グラフィクスとCAD29－9
(1987. 10. 1)

コンピュータグラフィックスによる景観シミュレーション

Computer Graphics for Scene Simulation

秋葉 澄伸　山本 強　青木 由直

Suminobu AKIBA　Tsuyoshi YAMAMOTO　Yoshinao AOKI

北海道大学工学部

Faculty of Engineering , HOKKAIDO univ.

あらまし　都市景観のシミュレーションを行なう場合、都市モデルが広範囲かつ大量のデータを必要とするため、単純なレンダリング手法では記述・計算効率が良くない。さらに、人工物だけではなく街路樹や遠景といった景観に大きな役割を持つものを効果的に作画する必要がある。　本システムでは、都市データは簡潔な階層構造に抑え、写真などのマッピングデータをリンクさせたデータ構造を用いた。複数のレンダリングアルゴリズムを適当に合成して、建築物や樹木を精密かつ効果的に表現できる。ＵＮＩＸワークステーション上の試作システムにより、広範囲の都市モデリングと景観の作画を行なった。

Abstract　It is laborious to simulate a city scene with the large number of data which coverd a wide range of city . Further , street trees and other natural matter play an important role in a city scene .

We use simplified models of city and link some locaton data to picture data for mapping ; and combine several result from different rendering algolithm to express natural matter and buildings minutely or efficiently . We will introduce an implementation of pilot system on UNIX machine and a few of our result .

1　はじめに

都市の再開発などは、費用や社会的影響の面から再試行が困難であり、そのシミュレーションには模型などを作成する。模型は作成のための費用・時間が大きく、検討のフィードバックがむつかしい。このような分野に、コンピュータグラフィックスの有効利用がより望まれる。

景観の検討には、ＣＧを使ったモンタージュ技法を用いたこころみがあり、実際に使用されている。これは、現存する背景の一部に建造物を追加する場合などには効果的である。しかし、広範囲のプランニングの全体像を必要とする場合や、現在存在しない景観を設計する場合には、景観検討のツールとして不十分である。また、モンタージュ技法は一般に２次元画像の合成により行なわれているものがおおく、そのままでは自由な照明や視点の移動について対応することはむずかしい。

都市構造のデータをつくり、ワイヤーフレーム、スキャンラインアルゴリズムなどで作画したものや、更にレイトレーシングアルゴリズムなどを用いてより微細な景観をシミュレートしようとする試みも見られる。ワイヤーフレームなどによる作画は処理時間の面で有利であり、得られる画像からは都市構造の全体像などを知

ることができるが、景観の検討には画質として不十分である。サーフェースおよびソリッドモデルを用いて、精密な都市構造のモデリングを行なえば高品位の作画を行うことができる。この場合、モデリングデータの量が増加するにつれてシステムの規模が膨大となり、実用的には使用環境が限られている。

さらに、レートレーシングを用いると、表面材質や陰影処理をもちいた精密な作画が可能である。しかし、都市景観のようにデータが広範囲に分散するようなモデルでは効率的な計算法がまだ確立されていない。このため、局所的な環境設計にはきわめて有効であるが、景観のシミュレーションには低効率である。

2　本システムの基調

本システムは、ＵＮＩＸワークステーション程度の小規模なハードウエア上で稼働できることを前提として設計した。その計算速度・記憶容量の制限からモデリングデータは簡便なものとならざるを得ないが、複数の作画アルゴリズムを合成して精密な都市景観を作画する事をことを試みた。

本システムでは複数の作画アルゴリズムを使用しているが、都市モデルのデータベースは一本化されており、その簡便なデータ構造と相まって管理は容易である。都市モデルのデータベースの一部には位置情報のほかにマッピングピクチャデータへのリンクを持っている。このデータから、スキャンラインアルゴリズムによる建築物・地形の作画、テクスチャマッピングによる自然構造物（樹木、複雑な壁面）の作画を独立に行い、各プロセスからの結果を統合して最終的な出力画像を得ている。

本システムを用いて実在する都市もモデリングを行い作画を行った結果、実用上十分な品質のシミュレーション画像が作成できることがしめされた。

3　景観シミュレーションの問題点

都市景観シミュレーションを行う際の問題点の一つは「都市構造のモデリング方法」である。

ある街区を対象にデータ化を試みると、その面積は数ｋｍ^2、家屋数は数１００軒に及ぶ。人間の視点から景観の作画を行なうことを考えれば、個々の家屋の構造もある程度記述する必要がある。景観のなかで重要な構造物の実物感を表現するためには、かなり精密なモデリングが必要である。このように考えると、標準的なサーフェースモデルによってモデリングするとして、家屋あたり１００パッチ以上必要となると思われる。全データ量は数十万パッチ以上のオーダがないと十分な景観シミュレーションとはならないであろう。

もう一つの問題点は、樹木などの自然造形物や遠景といった非人造物の作画方法である。街路樹などは都市景観の検討において、重要な位置を占めているが、これをサーフェースモデルによって精密にモデリングすることはほとんど不可能である。フラクタル技法を用いてアルゴリズム的に作画することは可能であるが実在の樹木を高速に作画することは困難である。

また、遠景を作画するためには、数１００ｋｍ^{2}にわたって地形の状態をモデリングする必要がある。通常のサーフェースモデルでモデリングを行なえばきわめて多量の情報が必要となるため、地形情報の性質にあったモデリング手法が必要である。

4　本システムのモデリング

本システムでは、都市構造のデータは木構造を用い、しかもデータ量を削減するために必要最小限の簡潔な構造表現にとどめた。単純なモデルから精密な作画を行なうために、テクスチャマッピングを併用したモデリングを採用した。

複雑な表現構造を持つ壁面や、街路樹などの自然造形物は、あらかじめ正面からの写真をビットイメージとしてデータベース化しておく。これらのものは、都市構造のデータのなかでは１枚の長方形要素として取り扱われ、構造的には単純なモデルとなっているので立体感は無いが極めて精密な画像がえられる。建造物の表面の構造的な立体感が意味を持つ距離まで視点を近づけると問題が生ずるが、そのような視点は景観という概念とかけ離れているため考慮していない。

一般にビットイメージはデータ量が膨大で、

二次記憶装置の負担となることが危惧されるが、多くのテクスチャデータは共有されることが多く、景観に重要なイメージほど高い解像度で、重要でないものは低い解像度で記録しておけばよいためＵＮＩＸシステムのファイル装置で取り扱える範囲に十分納まっている。

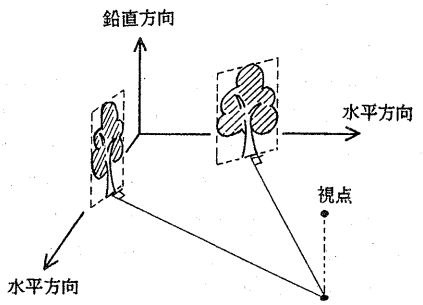


図１　樹木等の姿勢決定

5　本システムのレンダリング

基本的には従来のテクスチャマッピング技法を用いている。しかし、本手法のモデリングに適応するため、いくつかの改善がなされている。

樹木などの複雑な形状を持つものも、壁面と同様に一枚の長方形要素と考えて、テクスチャマッピングによって作画する。樹木などは単純な壁面とは異なり、その姿勢は鉛直方向に１軸が固定されており、もう１軸は作画時に視点とその樹木を結ぶ直線に垂直になるよう動的に決定される。樹木は実体は平面であるが、姿勢を常に視点方向へ向けるため比較的に自然な状態で作画される。電信柱などの回転対称である構造物も同様にして作画される。

この方法では、上空から樹木を見おろした時には樹木に厚みがなく不自然であるが、人間の視点の高さは２ｍ以下なので、街路樹などの景観の検討の場合は問題とはならない。

6　本システムの処理の流れ

6．1　構造データの構築

前述したように、都市構造を記述するデータベースはそれ自体に階層構造を持っている。本システムではその特性を利用して、トリー状のデータ表現の用いて、構造・テクスチャ・地形データを記述する。図１はデータ記述の形式を記述したものである。各レベルのマクロ構造は、それぞれのローカルな原点を中心に記述されており、絶対位置はルートからトップダウンに展開された時点で最終的に決定される。また、各レベルの記述様式は基本的には同じであり再帰的な定義も可能であるが、都市景観データ階層とデータベースのレベルの対応を明確にするためデータベースの各レベルには city, town, block, house といった名称がつけられている。

地形情報は構造的に特殊であり、トップレベルに対する付加的な情報として取り扱われる。具体的にはトリーのトップレベルの記述において ground 要素として一定間隔の標高データのマットリックスを記述してある。地形を標高マットリックスで表現するのは精度の点で問題があるが、このようなデータ構造に対しては特別なアルゴリズムで高速のレンダリングを行なう事も可能である。

```
City_file   :=  $ground  GROUND_FILE_NAME   Offset_X Y     ...
            |   $town    TOWN_FILE_NAME     Offset_X Y Z  Theta   ...

Town_file   :=  $block   BLOCK_FILE_NAME    Offset_X Y Z  Theta   ...

Block_file  :=  $house ID FILE_NAME/HOUSE_TYPE Offset_X Y Z  Theta   ...

House_file  :=  Type  rect      X0 y0 z0  x1 y1 z1  x2 y2 z2  color_code
                      triangle  X0 y0 z0  x1 y1 z1  x2 y2 z2  color_code
                      texture   x0 y0 z0  x1 y1 z1  x2 y2 z2  texture_file
                      plate     x0 y0 z0  x1 y1 z1  x2 y2 z2  texture_file
                :
```

図２　都市データの構造

6．2　構造ベースの展開

作画段階でデータベースはルートから展開により、基本プリミティブまで分解され、各プリミティブの絶対座標が決定される。各基本プリミティブは展開後、対応する作画アルゴリズム別に3個の中間ファイルに分解される。基本プリミティブのうち rect（四角形パッチ）、triangle（三角形パッチ）はスチャンラインアルゴリズムによって処理される。また、基本プリミティブ texture および tree は、対応するビットイメージデータをデータベース中に用意しておく。

6．3　作画

本システムは構造物に対するスキャンラインアルゴリズムによる作画、テクスチャマッピング及び地形データの処理の3プロセスが独立に行なわれる。それぞれのプロセスからの出力を合成して最終的な画像を得る。一連のデータの流れと各プロセスの関連を図2に示す。各プロセスはおのおの専用の中間データと、同一の視点・視軸・画角および照明状態（太陽光線の方向）を作画パラメータとして与えられる。各プロセスが使用するモデリングデータの中間形式は異なるが、それらは全て同一のデータベースから抽出されたものである。また、各プロセスの出力ファイルも最終段階での合成のためにフォーマットが統一されている。各プロセスの処理内容を大まかに説明する。

●スキャンライン処理

データベースの図形要素のうち rect,triangle の2種のプリミティブはスャンラインアルゴリズムによって、隠れ面処理およびシェーディング処理が行なわれる。計算時間を節約するために、現在のところ影付けは行なっていない。

●テクスチャマッピング処理

texture 及び tree 要素はテクスチャマッピングプロセスが処理を行なう。それらが参照するビットイメージは長方形マットリックスであるが、不定形の対称物（樹木など）への張り込みを可能とするために、ビット属性として透明を与えることができる。その結果、予めビットイメージに対して編集を施すことによって、三角形の壁面へのマッピングや樹木などの複雑な形状の自然な表現が可能となる。

マッピングアルゴリズムは、張り込まれる長方形パッチの作画スクリーンへ投影した各ピクセルから、ビットイメージに対して逆投影をほどこし、対応するピクセルの値をスクリーンに反映させるというものである。したがって処理時間はマッピングされるスクリーン上のピクセル数にほぼ比例する。

●地形情報の作画

ground要素は標高マットリクスで地形情報を与える。現在は地形情報も三角形パッチに変換しズキャンラインアルゴリズムにより作画を行なっている。しかし、地形に関しては任意の構造物とは異なり、鉛直方向の重なりが無いなどの拘束があるためより高速の処理が可能であるが、これは今後の課題である。

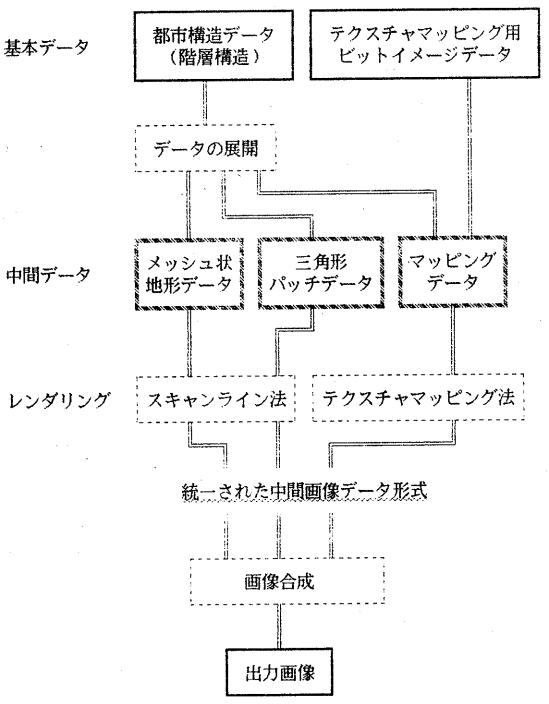
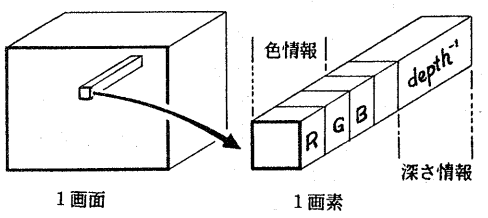


図3　システムの構成

各プロセスはそれぞれ、図４に示す形式の画像データを出力する。このデータの持つ情報はピクセルの輝度情報と「深さ」情報である。これはZ-buffer法のDepth情報にあたるものである。たいていのレンダリング手法は隠面処理のために視点からモデリング要素までの距離情報を使用するが、これを作画出力に残したものである。（ただし、透視変換後にＤＤＡを使用して深さ情報を計算するプロセスが多いため、単純なDepth情報ではなく、1/Depth、つまり深さ情報の逆数が残されている。）




```
struct color {
    BYTE red, blue, green, alias ;
} ;

struct pixel {
    struct color intensity ;
    float depth ;
} screen [480][640] ;
```

図４　画像データの中間形式

各プロセスから得られた出力は、各画素のDepth情報を比較することで、まったく異なる作画アルゴリズムによるものでありながら、容易に合成することができる。これが最終的な出力画像となる。

7　応用例

小樽市を対象にしておこなった都市景観のシミュレーションを例として、作画結果と処理の概要をしめす。（小樽市は北海道の地方都市としては長い歴史を持ち歴史的な建造物が数多く存在するため、それらを活かした都市再開発が求められている。）

地形データは５０ｍグリッドの標高値を入力し、中心部の建造物、街区の構造データを作成した。さらに、歴史的な建造物と重点的にシミュレーションを行なう街区について、建築物の正面写真のビットイメージ化を行いそれらを総合して小樽市に関する都市景観データベースを構築した。データベースは構造データ・地形データに関してはテキストベースであり、容量は約８４０ＫＢであった。テクスチャマッピングで使用されるビットイメージは平均２００×２００ｐｉｘｅｌ程度のイメージを約１００枚使用し、全体で５ＭＢ程度の容量となった。

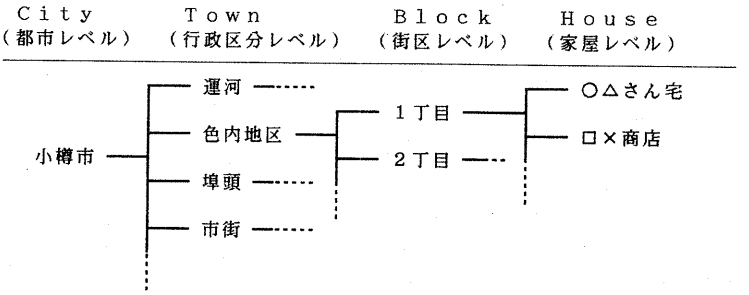


図５　小樽市のデータ構造例

いくつかの作画例についての作画時間の一覧を表１に示す。使用マシンは　ＤＣＬ社製　Ｕｓｔａｔｉｏｎ－Ｅ１５　という、ＭＰＵにＭＣ６８０００を使用した古典的なＶｅｒｓａバスマシーンであり、Ｓｋｙ社のＦＰＰを装着している。システムの記述言語はＵｎｉｐｌｕｓのＣである。表示ハードウエアはＶｅｒｓａバス直結のビットマップディスプレーのほかに、独立したフレームバッファと画像用ストリーマがある。

処理時間は一画像につき数１０分を要すが、これはマシンの処理能力の問題でもあり、最新の３２ｂｉｔ－ＭＰＵを使用すれば現在の数分の１まで短縮することができる。

８　今後の課題

今回の実験システムによる画像生成を行なった結果、今後改善されなければならないいくつかの問題点があきらかとなった。

第１に、現在のところ本システムではいずれのレンダリング手法についても、アンチエーリアシング処理をおこなっていない。精密な作画を行なうほどアンチエーリアシングの要求が大きくなる。

テクスチャマッピング時のアンチエイリアシング及びスムージングは特に問題となる。現在用いているマッピング処理は、単純にスクリーンからのビットイメージへの逆投影を行なっているものである。このような処理では、視点近くにマッピングされた壁面がある場合にビットイメージの１ｐｉｘｅｌがスクリーン上でかなり大きな面積を占め、ブロック状になって不自然である。同様に、遠方にマッピングされた場合もエイリアシングによりモアレが発生する。アニメーションを作成した場合などは、とくに画質が劣化する。

近景へのマッピング時のブロックの発生に対してはスムースシェーディングと同様の補間によって、遠景時のエイリアシングに対してはアベレージング処理によって対応できる。しかし、視点の移動に対する自由度が大きいため、マッピング平面が視軸に対して平行に近い場合などピクセルのｘ方向がブロック状でｙ方向がエイリアシングを生じるという場合もあり、単純ではない。こういった状況にも対応できる統一的なスムージングアルゴリズムの開発が必要である。

第２に、データベースが巨大であるため、その構築を支援するシステム全体の環境整備にたいする要求は大きい。今回は作画アルゴリズムの開発が主体であったためデータベース記述はテキストで行なわれたが、シミュレーション規模が拡大してくると対応は困難となる。

「景観」は構造モデリングとは異なり、厳密さが要求される部分と雰囲気が表現されればよい部分とが混在している。従ってあらかじめ標準的な景観の要素、例えば各種の並木・緑地・家屋などを用意しておき必要に応じて引用できるようなデータベースの整備が有効である。

以上のように実用システムとしての課題は多いが、本システムのモデリング及び作画手法の本質に影響するものではない。

９　むすび

都市景観のシミュレーションを目的とした作画システムについて報告した。本システムで提案した、テクスチャマッピングを併用したモデリングは、微細な構造を違和感なく表現でき、マクロな視点からミクロな視点にまで、ある程度柔軟に対応できることが示された。ＵＮＩＸワークステーション上で実験システムを構築し、実際の問題に適用した結果、モデリングデータ量、処理時間とも現実的な範囲であることが確認された。

都市景観のシミュレーションでは膨大なデータを取り扱うため、データの管理、作成についてはまだ多くの問題が残されているが、それらを充実させることによって本システムがより実用的なシステムになり得る可能性があることが確認できた。

謝辞

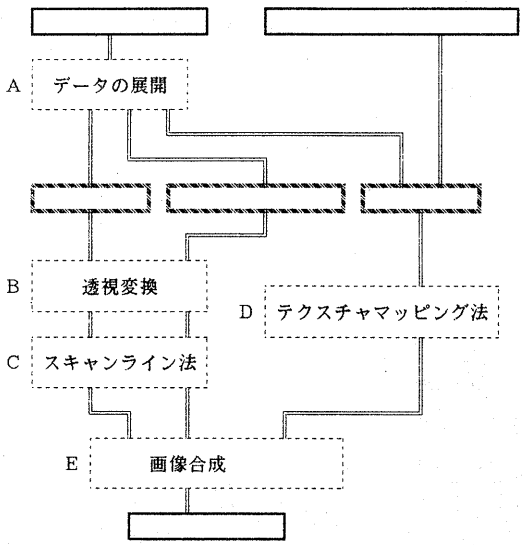
データの入手・作成等で
柳田石塚建築計画事務所　に
デモリールの作成等で
（株）アドビデオ北海道　に
それぞれ多大な御助力をいただいた。

そのほか協力をいただいている方々に深く感

謝する。

参考文献

1)中前、ｼﾐｭﾚｰｼｮﾝの視覚化、情報処理、Vol.28、No.3、1987
2)山本、The 3-Dimensional Computer Graphics、CQ、1983



	A データ展開	B 透視変換	C スキャンライン処理	D テクスチャマッピング	E データ合成	合計	三角形パッチ数	家屋数	マッピング数
写真1	211.5	63.8	825.2	0.0	48.5	1149.0	13955	946	0
写真2	284.8	102.2	973.5	520.1	113.8	1994.4	22597	997	1319
写真3	53.3	11.5	315.0	174.6	111.4	665.8	2520	346	14
写真4	55.2	11.3	297.6	827.0	123.0	1314.7	2475	305	346
						単位（秒）			単位（個）

表1　作画時間とデータの規模

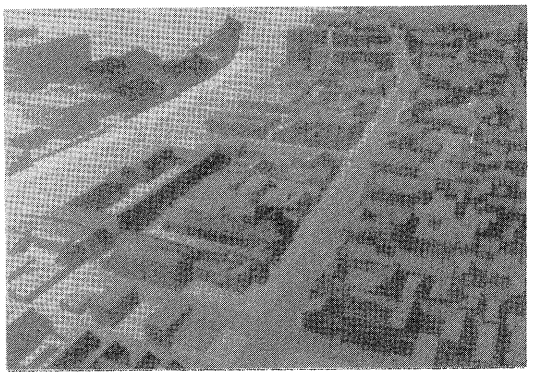


写真１

街の構造を見渡したもの
テクスチャマッピングは用いていない
各家屋は最低限のモデリングにより表現されていることがわかる。

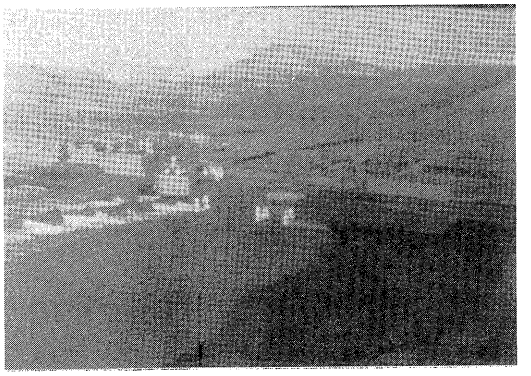


写真２

十分に広い範囲の地形データを持っているため遠景の「山なみ」もよく表現される。

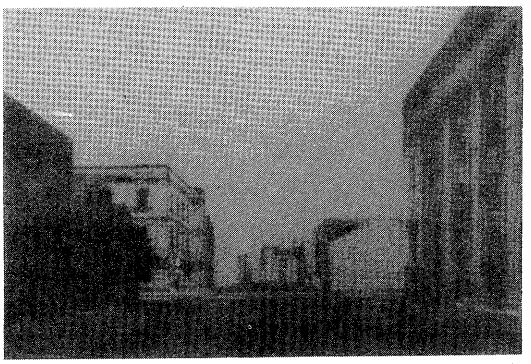


写真３

景観に重要な役割をしめる歴史的な建築物にのみテクスチャマッピングをほどこす。

写真４

写真３とおなじ視点から、街路樹や歩道の整備をシミュレートする。

盛光印刷所

87-CAD-27（1987/7/10） 27-4

n次元ディジタル画像の可逆的幾何変換法

志沢雅彦（ＮＴＴ通研）

正誤表

以下の様に，資料中に誤りがありましたので，お詫びして訂正いたします．

定理５－１の式

誤

$$A^{(k+1)}=R_{ku}A^{(k)}R_{kv}Q(\delta)\cdot P_{wk}((1-\delta a_{uv}^{(k)})/a_{uw}^{(k)})\cdot P_{kv}(-a_{uk}^{(k)})\prod_{\substack{j\neq v\\ j\neq k}}P_{kj}(-a_{uj}^{(k)})$$

正

（$v\neq k$，$v\neq w$のとき）

$$A^{(k+1)}=R_{ku}A^{(k)}R_{kv}Q(\delta)\cdot P_{wk}((1-\delta a_{uv}^{(k)})/a_{uw}^{(k)})\cdot P_{kv}(-a_{uk}^{(k)})\prod_{\substack{j\neq v\\ j\neq k}}P_{kj}(-a_{uj}^{(k)})$$

（$v=k$のとき）

$$A^{(k+1)}=R_{ku}A^{(k)}Q(\delta)\cdot P_{wk}((1-\delta a_{uk}^{(k)})/a_{uw}^{(k)})\cdot\prod_{j\neq k}P_{kj}(-a_{uj}^{(k)})$$

（$v=w$のとき）

$$A^{(k+1)}=R_{ku}A^{(k)}R_{kw}Q(\delta)\cdot P_{wk}((1-\delta a_{uw}^{(k)})/a_{uk}^{(k)})\cdot P_{kw}(-a_{uk}^{(k)})\prod_{\substack{j\neq w\\ j\neq k}}P_{kj}(-a_{uj}^{(k)})$$

定理５－３　の式

誤

$$d_j^{(k)}=d_j^{(k-1)}+|a_{uj}^{(k)}|d_v^{(k-1)}+|a_{uj}^{(k)}(a_{uv}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+(|a_{uj}^{(k)}|+1)\gamma\quad(j\neq v,\ j\neq k)$$

$$d_v^{(k)}=d_k^{(k-1)}+|a_{uk}^{(k)}|d_v^{(k-1)}+|a_{uk}^{(k)}(a_{uv}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+(|a_{uk}^{(k)}|+1)\gamma$$

$$d_k^{(k)}=d_v^{(k-1)}+|(a_{uv}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+\gamma$$

正

（$v\neq k$，$v\neq w$のとき）

$$d_j^{(k)}=d_j^{(k-1)}+|a_{uj}^{(k)}|d_v^{(k-1)}+|a_{uj}^{(k)}(a_{uv}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+(|a_{uj}^{(k)}|+1)\gamma\quad(j\neq v,\ j\neq k)$$

$$d_v^{(k)}=d_k^{(k-1)}+|a_{uk}^{(k)}|d_v^{(k-1)}+|a_{uk}^{(k)}(a_{uv}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+(|a_{uk}^{(k)}|+1)\gamma$$

$$d_k^{(k)}=d_v^{(k-1)}+|(a_{uv}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+\gamma$$

（$v=k$のとき）

$$d_j^{(k)}=d_j^{(k-1)}+|a_{uj}^{(k)}|d_k^{(k-1)}+|a_{uj}^{(k)}(a_{uk}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+(|a_{uj}^{(k)}|+1)\gamma\quad(j\neq k)$$

$$d_k^{(k)}=d_k^{(k-1)}+|(a_{uk}^{(k)}-\delta)/a_{uw}^{(k)}|d_w^{(k-1)}+\gamma$$

（$v=w$のとき）

$$d_j^{(k)}=d_j^{(k-1)}+|a_{uj}^{(k)}|d_w^{(k-1)}+|a_{uj}^{(k)}(a_{uw}^{(k)}-\delta)/a_{uk}^{(k)}|d_k^{(k-1)}+(|a_{uj}^{(k)}|+1)\gamma\quad(j\neq w,\ j\neq k)$$

$$d_w^{(k)}=d_k^{(k-1)}+|a_{uk}^{(k)}|d_w^{(k-1)}+|a_{uk}^{(k)}(a_{uw}^{(k)}-\delta)/a_{uk}^{(k)}|d_k^{(k-1)}+(|a_{uk}^{(k)}|+1)\gamma$$

$$d_k^{(k)}=d_w^{(k-1)}+|(a_{uw}^{(k)}-\delta)/a_{uk}^{(k)}|d_k^{(k-1)}+\gamma$$

アルゴリズム５－２の式

誤

$$V_k=\{\prod_{j\neq v,\ j\neq k}P_{kj}(a_{uj}^{(k)})\}P_{kv}(a_{uk}^{(k)})\cdot P_{wk}((1-\delta a_{uv}^{(k)})/a_{uw}^{(k)})Q(\delta)R_{kv}$$

正

（$v\neq k$，$v\neq w$のとき）

$$V_k=\{\prod_{j\neq v,\ j\neq k}P_{kj}(a_{uj}^{(k)})\}P_{kv}(a_{uk}^{(k)})\cdot P_{wk}((1-\delta a_{uv}^{(k)})/a_{uw}^{(k)})Q(\delta)R_{kv}$$

（$v=k$のとき）

$$V_k=\{\prod_{j\neq k}P_{kj}(a_{uj}^{(k)})\}\cdot P_{wk}((1-\delta a_{uk}^{(k)})/a_{uw}^{(k)})Q(\delta)$$

（$v=w$のとき）

$$V_k=\{\prod_{j\neq w,\ j\neq k}P_{kj}(a_{uj}^{(k)})\}P_{kw}(a_{uk}^{(k)})\cdot P_{wk}((1-\delta a_{uv}^{(k)})/a_{uk}^{(k)})Q(\delta)R_{kw}$$