mathscr{D}_1, \mathscr{D}_2, \mathscr{D}_3$ に対応する次元変換後の場所にノードの持つ平均値に対応する色をつけることで，多解像度化された結果の描画が完了する．

また，I_{foot} 以外に保存している木構造 $Tree_{T_{D_1},T_{D_2},T_{D_3}}$ は後の小節で述べる指定が閾値変化した際の木構造の再構築に利用する．

5.4.3　木構造の再構築

本節で紹介した次元別次元変換手法のための木構造構築の手法では，閾値とエラーの大小関係に基づいて木構造を構築するため，閾値の変化を行った場合には木構造をその閾値に従って再構築する必要がある．木構造を再構築する際には，すでに作られてある木構造 *Tree* の持つ情報を再利用することで，構築時間を減らすことができる．以下では，木構造 *Tree* を利用しながら，新たな木構造 *Tree′* を作成する方法を述べる．

木構造再構築を行うには，木構造 *Tree* 上を走査しながら，閾値よりエラーが低いノードの持つ子ノードの削除，閾値よりエラーが大きいノードに子ノードを作成するといった処理を行う．その際には以下のルールに基づいて処理を行う (2 種類の次元種類の場合を載せている).

「木構造再構築のためのルール」

1. 木構造 *Tree* の最上層のノードからスタートする．
2. 現在いるノードの持つエラー Err_{D_1}, Err_{D_2} の値と新たな閾値 T'_{D_1}, T'_{D_2} を比較した結果 (a) から (d) に応じた処理を行う．
   (a)　$Err_{D_1} \geq T'_{D_1}$ かつ $Err_{D_2} \geq T'_{D_2}$ のとき，子ノードが D_1，D_2 両方向に分割されてできたものである場合は子ノードの方へ移動して，「2 種類の次元群を多解像度化するための木構造作成ルール」の 2 番で書かれてある再帰的ルールを適用しながら走査を続ける．子ノードが D_1，D_2 両方向に分割されてできたものでない場合は，今のノードから「2 種類の次元群を多解像度化するための木構造作成ルール」の 2 番で書かれてある再帰的ルールに基づいて子ノードを再構築する．
   (b)　$Err_{D_1} \geq T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ のとき，子ノードが D_1 方向だけに分割されてできたものである場合は，子ノードの方へ移動して，「2 種類の次元群を多解像度化するための木構造作成ルール」の 2 番で書かれてある再帰的ルールをを適用しながら走査を続ける．子ノードが D_1 方向だけに分割されてできたものでない場合は，現在持っている子ノードを削除してから，今のノードから「2 種類の次元群を多解像度化するための木構造作成ルール」の 2 番で書かれてある再帰的ルール基づいて子ノードを再構築する．子ノード数が 0 の場合は，今のノードから木構造構築の際のルール 2 に基づいて子ノードを再構築する．
   (c)　$Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} \geq T_{D_2}$ のとき，子ノードが D_2 方向にだけ分割されてできたものである場合は，子ノードの方へ移動して，「2 種類の次元群を多解像度化するための木構造作成ルール」の 2 番で書かれてある再帰的なルールを適用しながら走査を続ける．子ノードが D_2 方向にだけ分割されてできたものでない場合は，現在持っている子ノードを削除してから，今のノードから「2 種類の次元群を多解像度化するための木構造作成ルール」の 2 番で書かれてある再帰的ルールに基づいて子ノードを再構築する．子ノード数が 0 の場合は，今のノードから木構造構築の際のルール 2 に基づいて子ノードを再構築する．
   (d)　$Err_{D_1} < T_{D_1}$ かつ $Err_{D_2} < T_{D_2}$ のとき，子ノードがある場合は，子ノードを削除する．

3. 現在のノードがすべてルール 2 の条件 (d) の場合になり，走査がストップした場合は，木構造再構築をストップし，できあがった新たな木構造 $Tree'$ とと子ノード作成が行えなくなったノードの情報 I_{foot} を保持しておく．

このルールに基づきできた木構造 $Tree'$ を利用して再描画を行えば，閾値変化があった場合の多解像度化処理が完了する．ここで載せたルールは 2 種類の次元種類の場合を載せたが，3 種類の場合も再帰的ルールの分岐数が 4 から 8 に増えるだけで，基本的な処理方法は 2 種類の場合と同じである．Fig.5.8 に Fig.5.5 で作成された木構造をもとにして，閾値変化があった場合の木構造の再構築の様子を載せている．各ノードに書かれたアルファベットは，上記の再構築の際の再帰的なルールにおける分岐 (a) から (d) に対応している．また，ここでは空間次元に対する閾値の値を小さくし，時間次元に対する閾値の値を大きくした場合に対応した再構築を行っている．図中の赤い枝とノードは新たに構築された箇所，破線で描かれた枝とノードは削除された箇所であることを表している．

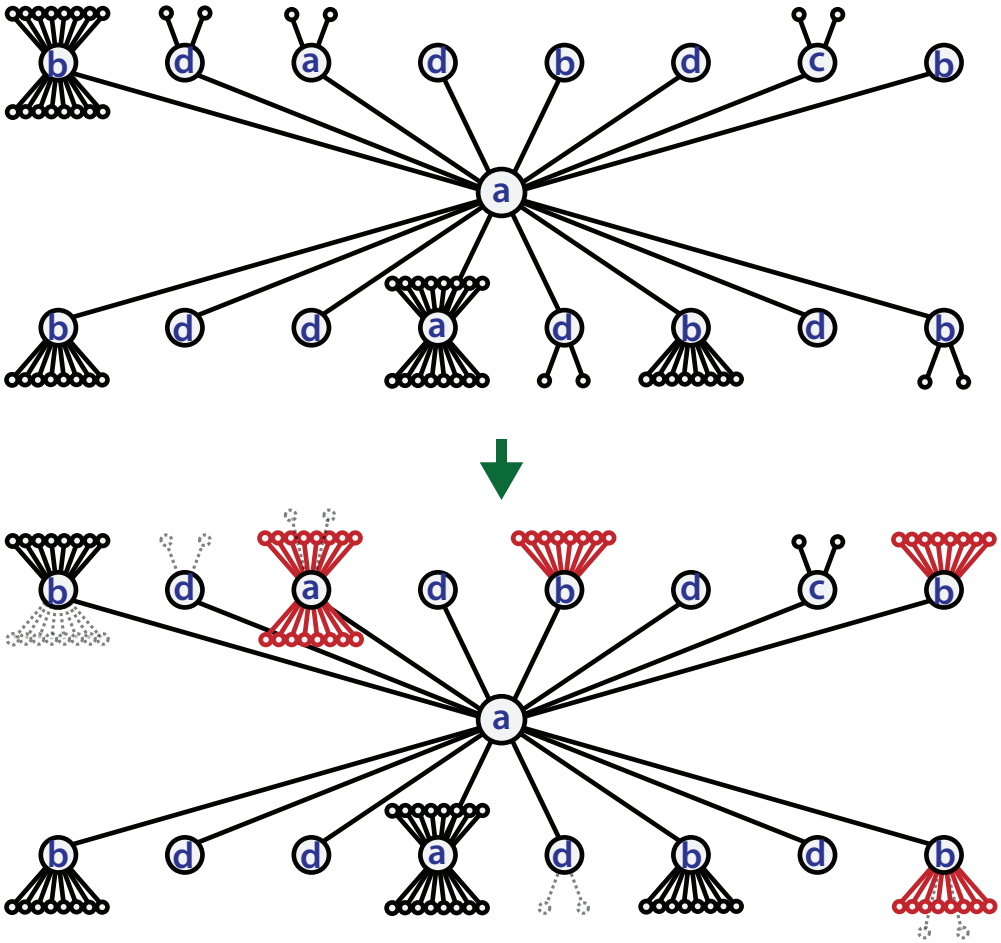


Fig. 5.8: Reconstruction of tree structure.

5.5　次元別の次元変換手法のまとめ

本章では，次元別に次元変換を行って低次元化・高次元化を行う手法を説明した．また，次元別に次元変換を行った場合には，次元別次元変換のための多解像度化の方法が必要であり，その方法についても説明した．次元別次元変換を行うことで，各次元における変化を把握しやすい形で表示できるようになった．

第 6 章

マッピングルール

6.1　自己相似図形とマッピングルールの再帰的適用

本研究手法では，もとの次元のデータと次元変換後の自己相似図形の間に解像度 1 のときのマッピングルールを設け，高解像度または多解像度でデータを表示する際には，自己相似図形の分割ルールとマッピングルールを再帰的に適用することで次元変換を実行することを説明した．自己相似図形上で決められたマッピングルールを再帰的に適用するため，解像度 1 のマッピングルールを再帰的に適用してできた高解像度用のマッピングルールはそのマッピングルール自体が自己相似性を持つルールとなる．自己相似性を持つルールは，解像度 1 のときのルールの持つ性質が局所的にも全体的にも見られる．そのため，解像度 1 のときのマッピングルールによって，可視化結果の全体的な傾向および局所的な傾向が決められる．解像度 1 のときのマッピングルールを適切に決めることによって，効果的な可視化結果が得られるようになるため，本章ではいくらかの特徴のあるマッピングルールを取り上げ，その特徴を考察する．

6.2　有用なマッピングルール例

以下では，有用なマッピングルール例を挙げる．ここでは，3 次元データを 2 次元化する際のマッピングを中心に取り上げる．しかし，ここに挙げるルールの多くはその考え方を利用すれば，他の n 次元データを次元変換する場合にも応用できる．

6.2.1　着目次元の座標値に基づくルール

ある着目している次元の座標値に基づいて作られたマッピングルールについて述べる．ここでは．3 次元データを 2 次元化する場合を例として取り上げて説明する．

例えば，z 座標に着目してルールを決める場合を考える．3 次元の場合のマッピングに用いる自己相似図形は，解像度 1 のとき 8 領域を持つ．一方，解像度 1 の 3 次元データは，各次元方向に 2 分割ずつされており，z 座標が小，大となる領域はそれぞれ 4 領域となる．そのため，z 座標の小，大によって 2 次元上でのマッピング位置を決めるには，2 次元自己相似図形上の 8 領域を 4 領域ずつの 2 グループに分け，それぞれに z 座標が小，大のものをマッピングする．2 次元自己相似図形の 8 領域を 4 領域ずつの 2 グループに分ける方法は多数あるが，同グループに所属している領域がどれであるか認識し易くするためには，

同グループ内にある領域は出来る限り近接しているほうがよい．この方針に基づいて 8 領域を 2 グループに分割する方法は，Fig.6.1 に挙げるようなものがある．Fig.6.1 の (a) のように 8 領域を 2 グループに分け，各グループに z 座標が小，大のものをマッピングするように決められた解像度 1 の場合のマッピングルール例を Fig.6.2 に載せる．また，Fig.6.2 のルールを再帰的に適用してできた解像度 1,2,3 のマッピングルールにおける z 座標値の大小を色を用いて表したものを Fig.6.3 に載せる．Fig.6.3 では，z 座標値が大きい領域ほど赤色で，小さい領域ほど青色で表現されている．Fig.6.3 の左図 (解像度 1) で見られた z 座標値が大きいものが左上，小さいものが右下に集まるといった傾向は，中央図 (解像度 2) や右図 (解像度 3) でも全体的な傾向として見られる．さらには，中央図や右図から自己相似な矩形領域の一部を取り出して観察しても，同様な傾向が見られるのがわかる．この結果からも，解像度 1 のときのルールが持つ傾向が，高解像度の場合の全体的な傾向にも局所的な傾向にも表れるということが分かる．

着目次元の座標値に基づいてルールを決定すれば，その着目次元方向の変化を可視化結果からとらえやすくなる．例えば，例で上げた z 座標値に基づいてルールを決めた場合には，z 座標値が大きいところにあるデータは左上，小さいところにあるデータは右下に集まる傾向があるため，z 座標値による値の変化が観察しやすくなる．

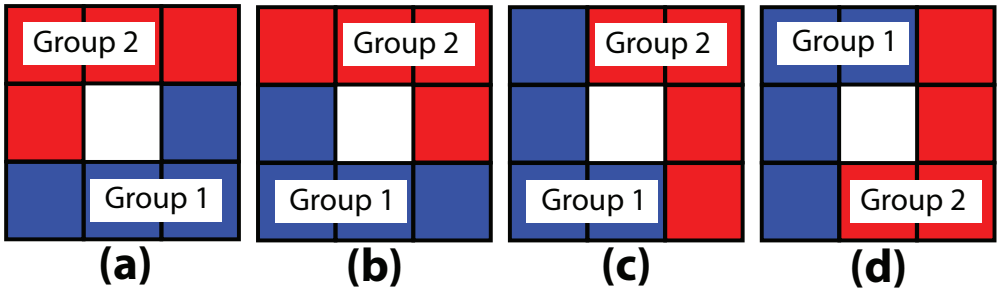


Fig. 6.1: An example of division of 2 groups for 8 regions.

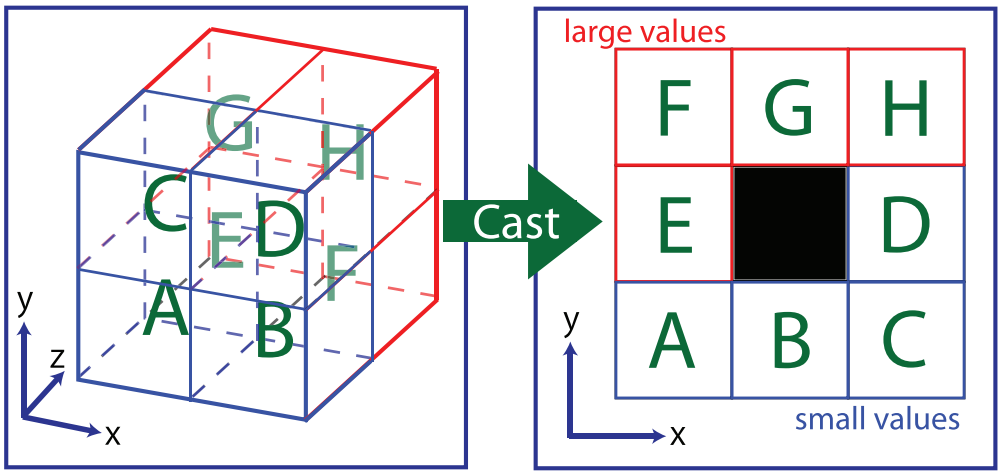


Fig. 6.2: Mapping rule example determined by the z value. The regions of small z values are at the lower right, and the regions of large z values are at the upper left.

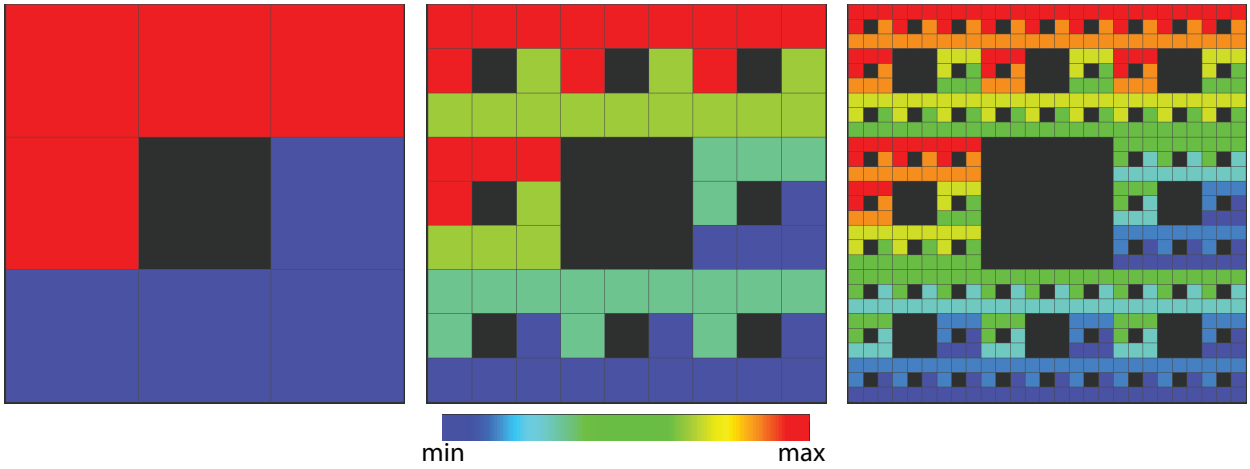


Fig. 6.3: Mapping results for resolution 1, 2, 3 with the expansion rule shown in Fig.6.2. The regions of small z values are colored blue and the regions of large z values are colored red.

6.2.2　ある位置からの距離に基づくルール

2 つめのルール例として，ある位置からの距離に基づいて作られたマッピングルールについて述べる．ここでは，3 次元データを 2 次元化する場合を例として取り上げて説明する．

例えば，原点からの距離に基づいてマッピングルールを決める場合を考える．以後の説明をしやすくするために，もとのデータの持つ領域の 3 次元空間での座標を次のように表現することにする．もとのデータが解像度 R を持ち，各次元方向に 2^R 分割されているときに，各次元方向の座標値が小さい順に $0,1,2,\cdots,2^R-1$ と座標を割り当てる．例えば，Fig.6.4 の左図の 3 次元上の各領域 A から H は，それぞれ座標 $A(0,0,0),B(1,0,0),C(0,1,0),D(1,1,0),E(0,0,1),F(1,0,1),G(0,1,1),H(1,1,1)$ となる．このとき原点からの距離は，それぞれ $0,1,1,\sqrt{2},1,\sqrt{2},\sqrt{2},\sqrt{3}$ となる．この距離に基づいてマッピングルールを決定するが，まずマッピング先の 2 次元画像の 8 領域のうち，左下の領域を 3 次元データで原点にある領域 A をマッピングする領域とする．できる限りその左下の領域から近くにある領域ほど，もとの 3 次元上での原点からの距離が近い領域がマッピングされるようにルールを決めると，もとの 3 次元上で原点から近かった距離が 2 次元上でも原点 (左下) から近くにある領域に表示されるようになるため，自然な可視化結果をもたらす．このように解像度 1 のときのマッピングルールを決めたものが，Fig.6.4 である．また，他の解像度にも適用した場合の例を Fig.6.5 に載せる．Fig.6.5 では，3 次元上での距離が原点に近いものほど青色で，遠いものほど赤色で表現されている．この図の結果も，前述の「着目次元の座標値に基づくルール」と同様に高解像度のときも，解像度 1 のときのルールが持つ原点からの距離に対する傾向が全体的・局所的に見受けられる．

ある位置からの距離に基づいてルールを決定すれば，その位置からの距離に応じて値が変化する現象 (例えば，拡散現象など) を効果的に可視化できる．

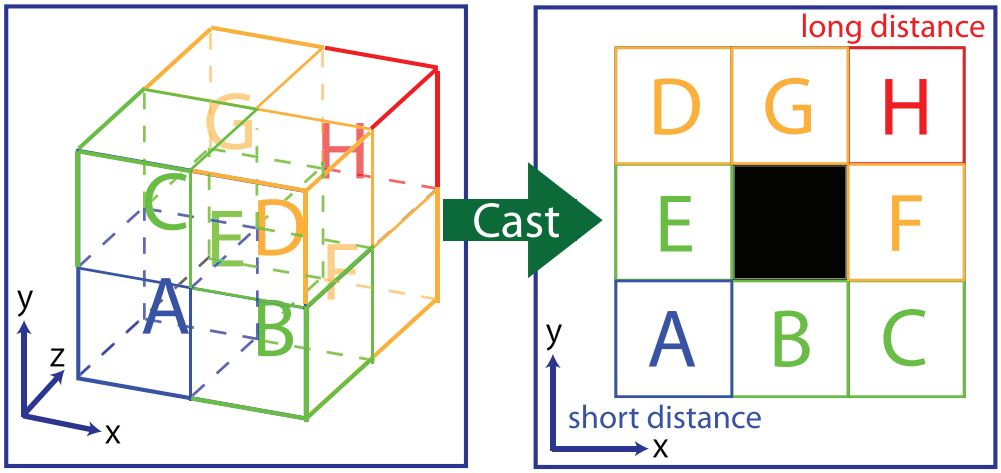


Fig. 6.4: Mapping rule example determined by distance from region A. The short-distance regions are at the lower left, and the long-distance regions are at the upper left.

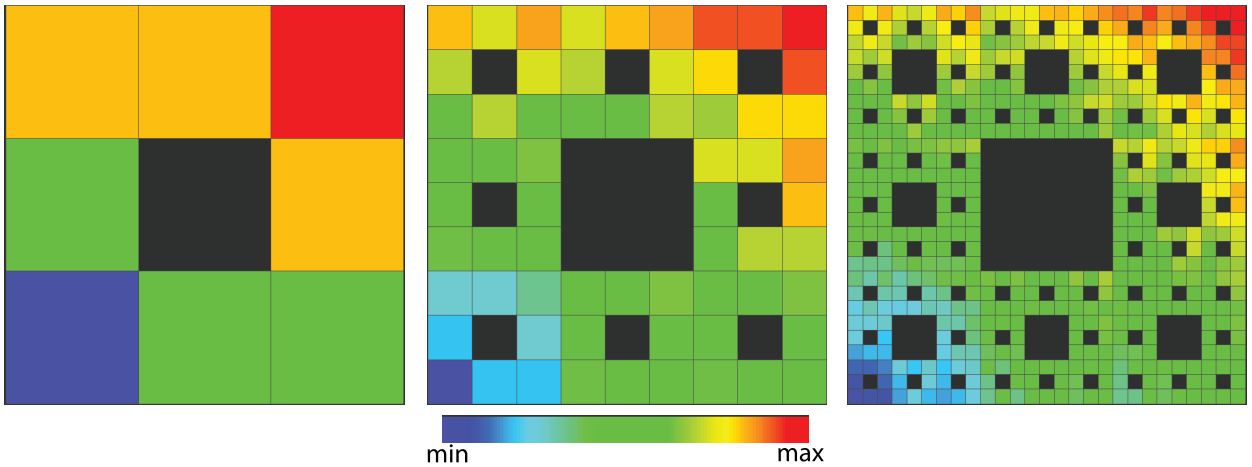


Fig. 6.5: Mapping results for resolution 1, 2, 3 with the expansion rule in Fig.6.4. The regions at a short distance from A are colored blue and the regions at a long distance are colored red.

6.2.3　点対称性に基づくルール

3 つめのルール例として，もとのデータの持つ点対称性を維持するようにマッピングルールを決める場合を取り上げる．ここでは，3 次元データを 2 次元化する場合を例として取り上げて説明する．

もとのデータの 3 次元上で中央点を点の中心として点対称な位置にあるデータをマッピング先の 2 次元上でも中央点を点の中心として点対称な位置にあるように解像度 1 のときのマッピングルールを Fig.6.6 のように定める．このルールでは，前述の 2 つのルールと異なり，3 次元上で中央点を点の中心として点対称な位置は 2 次元上でも中央点を点の中心として点対称な位置にその点対称性完全に維持してマッピングされる．Fig.6.6 に，Fig.6.7 のルールを高解像度の場合に再帰的に適用した結果を載せる．Fig.6.7 で

は，点対称な位置にある領域のペアは同じ色で塗られている．高解像度になった場合も，中央点を点の中心とした点対称性は維持されている (全体的な傾向)．また，局所的にも局所的な領域の中心点を点の中心とした点対称性が保持される (局所的な傾向)．この局所的な傾向は，Fig.6.8 で観察できる．Fig.6.8 は解像度 3 のときのマッピングルールであり，各領域の対応するもとの 3 次元データの位置の座標値をその領域内に 3 つの数字 (x,y,z 座標) として表示している．例えば，座標値 $(0,0,0),(1,0,0),(0,1,0),\cdots,(1,1,1)$ を持つ領域が含まれる局所的な領域では，もとの 3 次元データ上で点 $(1/2,1/2,1/2)$ を点の中心とした点対称な位置にある点を，中央のブランクスペースを点の中心とした周りの 8 領域に点対称に配置していることが分かる．

点対称性に基づくルールは，もとの次元上の位置関係を次元が変化した後も維持できるといった点でも有意義なルールである．

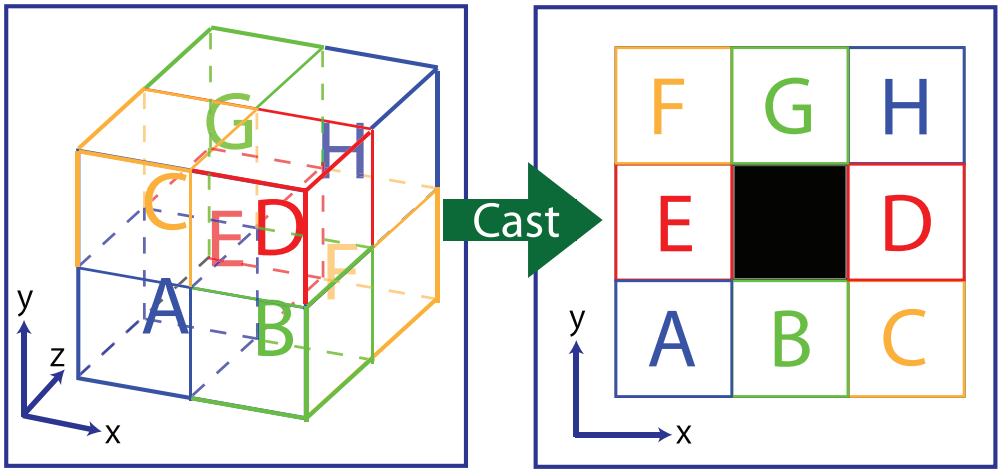


Fig. 6.6: Mapping rule example determined by point symmetry around the center. Point symmetric cells in 3D are mapped on point symmetric squares in 2D. Corresponding regions have the same color.

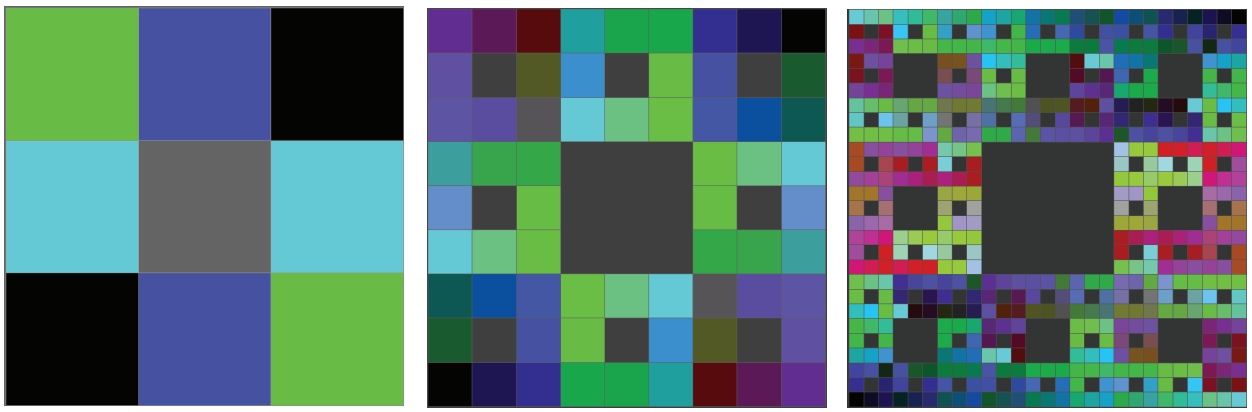


Fig. 6.7: Mapping results for resolution 1, 2, 3 with the expansion rule in Fig.6.6. 3D point symmetry cells have the same color. A 2D point symmetry can be observed.

さらに，この点対称に基づくマッピングルールを用いると解像度 1,2,3 のときの中心点からの距離が

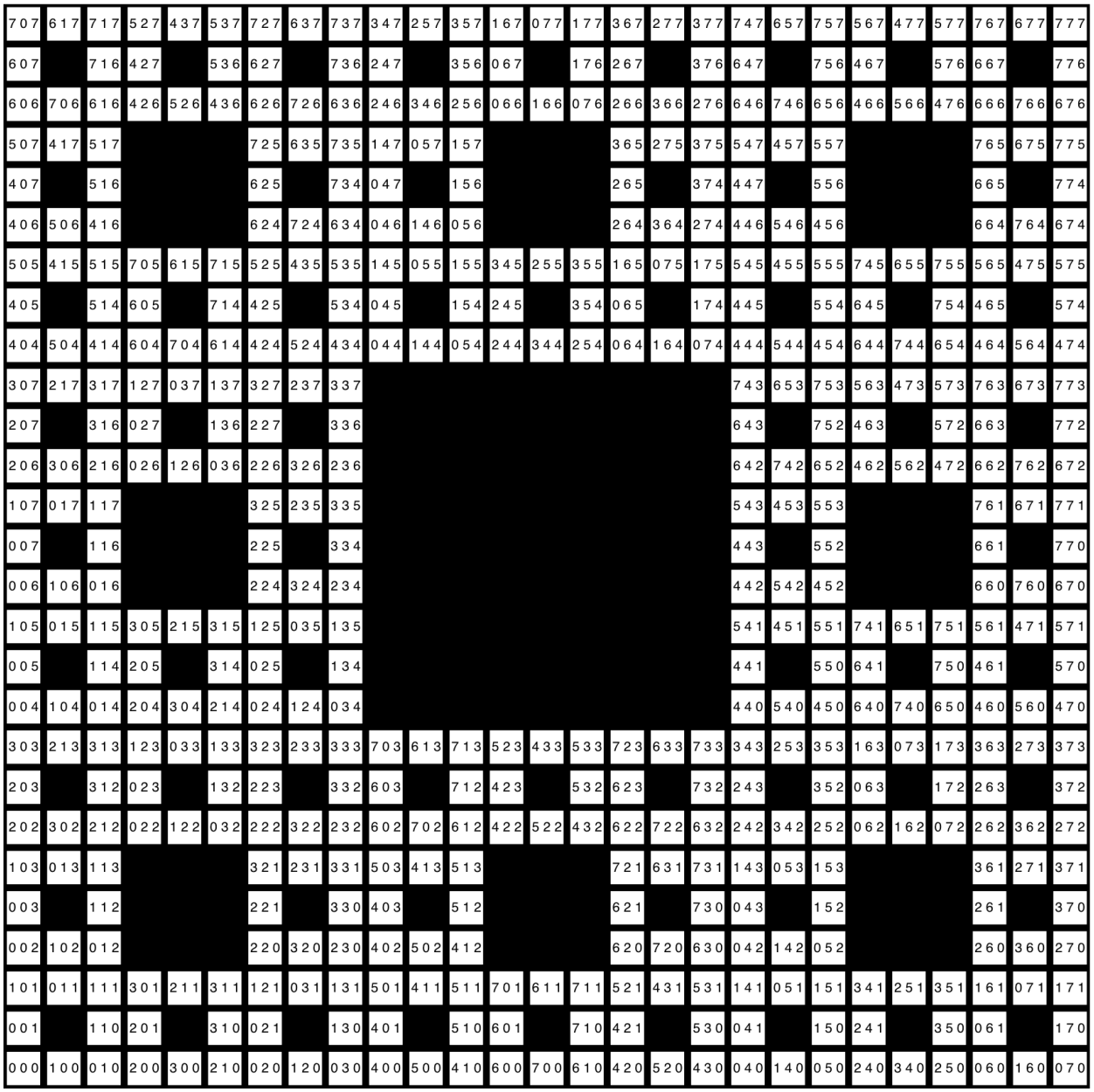


Fig. 6.8: Mapping results for resolution 3 with the expansion rule in Fig.6.6. The numbers represent corresponding 3D positions with the squares.

Fig.6.9 のようになる．Fig.6.9 では，もとの 3 次元上の中心点からの距離が近いものほど青色で，遠いものほど赤色で表されている．この図を見ると，3 次元上で中心点から最も近い位置にあるものは，中央のブランクスペースの近傍に配置され，最も遠い位置にあるものは，最も外郭の位置に配置されている事がわかる．ただし，ここで注記しておかなければならないことは，3 次元上で最も遠い位置にあるものが 2 次元上で最も外郭の位置に配置されるのは，どのようなマッピングルールを選択しても見られる性質であるということである (Fig.6.10)．これは，3 次元データのマッピング先であるシェルピンスキーのカーペットは解像度 1 のときすべての領域が空の領域 (中央点) の近傍に存在していることによる．マッピン

グ先の自己相似図形がすべての領域が中央点と近接していない場合は，もとの次元上で最も遠い位置にあるデータはどのようなマッピングルールを定めても，マッピング先で最も外郭に存在するというルールは成り立たない．

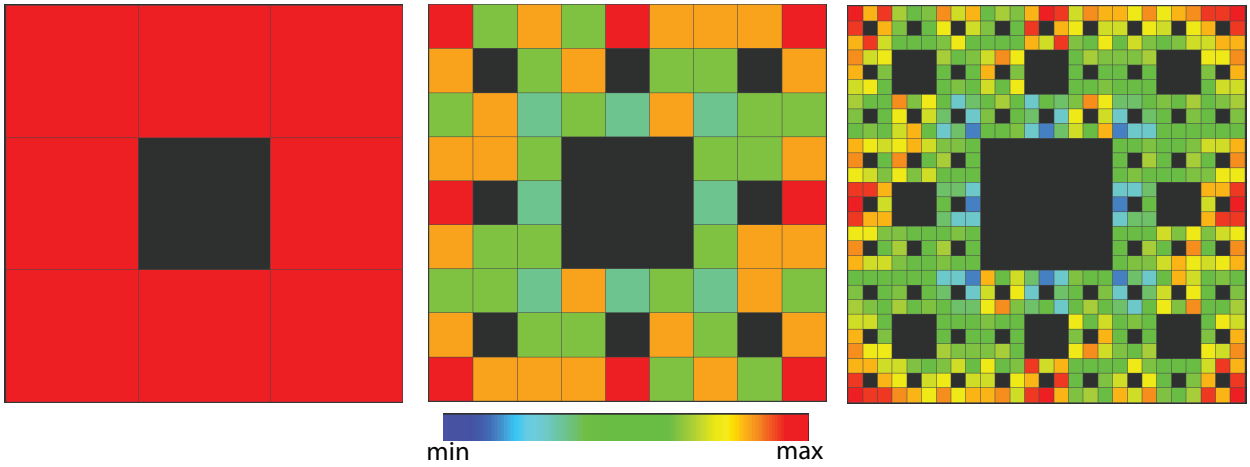


Fig. 6.9: Mapping results for resolution 1, 2, 3 with the expansion rule in Fig.6.6. The regions at a short distance from center point are colored in blue and the regions at a long distance from center point are colored in red.

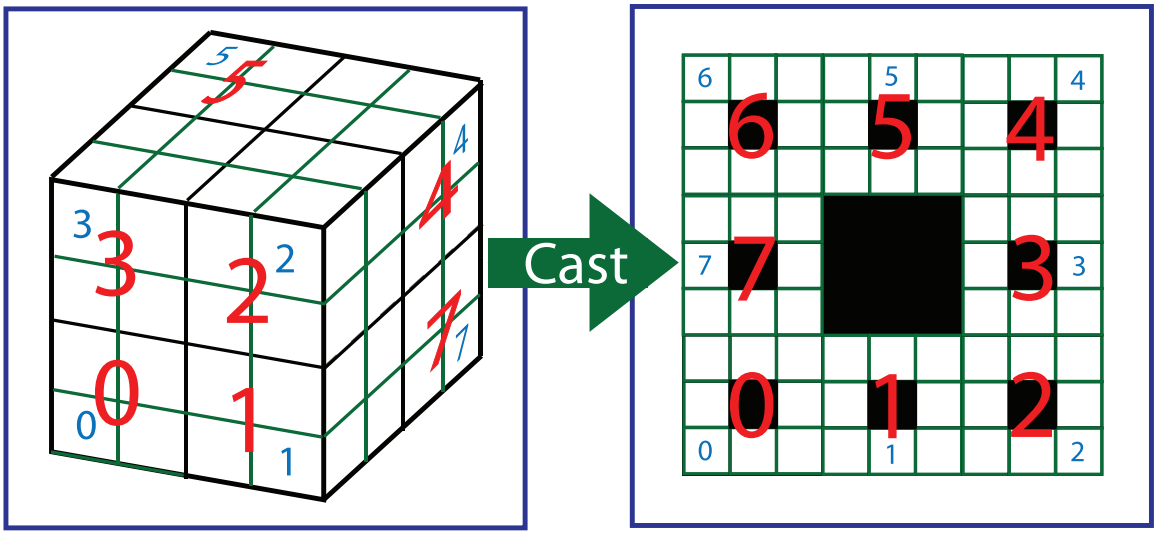


Fig. 6.10: Small Region X in Region X is in same direction from center point (X:0, 1, 2, 3, 4, 5, 6, 7).

6.2.4　組み合わせ自己相似図形の性質に基づくルール

組み合わせ自己相似図形はすでに作られてある自己相似図形を組み合わせて作られるため，ある次元用に利用される自己相似図形を他の次元用の自己相似図形の各領域が持つようになる．例えば，Fig.6.11 は 5 次元用に用意する自己相似図形であるが，この自己相似図形は 3 次元用の自己相似図形の各点が 2 次元

用の自己相似図形を持っていると見なすことができる．この状態を 2 次元空間がたたみ込まれて，3 次元上の各点に存在しているということとする．そのため，Fig.6.12(図中の各 5 桁の数字は，マッピング前の 5 次元空間での座標位置 x_1,x_2,x_3,x_4,x_5 を表している) のようにマッピングルールを定めると，全体としては x_1,x_2,x_3 の 3 次元的な変化を表示し，その 3 次元上の各点において x_4,x_5 の 2 次元的な変化が見られるような可視化結果をもたらすことができる．組み合わせ自己相似図形では，更に様々な次元がたたみ込まれて，様々な次元上の各点に存在しているような状態を作ることが可能である．例えば，Fig.6.13 は 8 次元に用いる自己相似図形の例であるが，5 次元空間がたたみ込まれて，3 次元上の各点に存在している．また，たたみ込まれている 5 次元空間自体も，2 次元空間がたたみ込まれて，3 次元上の各点に存在して作られた空間と見なすことができる．

組み合わせ自己相似図形の性質により次元がたたみ込まれているようなマッピングルールが作成可能であり，そのマッピングルールを用いることで，全体的な変化ではある次元方向の変化が見られ，自己相似図形が持つ各領域では他の次元方向の変化が見られるといった次元ごとに可視化結果に表れる状態に差をつけることが可能となる．

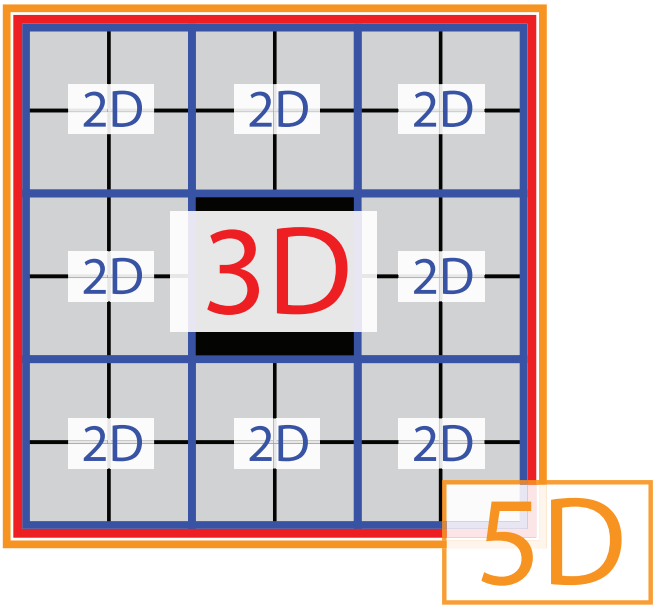


Fig. 6.11: 5D self-similar object includes 2D self-similar objects in each region of 3D self-similar object.

6.2.5　複数の性質を保持するルール

ここまでの小節では個別にある性質を有するマッピングルールを取り上げてきたが，1 つのマッピングルール内で複数の性質を持たせることもことも可能である．例えば，すでに取り上げた Fig.6.6 のマッピングルールは，1.z 座標値が大きい領域がマッピング先で左上に集まり，小さい領域は右下に集まっている，2.y 座標値が大きい領域は右上に集まり，小さい領域は左下に集まっている，3. 原点 (A 領域) から距離の遠いものほど右上に位置している，4. 中心点を点の中心として点対称になっている，といった 3 つの性質を同時に併せもち，「着目次元の座標値に基づくルール」かつ「ある位置からの距離に基づくルール」

10101	10111	01101	01111	11101	11111
10100	10110	01100	01110	11100	11110
00101	00111			11001	11011
00100	00110			11000	11010
00001	00011	10001	10011	01001	01011
00000	00010	10000	10010	01000	01010

Fig. 6.12: Mapping rule example for 5D. 2D spaces are folded in each point of 3D spaces.

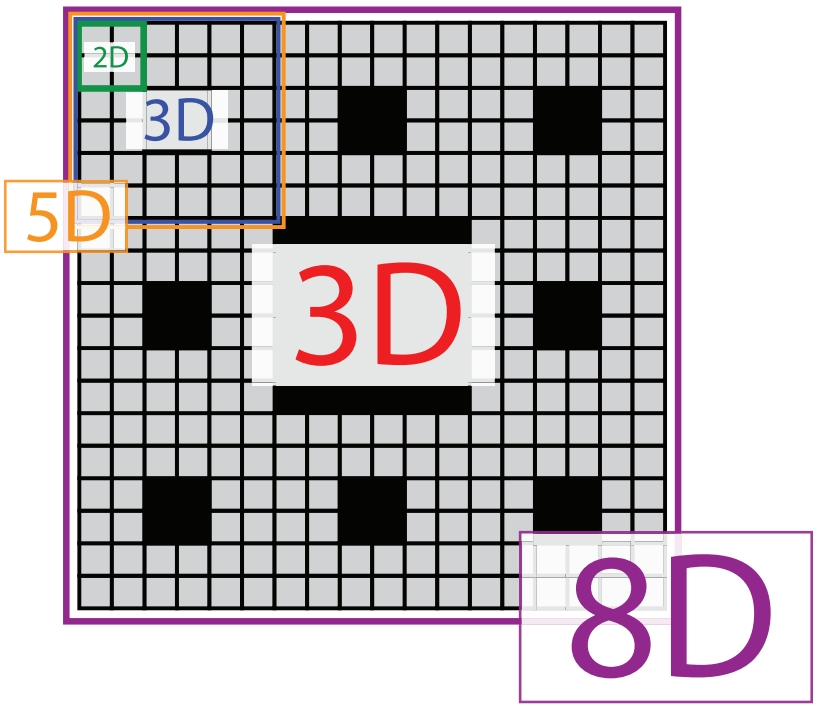


Fig. 6.13: 8D self-similar object includes 5D self-similar objects in each region of 3D self-similar object and 5D self-similar objects include 2D self-similar objects in each region of 3D self-similar object.

かつ「点対称性に基づくルール」となっている．このような複数の性質を 1 つのマッピングルールに持たせると，複数の性質に基づいた可視化結果が得られるため，より効果的な次元変換可視化が行える．

6.2.6　マッピングルールのまとめ

本章では，解像度 1 のときのマッピングルールが可視化像全体の特徴を左右することを説明し，解像度 1 のときのマッピングルールの決め方として有益だと思われるものを説明した．可視化対象に応じて適切にマッピングルールを定めることで，効果的な可視化結果を得ることができるようになる．

第 7 章

可視化事例

7.1　時系列ボリュームデータの可視化

可視化事例として時系列ボリュームデータの可視化を取り上げる．時系列ボリュームデータは空間 3 次元と時間 1 次元を持つ 4 次元データである．時系列ボリュームデータは数値シミュレーションから得られることが多いデータであり，可視化研究の主対象の 1 つとなっている．時系列ボリュームデータに本手法を適用し，様々な次元変換による結果例を示し，効果的な可視化結果が得られることを示す．ここであげる例では，数値シミュレーションから得られた自動車モデル後部周辺の時間変化する 3 次元圧力分布 [27] を可視化対象としている．この可視化対象は既存研究 [34] の中で扱われていたものと同様のものであり，Fig.7.1 において白枠で囲まれた立方体領域を本研究でも可視化を行う．Fig.7.1 では，ボリュームレンダリングではなく，Scatterplot を用いて圧力値の分布が表示されている．また，今回可視化する立方体領域は，解像度 5 のボリュームデータであり，合計 $8^5 = 32768$ 要素を持つ．ただし，ここで示す可視化の際には解像度を下げた状態で扱うこともある．例えば，解像度 4 のデータして扱う際には，Octree の最下層にある子ノード 8 個ずつの平均を取ったものを親ノードの値とし，親ノード 8^4 個を対象とする．

まずはじめに提案手法との比較を行うために，簡単に既存手法であるボリュームレンダリングとアニメーションによる可視化，自己相似図形を用いた 3 次元データの 2 次元化とアニメーションによる可視化による結果を確認する．さらにその後で，提案手法によって可視化を行い，既存手法との比較を行う．

7.1.1　3 次元空間表示とアニメーションによる可視化

時系列ボリュームデータの可視化方法として，一般的に利用されているボリュームレンダリングとアニメーションによる可視化の結果例を示す．各時間において空間解像度 5 の対象領域をボリュームレンダリングで 3 次元表示したものをアニメーションとして連続表示することで時系列ボリュームデータの可視化が行える．アニメーションから一部の時間を取り出したものを Fig.7.2 に挙げる．赤，緑，青軸の方向がそれぞれ x,y,z 方向に対応している．Fig.7.2 では多解像度化処理は施していない．

ボリュームレンダリングとアニメーションを利用して可視化した場合は，空間的な形状変化を確認しながら，ボリュームデータを見ることができる．しかし，一方で空間的遮蔽や視点依存性，さらにはアニメーションを利用して表示することによる時間的遮蔽が生じてしまう．これらの問題を解消することが本

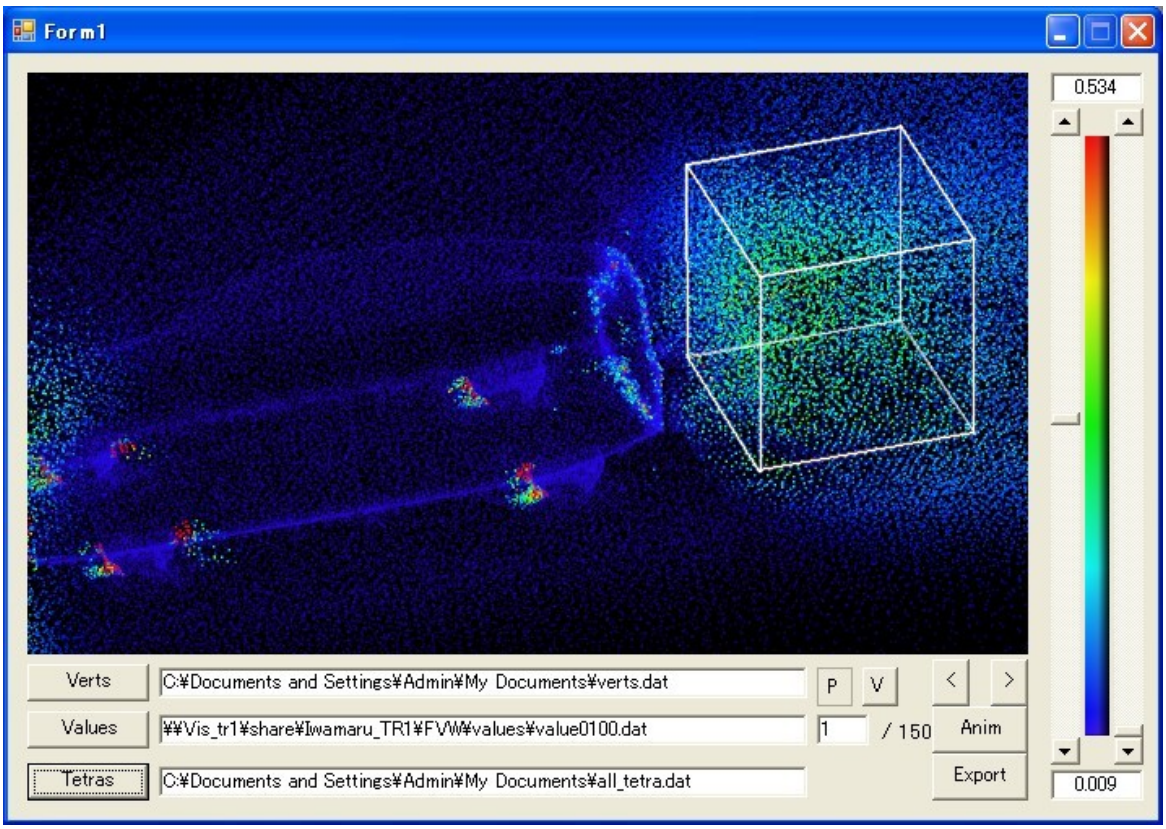


Fig. 7.1: Visualization target of our method[34].

研究の目的の 1 つであった．次の例からは，自己相似図形を用いて低次元化を施した可視化結果を確認する．

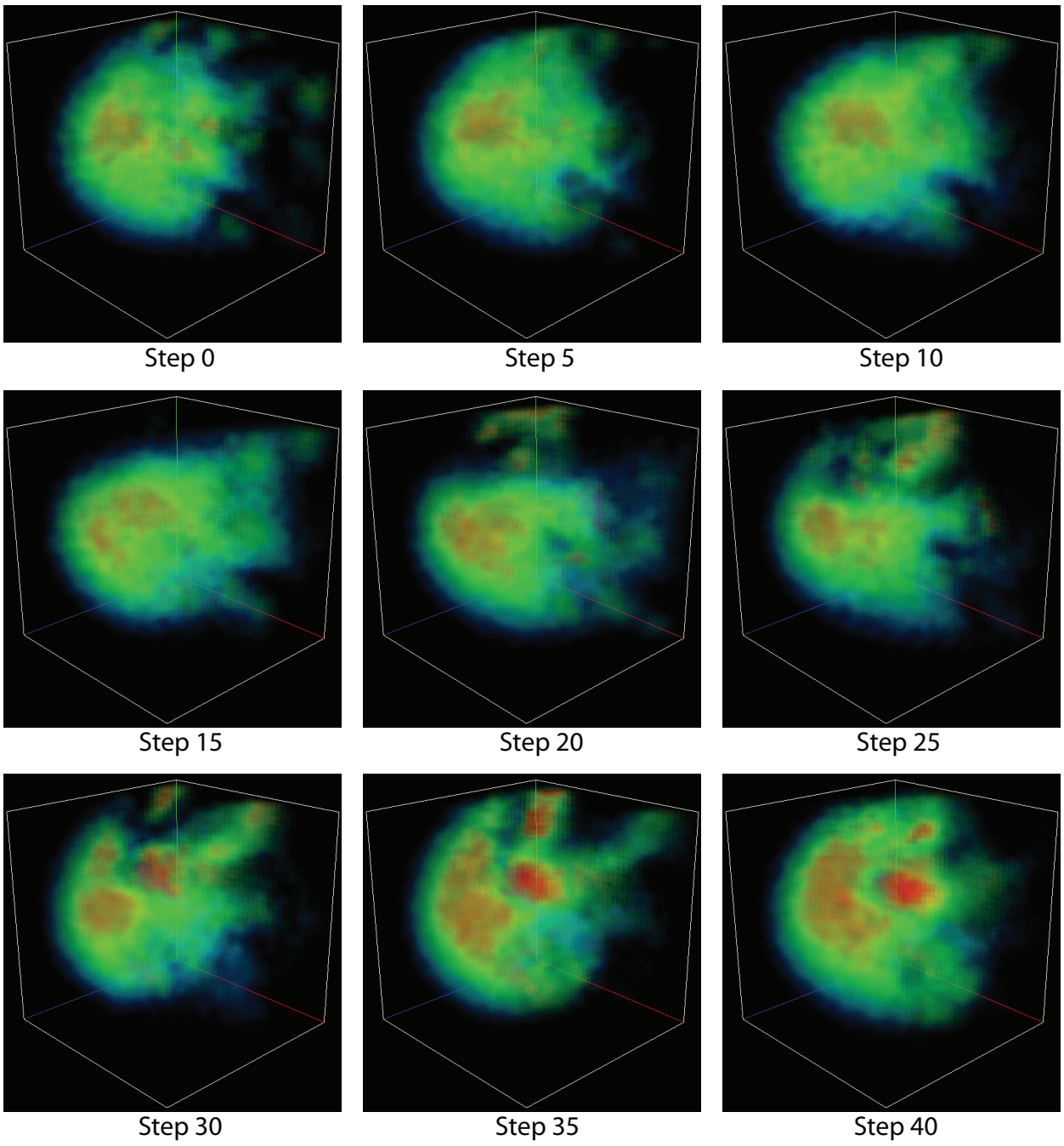


Fig. 7.2: Visualization results of time-varying volume data with volume rendering method and animation.

7.1.2　3 次元空間の 2 次元化表示とアニメーションによる可視化

提案手法に内包されている岩丸らの既存手法 [34] を用いることで，3 次元空間を低次元化して 2 次元化し，その 2 次元図形のアニメーションにより時系列ボリュームデータを可視化できる．ここでは，空間解像度 5 の領域を Fig.7.3 のルールを用いて 2 次元化することを選択した．Fig.7.3 の 3 つの値はそれぞれ，x,y,z の座標値を現しており，解像度 1 のときの座標が小さいほうを 0，大きい方を 1 としている．このルールでは，z 値が大きいところは左上，小さいところは右下にに，y 値が大きいところは右上に，小さいところは左下に配置されるようになる．また，原点からの距離が遠いものほど右上に配置される．さらに，3 次元上で中心に対して点対称な位置にある領域の組は，2 次元上でもブランクスペースを対称の中心にした点対称な位置に配置される．既存手法を用いて 2 次元化するメリットとして，3 次元空間をそのままボリュームレンダリング等によって可視化したものに含まれていた空間的な遮蔽や視点依存性を解消することができる．ただし，この可視化方法では前小節 7.1.1 と同じくアニメーションを利用して時系列変化を表示するため，ある時間のデータを見るにはその時間にならなければ見ることができないといった時間的な遮蔽性が含まれたままになってしまっている．

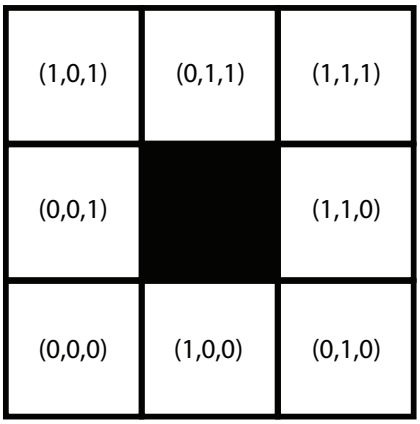


Fig. 7.3: Mapping rule for 3D space to 2D image.

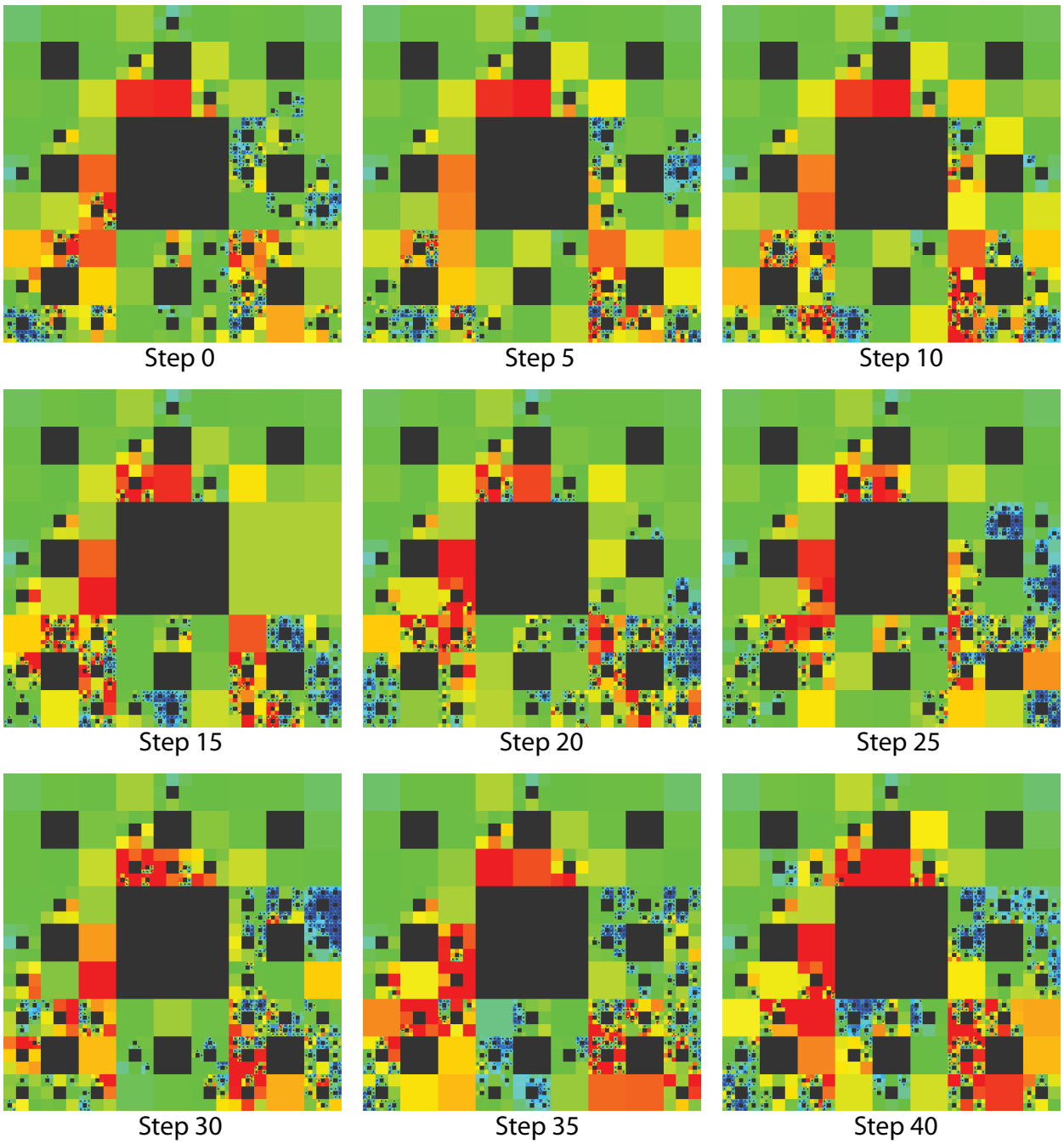


Fig. 7.4: Visualization results of time-varying volume data with 2D expanding method of 3D space and animation.

7.1.3　4 次元を同等に扱った 2 次元化可視化

提案手法を用いることで時系列ボリュームデータの 4 次元すべてを同等に扱って 2 次元化して可視化することが可能である．時間次元も含めて 2 次元化されることによって前小節 7.1.1, 7.1.2 で問題となっていた時間的遮蔽を解消できる．

今回用いるマッピングルールは，Fig.7.5 に載せたものを用いる．このマッピングルールは，6 章で紹介した「着目次元の座標値に基づくルール」かつ「組み合わせ自己相似図形の性質に基づくルール」かつ「点対称性を維持するルール」かつある程度の「原点からの距離関係を反映したルール」になっている (Fig.7.6)．2 次元空間がたたみ込まれて，2 次元空間の各点に存在しており，たたみ込まれている 2 次元空間では，横軸方向で x 方向の変化，縦軸方向で y 方向の変化が表されており，2 次元空間を含んでいる 2 次元空間では，横軸方向で t 方向の変化，縦軸方向で z 方向の変化が表されている．このルールが解像度 2 になった場合には，Fig.7.7 のようになる．Fig.7.7 の各領域に書かれてある 4 桁の数値は (x,y,z,t) 座標値を示しており，色はその t 座標値の大きさに基づいてつけられている．

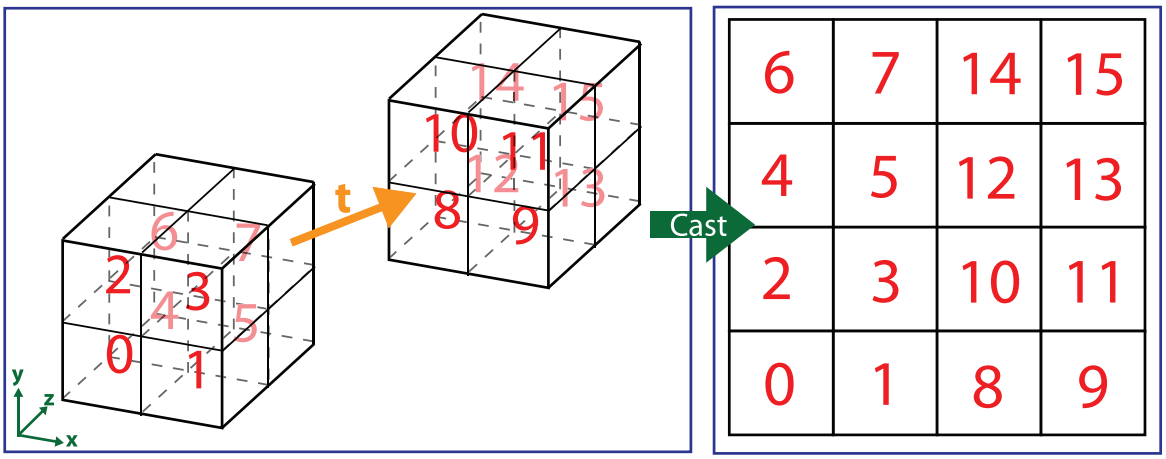


Fig. 7.5: A mapping rule for time-varying volume data.

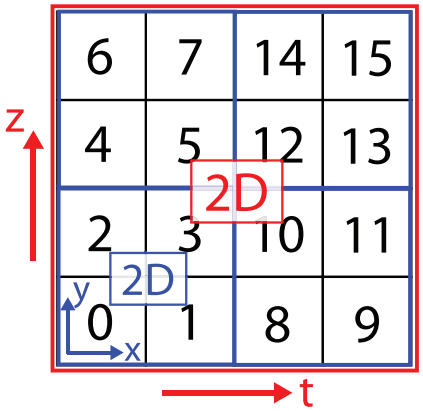


Fig. 7.6: Character of 4D mapping rule. 2D space folded in 2D points.

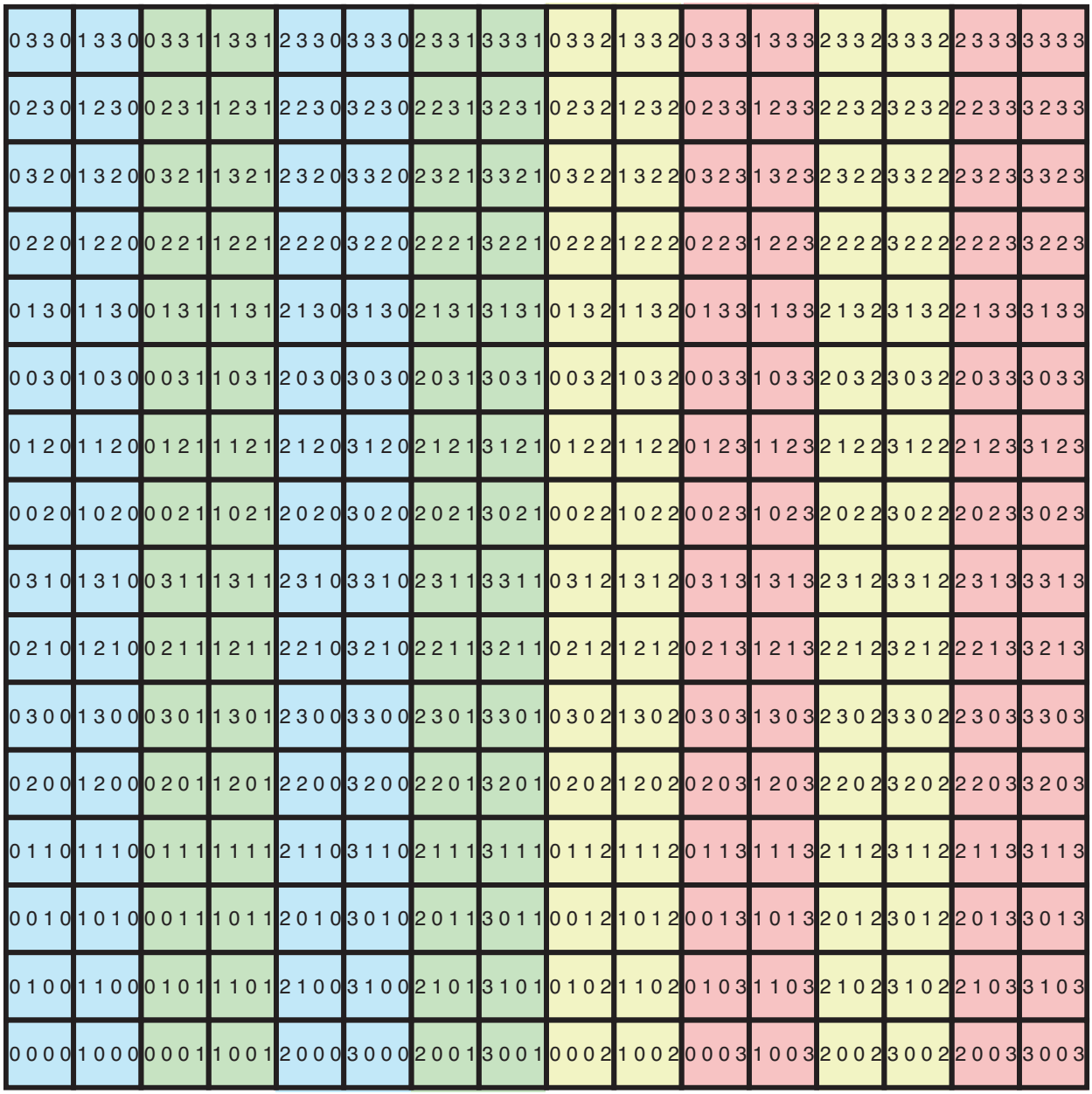


Fig. 7.7: A mapping rule for resolution 2. Regions are colored by their t coordinate value.

このルールを用いて x,y,z,t 方向すべて 16 要素分の領域 (空間解像度 4 のものが 16 ステップ分変化したもの) を可視化した例が，Fig.7.8 の右側のウィンドウで表示されてある結果である．この例では，提案手法による可視化結果と並んで，左上側のウィンドウで時系列ボリュームデータの各 16 ステップにおける状態をボリュームレンダリングを可視化している.16 ステップの並べ方は図中に $t = a(a : 0,1,\cdots,15)$ の形で示してある．また，この可視化例では，右側の可視化結果の領域にマウスポインタを置くと，マウスポインタが指す位置にある領域と同じステップ間のボリュームレンダリングによる結果が黄色の枠で囲まれ，同じ空間・ステップ間にあるところはピンクの枠で囲まれる．さらに，マウスをクリックすることで現在黄色の枠で囲まれている領域だけをズームアップして表示することができる (Fig.7.9)．このように

既存の 3 次元可視化手法と提案手法を連携させることで，提案手法だけでは把握しにくい空間的な位置を既存手法の可視化結果で確認しながら，データを観察することが可能になる．既存手法の持つ空間的遮蔽と時間的遮蔽の問題は提案手法によって解決し，提案手法の持つ空間形状が把握しにくと行った問題は既存手法で解決するといった相互補完的な可視化が行われている．さらに，左下に用意されてある 3 つのスライドバーを利用することで可視化結果を変化させることができる．一番上の白いスライドバーの位置を変えると高解像度な箇所と低解像度な箇所の割合を操作することができる．右にずらすと低解像度な箇所が増え，左にずらすと高解像度な箇所が増える．Fig.7.10 は，Fig.7.8 の状態からスライドバーを左にずらして高解像度な箇所を増やした結果である．2 番目と 3 番目の赤と青のスライドバーの位置を変えると色の表示の仕方を捜査できる．3.8 節の色を決定する式 3.8.1 で利用される v_{max} が赤色のスライドバーで操作され，v_{min} が青色のスライドバード操作される．赤，青のスライドバーを右に動かすとそれぞれ v_{max}, v_{min} の値が増え，左に動かすと v_{max}, v_{min} の値が減る．$v_{max} - v_{min}$ の値を小さくなるようにスライドバーの位置を変化させた結果を Fig.7.11 に載せている．

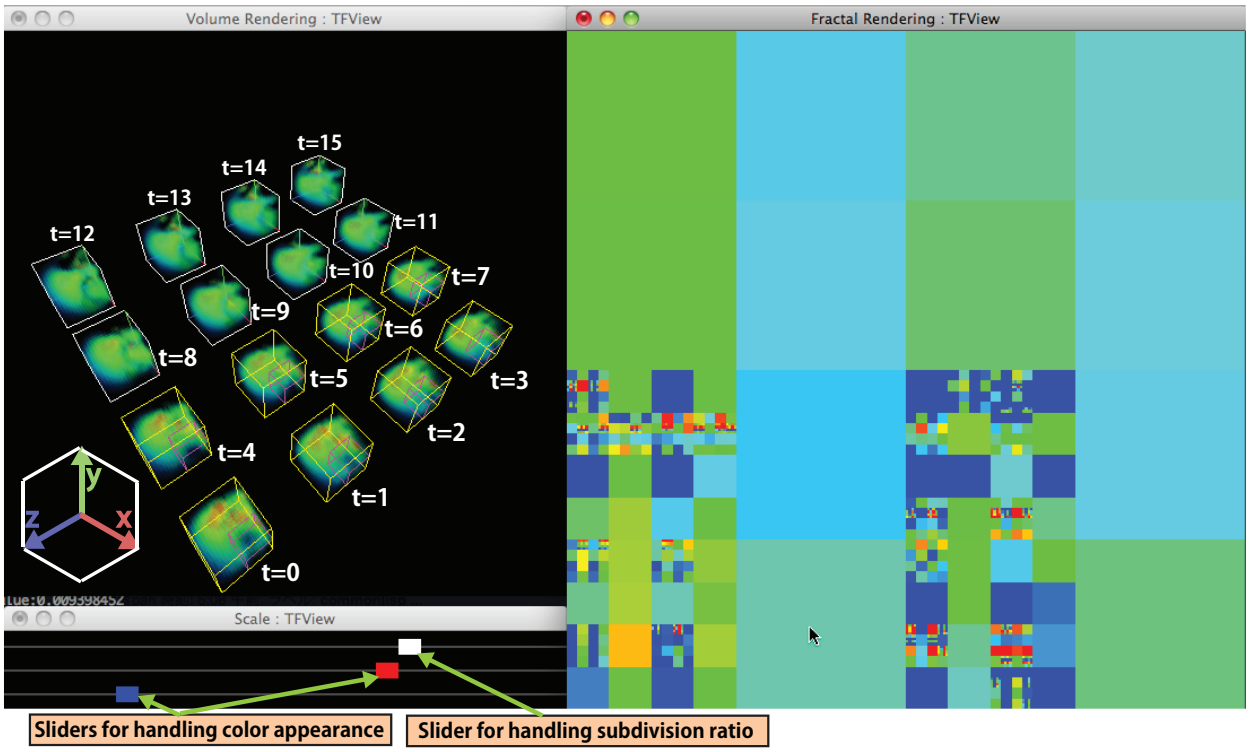


Fig. 7.8: Visualization application example of time-varing volume data as 2D image.

これらの結果から観察できるデータの特徴例を Fig.7.12 の中で示しており，各英字 (a) から (d) の領域における特徴として，

(a)　z 座標が大きな領域ではデータの値が小さなところが多く，分散が少ない．
(b)　y 座標が大きな領域でもデータの値が小さなところが多く分散が少ない．
(c)　z 座標が大きな領域でも一部大きな値を取るところがあり，そこでは値の分散が大きい．
(d)　x 座標が小，y 座標が大，z 座標が小の領域では値の分散が全体的に大きい．

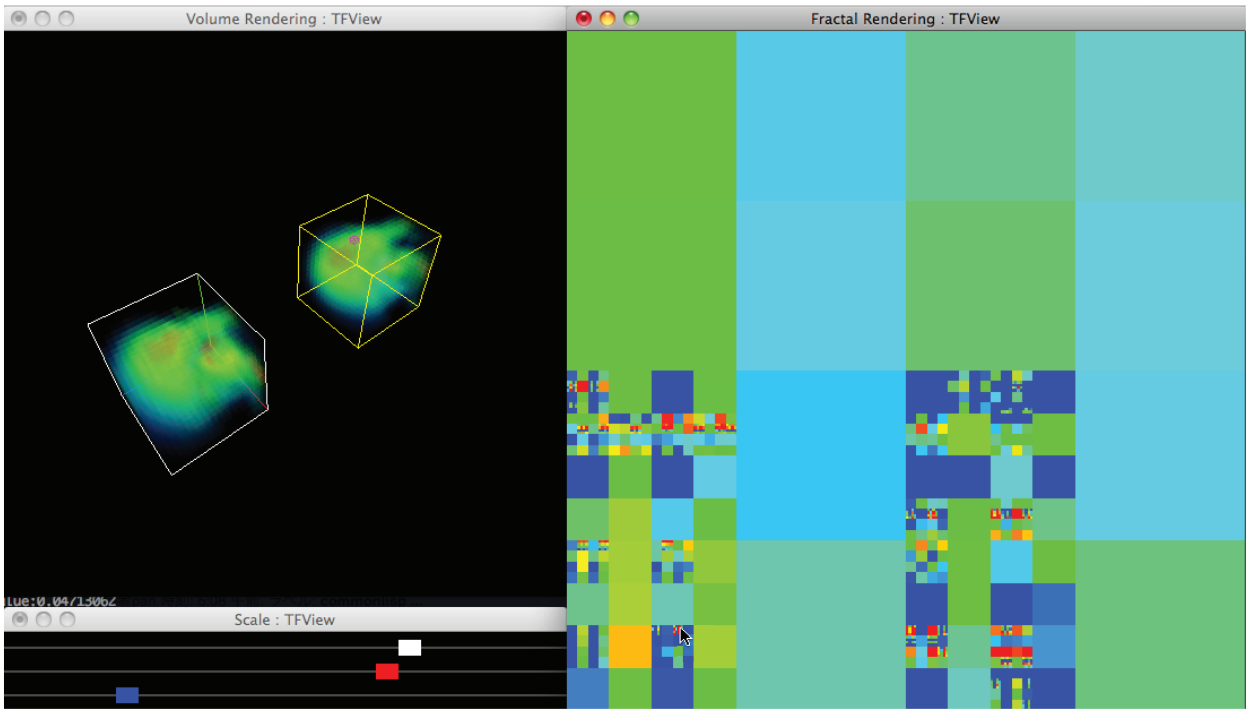


Fig. 7.9: Result with zoom up handling.

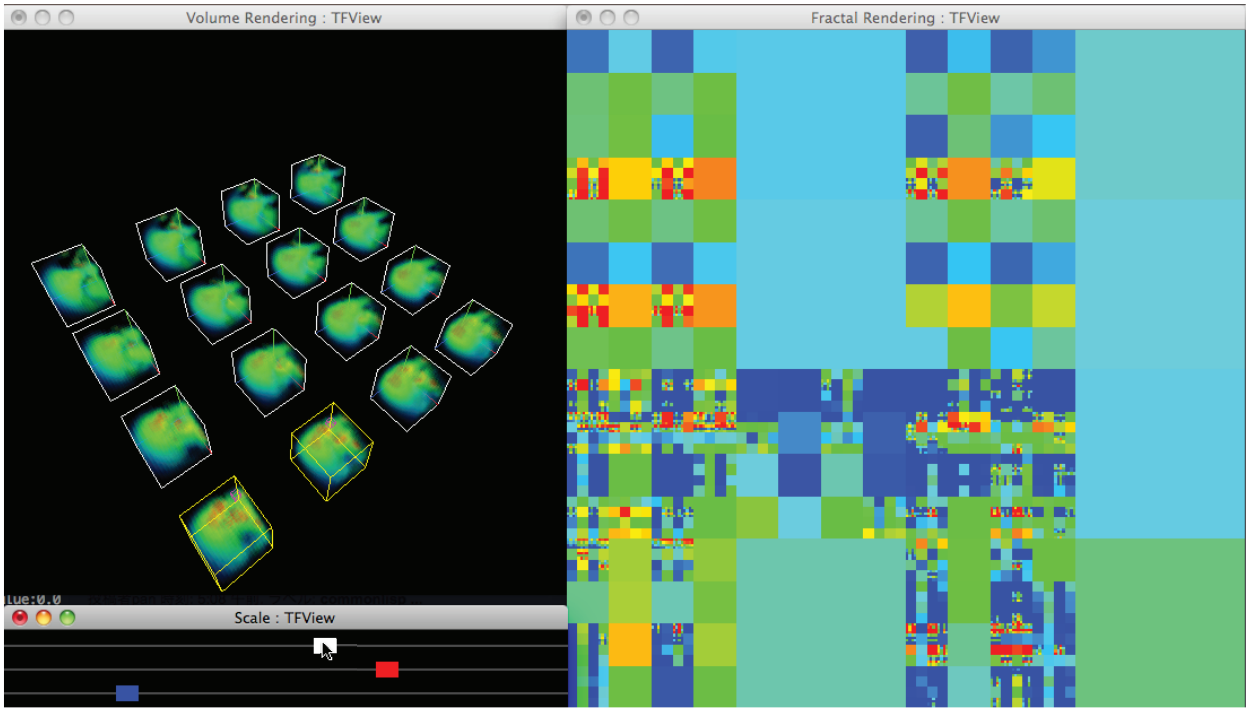


Fig. 7.10: Result by handling subdivision ratio with slider.

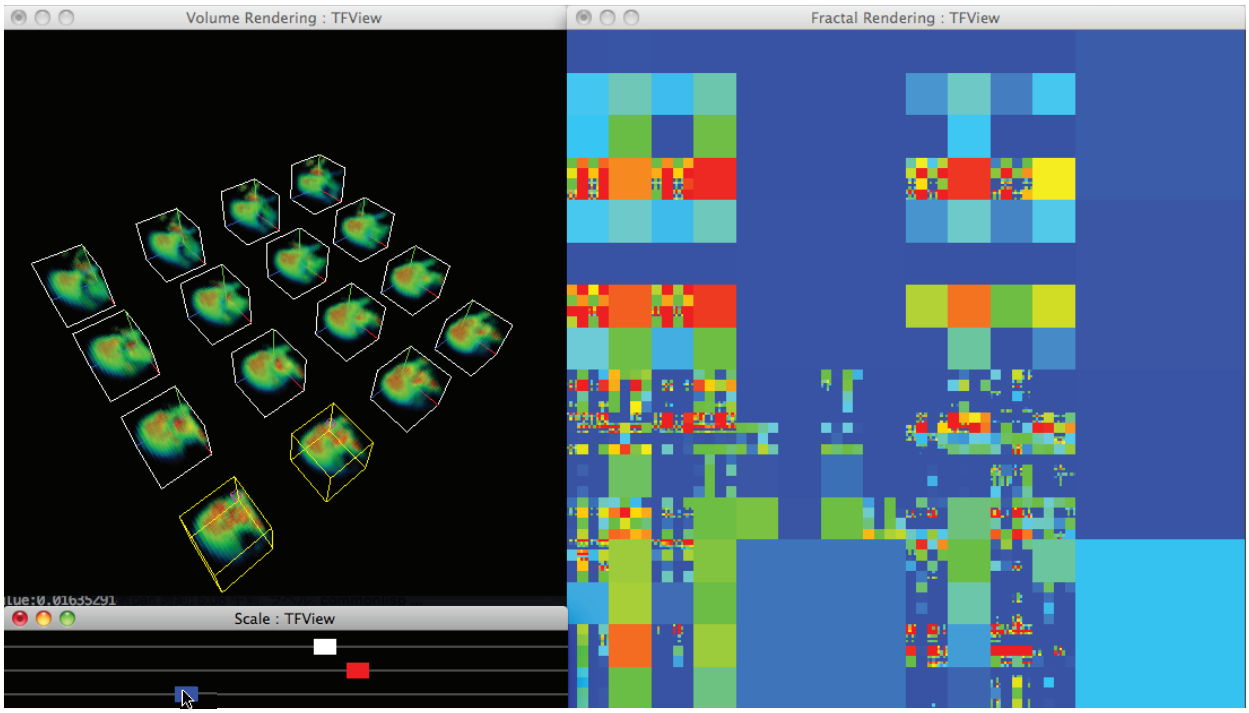


Fig. 7.11: Result by handling color appearance with slider.

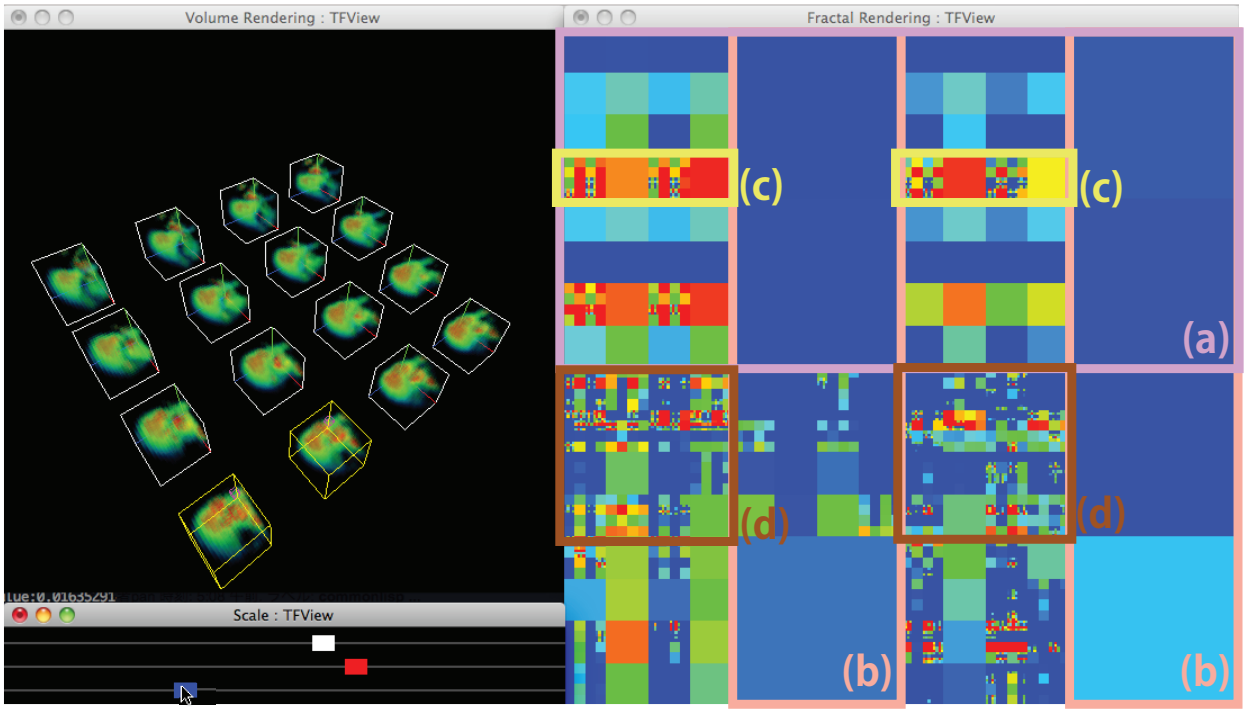


Fig. 7.12: Features of the time-varying pressure data.

などといったことが挙げられる．このようにして時系列ボリュームデータの全体的な特徴の把握を容易に行うことができる，これは，提案手法による可視化結果には空間的遮蔽・時間的遮蔽が含まれないなどといった特徴による．

しかし，この空間次元と時間次元を混ぜて 2 次元化した方法には問題がいくつか含まれる．その 1 つは，同じ空間領域における時間変化の結果が離れた箇所に表示されるため，各空間領域における時間変化を把握しにくいことである．また他にも，空間次元と時間次元を混ぜて多解像度化を行うため，その領域が高解像度の場合に空間次元方向においてデータの値の変化が大きいのか，時間次元方向において変化が大きいのか，それとも両方の次元方向において変化が大きいのか区別できない．低解像度な領域では，その逆でどの次元方向において変化が小さいのかが区別できない．また，次元を混ぜて次元変換しているため，x,y,z,t がすべて同じ解像度を持たなければならない (今回の可視化では解像度 4)．そのため，時間変化だけ細かく変化を見るといったようなことができない．

次小節では，これらの問題点を解決した次元別の次元変換による可視化の結果を示す．

7.1.4　3 次元空間と 1 次元時間を別々に扱った 2 次元化，3 次元化可視化

前小節では，提案手法によって時系列ボリュームデータを 3 次元空間と 1 次元空間を混ぜて 4 次元データとして扱い，2 次元化することで空間的遮蔽と時間的遮蔽を解決した可視化を行ったが，時間変化の把握が困難などといった問題点が存在した．本小節では，空間次元と時間次元で次元別に次元変換を行うことで，それらの問題を解決した可視化を行う．

時系列ボリュームデータの空間次元 3 次元を 1 種類目の次元群 S として，時間次元 1 次元を 2 種類目の次元群 T として 2 種類の次元群からなるデータとして扱う．扱う時系列ボリュームデータが空間次元群 S は解像度 R_S を持ち，時間次元群 T は解像度 R_T を持つとする．このデータを空間次元群を 1 次元し，時間次元群方向に並べて 2 次元化，空間次元群を 2 次元化し，時間次元群方向に並べて 3 次元化して表示する．はじめに空間次元群，時間次元群それぞれのための解像度 1 のときのマッピングルールを定める．ここでは，2 次元化用のマッピングルールは Fig.7.13 のように，3 次元化用のマッピングルールは Fig.7.14 のように定める．空間次元群のためのマッピングルールを R_S 回，時間次元群のためのマッピングルールを R_T 回再帰的に適用することで，すべての要素をマッピングすることができるようになる．

多解像度化を行うには，空間エラーと時間エラーを求める必要があるが，これは 5.4.1 小節で示した方法で求める．

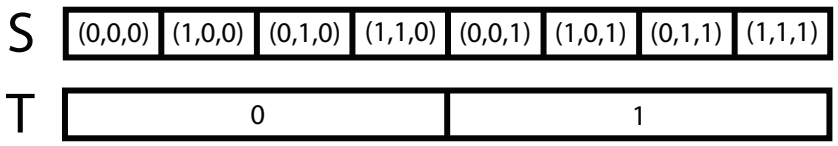


Fig. 7.13: 1D mapping rules for 3D space and 1D time.

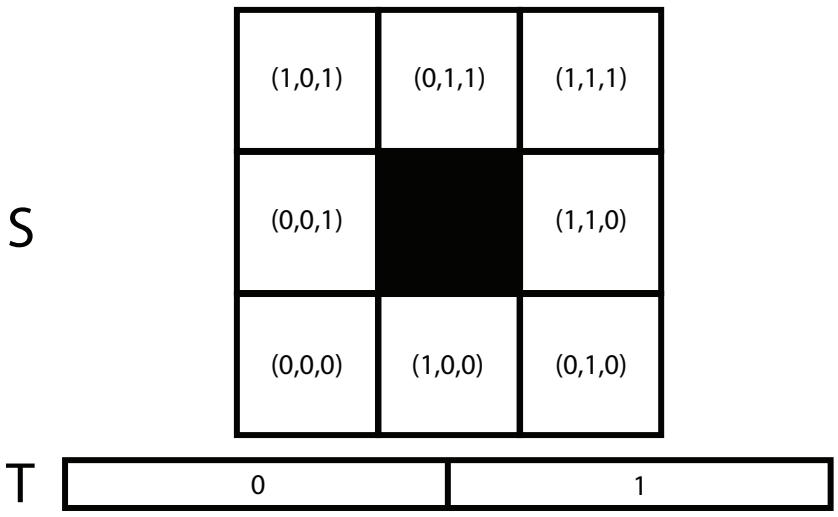


Fig. 7.14: 2D and 1D mapping rules for 3D space and 1D time.

次元別 2 次元化可視化例

まず，2 次元化して可視化した例を示す．ここでは，空間解像度 3 のデータが時間方向に 128 ステップ移動したものを可視化した (合計 65536 要素数)．多解像度化の際のエラーに対する閾値として，空間次元群は 0.2，時間次元群は 0.05 を定めて可視化を行った結果が Fig.7.15 である．可視化の際には，多解像度化により表示される領域数が約 4 分の 1 の 16542 領域に減少されている．Fig.7.15 では，横軸方向 (S 方向) を空間次元方向を表すことに，縦軸方向 (T 方向) を時間次元方向を表すことに用いている．マッピングルールとして，Fig.7.13 を用いているため，Fig.7.15 では右に行くほど z の値が大きい領域になり，さらに z 値で分けられる各領域の中において右に行くほど y の値が大きくなる．さらに，y 値で分けられる各領域の中でも右に行くほど x の値が大きくなっている．各位置における x,y,z の値を把握しやすくするため，可視化像の上部にその S 方向の位置における x,y,z の値の大きさを青から赤の 8 段階の色で表示している (それぞれ x,y,z 値 0 から 7 に対応する)．この結果から観察できるデータの特徴例を Fig.7.16 の中で示しており，各英字 (a) から (d) の領域における特徴として，

(a)　z 座標が大きな領域ではデータの値が小さなところが多く，分散が少ない．
(b)　y 座標が大きな領域でもデータの値が小さなところが多く分散が少ない．
(c)　z 座標が大きな領域でも一部大きな値を取るところがあり，そこでは他の z 座標が大きい領域に比べ値の分散が大きい．
(d)　x 座標が小，y 座標が大，z 座標が小の領域では値の分散が全体的に大きい．

などといったことがまず挙げられる．これらの特徴は，前小節の例 Fig.7.12 と同じく観察できる (前小節は 16 ステップ分を可視化していたので，Fig.7.16 中の下部 8 分の 1 の領域と対応する)．また Fig.7.12 と Fig.7.16 とを比べると，Fig.7.16 のほうが時間方向の変化が連続して表示されているため，これらの特徴をすぐに把握することができる．今回の可視化では，次元別に次元変換をしているため，空間次元と時間次元の解像度が大きく違っていても可視化できるため，時間方向の変化をより多いステップ分観察できる．そのため，時間変化の連続表示ができるという特徴と合わせると Fig.7.12 からは明確には分からなかった特徴，

(b)　y 座標が大きな領域では圧力が開始時よりも大幅に低くなる箇所が存在する．特に (b') の領域においては，圧力の増減が繰り返し見られ振動に似た挙動が現れている．
(d)　x 座標が小，y 座標が大，z 座標が小の領域では，(b') の領域と同様に圧力の増減の繰り返しが見られ，すでに述べたこの領域では値の分散が大きいことも合わせて考えると，この領域では激しい圧力変動が起こっている．
(e)　x 座標が小，y 座標が小，z 座標が小の領域では，初めの方のステップにおいて，値の変動が (d) と比べ少ないが，時間が経つと (d) の領域と似た挙動を示す．
(f)　z 座標が大きい領域は値が緑で表される中間的な値の圧力である箇所が多いが，(f) 領域は常に圧力値が高い状態であり続けている．この傾向は (c) 領域の一部でも見られる．

が見られることが分かる．また，領域の表す圧力値以外に次のような値も把握できる．

(g)　z 値が 2 から 3，y 値が 2 から 3 を持つ領域で，x 値 0 から 1 の領域から 2 から 3 の領域へ変化している箇所などでは，同じ圧力値であると考えられる箇所が右上がりに移動していることが見て取れる．この右上がりの傾きによって x 方向の圧力移動の速度も把握できる (傾きが急であるほど速度は小さい)．同様に z 値が 2 から 3，x 値が 0 から 3 を持つ領域で，y 値が 2 から 3 の領域から 4 から 5 の領域へ変化している箇所や，y 値が 0 から 4，x 値が 0 から 4 の領域で，z 値が 0 から 1 の領域から 2 から 3 への領域へ変化している箇所でも右上がりの移動が見られるところがあり，その傾きによって速度が把握できるため，速度の値もある程度確認することができる．このように，同じパターンになっていると考えられる箇所を比較することで，速度の情報を得ることができる．

このようにして，次元別に次元変換による 2 次元化を行うことで，時間変化を理解しやすい形で可視化が行える，空間次元と時間次元を別々に扱って多解像度化でき，その領域が持つ空間的な分散と時間的な分散を別々に把握できる，空間次元と時間次元の解像度が異なる形で可視化できる，速度もある程度把握できる，といった特徴を持ち，複数の特徴や値を一画像から把握できるようになる．そのため，時間変化を把握しやすい形で全体俯瞰可能な可視化像を作成することができる．

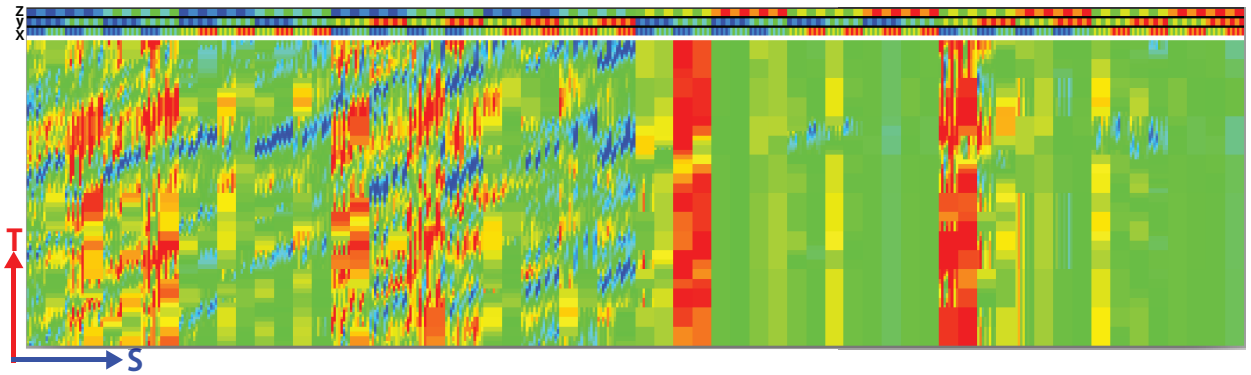


Fig. 7.15: 2D visualization result. Space dimension direction is displayed as 1D in horizontal axis. Time dimension direction is displayed as 1D in vertical axis.

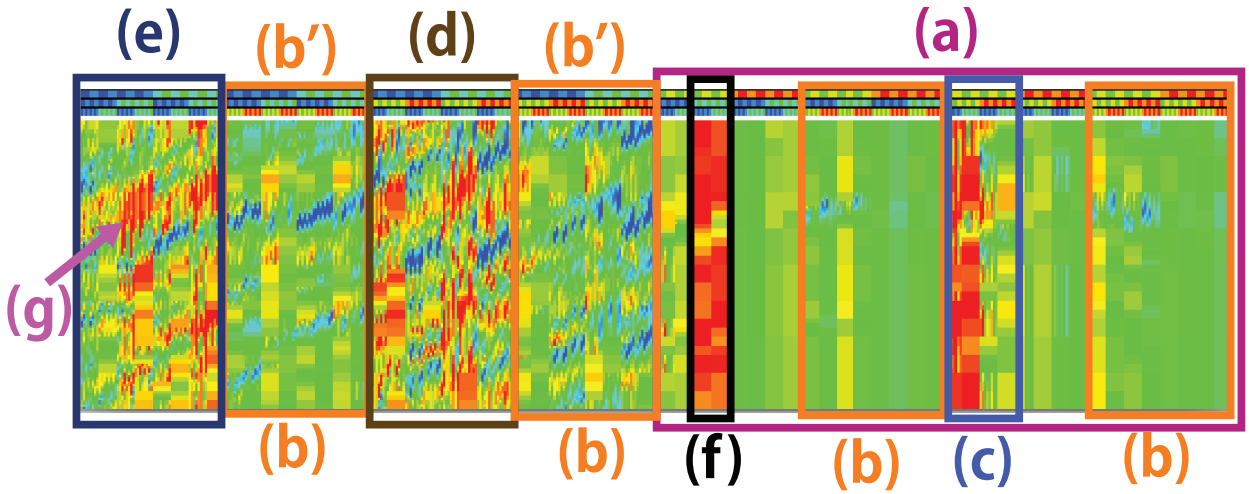


Fig. 7.16: 2D visualization result and the regions which have some features are sorrounded with colored rectangles.

次元別 3 次元化可視化

次に，3 次元化可視化例を示す．可視化するデータの解像度および多解像度の際に用いるエラーに対する閾値は 2 次元化可視化のときと同じものを使う．またマッピングルールは，すでに Fig.7.14 で示したものを利用する．すなわち，空間解像度 3 のデータが時間方向に 128 ステップ移動したものを，空間次元群は 0.2，時間次元群は 0.05 の閾値を利用して多解像度化して可視化を行った．その結果を Fig.7.17 から Fig.7.20 に載せている．シェルピンスキーのカーペットの枠線が描かれてある面で空間次元の変化を表し，シェルピンスキーのカーペットから手前方向に伸びる青色の軸方向で時間次元の変化を表している．Fig.7.17, 7.18, 7.19 は同じデータをデータの値に対する色を決定する式における上限値・下限値を異なる値に取ったもので．それぞれ上限値を高く・下限値を低く，上限値を高く・下限値を高く，上限値を低く・下限値を低く設定してある．また，Fig.7.17, 7.18 では値の大きい部分の色が，Fig.7.19 では値の小さい部分の色がはっきり見えるように値の低い部分ほどアルファ値が高くなるように伝達関数を設定して可視化を行った．Fig.7.20 は，(a), (b), (c) にそれぞれ Fig.7.17, 7.18, 7.19 を時間軸の正の方向からの角度で表示したものである．

これらの結果から，2 次元化可視化を行った場合と同様の特徴を観察することができるが，色の調整や結果像を見る角度を変化させなければ特徴を見つけるのは困難である．これは，ボリュームレンダリングによる 3 次元可視化を行うとデータの値を表す際に透明度を利用した表現になり，空間的遮蔽や視点依存性，各領域における単独のデータの値が見ることができないと行った問題があるからである．その他にも，この可視化方法には描画負荷が大きいといった問題もある．しかし，3 次元可視化のほうが 2 次元可視化よりも良い点もある．1 つは，一般的に 3 次元可視化を用いると，奥行き方向にもデータを置くことができるため表示できるデータポイント数が 2 次元可視化に比べて多くなるということである．2 つ目は，3 次元空間を 1 次元化するよりも 2 次元化したほうがもとの空間的な特徴を多く維持したマッピングルールを選ぶことができることである．

次元別次元変換の 2 次元可視化・3 次元可視化どちらにも一長一短があるため，可視化対象によって使い分けて利用するべきである．

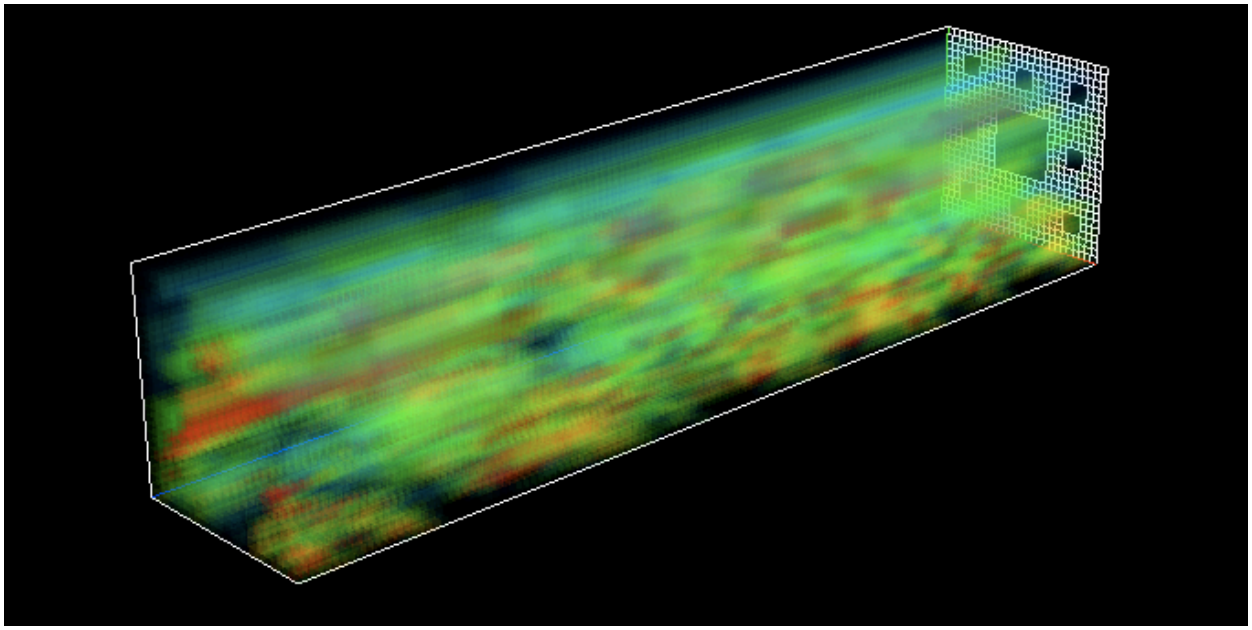


Fig. 7.17: 3D visualization result. Space dimension direction is displayed as 2D. Time dimension direction is displayed as 1D.

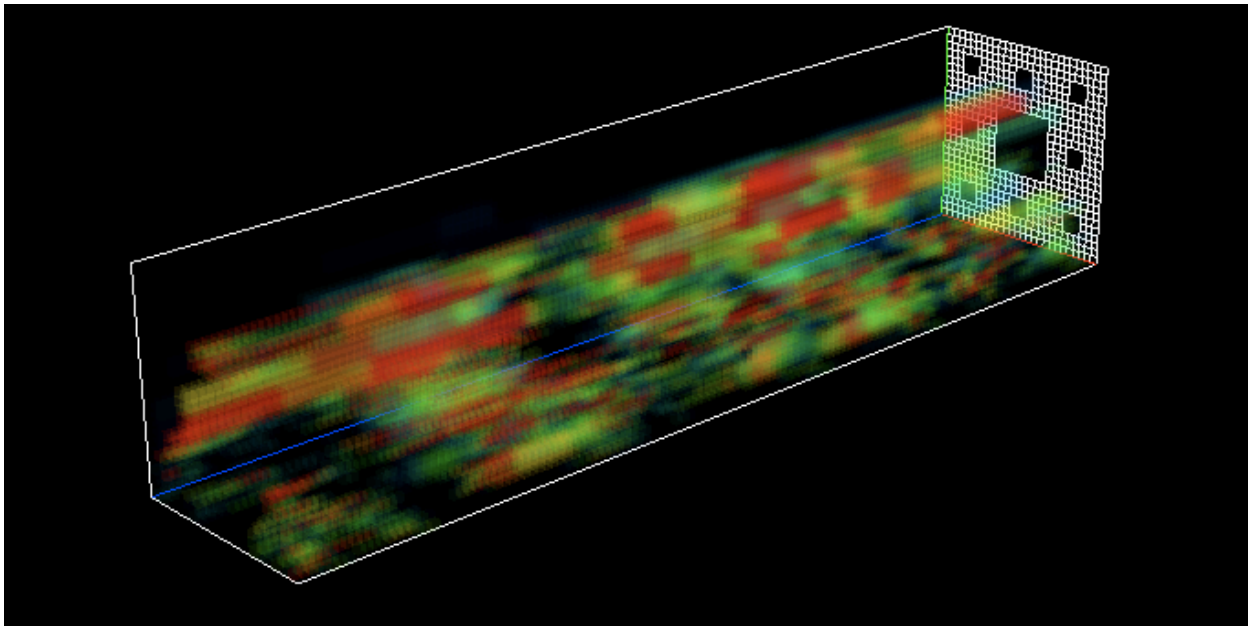


Fig. 7.18: The result highlighted with high pressure part.

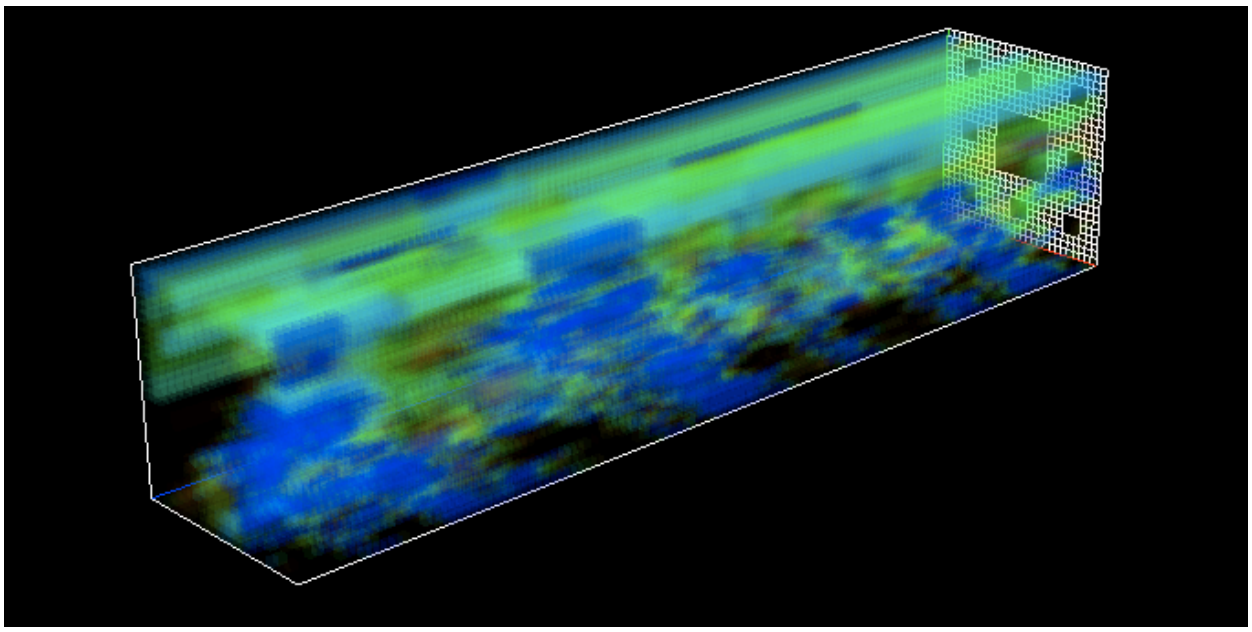


Fig. 7.19: The result highlighted with low pressure part.

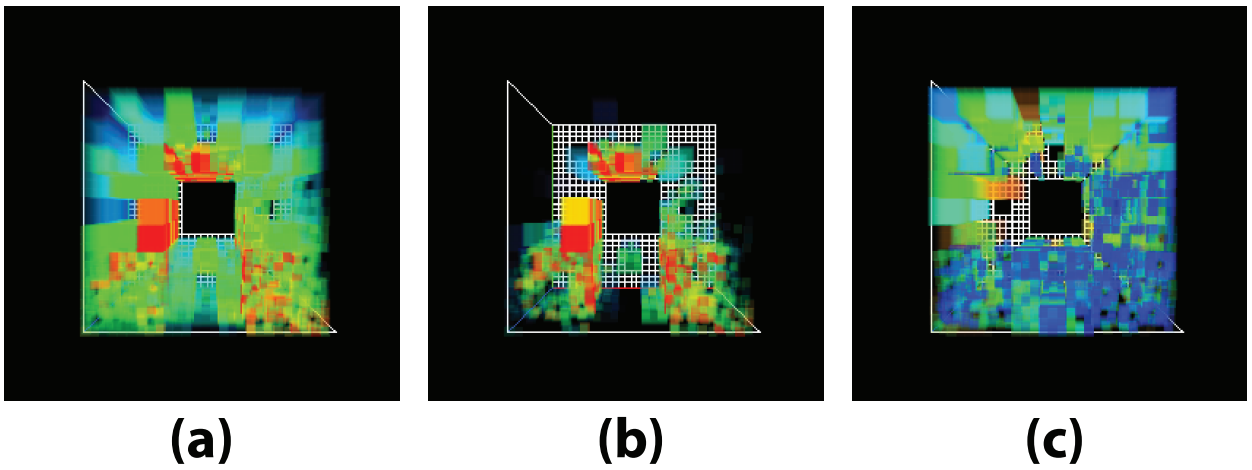


Fig. 7.20: The front faces of the results. Each face corresponds with (a) Fig.7.17, (b) Fig.7.18, (c) Fig.7.19.

7.1.5　各手法の比較

ここでは，今までで取り上げた時系列ボリュームデータの可視化手法の比較を行う．各手法の比較をまとめたものを Table.7.1 に載せる．Table.7.1 に書かれた内容について詳しく説明する．

形状表示　ボリュームレンダリングでは 3 次元形状を保ったまま表示を行うため，データの形状把握が容易である．他の手法は次元変換を行うため，空間形状が崩れ，形状把握が難しい．

空間的遮蔽　ボリュームレンダリングでは，3 次元空間を 3 次元のまま表示するため，空間的遮蔽の問題が生ずる．他の手法では，3 次元空間を 1, 2 次元化するため空間的遮蔽が解消する．しかし，次元別次元変換による 3 次元化では，2 次元化した空間を時間方向に並べるため再度 3 次元表示となり，他の時間における空間方向を２次元化したデータによって遮蔽されるといった問題が生ずる．

時間的遮蔽　アニメーションを用いる方法では，時間的な遮蔽が存在してしまう．提案手法により時間変化も含めて画像化することで，時間的遮蔽は解消する．しかし，次元別次元変換による 3 次元化では，アニメーションは利用しないものの，他の時間のデータによって表示結果に空間的な遮蔽が生じ，個々の時間変化が把握しにくくなってしまう．

時間変化の把握　アニメーションを用いた可視化や時間次元も含めた低次元化を行った可視化結果であると，時間変化による値の変化の把握が難しい．次元別次元を行うことで，各空間領域における時間変化が分かりやすい形で表示される．

表示可能データ数　3 次元可視化を行うと奥行き方向を用いたデータの表示ができるため，表示可能なデータ数は多い．2 次元可視化の場合は，表示可能なデータ数は少なくなる．また，提案手法では，自己相似図形を用いるが，シェルピンスキーのカーペットのように空の領域が存在するものを利用すると，さらに表示可能なデータ数は減少する．

描画負荷　3 次元可視化では，奥行き方向の値を反映した可視化結果を得るために，透明度などを考慮した高度なレンダリング方法を行う．そのため，描画負荷は大きい．一方で，2 次元可視化では，簡易な方法を用いるため，描画負荷は小さい．

このように，どの手法にも一長一短があるため，可視化対象によってどの手法を用いるのかを選択するべきである．また，提案手法はボリュームレンダリングのような他の手法と連携して用いることができることから，互いに相手の短所を補うことのできる提案手法と他の可視化手法や解析手法を併用することで，より効果的な可視化やデータの把握が可能になると考える．

Table. 7.1: Comparison of visualization methods for time-varying volume data.

	ボリュームレンダリングとアニメーション	空間 2 次元化とアニメーション	全次元をまとめて 2 次元化	次元別次元変換による 2 次元化	次元別次元変換による 3 次元化
形状表示	○	×	×	×	×
空間的遮蔽がない	×	○	○	○	△
時間的遮蔽がない	×	×	○	○	△
時間変化の見易さ	×	×	×	○	○
表示可能データ数	○	×	×	×	○
描画負荷の低さ	×	○	○	○	×

7.2　4 次元以上の空間における拡散現象の可視化

提案手法によって，通常の可視化手法では表示不可能な 4 次元以上の空間現象も可視化可能となることが，提案手法を利用する利点の 1 つである．ここでは，簡単な 4 次元以上の空間内の現象の可視化例として拡散現象の可視化を行う．

7.2.1　拡散方程式に基づく高次元拡散現象

1 つめの拡散現象の例として 5 次元の拡散方程式に基づく拡散現象 (時系列 5 次元データ) を扱い，次元別の次元変換による可視化を扱う．時系列 5 次元データは空間 5 次元，時間 1 次元を併せ持つデータであり，ここでは空間を 2 次元化し，時間 1 次元方向に並べて 3 次元化して表示する．以下の式に，今回計算を行った 5 次元の拡散方程式を載せる．

$$\frac{\partial f}{\partial t} = k\left(\frac{\partial^2 f}{\partial x_1^2} + \frac{\partial^2 f}{\partial x_2^2} + \frac{\partial^2 f}{\partial x_3^2} + \frac{\partial^2 f}{\partial x_4^2} + \frac{\partial^2 f}{\partial x_5^2}\right) \tag{7.2.1}$$

$f(x_1,x_2,x_3,x_4,x_5,t)$ は拡散物質の密度，x_1,x_2,x_3,x_4,x_5 は 5 次元空間の各次元方向の座標，t は時刻，k は拡散係数である．また，初期条件や境界条件等の設定を Fig.7.21 に載せている．空間次元の格子点数は各次元方向に 8 点の合計 8^5 点を取っており，初期条件として座標 (4,4,4,4,4) から距離 1 以内に位置するところは f の値を 1 に，その他の領域では f の値を 0 に設定し，境界条件として空間の最も外側にある 4 次元壁の位置では f の値が 0 にあるようにしてある．可視化範囲はこの $8\times8\times8\times8\times8$ の領域のうちの各座標値が 4 以上の $4\times4\times4\times4\times4$ 領域を可視化する．また時間変化は 32 ステップ分を可視化している．さらに，5 次元空間を 2 次元化する際の展開ルールとしては，Fig.7.22 のものを用いる．この解像度 1 の展開ルールは 5 次元上で原点から離れた距離にあるものほど，2 次元上でも原点から遠い位置に展開されるように決められているものである．図中の数字列はその場所に展開されるもとの 5 次元上での座標位置を示している．

これらの条件，展開ルールのもとで可視化を行った結果が Fig:7.23 である．自己相似図形が描かれている面方向で 5 次元空間を表すのに用いられている．また，時間は奥 (右上) 側に行くほど経過する．また f の値が完全に 0 である箇所はアルファ値を 0 としてして，色が表示されないように調整して可視化している．この図を見ると，徐々に値が原点の箇所から拡散していっている様子が分かる．この可視化結果で特徴的なところとして，5 次元現象を表示しているにも関わらず，2 次元の現象を表示した場合の結果と比べても違和感のない可視化結果が得られているところが挙げられる．これは，5 次元の展開ルールを現象に合わせてうまく決めたことによる (今回は拡散現象であったので距離に基づいてルールを決めた)．また空間を 1 次元化ではなく 2 次元化したため，1 次元化した場合よりも 5 次元の距離関係がある程度維持することができていることも理由の 1 つとして考えられる．より距離関係をうまく可視化する方法として，非放射対称性の自己相似図形を用いるという方法も考えられる．

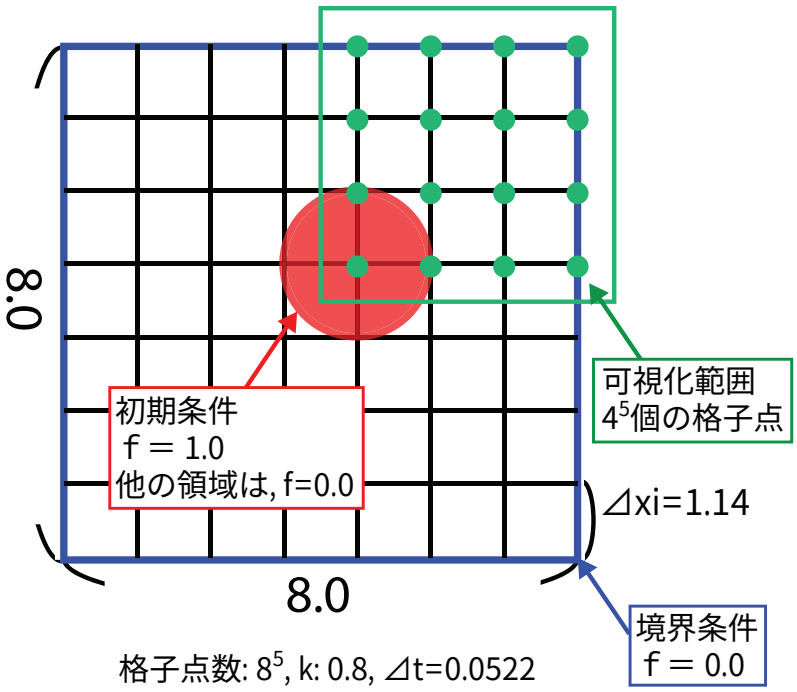


Fig. 7.21: 2D image of the conditions for 5D diffusion simulation.

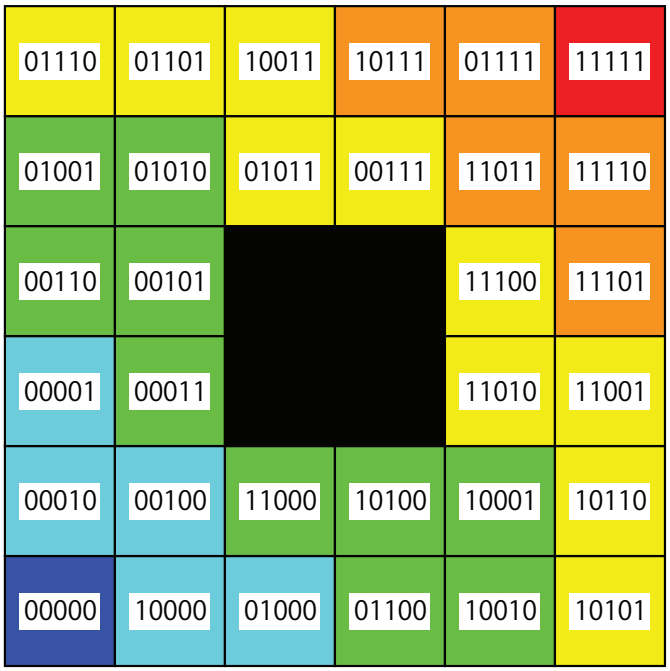


Fig. 7.22: The mapping rule for 5D diffusion simulation. The sequences of number represent their 5D position with resolution 1. The colors represent the distance from the $(0,0,0,0,0)$.

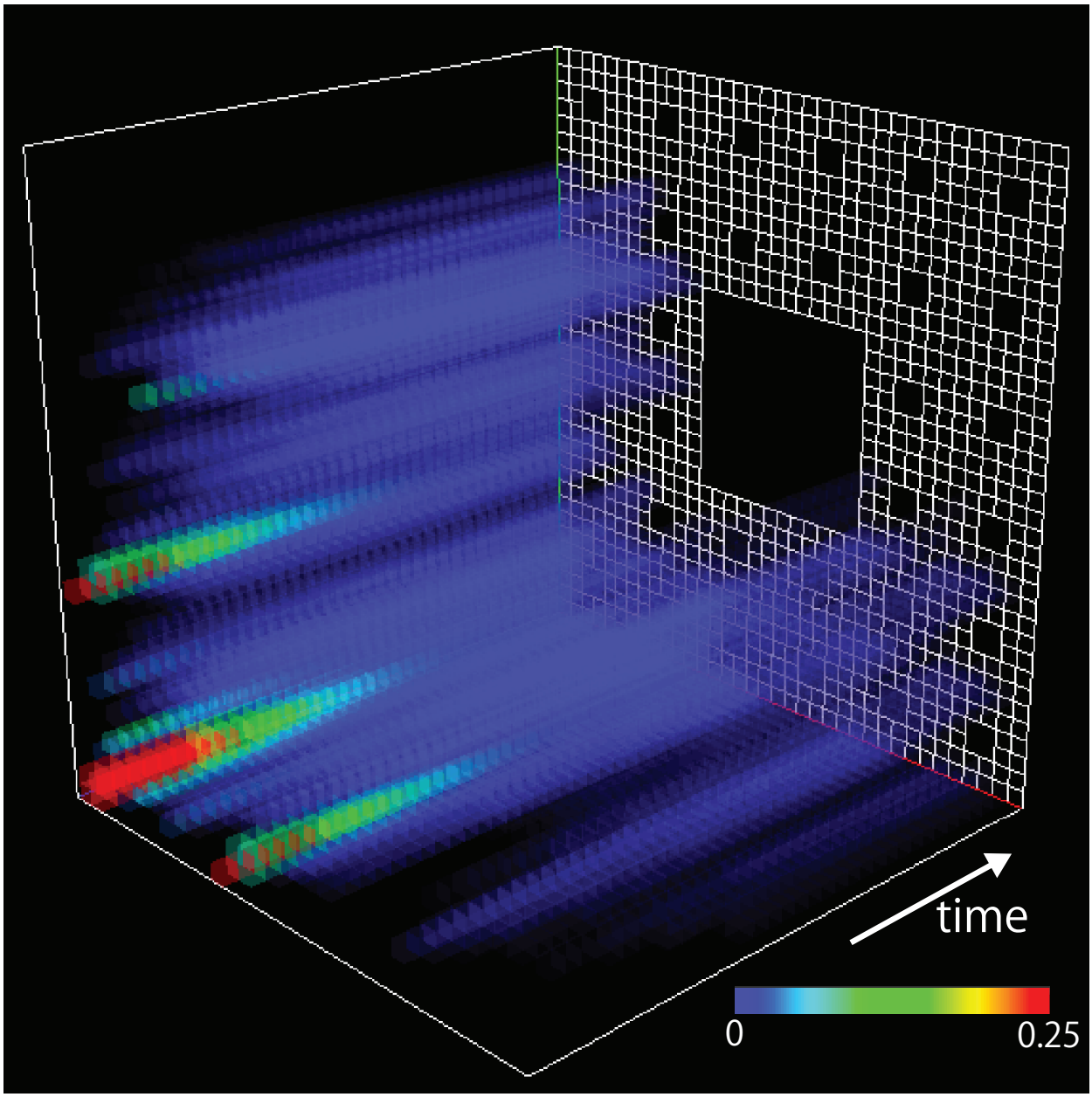


Fig. 7.23: Visualization result of 5D diffusion simulation.

7.2.2　高次元における量子ウォーク

もう一つの高次元現象の可視化事例として，量子ウォークを取り上げる [39]．量子ウォークはランダムウォークの量子版であり，2000 年前後から盛んに研究が行われている．量子ウォークの研究は数式によるものが中心であり，量子ウォークの原理，さらにその原理・数式から導かれる無限大の時刻における確率分布の収束の有無や分布の一様性や局所性などといった容易に計算できることや単純な設定によるものについて議論されている [10]．また，数式による量子ウォークの特性把握は 1 次元空間におけるものが中心ではあるが，高次元空間やツリー・グラフにおけるものなど多岐に及んでいる [22], [24]．一方でシミュレーションによる量子ウォークの研究はあまり行われておらず，1，2 次元空間におけるシミュレーションまではなされているものの [10]，3 次元以上の空間における量子ウォークのシミュレーションはなされていない．さらに，こういった状況から 3 次元以上の空間における量子ウォークの可視化ももちろん行われていない．シミュレーション結果に基づく可視化を行うことで，高次元量子ウォークの全体像・局所的な値の把握，さらにそこから得られる特徴を視覚的に把握することができる．

ここでは，高次元量子ウォークの可視化に本手法が有効であることを示す．そのために，量子ウォークの原理の説明をはじめに行い，複数の簡単な設定のもと 2 次元空間と 4 次元空間における量子ウォークのシミュレーション結果の可視化を行う．さらに 4 次元量子ウォークの可視化結果について，2 次元の場合との比較も行いながら考察を行う．

原理

はじめに，1 次元 (線上) における 2 状態を持つ量子ウォークの場合について説明する．

1 次元の各格子点上に，複素数を成分に持つ長さ 2 のベクトルを考える (Fig.7.24)．時刻 t のときに座標 x にあるこの複素数の行列を $\psi_t(x)$(量子ビットと呼ばれる) と書くこととする．このとき，$\psi_t(x)$ は以下の式 7.2.2 で表される．

$$\psi_t(x) = \begin{bmatrix} \psi_t^R(x) \\ \psi_t^L(x) \end{bmatrix} \in \mathbb{C}^2 \tag{7.2.2}$$

ただし，$\psi_t(x)$ は以下の式 7.2.3 を満たす．

$$\sum_{x=-\infty}^{\infty} ||\psi_t(x)||^2 = \sum_{x=-\infty}^{\infty} \left(|\psi_t^R(x)|^2 + |\psi_t^L(x)|^2 \right) = 1 \tag{7.2.3}$$

この $\psi_t(x)$ は，場所 x における時刻 t の量子ウォーカーの確率振幅を定めるベクトルとなっている．また，$\psi_t^R(x)$ は右向きの量子状態，$\psi_t^L(x)$ は左向きの量子状態に対応する．時刻 t において座標 x に粒子が存在する確率 $P_t(x)$ は，以下の式 7.2.4 ように定義する．

$$P_t(x) = ||\psi_t(x)||^2 = |\psi_t^R(x)|^2 + |\psi_t^L(x)|^2 \tag{7.2.4}$$

さらに，量子ウォークを時間発展させるために利用する 2×2 のユニタリ行列 U を導入する (式 7.2.5)．

$$U = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix} \tag{7.2.5}$$

ただし，$u_{ij}(i,j$ は自然数の添字番号) は複素数である．このユニタリ行列 U の成分を用いて，量子ウォークの時間発展は式 7.2.6 のように定義される．

$$\psi_{t+1}(x) = \begin{bmatrix} \psi^R_{t+1}(x) \\ \psi^L_{t+1}(x) \end{bmatrix} = \begin{bmatrix} u_{11}\psi^R_t(x-1) + u_{12}\psi^L_t(x-1) \\ u_{21}\psi^R_t(x+1) + u_{22}\psi^L_t(x+1) \end{bmatrix} \tag{7.2.6}$$

さらに，量子ウォークの初期条件と境界条件を定める．ここでは，初期条件は原点に φ，他の座標には 0 ベクトルを置くこととする．ただし，φ は以下の式で表される．

$$\varphi = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, ||\varphi||^2 = |\alpha|^2 + |\beta|^2 = 1 \tag{7.2.7}$$

この初期条件によると，はじめの確率の分布は以下のようになる．

$$P_0(0) = 1, P_0(x) = 0 (x \neq 0) \tag{7.2.8}$$

また，境界条件の設定の例として，周期境界条件が挙げられる．

ここまでが，1 次元量子ウォークの原理である．理解を促すために，1 次元量子ウォークの例として，各値を以下の式 7.2.9 に設定した場合の時間変化を Fig.7.24 に載せる (赤字で書かれた数値は確率 $P_t(x)$ を表す).

$$U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \varphi = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix} \tag{7.2.9}$$

1 次元の場合について，ここまでで述べたが，より高次元になった場合も基本的な原理は 1 次元の場合と同様である．1 次元の場合と異なるのは，n 次元になると量子ビットが $\psi_t(x_1, x_2, \cdots, x_n)$ と表され，n 個の座標値で決定されるようになること，また各位置における $\psi_t(x_1, x_2, \cdots, x_n)$ がそれぞれ $2n$ 個の状態を持つこと，ユニタリ行列が $2n \times 2n$ の大きさになることである．例えば，2 次元の場合は，量子ビットは $\psi_t(x,y)$ となり，右方向，左方向，上方向，下方向に対応した 4 つの状態を持つ．さらにユニタリ行列は 4×4 の大きさになる．

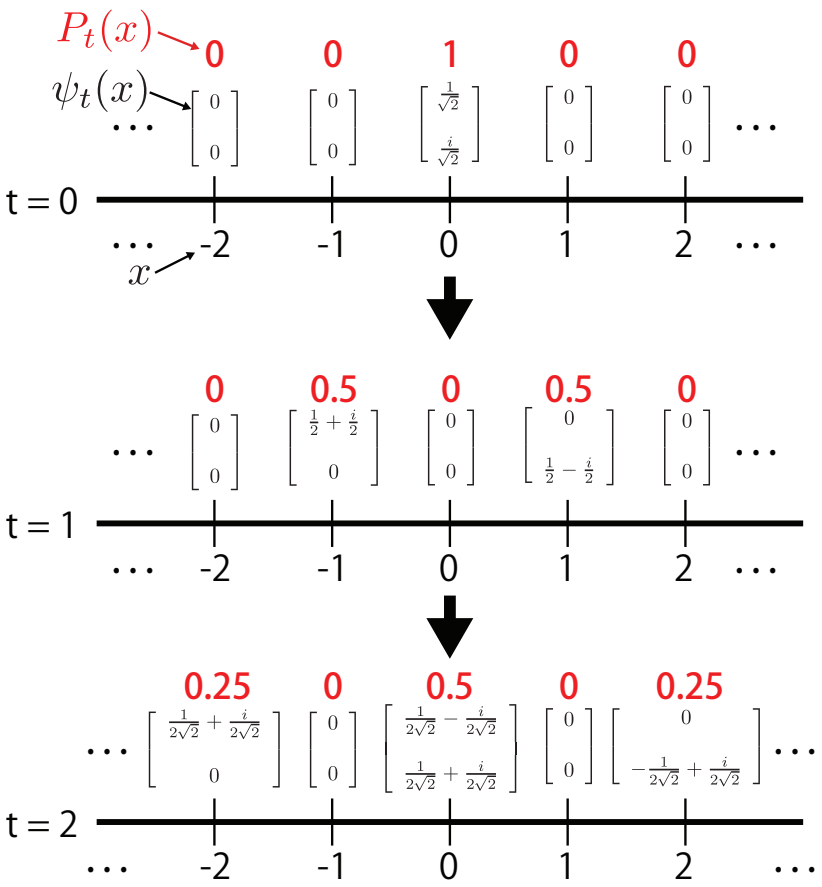


Fig. 7.24: An example 1D Hadamard quantum walk.

2 次元量子ウォークの可視化

可視化例として，本研究手法を用いずに可視化することのできる 2 次元の量子ウォークを取り上げる．2 次元の量子ウォークを観察することで，各設定のもとでの量子ウォークの 2 次元上での特徴を捉える．ここでは，3 つのユニタリ行列 (アダマール変換，グローヴァーウォークを実現する行列，離散フーリエ変換) とそれぞれの場合において 2 つの異なる初期条件による変化を可視化する．

■アダマール変換　1 つめの 2 次元量子ウォーク例として，アダマール変換による量子ウォークを取り上げる．ここで用いる 2 次元におけるアダマール変換 U_H は，以下の式 7.2.10 で与えられる．

$$U_H = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{7.2.10}$$

U_H を量子ウォークの時間変化に用いるユニタリ行列として使用し，以下の式 7.2.11, 7.2.12 で表される初期条件 1，初期条件 2 を初期条件として用いる．また境界条件としては，周期境界条件を用いる．この設定のもと，縦横ともに格子点の数が 31($-15 \leq x \leq 15, -15 \leq y \leq 15$) のもとで可視化を行った例が Fig.7.26(初期条件 1), Fig.7.27(初期条件 2) である (横軸方向が x, 縦軸方向が y 方向の変化を表し，中心が $x = 0, y = 0$ である)．色で表されているのはその時間の座標における確率 $P_t(x,y)$ であり，値と色の対応はは Fig.7.25 に載せたものを用いた．

「初期条件 1」

$$\psi_0(x,y) = \begin{cases} \varphi_1 & (x = y = 0) \\ 0 & (x \neq 0 \,\mathrm{or}\, y \neq 0) \end{cases}, \varphi_1 = \frac{1}{2}\begin{bmatrix} 1 \\ i \\ i \\ -1 \end{bmatrix} \tag{7.2.11}$$

「初期条件 2」

$$\psi_0(x,y) = \begin{cases} \varphi_2 & (x = y = 0) \\ 0 & (x \neq 0 \,\mathrm{or}\, y \neq 0) \end{cases}, \varphi_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.2.12}$$

アダマール変換による量子ウォークの確率分布の特徴として，確率の変化が複雑であり，値の振動のようなものが見られること，矩形的な形を比較的保持しながら外郭が広がっていく，また $y = x$ の直線上付近にある座標の確率が高い，確率の高いところの一部がまとまって外側に広がっていっている (特に初期条件 1 であると，$y = x$ 上の確率が広がる先端部分に集まっている) といったことがあげられる．また，量子ウォーカーの持つ初期値によって，確率の分布の様子が大きく異なっている．初期条件 1 では，$y = x, y = -x$ に対して線対称，中心座標に対して点対称などの対称性があるが，初期条件 2 ではそれらの対称性が大きく崩れている．古典的なランダムウォークでは，確率分布は 2 項分布タイプであること，広がりが狭いこと，値の振動のようなものがないことなどと考え合わせると，量子ウォークには古典的なランダムウォークと異なる特徴が多く存在する．

Fig. 7.25: Corresponding color and probability value for quantum walk visualization.

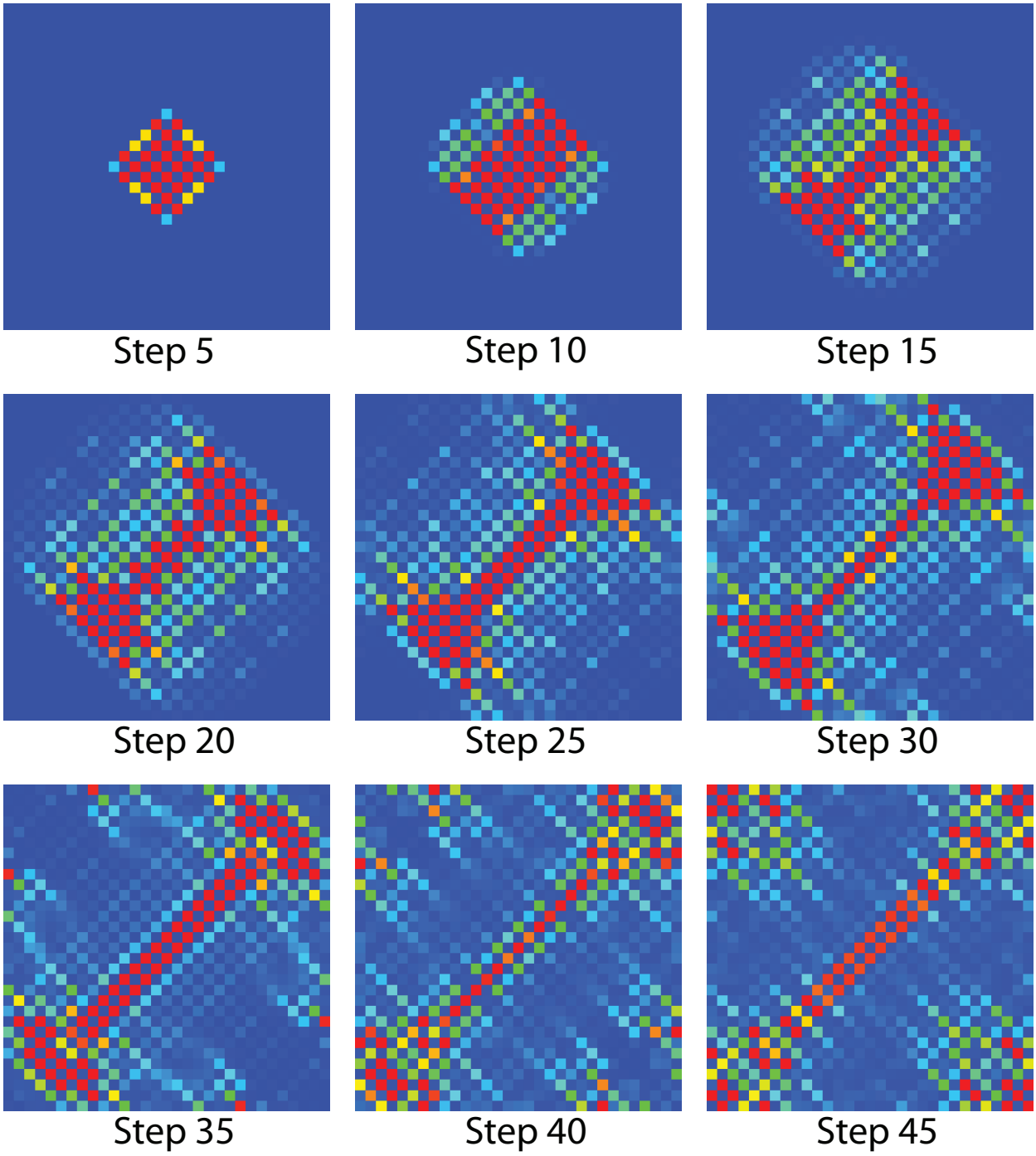


Fig. 7.26: 2D Hadamard quantum walk results with initial condition 1.

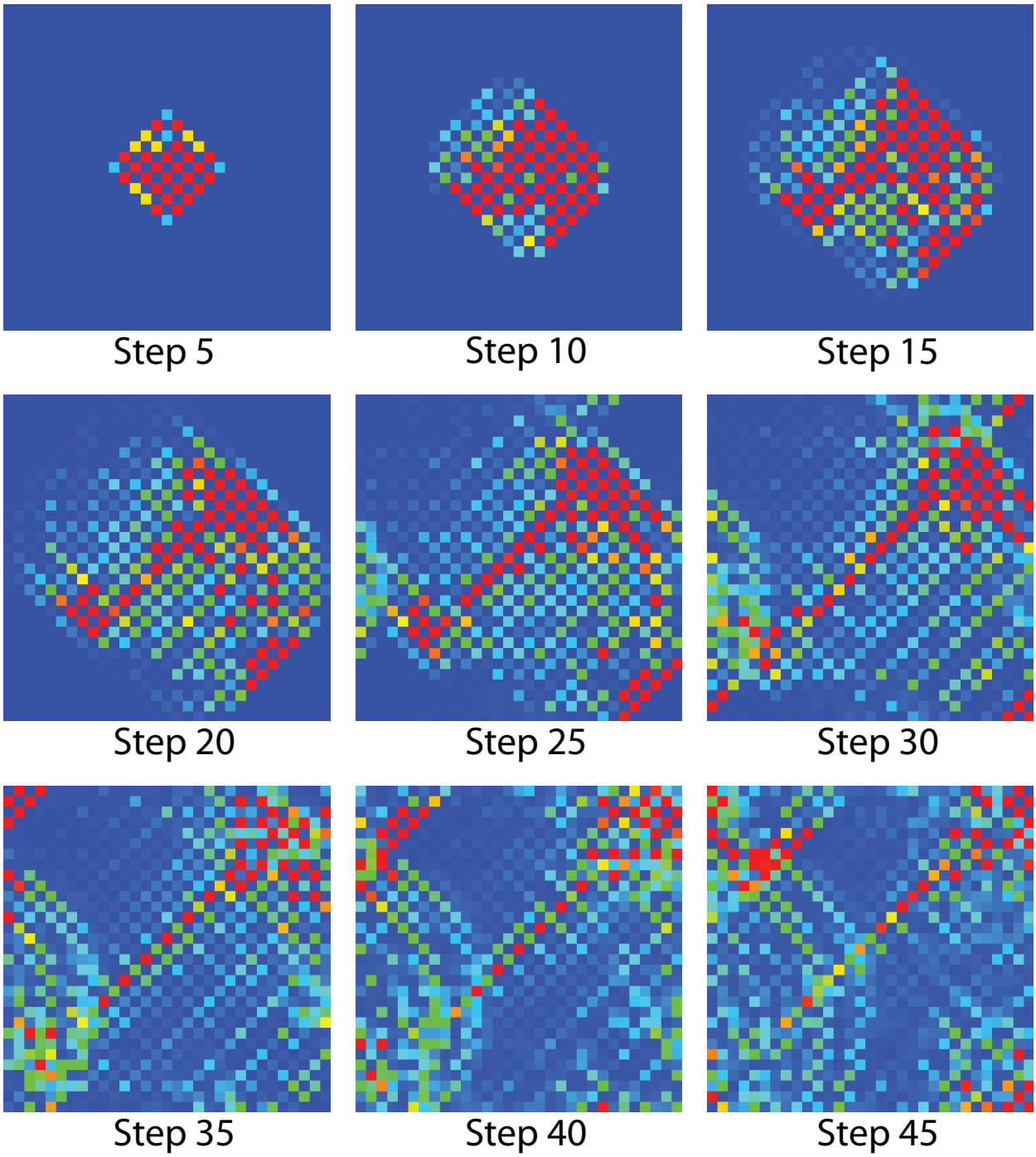


Fig. 7.27: 2D Hadamard quantum walk results with initial condition 2.

■グローヴァー　2 つめの 2 次元量子ウォーク例として，グローヴァーウォークを取り上げる．ここで用いる 2 次元におけるグローヴァーウォークを実現する行列 U_G は，以下の式 7.2.13 で与えられる．

$$U_G = \frac{1}{2}\begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix} \tag{7.2.13}$$

U_G を量子ウォークの時間変化に用いるユニタリ行列として使用し，その他の設定はアダマール変換の例のときと同様のものを用いて可視化を行った例が Fig.7.28(初期条件 1), Fig.7.29(初期条件 2) である．

グローヴァーウォークにおける確率分布特徴として，アダマール変換によるものと比べて広がり方が円形に近い，アダマール変換と同様に確率の高いところが外側に大きく広がっていっている，初期条件 1, 2 によって分布は変わるがどちらでも線対称性などが見られるといったことがあげられる (初期条件 1 の場合は 3 方向に値の高い部分が広がっていくが，初期条件 2 では 2 方向である)．最もグローヴァーウォークで特徴的なのは，中心座標およびその周辺での確率の高さである．グローヴァーウォークでは，中心座標における確率の時間平均が他の場所と比べて非常に大きく，確率分布の局所化が起こる (文献 [10] で詳しく述べられている)．今回の可視化像では，確率が 0.005 以上のところはすべて赤色になるため，他の赤色の箇所と比べても値の違いが分からないが，中心座標の確率の 100 ステップの時間平均を計算すると 0.175(約 6 分の 1) ととても大きな値をとる．

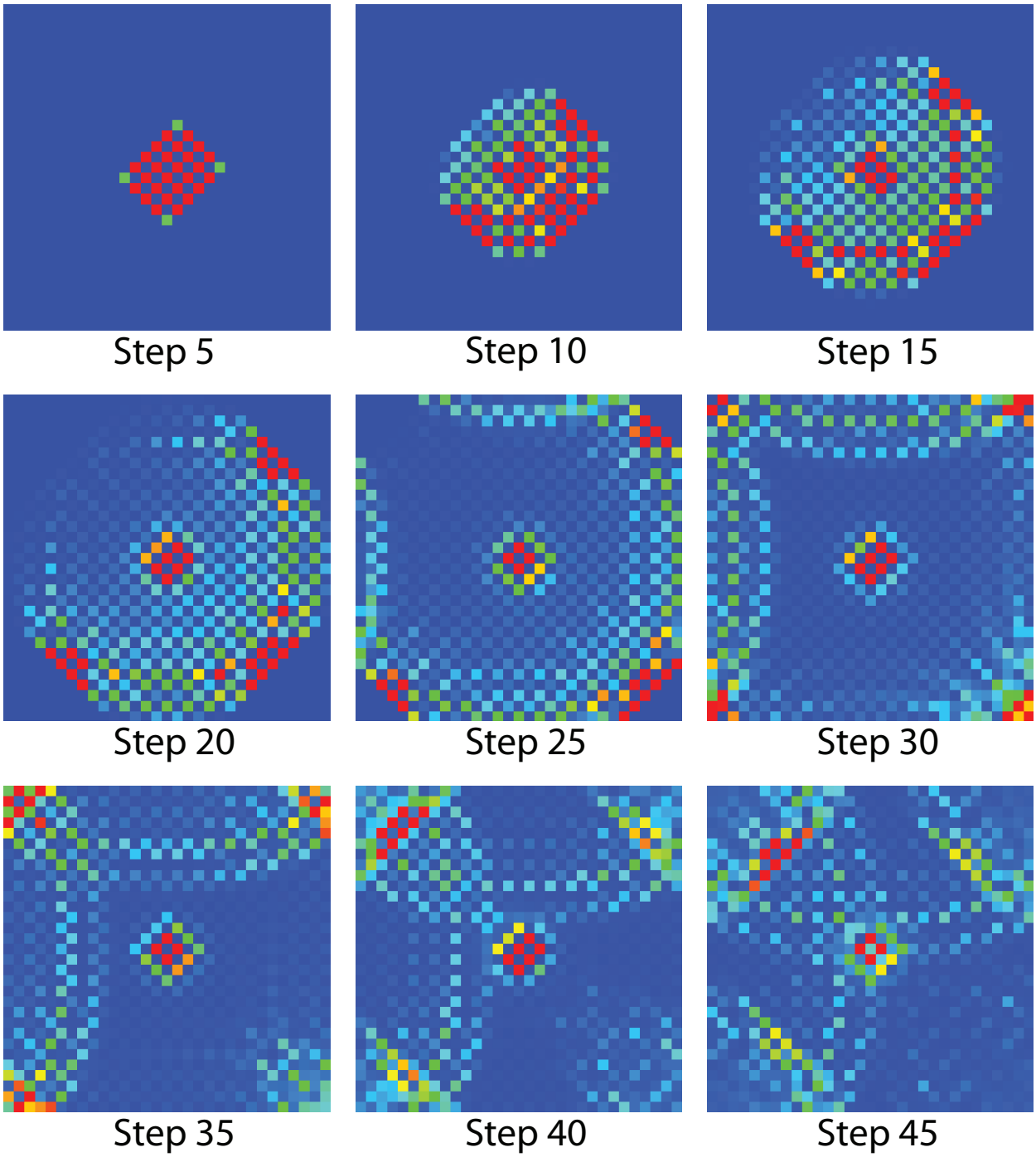


Fig. 7.28: 2D Grover quantum walk results with initial condition 1.

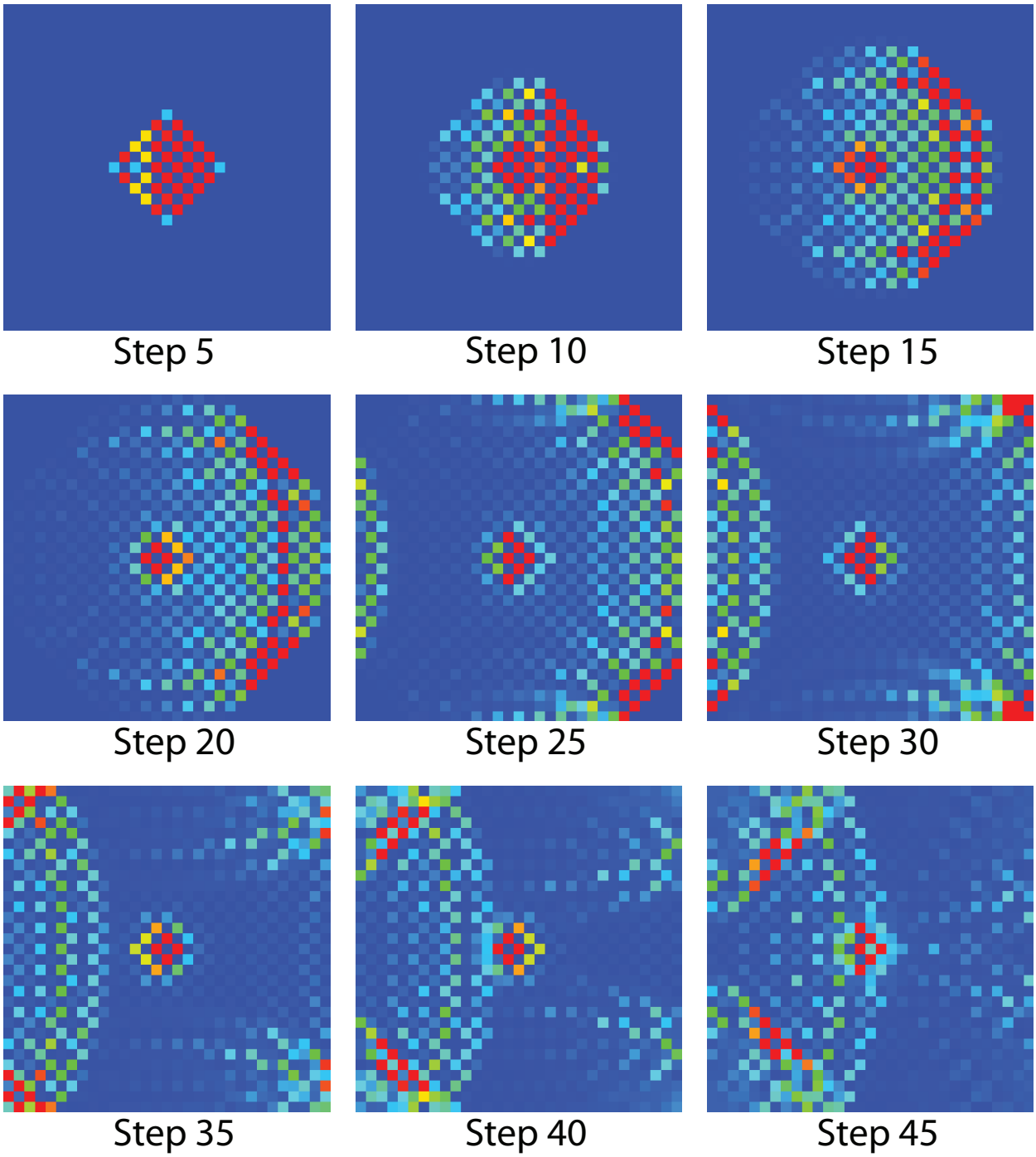


Fig. 7.29: 2D Grover quantum walk results with initial condition 2.

■離散フーリエ変換　3 つめの 2 次元量子ウォーク例として，離散フーリエ変換による量子ウォークを取り上げる．ここで用いる 2 次元における離散フーリエ変換 U_F は，以下の式 7.2.14 で与えられる．

$$U_F = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \quad (\omega = e^{i\pi/2}) \tag{7.2.14}$$

U_F を量子ウォークの時間変化に用いるユニタリ行列として使用し，その他の設定はアダマール変換の例のときと同様のものを用いて可視化を行った例が Fig.7.30(初期条件 1), Fig.7.31(初期条件 2) である．

離散フーリエ変換による量子ウォークでは，アダマール変換，グローヴァーウォークの場合と比べて，確率分布の広がり方にはっきりと分かる対称性や規則性が見られないといったことがあげられる．ただし，$y = -x$ 上に確率が高いところが存在し，特に $+x$ 方向では確率が高いところが集まっている箇所が見られることは確認できる．

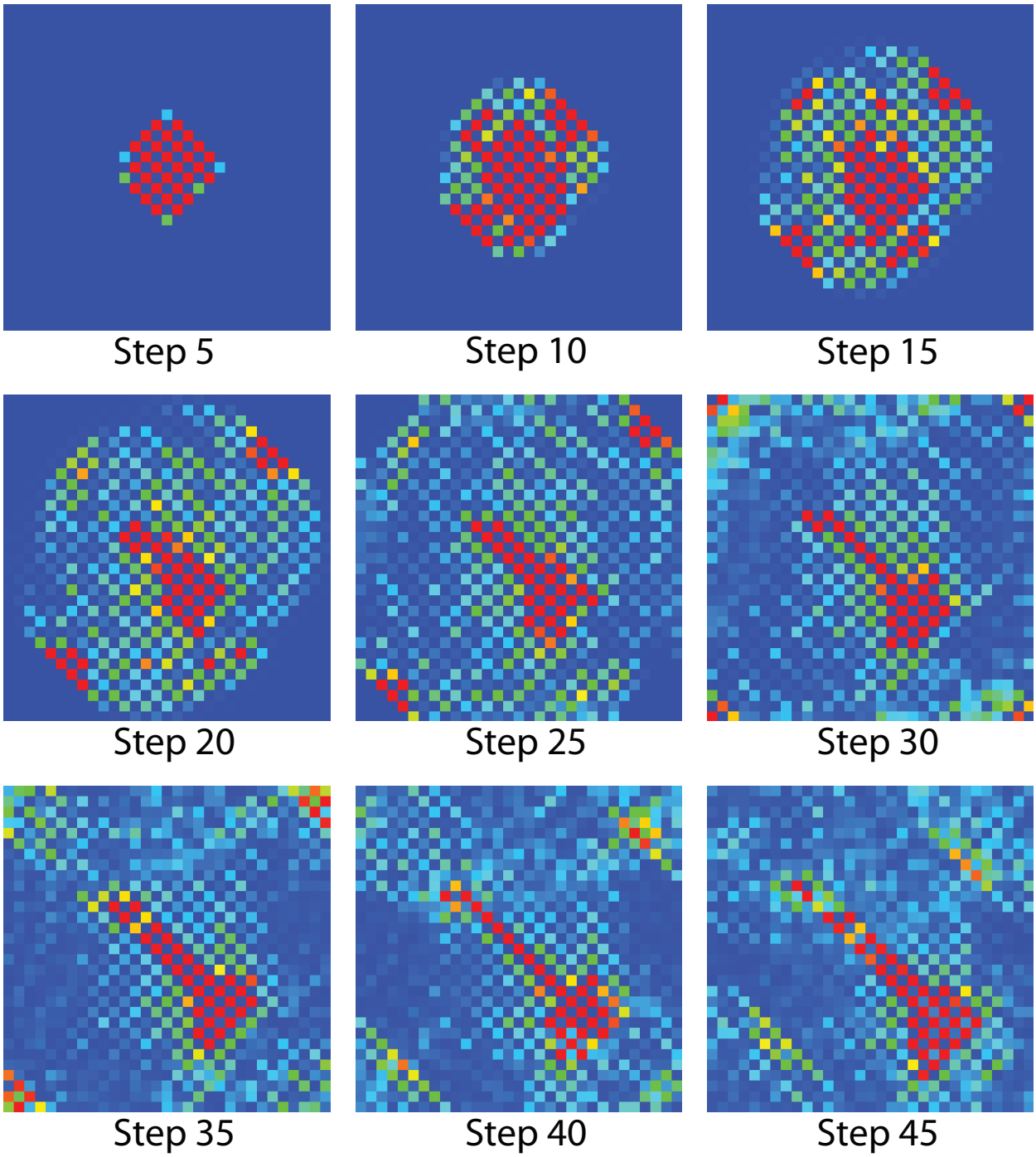


Fig. 7.30: 2D DFT quantum walk results with initial condition 1.

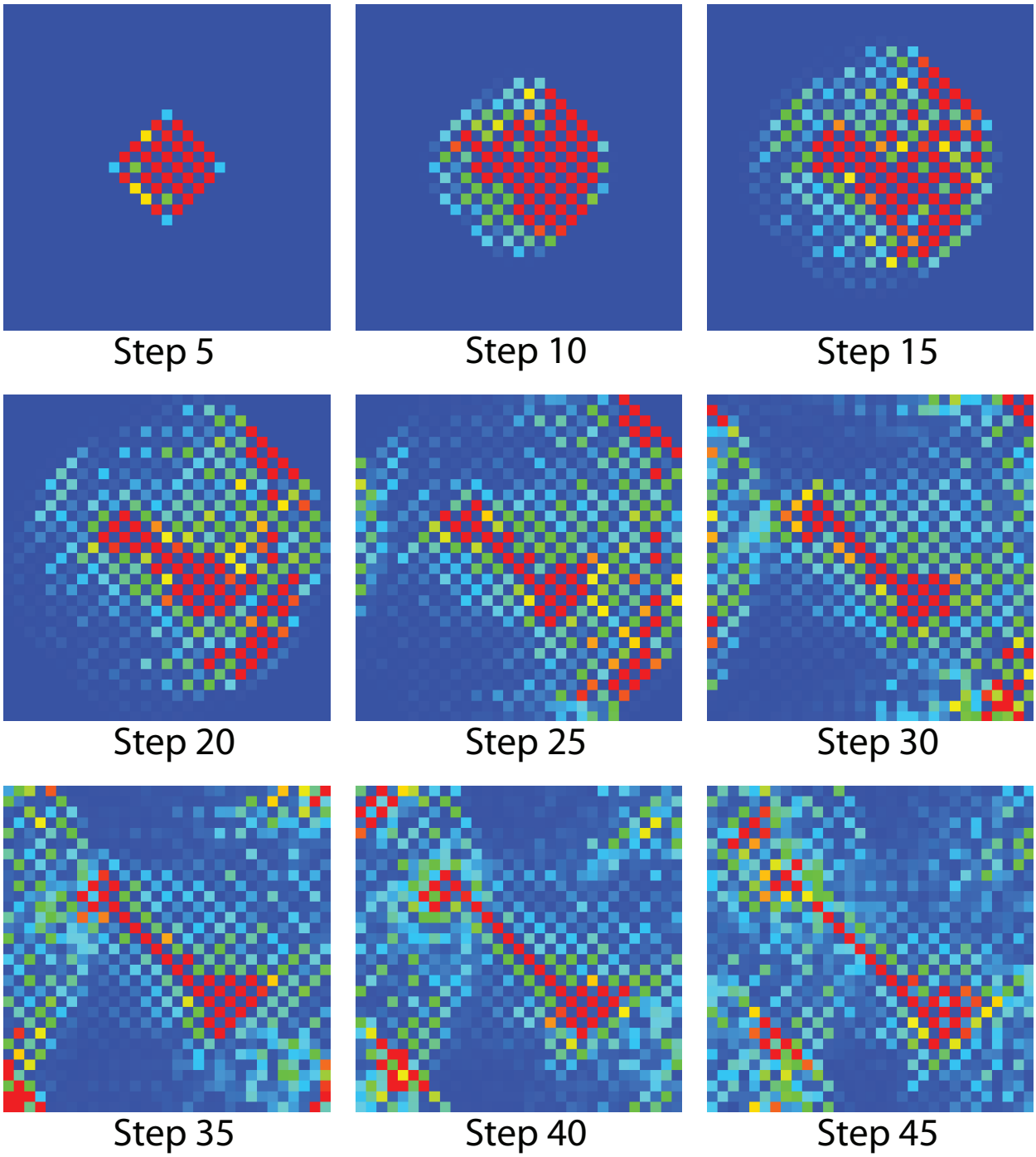


Fig. 7.31: 2D DFT quantum walk results with initial condition 2.

4 次元量子ウォークの可視化

ここでは，4 次元空間における量子ウォークの可視化を行う．提案手法によって低次元化を行うことで，4 次元空間における量子ウォークが可視化可能になる．低次元化の方法としては，4 次元すべてを同様に扱った 2 次元化を選択した．

マッピングルールには，Fig.7.32 に挙げたものを利用する．このルールは，点対称性，次元のおりたたみ (x_3, x_4 の大小によって分けられる 4 領域の中に x_1, x_2 の大小によって分けられる 4 領域が存在する) などの特徴を有し，またある程度距離に基づいたルールになっている．また，解像度 1, 2, 3 のときの各領域にマッピングされるデータのもとの 4 次元空間における原点からの距離の様子を Fig.7.33 に載せる．もとの図形において原点に近いものほど 2 次元上の原点に近いところに，遠いものほど 2 次元上の原点から遠いところに大まかにマッピングされることが分かる．

さらに，今回は拡散の対称性をさらに見やすくするために，マッピングの際に特殊な処置を行う．まず，マッピングルールの再帰的な適用を通常通りに 1 回行い 16 分割された 4 次元領域を 2 次元図形上の 16 領域にマッピングする箇所を決める．2 回目以降の再帰的なマッピングルールは，次のような座標変換を行ってから適用する．4 次元データの位置が 4 つの座標値 (x_1, x_2, x_3, x_4) で決定するとき，それぞれの軸における正の方向を $+x_1, +x_2, +x_3, +x_4$，負の方向を $-x_1, -x_2, -x_3, -x_4$ と表すこととすると，中央点 (m_1, m_2, m_3, m_4) から $+x_1, +x_2, +x_3, +x_4$ 方向にある領域には，座標変換を行わず，通常の再帰的なマッピングを行う．そうでない領域では，$-x_i (i = 1, 2, 3, 4)$ 方向になっている次元の値を $2m_i - x_i$ に変換し，中央点から正方向の座標に変化させる．すべての座標値を中央点から同じ距離にある正方向の座標値に変化させ，その後再帰的なマッピングを行う．上記のマッピングを 3 次元データに行った場合の例を Fig.7.34 に載せる．このようにマッピングを行うことで，中央点から各領域方向への変化が同じ領域では，まったく同一の写像結果が得られるようになる．この性質とすでに決めた原点からの距離関係を反映するマッピングルールにより，中央からの変化の対称性や拡散の様子が容易に観察できるようになるため，今回の可視化対称である高次元量子ウォークには適したマッピングとなっている．

4 次元量子ウォークが行われる領域の大きさは各軸方向の格子点数が奇数の $15 \times 15 \times 15 \times 15$ である．この領域を可視化する際には，中央点に対して対称に領域を分割するために，1 回目の分割で得られる 16 領域はすべて中央点を含む $8 \times 8 \times 8 \times 8$ の大きさで取ることとする (Fig.7.35 に 2 次元での中央点を含ませた分割イメージを載せている)．そのため可視化に使用する領域数は $16^4 = 65536 (= 8^4 \cdot 16)$ となり，解像度は 4 である．また，色と確率の値との対応のさせ方は，2 次元量子ウォークのときと同じ Fig.7.25 のものを用いる．また境界条件としては，周期境界条件を用いる．

0 1 0 1	1 1 0 1	0 1 1 1	1 1 1 1
0 0 0 1	1 0 0 1	0 0 1 1	1 0 1 1
0 1 0 0	1 1 0 0	0 1 1 0	1 1 1 0
0 0 0 0	1 0 0 0	0 0 1 0	1 0 1 0

Fig. 7.32: 4D mapping rule for quantum walk.

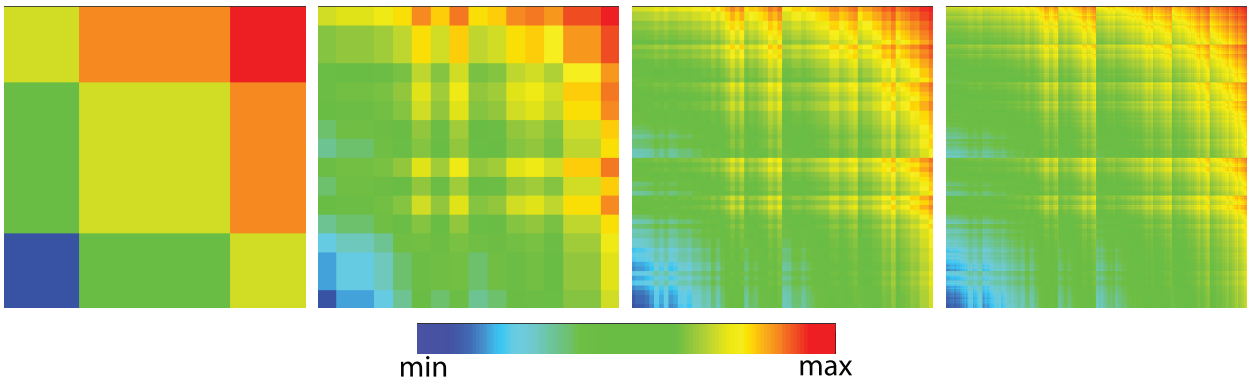


Fig. 7.33: Distance from origin of 4D mapping for resolution 1, 2, 3 and 4.

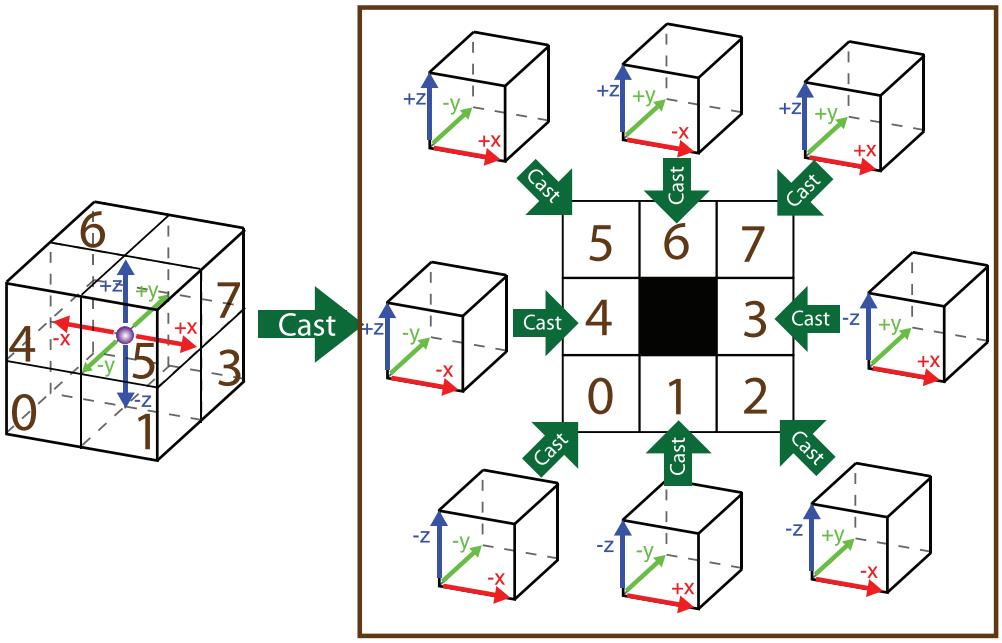


Fig. 7.34: An example of 2 step mapping rule for 3D data.

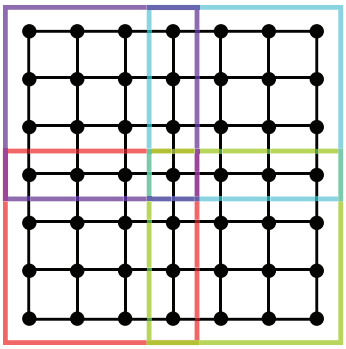


Fig. 7.35: 2D image of division including center position. The region is divided in 4 color regions.

■アダマール変換　1 つめの 4 次元量子ウォーク例として，アダマール変換による量子ウォークを取り上げる．ここで用いる 4 次元におけるアダマール変換 U_H は，以下の式 7.2.15 で与えられる．

$$U_H = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \tag{7.2.15}$$

U_H を量子ウォークの時間変化に用いるユニタリ行列として使用し，以下の式 7.2.16, 7.2.17 で表される初期条件 1，初期条件 2 を初期条件として用いる．提案手法を用いて可視化を行った例が Fig.7.36 から Fig.7.41(初期条件 1), Fig.7.42 から Fig.7.47(初期条件 2) である．

$$\psi_0(x,y) = \begin{cases} \varphi_1 & (x=y=0) \\ 0 & (x \neq 0 \,\mathrm{or}\, y \neq 0) \end{cases}, \varphi_1 = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 \\ i \\ i \\ -1 \\ i \\ -1 \\ -1 \\ -i \end{bmatrix} \tag{7.2.16}$$

$$\psi_0(x,y) = \begin{cases} \varphi_2 & (x=y=0) \\ 0 & (x \neq 0 \,\mathrm{or}\, y \neq 0) \end{cases}, \varphi_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7.2.17}$$

アダマール変換による 4 次元の量子ウォークの確率分布の 1 つ目の特徴として，2 次元の場合と同様に各座標において確率の値が複雑で振動していることが挙げられる．この性質は一般的な量子ウォークと同様である．また，同じ拡散の挙動を示す領域があり，対称性が見られる．例えば，白線で区切られた 16 領域の呼び方に Fig.7.6 に書かれてある 4 桁の数字を用いると，初期条件 1 の場合は，領域 0000 と 1111, 1000 と 0111，0100 と 1011 など 4 桁の数字の 0 と 1 が入れ替わっている領域どうしが同じ拡散の仕方をしている．これは，2 次元で $y=x, y=-x$ に関して線対称な拡散の仕方をしていたことと対応し，2 次元中で見られた特徴が 4 次元中でも観察できる．初期条件 2 に初期値を変更した場合は，対称性が大きく崩れ，このような対称な特徴ははっきりとは観察できない．しかし，初期条件 2 の場合でも拡散の仕方が似ている領域の組を見ることはできる．さらに，2 次元のときに $y=x$ の直線上の広がりの先端部分に確率の高い領域が集まっていたことと似た現象が 4 次元でも観察できることが次の結果から分かる．4 次元

の初期条件 1 の場合のステップ 25 のときの結果 (Fig.7.40) を見ると，白線で囲まれた領域 0000，0110，1001，1111 をさらにそれぞれを 16 領域に分けた領域のうち 0000，0110，1001 で他の領域よりも確率の高い箇所が多く存在することが分かる．2 次元の量子ウォークの場合と比較すると，領域 0000 は原点近く，0110，1001 は $y = x$ 上の確率の高い先端部分に対応していると考えられる．2 次元の場合は，確率の高い先端部分が 2 つであったが，4 次元の場合は 8 個あることが分かる．他の次元の場合も調べる必要があるが，この結果から，初期条件 1 でアダマール変換による量子ウォークを行うと，一般の n 次元のとき，2^{n-1} 個の確率の高い先端部分が存在するということが予想できる．このように 4 次元の場合であっても本手法をマッピングルールを適切に選んで用いることで，現象が把握しやすい結果を得ることができ，2 次元などの場合と比較が容易に可能であることが分かる．

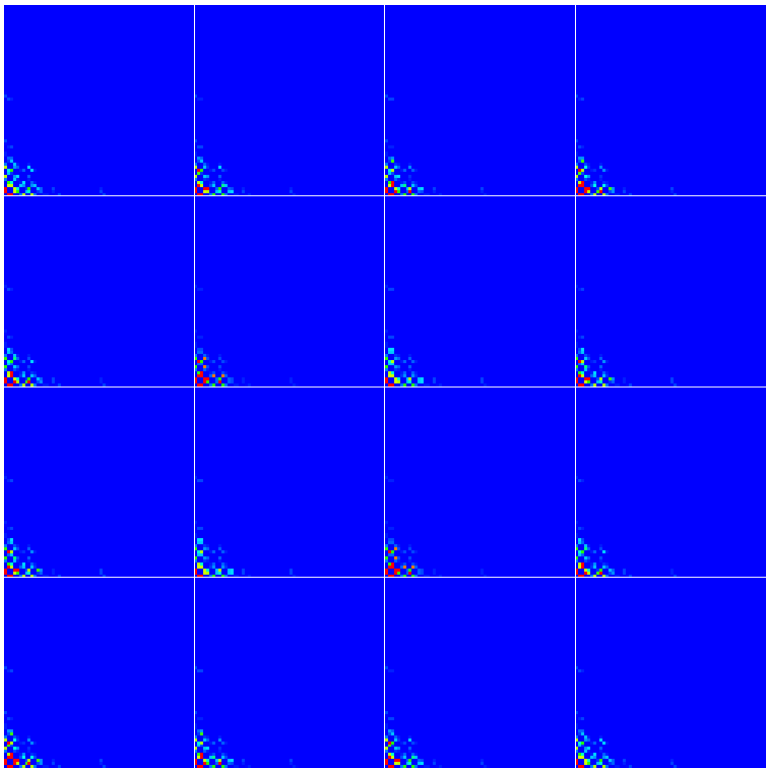


Fig. 7.36: 4D Hadamard quantum walk result with initial condition 1 (step 5).

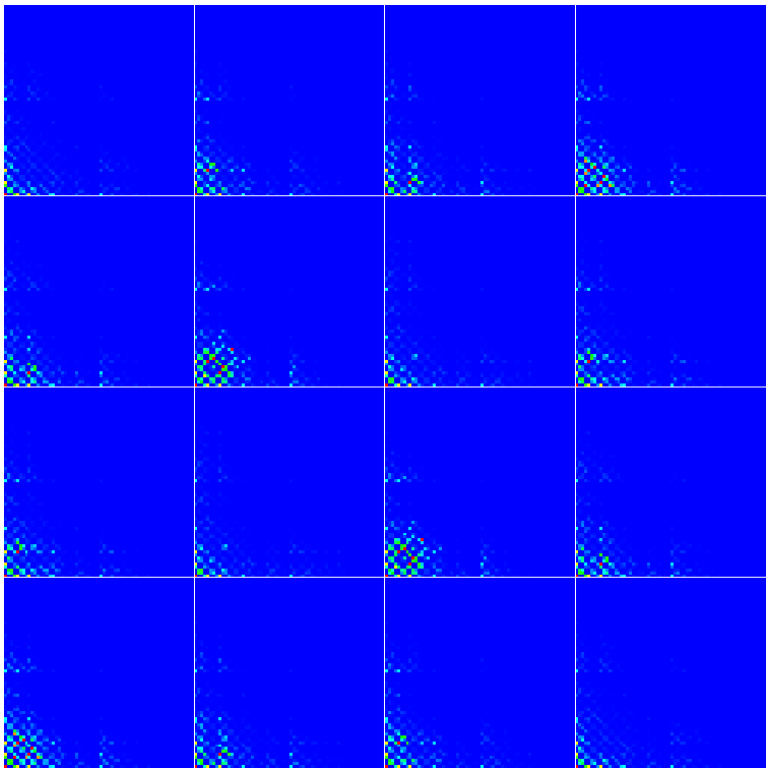


Fig. 7.37: 4D Hadamard quantum walk result with initial condition 1 (step 10).

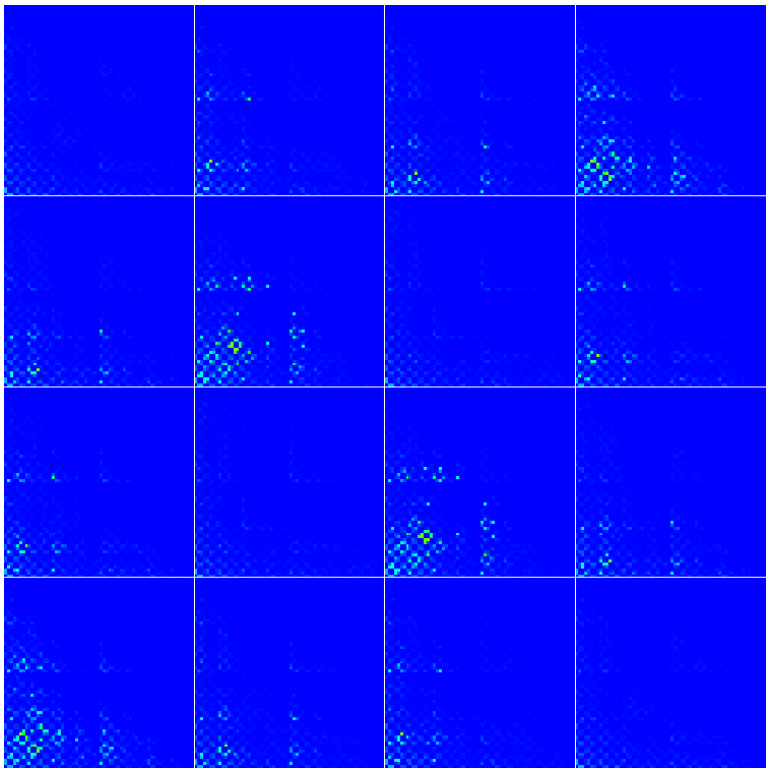


Fig. 7.38: 4D Hadamard quantum walk result with initial condition 1 (step 15).

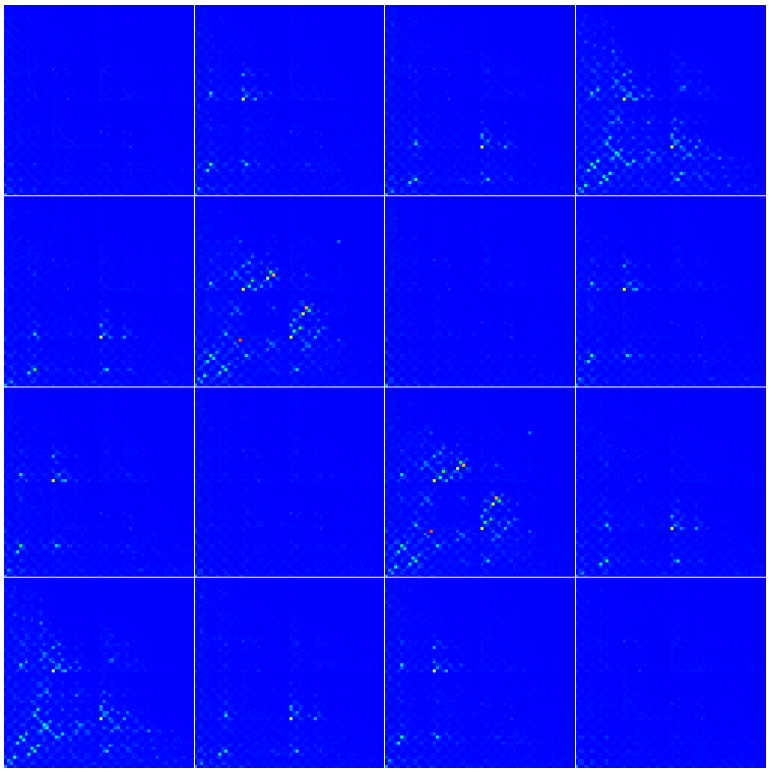


Fig. 7.39: 4D Hadamard quantum walk result with initial condition 1 (step 20).

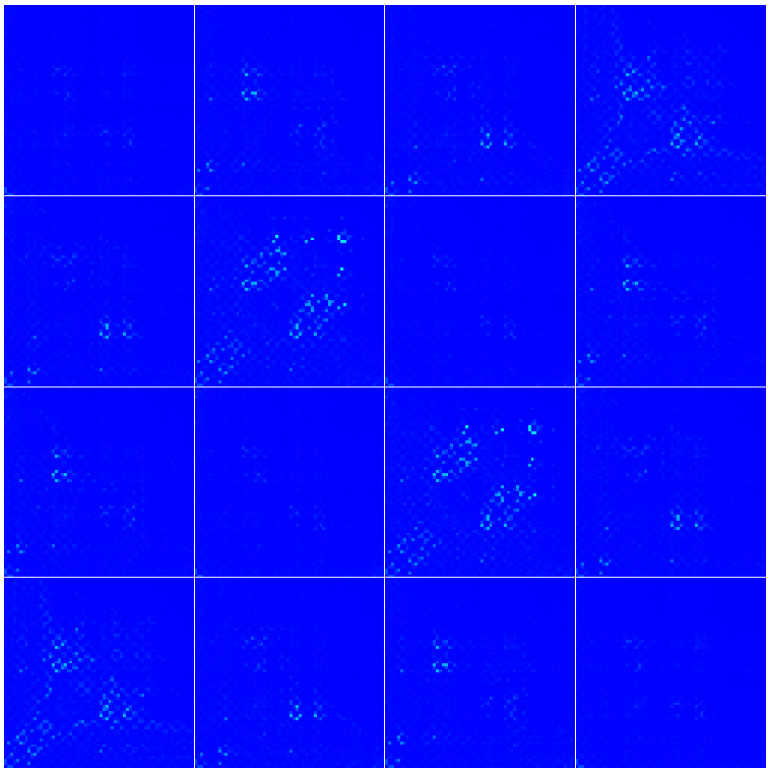


Fig. 7.40: 4D Hadamard quantum walk result with initial condition 1 (step 25).

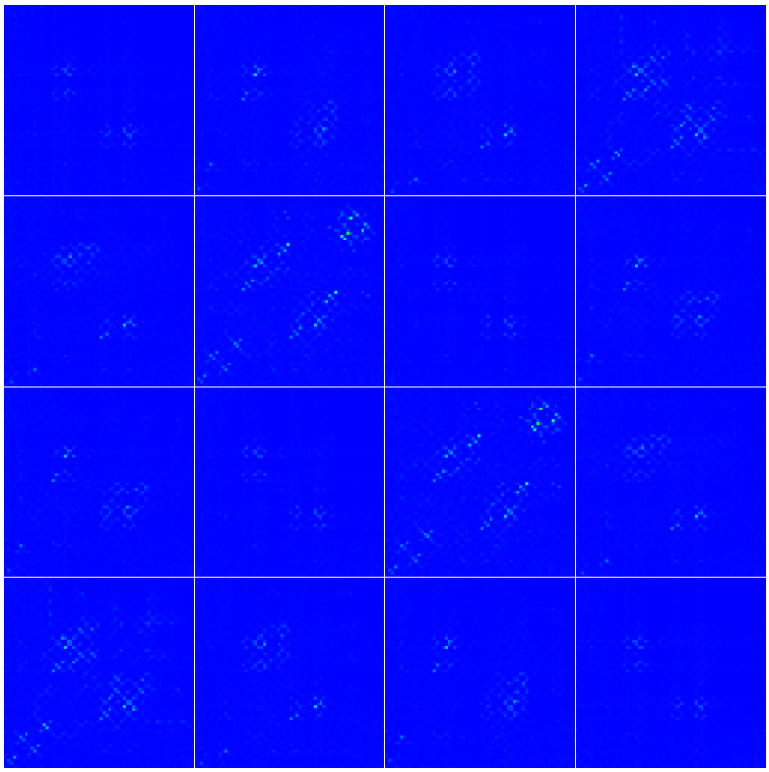


Fig. 7.41: 4D Hadamard quantum walk result with initial condition 1 (step 30).

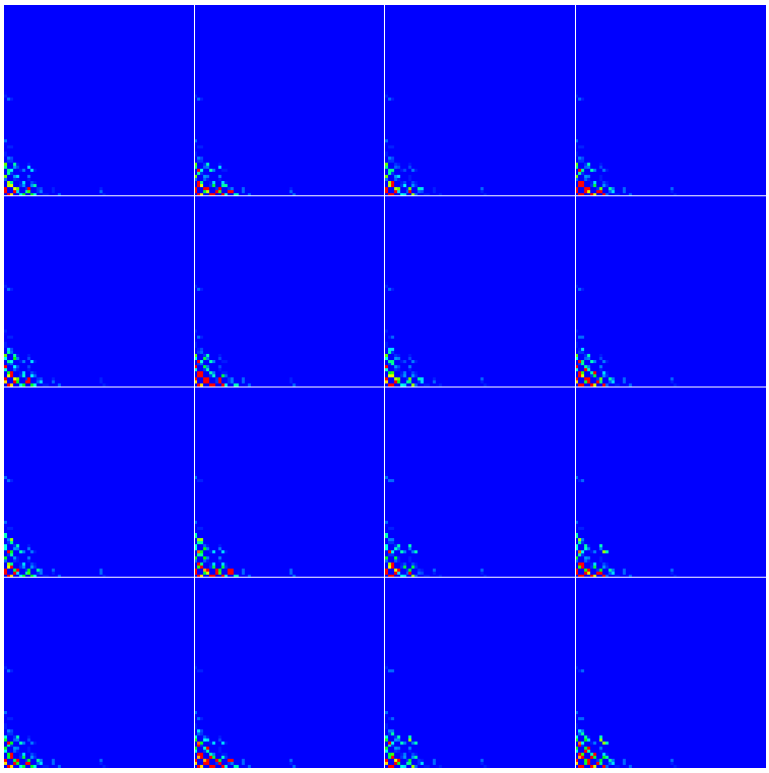


Fig. 7.42: 4D Hadamard quantum walk result with initial condition 2 (step 5).

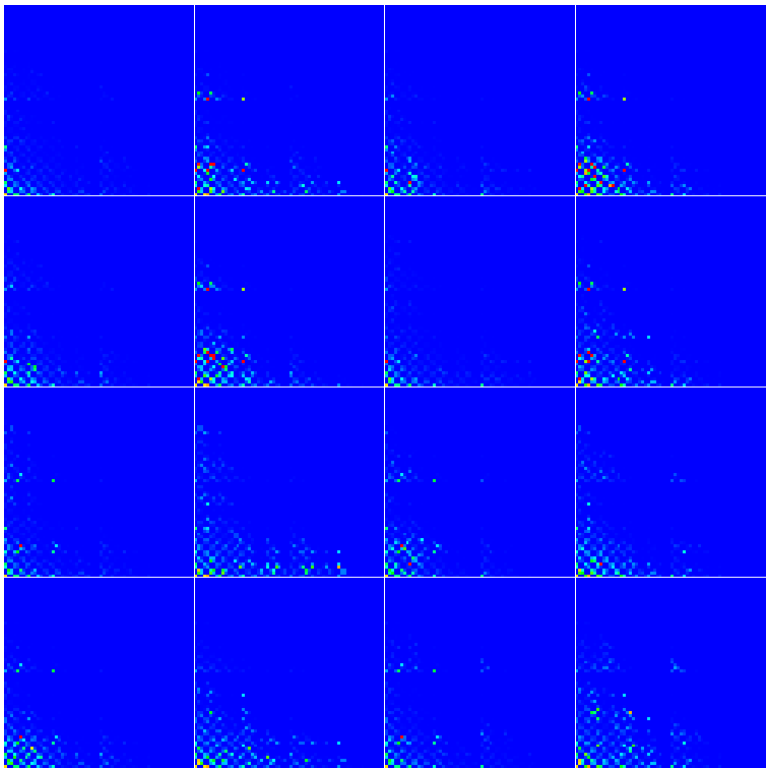


Fig. 7.43: 4D Hadamard quantum walk result with initial condition 2 (step 10).

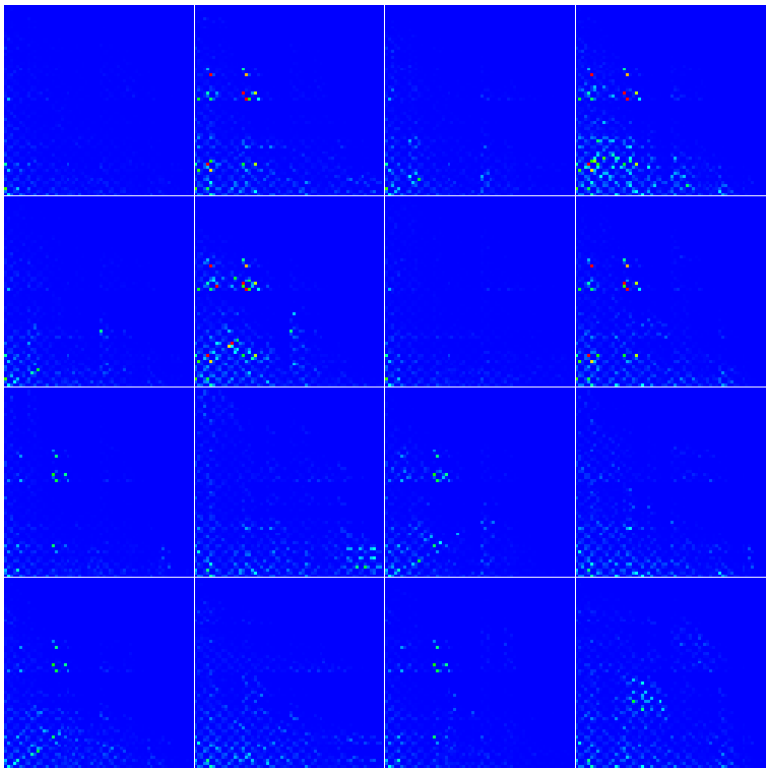


Fig. 7.44: 4D Hadamard quantum walk result with initial condition 2 (step 15).

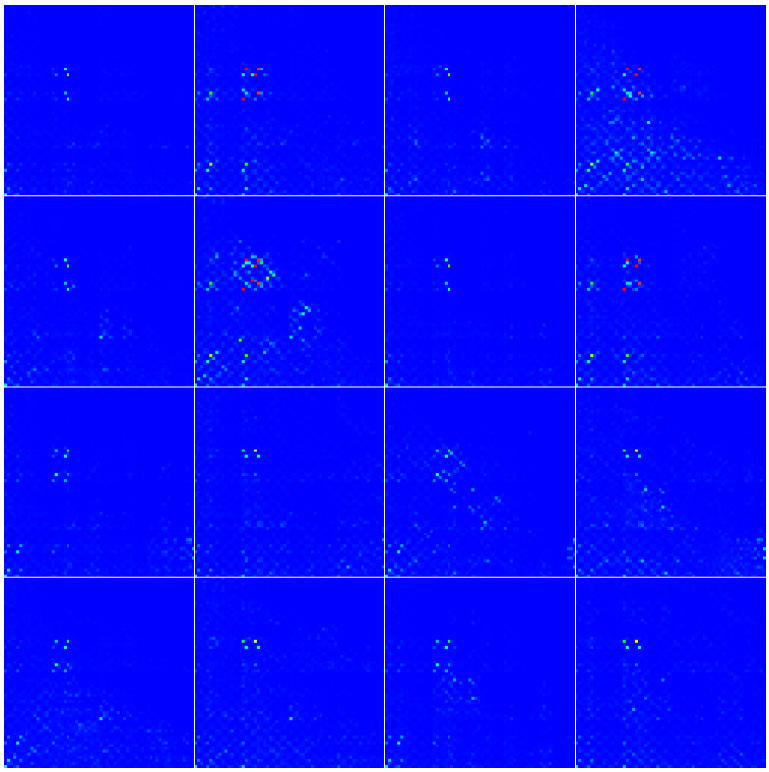


Fig. 7.45: 4D Hadamard quantum walk result with initial condition 2 (step 20).

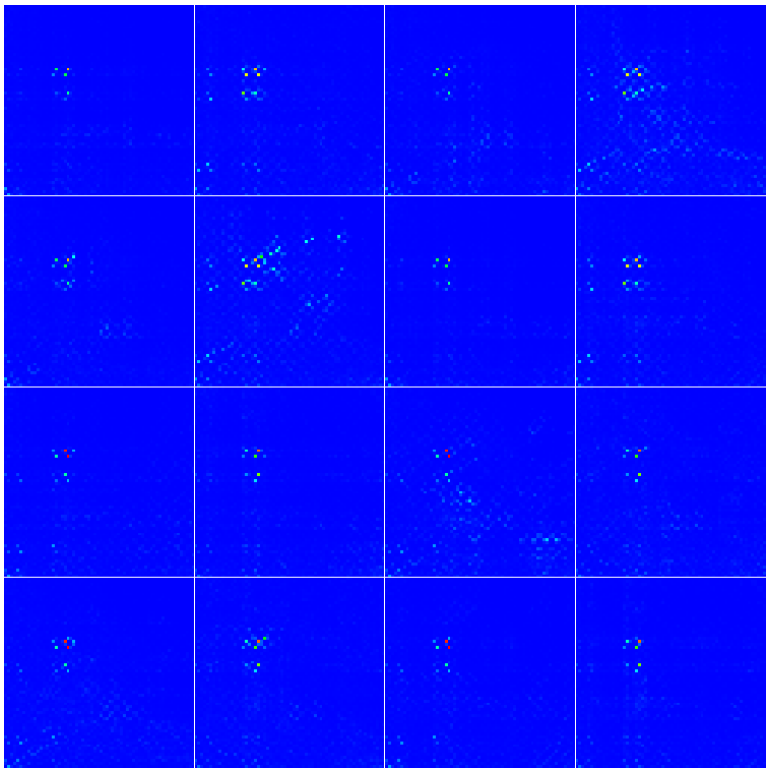


Fig. 7.46: 4D Hadamard quantum walk result with initial condition 2 (step 25).

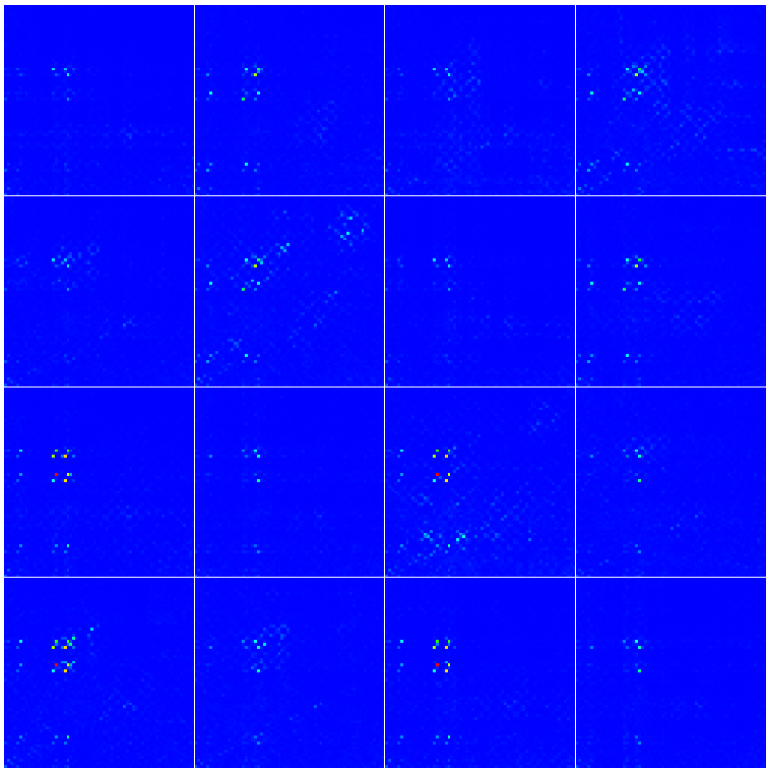


Fig. 7.47: 4D Hadamard quantum walk result with initial condition 2 (step 30).

■グローヴァー　2 つめの 4 次元量子ウォーク例として，グローヴァーウォークを実現するユニタリ行列による量子ウォークを取り上げる．ここで用いる 4 次元におけるグローヴァーウォークを実現するユニタリ行列 U_G は，以下の式 7.2.18 で与えられる．

$$U_G = \frac{1}{4}\begin{bmatrix} -3 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -3 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -3 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -3 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -3 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & -3 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -3 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & -3 \end{bmatrix} \tag{7.2.18}$$

U_G を量子ウォークの時間変化に用いるユニタリ行列として使用し，アダマール変換のときと同様に式 7.2.16, 7.2.17 で表される初期条件 1，初期条件 2 を初期条件として用いる．提案手法を用いて可視化を行った例が Fig.7.48 から Fig.7.53(初期条件 1), Fig.7.54 から Fig.7.59(初期条件 2) である．

グローヴァーウォークの場合もアダマール変換のときと同様に 2 次元上で見られた特徴がいくらか観察できる．初期条件 1 の場合の結果を見ると，ステップ 5(Fig.7.48) において原点から距離が離れた位置に確率の高いところがすでに存在していること，ステップ 10(7.49) で原点からの距離が等しいと考えられる範囲に確率が他の箇所より高いことを示すラインができていることから，4 次元になってもグローヴァーウォークでは，超球形の外周に近い形で確率が広がる先端部分が確率が他の部分と比べて高いということが分かる．また，原点付近，特に原点における確率が他と比べて常に高い数値にあるといった確率の局所化の現象も 2 次元の場合と同様に観察できる．ステップ 15 のときには，2 次元化された 16 領域の座標において $y=0, x=0, y=x$ などの直線上に確率の高い箇所が乗っていることも観察できる．初期条件 2 の場合の結果を見ると，初期条件 1 の場合とは違った対称性が現れていることが分かる．各ステップの変化を見ると，x_1 の座標値の中心点からの距離や方向によって起こる現象の様子が分けられることが分かる．例えば，ステップ 10 の結果 (Fig.7.55) を見ると，中心から $+x_1$ の方向にある白線で囲まれた各領域で x_1 の値が小さいところに確率の高い領域が存在している．これは，2 次元のときのステップ 10 の結果 (Fig.7.29 の Step 10) と比べると，2 次元の場合と同様の結果だと言える．このことから，グローヴァーウォークの場合，初期値の設定の仕方によって確率の高い箇所が広がる方向が変化するということが予測できる (初期条件 2 の場合は，$+x_1$ 方向の状態にあたるところだけ数値 1 を入れ，他の箇所には 0 をいれていた)．また，ここに載せた結果以外のステップも連続で観察すると，ステップ 10 のとき白線で囲まれた各領域の中で，x_1, x_3 小の箇所に縦方向に伸びる柱状に分布する確率の高い領域は，15 ステップを周期とした振動的な変化が観察できる．これは，今回各軸方向に 15 格子点を持つ領域で量子ウォークシミュレーションを行ったことに由来すると考えられる．この他にも一般的な量子ウォークに見られる各位置における値の振動なども観察できる．

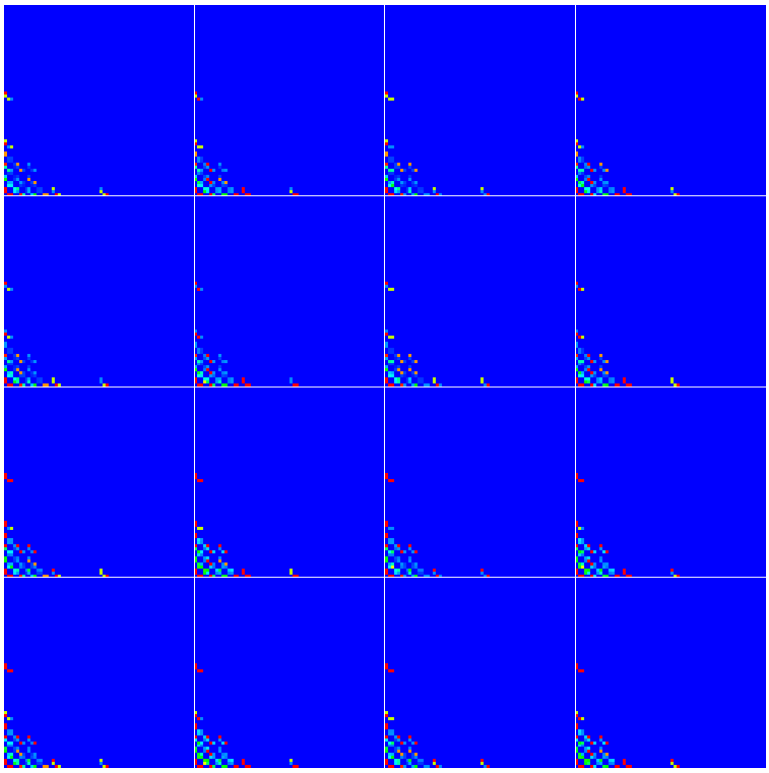


Fig. 7.48: 4D Grover quantum walk result with initial condition 1 (step 5).

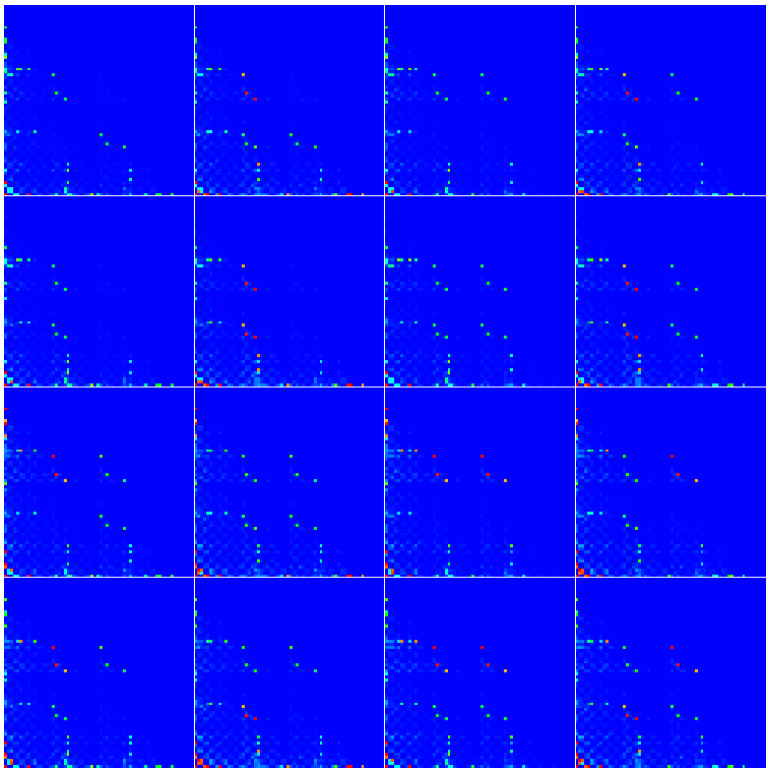


Fig. 7.49: 4D Grover quantum walk result with initial condition 1 (step 10).

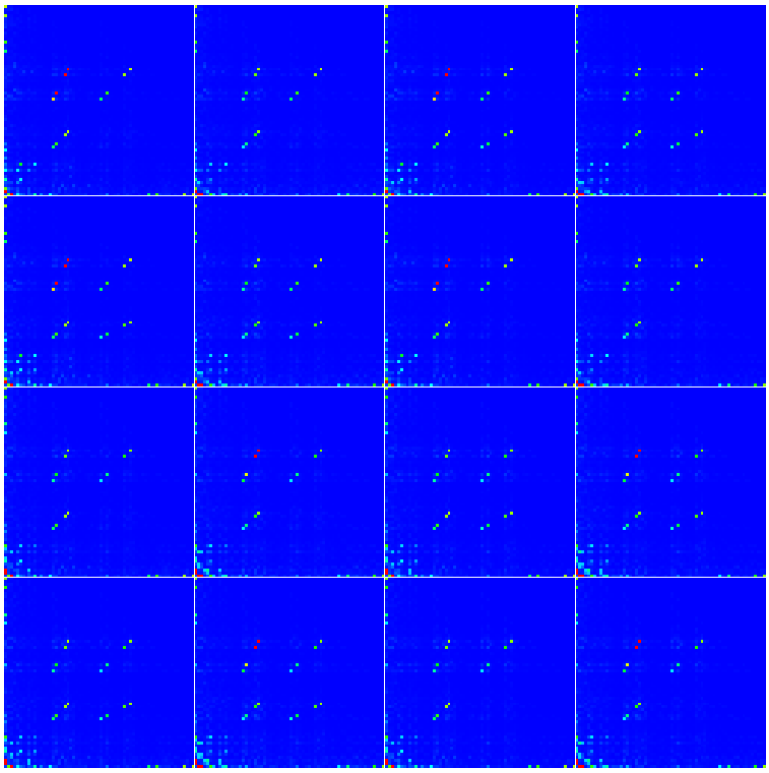


Fig. 7.50: 4D Grover quantum walk result with initial condition 1 (step 15).

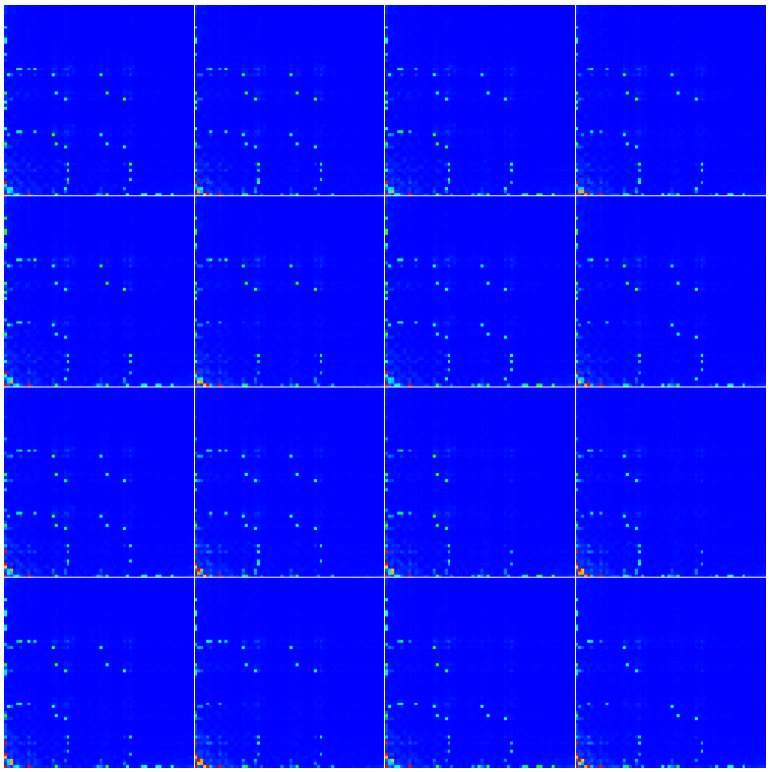


Fig. 7.51: 4D Grover quantum walk result with initial condition 1 (step 20).

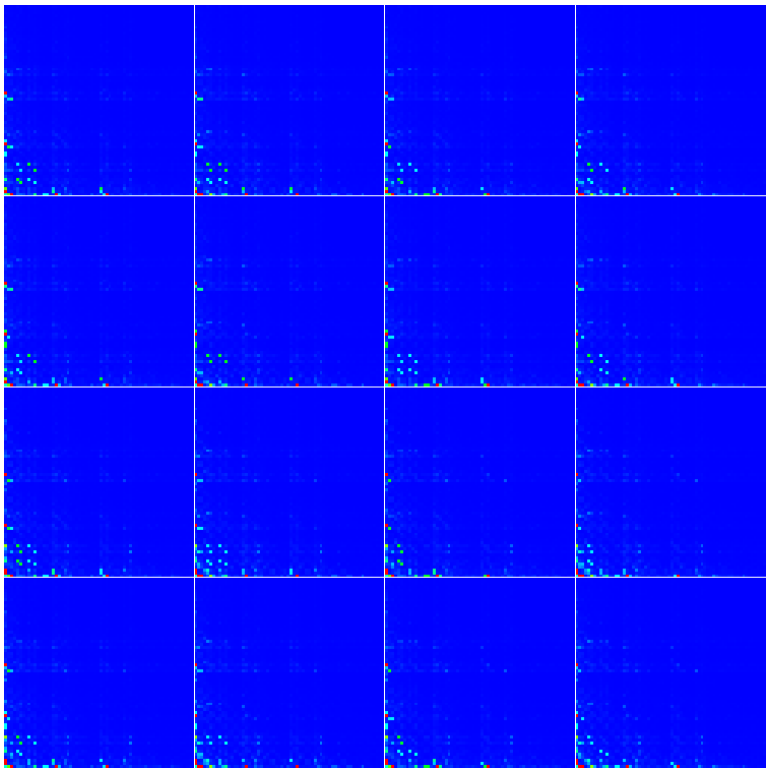


Fig. 7.52: 4D Grover quantum walk result with initial condition 1 (step 25).

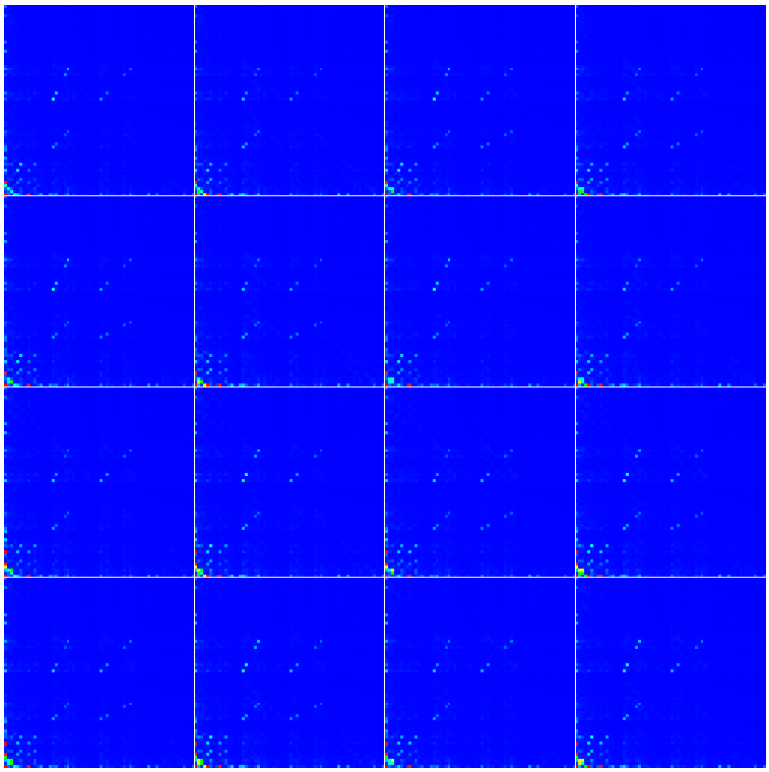


Fig. 7.53: 4D Grover quantum walk result with initial condition 1 (step 30).

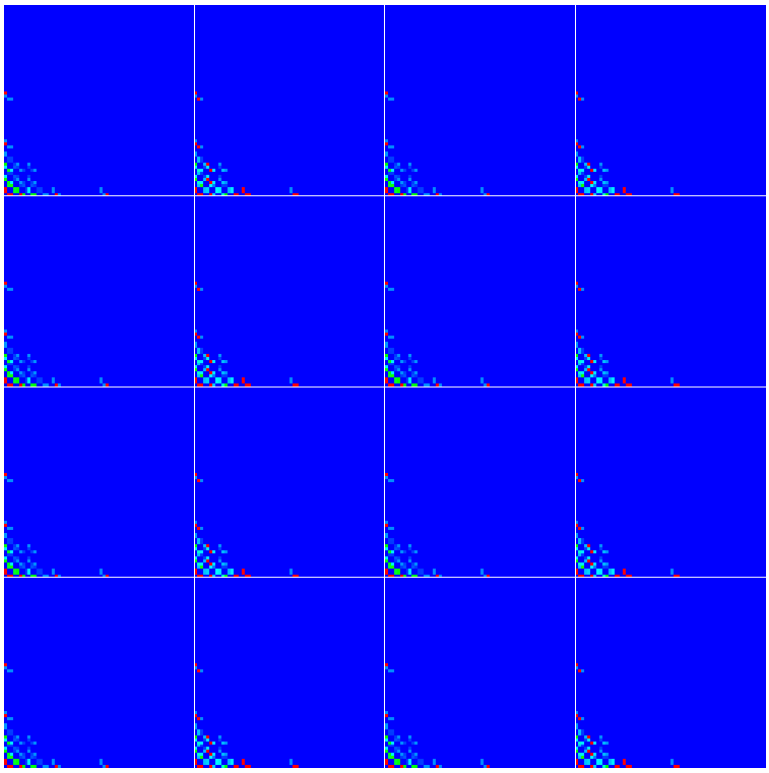


Fig. 7.54: 4D Grover quantum walk result with initial condition 2 (step 5).

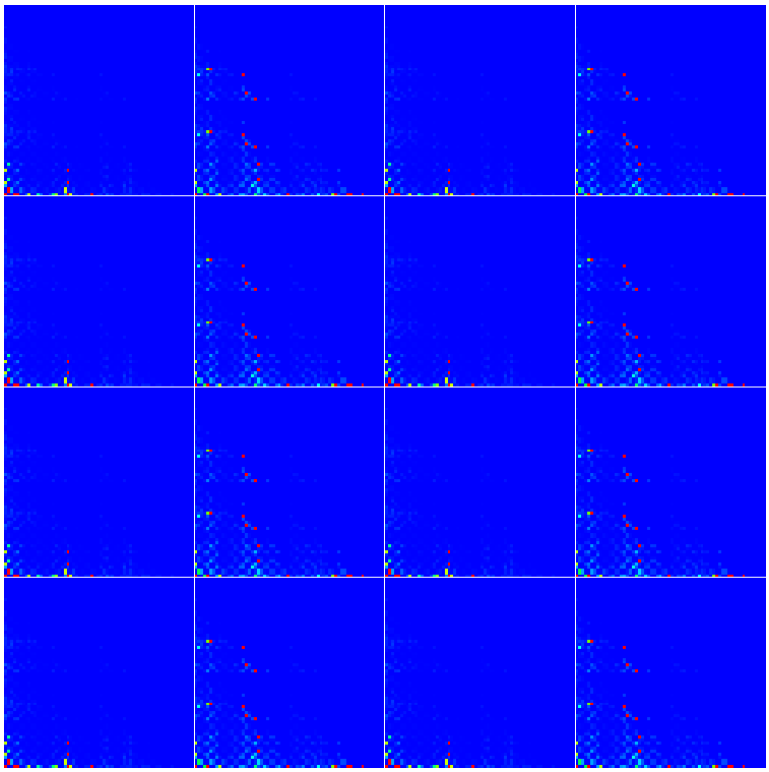


Fig. 7.55: 4D Grover quantum walk result with initial condition 2 (step 10).

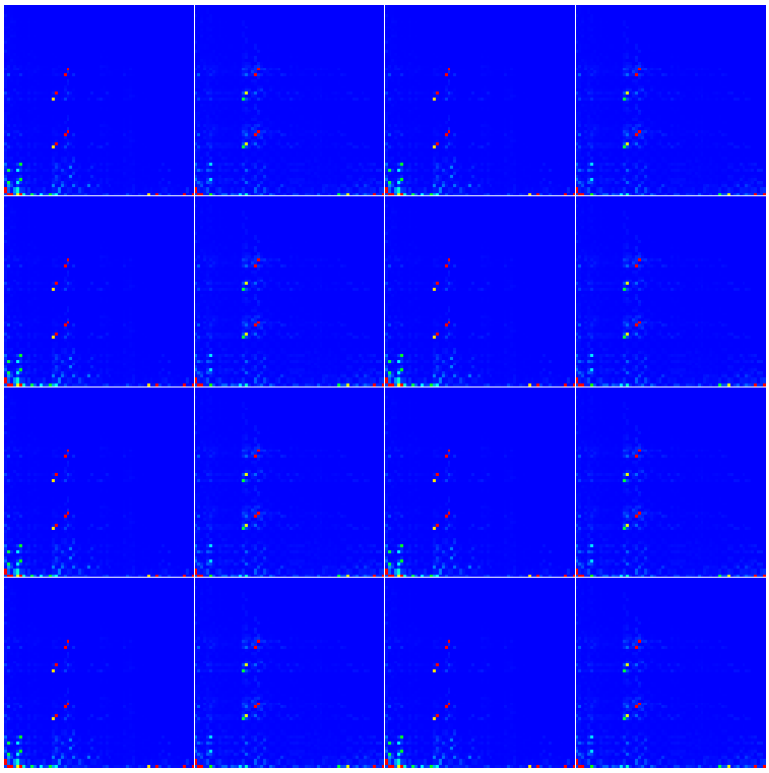


Fig. 7.56: 4D Grover quantum walk result with initial condition 2 (step 15).

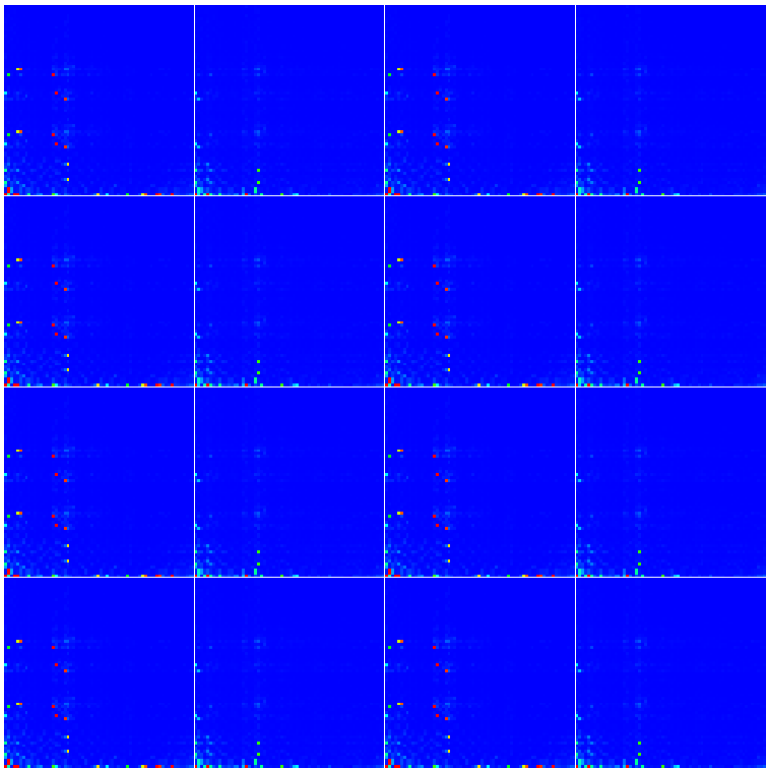


Fig. 7.57: 4D Grover quantum walk result with initial condition 2 (step 20).

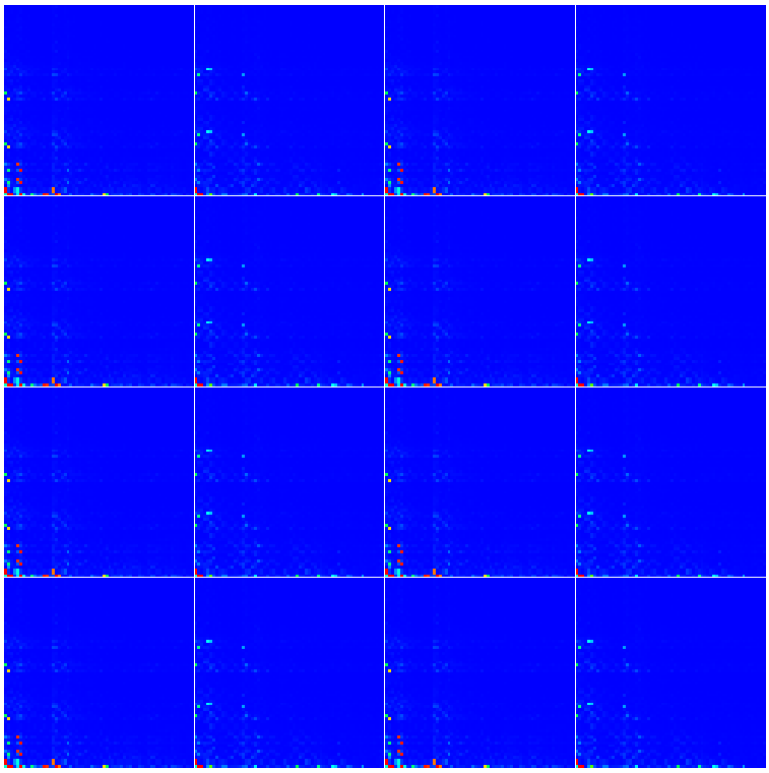


Fig. 7.58: 4D Grover quantum walk result with initial condition 2 (step 25).

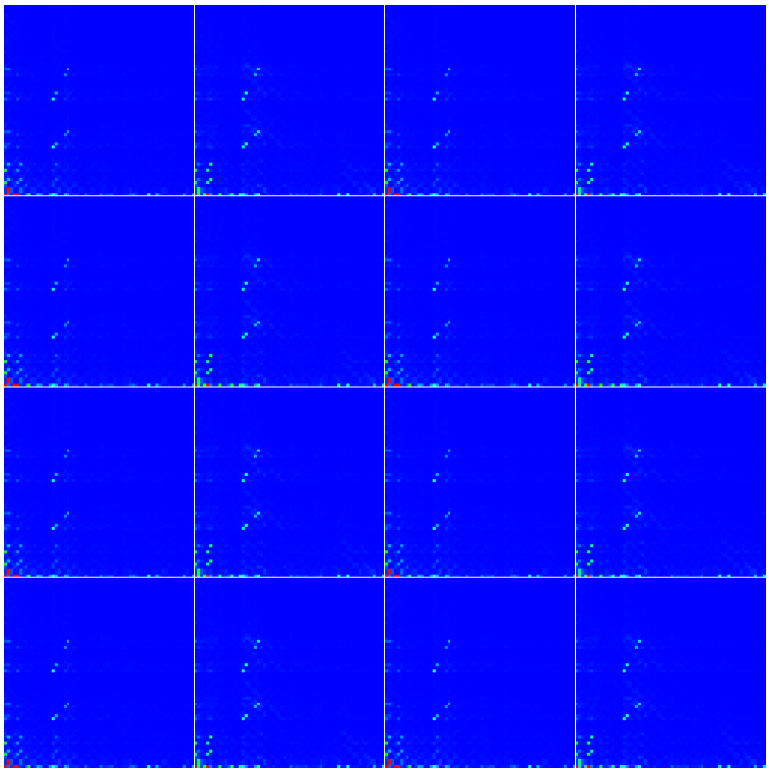


Fig. 7.59: 4D Grover quantum walk result with initial condition 2 (step 30).

■離散フーリエ変換　2 つめの 4 次元量子ウォーク例として，離散フーリエ変換による量子ウォークを取り上げる．ここで用いる 4 次元における離散フーリエ変換 U_F は，以下の式 7.2.19 で与えられる．

$$U_F = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} & \omega^{21} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} & \omega^{28} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} & \omega^{35} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} & \omega^{42} \\ 1 & \omega^7 & \omega^{14} & \omega^{21} & \omega^{28} & \omega^{35} & \omega^{42} & \omega^{49} \end{bmatrix}, (\omega = e^{i\pi/4}) \tag{7.2.19}$$

U_F を量子ウォークの時間変化に用いるユニタリ行列として使用し，アダマール変換のときと同様に式 7.2.16, 7.2.17 で表される初期条件 1，初期条件 2 を初期条件として用いる．提案手法を用いて可視化を行った例が Fig.7.60 から Fig.7.65(初期条件 1), Fig.7.66 から Fig.7.71(初期条件 2) である．

離散フーリエ変換による量子ウォークの確率分布の特徴としては，量子ウォーク一般に見られる分布の複雑性や値の振動が観察されること，アダマール変換やグローヴァーウォークと比べて，対称性があまり見られないこと (おおまかに見れば，対称といえるような領域の組も多く存在する)，初期条件 1, 2 どちらの場合でも，ステップ 20 以降では，$+x_1$ 方向で x_1, y_1 の値が小さい領域などに他の領域と比べて確率の高い箇所が集まっていることが挙げられる．しかし，離散フーリエ変換においては，グローヴァーウォークの場合ほどの値の大きな偏りは見られない．

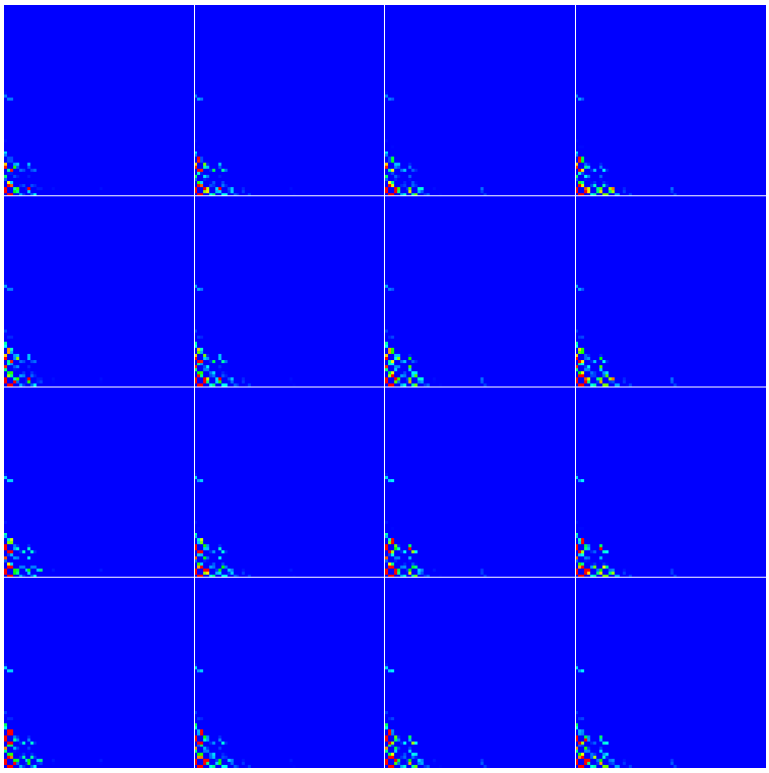


Fig. 7.60: 4D DFT quantum walk result with initial condition 1 (step 5).

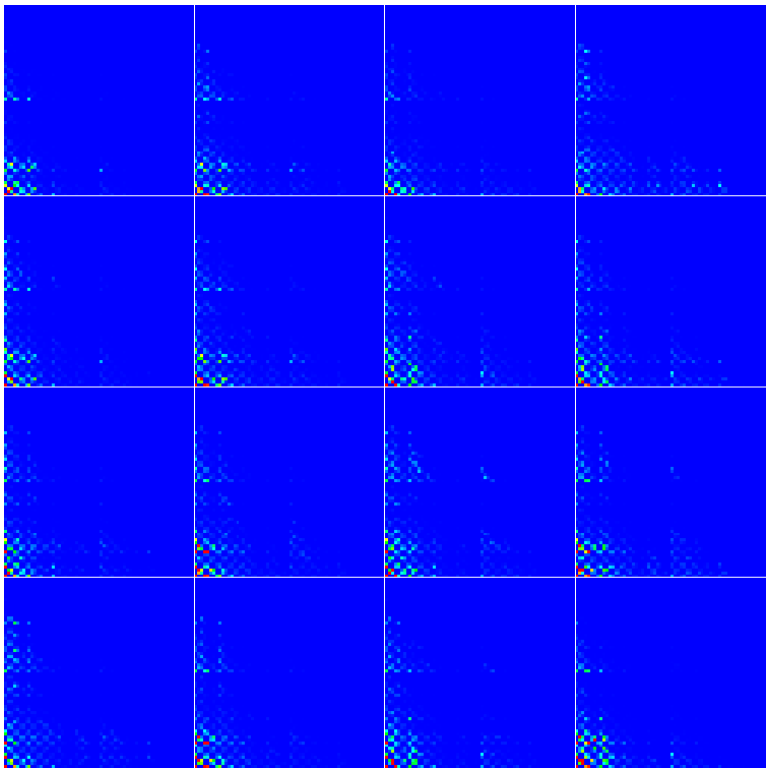


Fig. 7.61: 4D DFT quantum walk result with initial condition 1 (step 10).

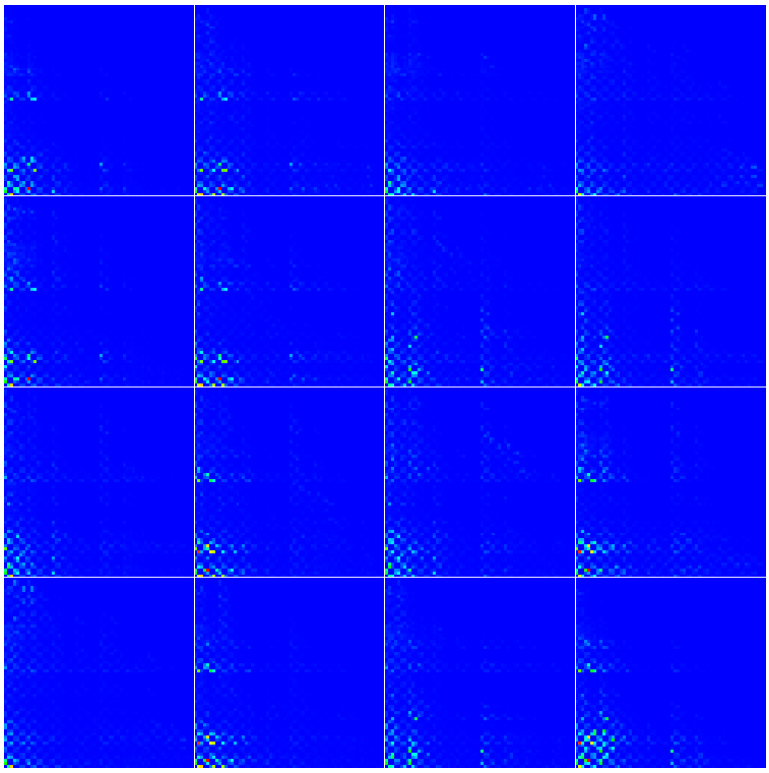


Fig. 7.62: 4D DFT quantum walk result with initial condition 1 (step 15).

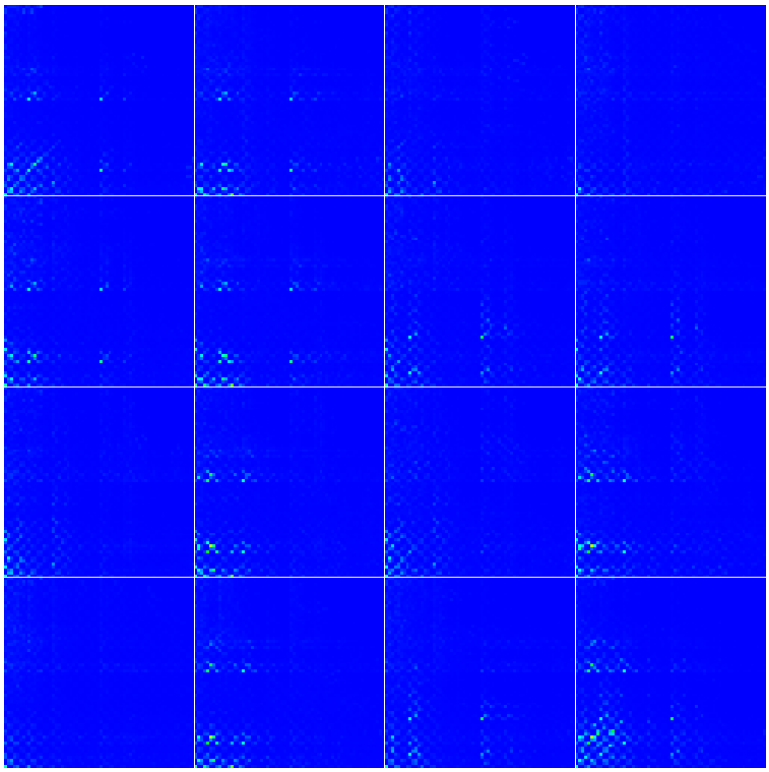


Fig. 7.63: 4D DFT quantum walk result with initial condition 1 (step 20).

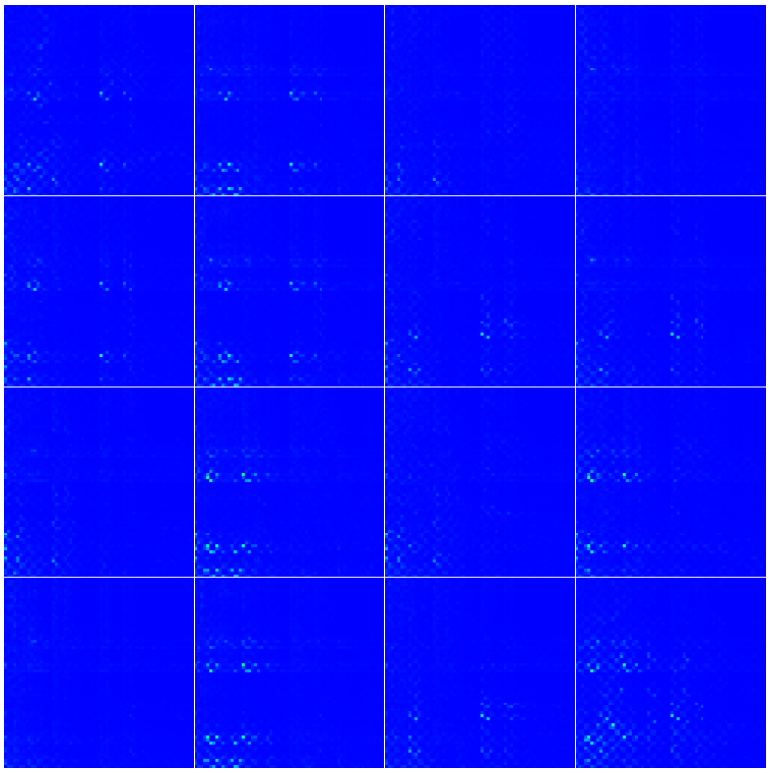


Fig. 7.64: 4D DFT quantum walk result with initial condition 1 (step 25).

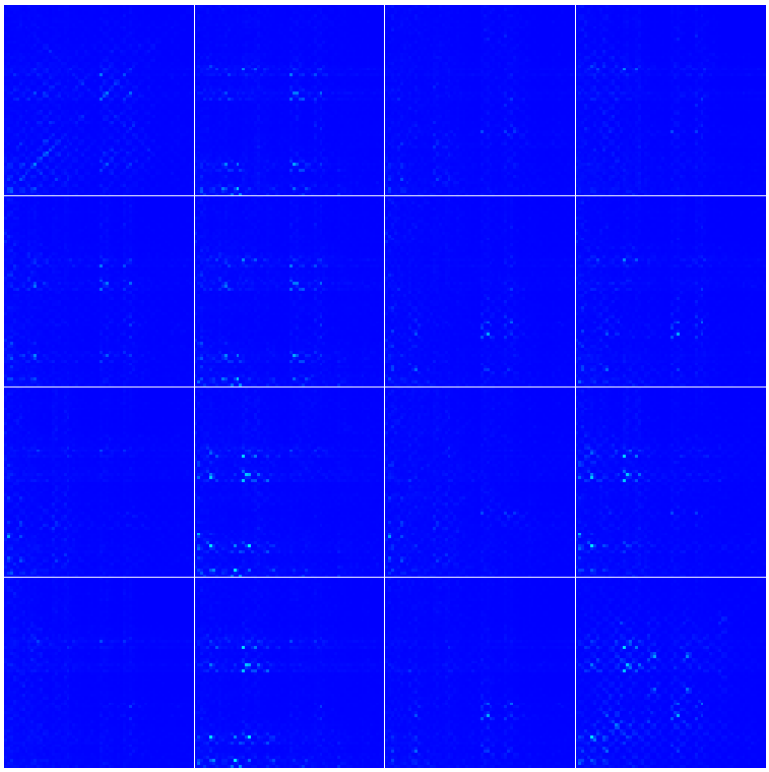


Fig. 7.65: 4D DFT quantum walk result with initial condition 1 (step 30).

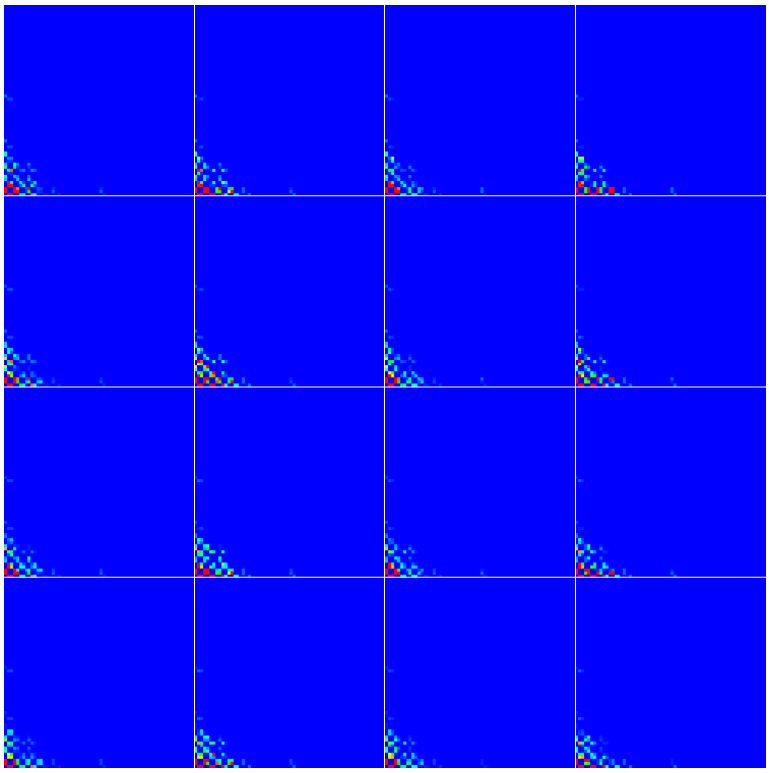


Fig. 7.66: 4D DFT quantum walk result with initial condition 2 (step 5).

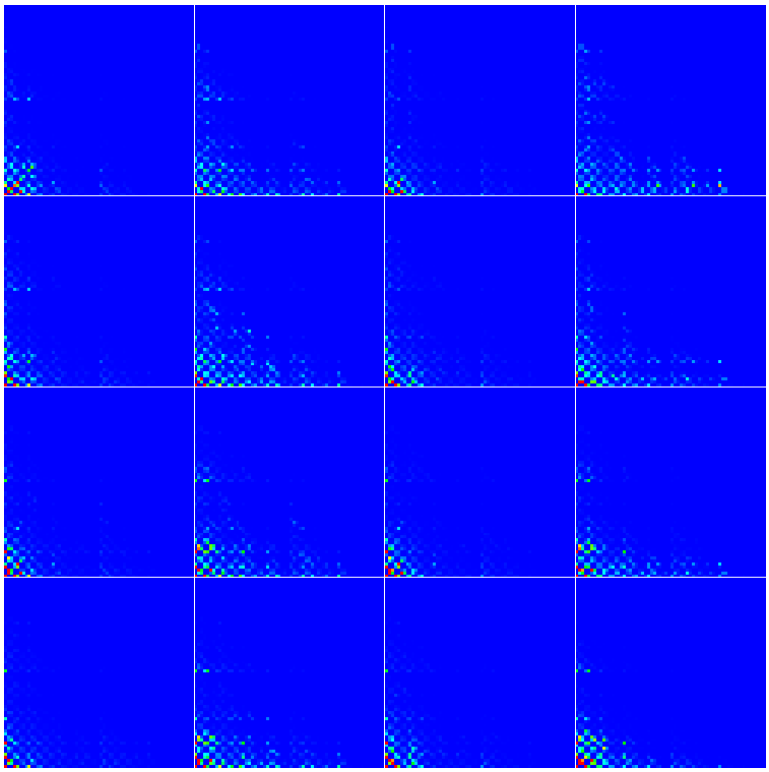


Fig. 7.67: 4D DFT quantum walk result with initial condition 2 (step 10).

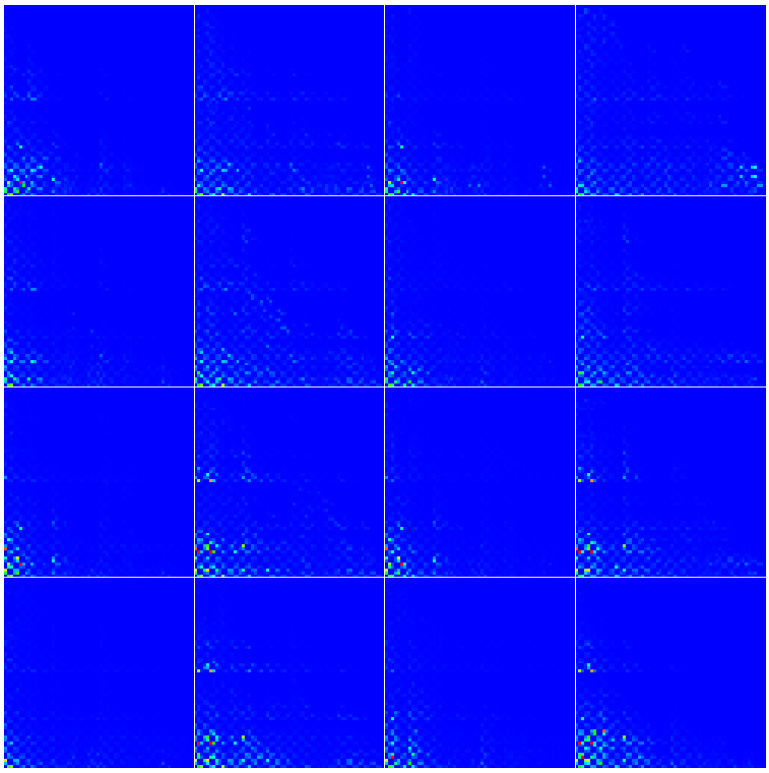


Fig. 7.68: 4D DFT quantum walk result with initial condition 2 (step 15).

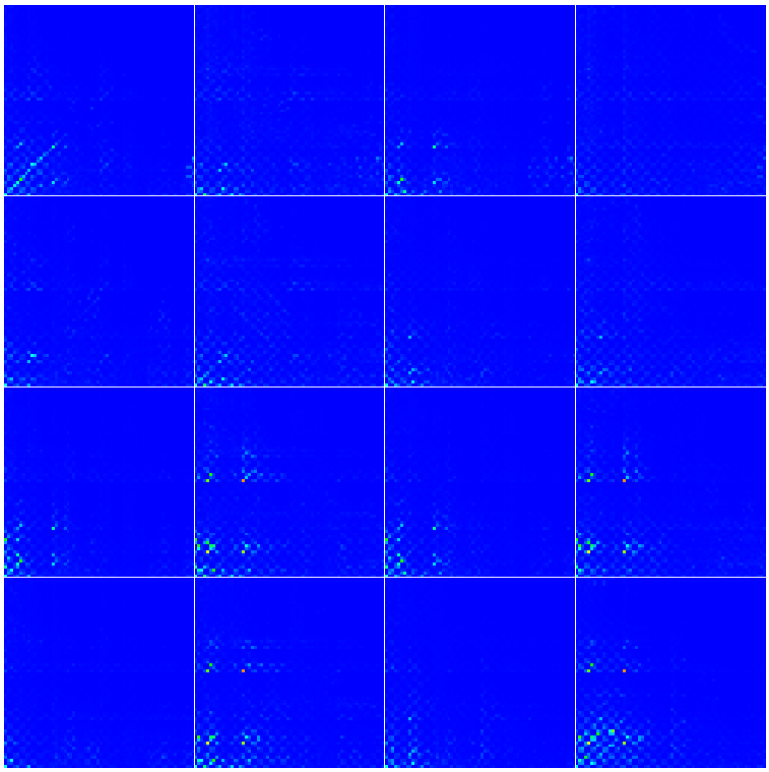


Fig. 7.69: 4D DFT quantum walk result with initial condition 2 (step 20).

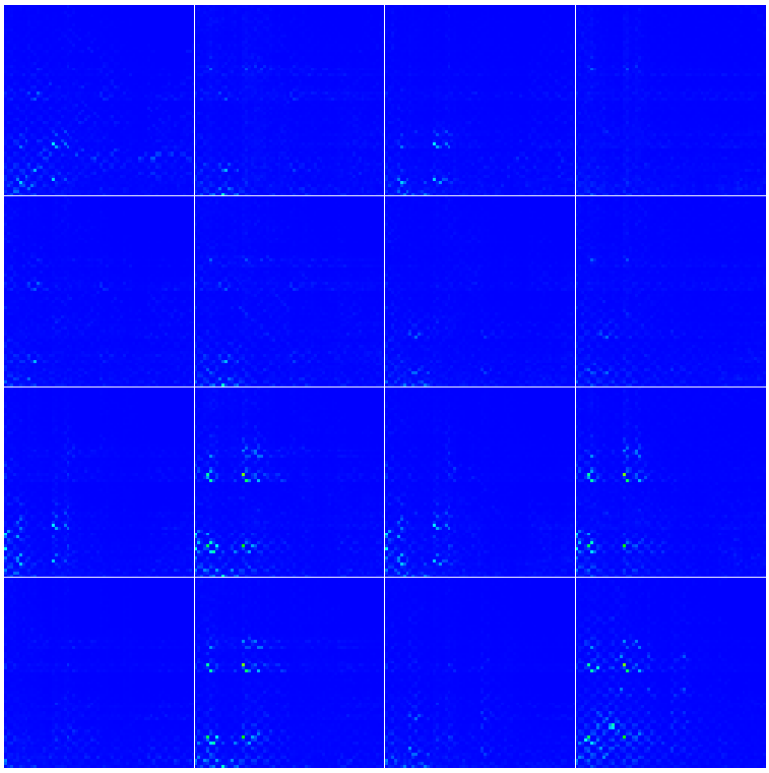


Fig. 7.70: 4D DFT quantum walk result with initial condition 2 (step 25).

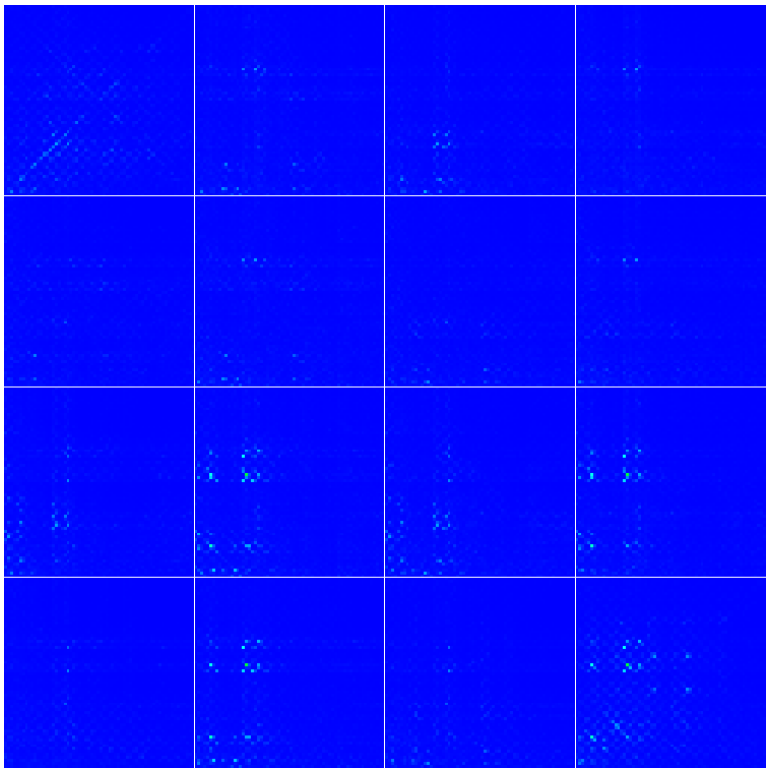


Fig. 7.71: 4D DFT quantum walk result with initial condition 2 (step 30).

本小節では，2 次元および 4 次元の量子ウォークのシミュレーション結果の可視化を行った．本手法によって 4 次元以上の量子ウォークの確率分布などが観測できるようになったことを可視化例を挙げて示した．また，マッピングルールを適切に設定することで，対称性や中心からの拡散の様子を容易に見ることができるようになり，2 次元の場合と比較が簡単にできることも示した．今回は，簡単な量子ウォークの設定での結果と 2 次元の場合との比較を中心に示したが，より多様な設定における比較やより複雑な設定によって数式からは導くことが困難な結果を簡単に観測できるようになる．こうしたことから，本手法は高次元の量子ウォークの理解を効果的に補助することができる可視化手法だと考えられる．

7.3　可視化事例のまとめ

本章では，時系列ボリュームデータの可視化例，4 次元以上の空間における現象の可視化例を示した．これらの可視化事例を通して，提案手法によって空間的遮蔽・時間的遮蔽を解消した要約化された 2, 3 次元画像を得ることができ，データの全体像の把握が容易になることや，高次元現象の可視化に本手法が有効であることを示した．

第 8 章

考察

8.1　本手法の有する特徴

ここでは，本手法の有する特徴を取り上げ考察を行う．本手法は，自己相似図形および再帰的なマッピングルールを用いた次元変換手法であり，次のような長所と短所を有する．

長所
1. あらゆる次元データを低次元化・高次元化できる．
2. 対象データの持つ各要素のもとの次元空間内での階層構造を保持した可視化ができる．
3. 解像度 1 のマッピングルールを決定すれば，全解像度のマッピングができ，なおかつマッピングの結果は解像度 1 のときに決めたルールの性質の傾向を全体的・局所的に持つ．
4. 多解像度化が可能であり，要約的な可視化，描画負荷を低減した可視化ができる．
5. 2 次元以下の次元で可視化を行った場合，オクルージョンのない可視化結果を得ることができる．
6. 次元種類別に次元変換・多解像度化を施した可視化を行うことができる．
7. もとのデータを操作するインターフェースとして利用することや他手法との連携が可能である．
8. データの全体俯瞰に適している

短所
1. もとの次元におけるデータの空間的構造が把握しにくい
2. マッピングルールを理解していなければ，可視化結果の各領域がもとの次元における位置が把握できない．
3. 利用する自己相似図形によってはブランクスペースが生じる

これらの長所・短所をそれぞれ個別に取り上げて，以下では説明する．

8.1.1　長所

あらゆる次元データを低次元化・高次元化できる

本手法を用いると対象データの次元を自在に変換し，低次元化・高次元化して可視化することができる．そのため，低次元化によって空間的遮蔽や時間的遮蔽を解消した可視化や既存手法では表示不可能な

次元データ (4 次元以上の空間を持つデータ) の 2 次元ディスプレイ上での表示を可能にする．高次元化によって低次元データを画面使用率の高い次元で表示することができ，より大規模なデータを限られたスペースで表示することが可能となる．

対象データの持つ各要素のもとの次元空間内での階層構造を保持した可視化ができる

対象データの持つ空間階層的なデータ構造 (例えば，3 次元データなら Octree 構造) をそのまま次元変換した結果に反映して可視化を行う手法になっているため，その空間階層構造を保持した可視化が行われる．そのため，単なるマッピングによる次元変換と違い，階層構造を把握しながらのデータの理解ができる．この性質は，データの大まかな位置や傾向を把握するのに適しているため，高次元や大規模の複雑なデータを理解する際に役立つ．

解像度 1 のマッピングルールを決定すれば，全解像度のマッピングができ，なおかつマッピングの結果は解像度 1 のときに決めたルールの性質の傾向を全体的・局所的に持つ

本手法では解像度 1 のマッピングルールだけを決定して，あとはそのルールを再帰的に適用することで次元変換が可能である．そのため，様々な解像度のマッピングルールを作成する必要がない．また，マッピングルールの再帰的な適用により，マッピングルールの持つ性質が再帰的に自己相似図形上に表れる．その結果，解像度 1 のときに決めたルールの特徴が可視化結果の全体的な傾向と局所的な傾向として表れる．そのため，解像度 1 のときのルールをうまく決めておくことで，効果的な可視化結果を得ることができる．

多解像度化が可能であり，要約的な可視化，描画負荷を低減した可視化ができる

本手法では，マッピングルールの再帰的な適用とマッピング先の図形が再帰的な分割ルールによって作られる自己相似図形であることによって，マッピングルールおよび自己相似図形の分割ルールを一部分だけに再帰的に適用することが可能である．多解像度化を行うともとデータのデータの持つ領域の細かさに応じた可視化 (level-of-detail, LOD 可視化) や関心の高い箇所は高解像度で，関心の低い箇所は低解像度で可視化 (degree-of-interest, DOI 可視化) を行うことが可能である．こうした可視化を行うとその結果が要約的な可視化となり，認知負荷を低減することができる．また，可視化する領域数も減少するため，描画負荷も低減することが可能である．

2 次元以下の次元で可視化を行った場合，オクルージョンのない可視化結果を得ることができる

本手法ではあらゆる次元のデータを 2 次元以下に低次元化することができる．2 次元以下に低次元化した場合には，全データが他のデータの遮蔽を受けることなく表示される．そのため，ボリュームデータのような通常 3 次元可視化手法で表示されるデータの理解を補助することができる．また，視点操作や伝達関数の細かい指定などの特別な操作なしにデータの様子を 1 画面で観察できる．

次元種類別に次元変換・多解像度化を施した可視化を行うことができる

本手法では，次元種類別に次元変換・多解像度化を行うことができる方法も有する．そのため，ある空間次元のデータが時間変化を可視化するような際には，空間と時間の変化を個別に把握しやすくなる．ま

た，各次元種類ごとに多解像度化の閾値を決めることができるので，関心の高い次元は高解像度で表示し，関心の低い次元は低解像度で表示するといった次元別の DOI 可視化や LOD 可視化が行える．

もとのデータを操作するインターフェースとして利用することや他手法との連携が可能である

本手法では，もとの次元における各データの存在領域とマッピング先の領域の間に 1 対 1 のマッピングを行っているため，マッピング先の領域を指定することで，もとの次元の領域にあるデータを指定することができる．指定したデータの値を取得したり，その値を利用して可視化結果に変化を与えるなどといったことが可能であり，もとのデータを操作するインターフェースとして利用できる．また，指定したデータの領域の位置を取得することができるため，他の手法で可視化された結果においてそのデータがどこの領域に表示されているのかといったことを把握しながらデータを見ることができる．このように他手法との連携が可能であり，本手法の持つ長所を活かしながら，他手法によって本手法の短所を補助するような可視化，またはその逆の他手法の短所を本手法によって補助するような可視化が可能である．

データの全体俯瞰に適している

上で述べた長所である次元変換，階層性の維持，マッピングルールの自己相似性，多解像度化を本手法は併せ持つため，任意の次元データをもとの空間位置や階層性を反映した要約的なが 1, 2, 3 次元画像結果を得ることができる．そのため，得られた一画像結果によって，複雑な高次元・大規模データの全体俯瞰ができる．本手法によって全体俯瞰したあと，個別の事象の調査には他の形状表示に強みを持つ可視化手法やその他の解析手法を用いることで，効率的な高次元・大規模データの理解を可能にする．

8.1.2　短所

もとの次元におけるデータの空間的構造が把握しにくい

本手法では次元変換を行うため，もとの次元におけるデータの空間的構造を維持したままの可視化は不可能である．そのため，複雑な空間形状などは本手法では直感的に把握しにくい．空間形状を把握したい場合には，他の可視化手法と連携して表示することを選択するべきである．

マッピングルールを理解していなければ，可視化結果の各領域がもとの次元における位置が把握できない

本手法では，解像度 1 のマッピングルールをもとに次元変換を行うため，解像度 1 のマッピングルールがどのように決められているか理解していなければ，可視化結果の各領域のもとの次元における位置が把握できない．値全体の散らばりなどはルールの理解なしでも把握できるが，位置による値の特徴などといった詳細なデータの理解はマッピングルールを把握していなければ不可能である．

利用する自己相似図形によってはブランクスペースが生じる

本手法では，変換前のデータの次元と変換後のデータの次元の組み合わせによっては空の領域を有する自己相似図形を利用しなければならないことがある．例えば，3 次元から 2 次元化を行うときに利用するシェルピンスキーのカーペットは解像度 1 のとき，全体の 1/9 のブランクスペースを有する．さらに，自己相似図形の性質上，解像度が増加すればするほどブランクスペースも増加する．シェルピンスキーの

カーペットの場合，解像度 2 のときは，$\frac{1}{9}+8\cdot\frac{1}{9}\cdot\frac{1}{9}=\frac{17}{81}$，解像度 3 のときは $\frac{1}{9}+8\cdot\frac{1}{9}\cdot\frac{1}{9}+64\cdot\frac{1}{9}\cdot\frac{1}{9}\cdot\frac{1}{9}=\frac{217}{729}$，解像度 n のときは，$\frac{1}{9}+\frac{1}{9}\cdot\frac{8}{9}+\frac{1}{9}\cdot\left(\frac{8}{9}\right)^2+\cdots+\left(\frac{8}{9}\right)^{n-1}=1-\left(\frac{8}{9}\right)^n$ となり，解像度が大きくなればなるほどブランクスペースの値は 1 に近づく．そのため，できるだけデータ数を表示したいため高解像度で表示を行おうとしたときに，逆にマッピング先はブランクスペースが増え，画面仕様率が低くなるといった状況になる．実際には，利用する解像度の範囲には上限があり，一般的なディスプレイにおける表示範囲では，シェルピンスキーのカーペットの場合解像度 6(729×729 pixels)，解像度 7(2187×2187 pixels) までしか利用しない．解像度 6,7 それぞれの場合，ブランクスペースの割合は 50.6%,56.1% となるため，100% 近くなることはないがかなりのブランクスペースが発生することがある．

8.1.3　本手法の可視化対象

本手法では，あらゆる次元をもつデータを可視化することができるが，上に記した長所・短所より次のようなデータに対して用いる際に特に効果を示すと考えられる．

■高次元または大規模なデータ　高次元データや大規模なデータはデータ数が多いことやその現象が複雑になりやすい．こういったデータは本手法によって高次元化を行い効率的にデータを配置したり，低次元化することによって表示可能にできることや，多解像度化により要約的な可視化を行うことで，データの全体俯瞰ができ，更なる可視化・解析方法の方針を与えることができる．

■ボリュームデータ，時系列ボリュームデータ　ボリュームデータや時系列ボリュームデータは既存研究によってその 3 次元的な可視化方法が多く研究されている．本手法では，そういった手法の持つ空間的遮蔽・時間的遮蔽を解消することができるため，既存手法と連携させて利用することで，既存手法の持つ欠点を補いながら可視化を行うことができる．本手法と連携することで，ボリュームデータ・時系列ボリュームデータのより正確な理解，多角的な視点による理解，容易なデータの操作ができるようになる．

■連続的または多数の離散的な数値を取る多変数を有する現象のデータ　多変数を有する現象のデータにおいて，各変数の変化をそれぞれ一つずつの次元方向とみなし，本手法で低次元化を行うことで，多変数を変化させた結果像を 1 枚の可視化結果内に要約的に表示することができる．さらに次元別に低次元化を行えば，変数を最大 3 つの変数群に分けて扱って表示することが可能である．

第 9 章

結論

本研究では，自己相似図形を利用した次元変換によって，多様な次元データの多解像度可視化を行える手法を開発した．本手法は，高次元データの低次元化の方法，低次元データの高次元化の方法，さらに次元別に次元変換を行う方法を含んでいた．低次元化によって高次元なデータをディスプレイ上で 1, 2, 3 次元画像として表示できるようになり，高次元化によって低次元な大規模データを高い画面仕様率で配置できるようになった．また，これら次元変換に加えて，可視化結果の多解像度化の方法も示した．多解像度化により認知負荷・描画負荷の少ない可視化を実現した．本手法を用いた可視化例として，時系列ボリュームデータに対しての低次元化可視化を行い，空間的遮蔽・時間的遮蔽のない可視化を実現した．また，既存手法では表示ができなかった高次元データの可視化を時系列変化する 5 次元の拡散現象と 4 次元量子ウォークを例として行った．提案手法によって任意の次元データの要約的な 1, 2, 3 次元可視化結果が得られるため，提案手法は高次元・大規模データの全体俯瞰に適している．そのため，本手法と他の形状把握に強みを持つ可視化手法や多様な解析手法とを併用することで，高次元・大規模データの効率的な把握が可能になる．

今後の課題として，第一にユーザビリティテストが挙げられる．本論文では，可視化結果をもとにして本手法の有用性を示したが，更なる本手法の特徴を理解するためにもユーザビリティテストが必要である．ユーザビリティテストの内容としては．既存手法との回答時間や使いやすさ，理解のしやすさなどの比較，本手法によって理解可能な空間形状の調査，同一データに対して多数の次元変換の方法による可視化結果を示し，各データに対して最も適切な可視化方法の確認，多解像度化による回答時間などの変化，などが挙げられる．また，他の可視化手法や解析手法の連携も今回は 7.1.3 節で簡単なものを取り上げたが，他手法との効果的な連携方法や，他手法と連携させることによる回答時間の変化など更なる研究が必要である．その他にも，本手法によって 4 次元以上の空間を持つ高次元データが可視化可能になったため，より多様な高次元現象の可視化を行い高次元データの可視化方法を探索することや，今回は取り上げなかった情報的可視化の対象となっているデータに対しての本手法の適用可能性，有用性の有無を確認することなどがあげられる．

参考文献

[1] H. Akiba and K. Ma. A Tri-Space Visualization Interface for Analyzing Time-Varying Multivariate Volume Data; Eurographics. In *IEEE-VGTC Symposium on Visualization*, 2007.

[2] C.L. Bentley and M.O. Ward. Animating multidimensional scaling to visualize N-dimensional data sets. In *Proceedings of the 1996 IEEE Symposium on Information Visualization (INFOVIS'96)*, p. 72. IEEE Computer Society, 1996.

[3] R.A. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. *ACM Siggraph Computer Graphics*, Vol. 22, No. 4, p. 74, 1988.

[4] Z. Du, Y.J. Chiang, and H.W. Shen. Out-of-core volume rendering for time-varying fields using a space-partitioning time (SPT) tree. In *Visualization Symposium, 2009. PacificVis' 09. IEEE Pacific*, pp. 73–80. IEEE, 2009.

[5] T. Fujiwara, R. Matsushita, M. Iwamaru, M. Tange, S. Someya, and K. Okamoto. Fractal Map: Fractal-Based 2D Expansion Method for Multi-scale High-Dimensional Data Visualization. *Advances in Visual Computing*, pp. 306–315, 2010.

[6] A.J. Hanson. A Construction for Computer Visualization of Certain Complex Curves. *Anthology Notices of the American Mathematical Society Year*, Vol. 41, No. 9, pp. 1156–1163, 1994.

[7] M.C. Hao, U. Dayal, D.A. Keim, and T. Schreck. *Multi-resolution techniques for visual exploration of large time-series data*. Citeseer, 2006.

[8] S. He, R. Dai, B. Lu, C. Cao, H. Bai, and B. Jing. Medial Axis Reformation:: A New Visualization Method for CT Angiography. *Academic Radiology*, Vol. 8, No. 8, pp. 726–733, 2001.

[9] W.H. Hsu, J. Mei, C.D. Correa, and K.L. Ma. Depicting Time Evolving Flow with Illustrative Visualization Techniques. *Arts and Technology*, pp. 136–147, 2010.

[10] N. Inui, Y. Konishi, and N. Konno. Localization of two-dimensional quantum walks. *Physical Review A*, Vol. 69, No. 5, p. 52323, 2004.

[11] B. Johnson and B. Shneiderman. Tree-Maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd conference on Visualization'91*, pp. 284–291. IEEE Computer Society Press, 1991.

[12] A. Kanitsar, D. Fleischmann, R. Wegenkittl, P. Felkel, and ME Gr
"oller-CPR. Curved planar reformation. In *IEEE Visualization*, pp. 37–44, 2002.

[13] A. Kaufman and K. Mueller. Overview of volume rendering. *The Visualization Handbook*, pp. 127–174,

2005.

[14] D.A. Keim. Pixel-oriented visualization techniques for exploring very large data bases. *Journal of Computational and Graphical Statistics*, Vol. 5, No. 1, pp. 58–77, 1996.

[15] D.A. Keim. Designing pixel-oriented visualization techniques: theory and applications. *Visualization and Computer Graphics, IEEE Transactions on*, Vol. 6, No. 1, pp. 59 –78, jan-mar 2000.

[16] D.A. Keim, M.C. Hao, U. Dayal, and M. Hsu. Pixel bar charts: a visualization technique for very large multi-attribute data sets. *Information Visualization*, Vol. 1, No. 1, pp. 20–34, 2002.

[17] D.A. Keim and H.P. Kriegel. VisDB: Database exploration using multidimensional visualization. *Computer Graphics and Applications, IEEE*, Vol. 14, No. 5, pp. 40–49, 2002.

[18] Hideki Koike. Fractal views: a fractal-based method for controlling information display. *ACM Trans. Inf. Syst.*, Vol. 13, No. 3, pp. 305–323, 1995.

[19] O.D. Lampe, C. Correa, K.L. Ma, and H. Hauser. Curve-Centric Volume Reformation for Comparative Visualization. *Visualization and Computer Graphics, IEEE Transactions on*, Vol. 15, No. 6, pp. 1235–1242, 2009.

[20] M. Levoy. Display of surfaces from volume data. *IEEE Computer graphics and Applications*, Vol. 8, No. 3, pp. 29–37, 1988.

[21] K.L. Ma. Visualizing time-varying volume data. *Computing in Science & Engineering*, Vol. 5, No. 2, pp. 34–42, 2003.

[22] T.D. Mackay, S.D. Bartlett, L.T. Stephenson, and B.C. Sanders. Quantum walks in higher dimensions. *Journal of Physics A: Mathematical and General*, Vol. 35, p. 2745, 2002.

[23] W.H. Mak, M.Y. Chan, Y. Wu, K.K. Chung, and H. Qu. VoxelBars: An Informative Interface for Volume Visualization. *Advances in Visual Computing*, pp. 161–170, 2008.

[24] C. Moore and A. Russell. Quantum walks on the hypercube. *Randomization and Approximation Techniques in Computer Science*, pp. 952–952, 2002.

[25] P. Sabella. A rendering algorithm for visualizing 3D scalar fields. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, p. 58. ACM, 1988.

[26] H.W. Shen, L.J. Chiang, and K.L. Ma. A fast volume rendering algorithm for time-varying fields using a time-space partitioning (TSP) tree. In *Proceedings of the conference on Visualization'99: celebrating ten years*, pp. 371–377. IEEE Computer Society Press, 1999.

[27] M. Tsubokura, T. Kobayashi, T. Nakashima, T. Nouzawa, T. Nakamura, H. Zhang, K. Onishi, and N. Oshima. Computational visualization of unsteady flow around vehicles using high performance computing. *Computers & Fluids*, Vol. 38, No. 5, pp. 981–990, 2009.

[28] C. Upson and M. Keeler. V-buffer: visible volume rendering. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, p. 64. ACM, 1988.

[29] C. Wang and H.W. Shen. Lod map-a visual interface for navigating multiresolution volume visualization. *IEEE Transactions on visualization and computer graphics*, pp. 1029–1036, 2006.

[30] M. Wattenberg. A note on space-filling visualizations and space-filling curves. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pp. 181–186. IEEE, 2005.

[31] C.C. Yang and F.L. Wang. Fractal summarization for mobile devices to access large documents on the web. In *Proceedings of the 12th international conference on World Wide Web*, pp. 215–224. ACM, 2003.

[32] C.C. Yang and F.L. Wang. An information delivery system with automatic summarization for mobile commerce. *Decision support systems*, Vol. 43, No. 1, pp. 46–61, 2007.

[33] Jing Yang, Daniel Hubball, Matthew O. Ward, Elke A. Rundensteiner, and William Ribarsky. Value and relation display: Interactive visual exploration of large data sets with hundreds of dimensions. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 13, pp. 494–507, 2007.

[34] 岩丸雅紀. フラクタル図形を利用した 3 次元構造データのマルチレゾリューション可視化に関する研究. 新領域創成科学研究科修士論文, 2009.

[35] 岩丸雅紀, 岡本孝司. フラクタル図形を用いたボリュームデータの 2 次元展開. 第 36 回可視化情報シンポジウム講演論文集, pp. 75–78, 2008.

[36] 岩丸雅紀, 岡本孝司. フラクタル図形を利用したボリュームデータの可視化手法とその応用. 可視化情報学会全国講演会講演論文集, pp. 167–168, 2008.

[37] 岩丸雅紀, 岡本孝司. ボリュームデータ可視化装置および方法並びにプログラム. 特開 2009-289144, 2009.

[38] 岩丸雅紀, 岡本孝司. 3 次元位置指定装置、3 次元位置指定方法および 3 次元位置指定用プログラム、ならびにボクセルモデリング装置、ボクセルモデリング方法およびボクセルモデリング用プログラム. *WO2010-029953*, 2010.

[39] 今野紀雄. 量子ウォークの数理. 産業図書, 2008.

[40] 藤原孝紀, 岩丸雅紀, 岡本孝司. 高次元データ可視化装置および方法ならびにプログラム. 特願 2009–237504, 2009.

[41] 藤原孝紀, 丹下学, 染矢聡, 岡本孝司. フラクタル図形を用いた高次元データの 2 次元展開可視化手法. 可視化情報学会全国講演会講演論文集, pp. 319–320, 2009.

[42] 藤原孝紀, 丹下学, 染矢聡, 岡本孝司. フラクタル図形を用いた時系列ボリュームデータの低次元化可視化手法. 第 38 回可視化情報シンポジウム講演論文集, 2010.

[43] 永田靖, 棟近雅彦. 多変量解析法入門. サイエンス社, 2001.

発表論文・特許リスト

国際会議

1. T. Fujiwara, R. Matsushita, M. Iwamaru, M. Tange, S. Someya and K. Okamoto, “Fractal Map: Fractal-Based 2D Expansion Method for Multi-Scale High-Dimensional Data Visualization”, Advances in Visual Computing, pp.306-315, 2010.

国内学会発表

1. 藤原孝紀, 丹下学, 染矢聡, 岡本孝司, 「フラクタル図形を用いた時系列ボリュームデータの低次元化可視化手法」, 第 38 回可視化情報シンポジウム講演論文集, 2010.
2. 藤原孝紀, 丹下学, 染矢聡, 岡本孝司, 「フラクタル図形を用いた高次元データの 2 次元展開可視化手法」, 可視化情報学会全国講演会論文集, pp. 319-320, 2009.

特許出願

1. 藤原孝紀, 岩丸雅紀, 岡本孝司, 「高次元データ可視化装置および方法ならびにプログラム」, 特願 2009-237504, 2009.

謝辞

なかなか正体がはっきりしない自己相似図形というものを使って，さらに正体がはっきりしない多次元データを可視化するという今回の研究を，なんとか今はっきりした形としてここに結果をまとめることができました．自己相似図形や多次元データにまいることなく研究を続け，論文を結ぶことができたのは，多くの方々にご指導・ご協力をいただいたおかげです．この場を借りて，お礼申し上げます．

岡本先生には，やり甲斐のある研究と自由に研究できる研究室の雰囲気・環境，的確な助言，幾度にも及ぶ刺激的な挑戦の機会を与えて頂き大変感謝しています．思い返せば，本当に多様な経験ができました．国内学会，国際学会，特許申請・紹介，学部生相手の講義，スウェーデンの可視化研究者との交流，ジャーナルへの挑戦，まだまだ色々あったと思いますが，どれも大変刺激的で今後の人生においても忘れられない貴重な経験であったと思います．本当にありがとうございました．

染矢先生には，研究の方向性を掴むのにいつも困っていた自分に対して，常にアドバイスをいただきました．研究面だけではなく，その他の面でも大きな迷惑をかけたと思います．染矢先生の熱心な研究姿勢とご指導により，弱音を吐くことなく研究が続けられたと思います．どうもありがとうございました．

また，この 2 年間岡本・染矢研でお世話になった先輩・後輩・同期の皆様にも感謝しています．人数の多い研究室は居室内も居室外でもいつもわいわいできるような環境で，とても充実した 2 年間だったと思います．飲み会，サークル，研究室旅行，どれも楽しめました．2 年の研究室旅行の深夜起きたことは忘れられません．

特に，衛生面でお世話になった緒方先輩には，大変感謝しています．

最後に、至らぬ面の多い自分をいつも支えてくれた家族・友人に感謝します。

2011 年　1 月　25 日

二羽の鳴き声を聞きながら

藤原　孝紀