(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3839871号
(P3839871)
(45)発行日　平成18年11月1日(2006.11.1)　(24)登録日　平成18年8月11日(2006.8.11)

(51)Int.Cl.　　F I
G06T 15/00 (2006.01)　　G06T 15/00 100A
G06T 11/40 (2006.01)　　G06T 11/40 200D

請求項の数 14　(全 32 頁)

(21)出願番号　特願平8-145822
(22)出願日　平成8年6月7日(1996.6.7)
(65)公開番号　特開平9-6967
(43)公開日　平成9年1月10日(1997.1.10)
審査請求日　平成15年1月29日(2003.1.29)
(31)優先権主張番号　488,474
(32)優先日　平成7年6月8日(1995.6.8)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(72)発明者　ジョン・エル・アッシュバーン
アメリカ合衆国80521コロラド州フォート・コリンズ、サマー・ストリート　1022

審査官　村松　貴士

最終頁に続く

(54)【発明の名称】三角形を表わす画素データを生成する装置

(57)【特許請求の範囲】
【請求項１】
コンピュータ・グラフィックス・システムで、三角形を表わす画素データを生成する装置であって、平面における座標をｘ、ｙで表わし、いずれかの画素データをｐで表わした場合に、三角形の平面方程式を
　ｐ＝ａｘ＋ｂｙ＋ｃ
と定義し、平面方程式データを、定数ａ、ｂ、ｃおよび定数ａ、ｂ、ｃを導くための頂点および辺に関するデータを含むデータと定義した場合に、三角形を表す頂点データに応じて平面方程式データの第１の部分を生成する手段と、ソートされた頂点データを供給するように、上記平面方程式データの上記第１の部分に応じて、上記平面方程式データの計算を容易にするソーティング規準に従って上記頂点データをソートする手段と、上記ソートされた頂点データに応じて、上記頂点データのソーティングに従って上記平面方程式データの第２の部分を生成する手段と、を含む、三角形を表わす頂点データに応じて平面方程式データを生成する処理回路と、上記平面方程式データの第１および第２の部分に応じて上記三角形を表わす画素データを生成する充填スキャン・コンバータと、を備える、装置。
【請求項２】
充填スキャン・コンバータが第１の軸および第２の軸を含むディスプレイの画素データを生成し、上記平面方程式データの上記第１の部分を生成する手段が、三角形の第１の辺上の第１の軸の座標値の差を表す、第１の符号をもつ第１の辺の変化量を生成する手段と、

三角形の第２の辺上の第１の軸の座標値の差を表す、第２の符号をもつ第２の辺の変化量を生成する手段と、三角形の第３の辺上の第１の軸の座標値の差を表す、第３の符号をもつ第３の辺の変化量を生成する手段とを含み、上記平面方程式データの上記第１の部分は、上記第１の辺の変化量、上記第２の辺の変化量および上記第３の辺の変化量を含む、請求項１に記載の装置。

【請求項３】

ソートする手段が、第１の符号、第２の符号および第３の符号に基づいて上記ソートされた頂点データを決定する論理回路を含む、請求項2に記載の装置。

【請求項４】

処理回路が、頂点データを格納するレジスタ・ファイルをさらに含み、論理回路がソートされた頂点データを供給するように、レジスタ・ファイルに格納された頂点データを参照するソートされたアドレスを供給する請求項3に記載の装置。

【請求項５】

ソートする手段が、第１の符号、第２の符号および第３の符号に基づいて上記ソートされた頂点データを決定する手段を含む請求項2に記載の装置。

【請求項６】

処理回路が、頂点データを格納するレジスタ・ファイルを含み、上記ソートされたデータを決定する手段が、ソートされた頂点データを供給するように、レジスタ・ファイルに格納された頂点データを参照するソートされたアドレスを供給する手段を含む、請求項5に記載の装置。

【請求項７】

処理回路が、頂点データを格納するレジスタ・ファイルを含み、上記ソートする手段が、ソートされた頂点データを供給するように、レジスタ・ファイルに格納された頂点データを参照するソートされたアドレスを供給する手段を含む、請求項1に記載の装置。

【請求項８】

ソートされたアドレスが、頂点の1つの表示パラメータを参照する第1の部分とソートされた頂点を参照する第2の部分とをそれぞれ含む請求項7に記載の装置。

【請求項９】

平面における座標をｘ、ｙで表わし、いずれかの画素データをｐで表わした場合に、三角形の平面方程式を

$$p = ax + by + c$$

と定義し、平面方程式データを、定数ａ、ｂ、ｃおよび定数ａ、ｂ、ｃを導くための頂点および辺に関するデータを含むデータと定義した場合に、平面方程式データに応じて三角形を表わす画素データを生成する充填スキャン・コンバータを含むコンピュータ・グラフィックス・システムで、平面方程式データを生成する方法であって、三角形を示す頂点データを受け取るステップと、頂点データを使用して、頂点データがソートされているかどうかに依存しない平面方程式データの第１の部分を生成するステップと、ソートされた頂点データを生成するように、平面方程式データの第１の部分に基づいて頂点データをソートするステップと、ソートされた頂点データを使用して平面方程式データの第２の部分を生成するステップと、平面方程式データの第１の部分および第２の部分を充填スキャン・コンバータに供給するステップと、を含む、方法。

【請求項１０】

充填スキャン・コンバータが第１の軸および第２の軸を含むディスプレイの画素データを生成し、平面方程式データの第１の部分を生成するステップが、三角形の第１の辺上の第１の軸の座標値の差を表す、第１の符号をもつ第１の辺の変化量を生成するステップと、三角形の第２の辺上の第１の軸の座標値の差を表す、第２の符号をもつ第２の辺の変化量を生成するステップと、三角形の第３の辺上の第１の軸の座標値の差を表し、第３の符号をもつ第３の辺の変化量を生成するステップとを含む、請求項9に記載の方法。

【請求項１１】

ソートするステップが、第１の符号、第２の符号および第３の符号に基づいて、ソートさ

れた頂点データを決定することを含む、請求項10に記載の方法。
【請求項１２】
頂点データを受け取るステップが、頂点データをレジスタ・ファイルに格納することを含み、ソートするステップが、ソートされた頂点データを供給するように、レジスタ・ファイルに格納された頂点データを参照するソートされたアドレス・データを生成することを含む、請求項11に記載の方法。
【請求項１３】
頂点データを受け取るステップが、頂点データをレジスタ・ファイルに格納することを含み、ソートするステップが、ソートされた頂点データを供給するように、レジスタ・ファイルに格納された頂点データを参照するソートされたアドレス・データを生成することを含む、請求項9に記載の方法。
【請求項１４】
ソートされたアドレスを生成するステップが、頂点の1つの表示パラメータを参照する第1の部分とソートされた頂点を参照する第2の部分とを生成することを含む請求項13に記載の方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、一般にディスプレイ・グラフィックスの生成に関し、特に、充填スキャン・コンバータに入力される三角形の平面方程式の生成に関する。
【０００２】
【従来の技術】
多角形は、表示画面上に表面を表示するためにグラフィックスで用いられる基本プリミティブの一つである。多角形の充填(fill)される領域は、グラフィックス・ディスプレイに３次元の物体の表面をレンダリング(rendering)する際に特に役に立つ。一般に多角形は、凸状または凹状のどちらかの任意の形状をもつ、結合され、閉じられた線の連続として定義される。多角形の辺の上および中に入る表示画素をアクティブにすることによって、表示画面上で多角形を充填することができる。辺の内側の画素は、内部(interior)の画素と呼ばれる。図18は、いくつかの多角形および充填される多角形をレンダリングするためにアクティブにされる画素を示している。
【０００３】
いくつかの技法を用いて、多角形を充填するためにどの画素をアクティブにするべきか決定する。これらの技法は、充填スキャン・コンバージョンといわれる。一般にこれらの技法は、画素の水平または垂直の行のいずれのスパン上の一組の辺の間で作用する。例えば図18で、それぞれの多角形についてそれぞれの水平スパンの始点および終点が陰付けされた画素によって印されている。充填スキャン・コンバータは、画素から画素へ、例えば図18のX方向へステップし、それぞれの画素についてパラメータの値を計算することによって、１つのスパン内の画素データを順に生成することができる。そして１画素下の行にY方向に下がることによって、画素を生成するための新しいスパンが選択され、新しいスパンについて画素から画素へステッピングを繰り返す。
【０００４】
図18から分かるように、多角形は複雑になりえる。多くの辺をもつ多角形、内部交差(intersecting)する多角形、および凹形の多角形のような複雑な多角形については、スキャン・コンバージョンは非常に複雑になりうる。従って、三角形を充填するように充填スキャン・コンバータを最適化し、三角形の組合せとして多角形をレンダリングすることが望ましい。さらに、通常グラフィック・データベースには四辺形が存在するため、２つの三角形に分割される四辺形を支持するように充填スキャン・コンバータを最適化することが望ましい。そして三角形の充填スキャン・コンバータは、結果の三角形を両方とも充填することができる。
【０００５】

図18は２次元で単一色で描かれているが、多角形は例えば赤、緑および青の色の値によって定義されるような色をもつこともできる。多角形は３次元であってもよく、したがって通常Z軸の座標として定義されるそれぞれの画素に関連する奥行き(depth)をもつ。それゆえ図18に示すそれぞれの画素は、実際上、X座標およびY座標、さらに赤、緑、青およびZ軸座標をもつことができる。

【０００６】

三角形を効率的に表示するために設計されるハードウェアおよびソフトウェアは、例えば加速器のブロックと充填スキャン・コンバータのブロックというようにいくつかのブロックに分割することができる。このような配置において、加速器は三角形の頂点(V0,V1,V2)に関するデータに基づいて平面方程式を生成する。これらの平面方程式は、三角形の中のそれぞれの画素に関連するパラメータの値を一意に定義するように設計され、結果として充填スキャン・コンバータは高速でシンプルな電子機器ハードウェアとなり、加速器から供給される平面方程式を「ステッピング(stepping)する」ことによって画素を生成する。例えば加速器は加算、乗算および除算を行うことができ、充填スキャン・コンバータは加算のみ行う。

【０００７】

【発明が解決しようとする課題】

四辺形を生成および充填するとき、それは２つの別々の三角形に分割することができ、それぞれの三角形に関するデータは別々に加速器に入力されて適当な平面方程式を生成することができる。しかしこの方法には、相当な量の処理資源を使用し、長々とした計算が必要となりうる。四辺形を充填するために必要な処理資源および時間を減少させることが要求されている。

【０００８】

さらに充填スキャン・コンバータは、頂点の指定に関してソートされた三角形情報が与えられる場合、より効率的に設計することができる。頂点のソートは平面方程式を生成するより前に実行することができるが、その方法のために付加的な処理資源および時間が用いられる。したがって、ソートされた三角形の頂点を用いて平面方程式を生成する際に必要とされる処理資源および時間を減少させることが要求されている。

【０００９】

【課題を解決するための手段】

この発明の一つの形態は、三角形を表わす画素データを生成するコンピュータ・グラフィック・システム装置に照準されている。装置は、三角形を表わす頂点データに応じて平面方程式データを生成する処理回路と、平面方程式データの処理回路に応じて三角形を表わす画素データを生成する充填スキャン・コンバータとを含む。処理回路は、頂点データに応じて平面方程式データの第１の部分を生成する手段と、ソートされた頂点データを供給するために、平面方程式データの計算を容易にするソーティング規準に従って頂点データをソートする手段と、ソートされた頂点データに応じて、頂点データのソーティングに従って平面方程式データの第２の部分を生成する手段とを含み、上記平面方程式の第１の部分は頂点データのソーティングと並行して生成される。充填スキャン・コンバータは、第１の軸および第２の軸を含むディスプレイの画素データを生成することができ、平面方程式データの第１の部分は、三角形の第１の辺上の第１の軸の座標値の差を表す、第１の符号をもつ第１の辺のデルタと、三角形の第２の辺上の第１の軸の座標値の差を表す、第２の符号をもつ第２の辺のデルタと、三角形の第３の辺上の第１の軸の座標値の差を表す、第３の符号をもつ第３の辺のデルタとを含むことができる。ソートする手段は、第１の符号、第２の符号および第３の符号に基づいて、ソートされた頂点データを決定する論理回路を含むことができる。処理回路は、頂点データを格納するレジスタ・ファイルを含むことができ、論理回路は、レジスタに格納される頂点データを参照するソートされたアドレスを与えることができる。さらにソートする手段は、第１の符号、第２の符号および第３の符号により、ソートされた頂点データを決定する手段を含むことができる。ソートされたアドレスは、一つの頂点の表示パラメータを参照する第１の部分と、ソートされた頂点

を参照する第２の部分を含むことができる。
【００１０】
この発明の別の形態は、充填スキャン・コンバータに供給される三角形を表わす平面方程式データを生成する方法に照準されている。方法は、三角形を示す頂点データを受け取るステップと、頂点データがソートされているかどうかに依存しない平面方程式データの第１の部分を生成するステップと、ソートされた頂点データを生成するために頂点データをソートするステップと、ソートされた頂点データを使用して平面方程式データの第２の部分を生成するステップと、平面方程式データの第１の部分および第２の部分を充填スキャン・コンバータに供給するステップとを含む。
【００１１】
この発明のもう一つの形態は、充填スキャン・コンバータに供給される三角形を表わす平面方程式データを生成する装置に照準されている。装置は、三角形を示す頂点データを受け取る手段と、頂点データがソートされているかどうかに依存しない平面方程式データの第１の部分を生成する手段と、ソートされた頂点データを生成するために頂点データをソートする手段と、ソートされた頂点データを使用して平面方程式データの第２の部分を生成する手段と、平面方程式データの第１の部分および第２の部分を充填スキャン・コンバータに供給する手段を備える。
【００１２】
【発明の実施の形態】
図１は、本発明に従って前処理を行うライン・スキャン・コンバージョンの取り込みに適したグラフィックス・システムの一つの実施例のブロック図である。図示される具体化は、基板およびチップの数、それらが区分される方法、バスの幅およびデータ転送速度に関して単なる典型であることを理解されたい。多数の他の具体化を使ってもよい。ここで図示され、説明される回路は単なる実例として与えられている。回路は、例えばSynopsysから市販されている論理統合ソフトウェアを使用する大規模なカスタム集積回路で実現されることが好ましい。論理統合ソフトウェアは、Veralogのような高級言語で論理ゲートに書き込まれる回路の説明を最適化および翻訳する。回路は、５ボルトで動作する１ミクロンのFETを生成するCMOSプロセス、3.3ボルトで動作する0.6ミクロンで引かれた(drawn)ゲート長のデバイスを生成するCMOSプロセス、またはデジタル回路を実行するに適した他のプロセスを使用して実現することができる。論理統合ソフトウェアへの入力は構造的というより機能的であるので、論理統合ソフトウェアによって生成される実際の回路は、ここで示されるものと異なってもよい。
【００１３】
図示されるように、システムはフロント・エンド基板10、テクスチャ・マッピング基板12およびフレーム・バッファ基板14を含む。フロント・エンド基板は、52ビットのバス16を通してホスト・コンピュータ15に連絡している。フロント・エンド基板は、レンダリングされるプリミティブをバス16を通してホスト・コンピュータから受け取る。プリミティブは、プリミティブが三角形である場合、例えば頂点のようなプリミティブの部分に関して、x,y,zベクトル座標のデータ、R,G,B色データおよびテクスチャS,T座標によって規定される。３次元のプリミティブを表すデータは、フロント・エンド基板10によって、85ビットのバス18を通してテクスチャ・マッピング基板12およびフレーム・バッファ基板14に供給される。テクスチャ・マッピング基板は、プリミティブを表す画面表示画素を計算するために受け取ったプリミティブ・データを補間し、プリミティブのそれぞれの画素について対応する結果のテクスチャ・データを決定する。結果のテクスチャ・データは、図をわかりやすくするために図１に単一のバスとして示される55ビットのバス28を通してフレーム・バッファ基板に供給される。
【００１４】
フレーム・バッファ基板14は、それぞれのプリミティブを表す表示画面上の画素を計算するためにフロント・エンド基板10から受け取ったプリミティブ・データを補間し、物体の色の値を決定する。フレーム・バッファ基板は、画素毎に、物体の色の値をテクスチャ・

マッピング基板から供給される結果のテクスチャ・データと組み合わせ、それぞれの画素について結果の画像R,G,B値を生成する。それぞれの画素のR,G,B色制御信号がR,G,B線29を通して供給され、表示画面の画素を制御し、表示画面上にテクスチャ・マッピングされたプリミティブを表す結果の画像を表示する。

【００１５】

フロント・エンド基板10、テクスチャ・マッピング基板12、およびフレーム・バッファ基板14はそれぞれパイプラインで結ばれ、複数のプリミティブについて同時に操作する。テクスチャ・マッピングおよびフレーム・バッファの基板はフロント・エンド基板から供給されるプリミティブについて操作し、一方、フロント・エンド基板は、基板12および14のパイプラインが満杯になるまで新しいプリミティブについて操作および供給を続ける。

【００１６】

フロント・エンド基板10は、分配器チップ30、３つの３次元(3D)加速器チップ32A、32Bおよび32C、２次元(2D)加速器チップ34、および集線器チップ36を含む。分配器チップ30は、バス16を通してホスト・コンピュータからX,Y,Z座標および色のプリミティブ・データを受け取り、３次元のプリミティブ・データを３Ｄ加速器チップ32A、32Bおよび32Cに一様に分配する。このように３つのグループのプリミティブが同時に操作されるので、システムの帯域幅は増大する。データは、40ビットのバス38Aを通して３Ｄ加速器チップ32Aおよび32Bに供給され、40ビットのバス38Bを通してチップ32Cに供給される。バス38Aおよび38Bは、60MHzの速度でデータを転送し、２つの３Ｄ加速器チップを支持するに十分な帯域幅を与える。２Ｄのプリミティブ・データは、44ビットのバス40を通して40MHzの速度で２Ｄ加速器チップ34に供給される。

【００１７】

それぞれの３Ｄ加速器チップは、受け取ったプリミティブを定義するx,y,z座標を対応する画面の空間座標に変換し、画面の空間座標について物体のR,G,B値およびテクスチャS,T値を決定し、プリミティブの四辺形を三角形に分解し、それぞれの三角形を定義する三角形の平面方程式を計算し、そしてR,G,B値を決定するために光源の計算を行う。またそれぞれの３Ｄ加速器チップは、複数のウィンドウが表示される場合またはプリミティブの部分が表示画面に表示される画像の量を越えてしまう場合、画像のクリッピング操作を実行して、結果の画像の正確な画面表示を保証する。３Ｄ加速器チップ32A、32Bおよび32Cのそれぞれからの出力データは、44ビットのバス42Aおよび42Bを通して60MHzの速度で集線器チップ36に供給される。２Ｄ加速器チップ34もまた、46ビットのバス44を通って45MHzの速度で集線器チップ36に出力データを供給する。集線器チップ36は、３Ｄ加速器チップ32Aないし32Cから受け取った３Ｄプリミティブを結合し、分配器チップ30によって分配される前のオリジナルの順序にプリミティブを並べ直し、結合されたプリミティブの出力データをバス18を通してテクスチャ・マッピングおよびフレーム・バッファの基板に供給する。

【００１８】

テクスチャ・マッピング基板12は、テクスチャ・マッピング・チップ46およびキャッシュ・メモリとして配置されることが好ましいローカル・メモリ48を含む。より詳しく後述するが、キャッシュ・メモリ48は、フレーム・バッファ基板でレンダリングされるプリミティブに関連するテクスチャMIPマップ・データを格納する。テクスチャMIPマップ・データは、ホスト・コンピュータ15の主メモリ17からバス40を通り、２Ｄ加速器チップ34、そして24ビットのバス24を通してダウンロードされる。

【００１９】

テクスチャ・マッピング・チップ46は、表示画面上にレンダリングされるプリミティブを表すプリミティブ・データを、バス18を介して連続して受け取る。上述のように３Ｄ加速器チップ32AないしCから供給されるプリミティブは、点、線および三角形を含む。テクスチャ・マッピング基板は、点および線のテクスチャ・マッピングを実行せず、三角形のプリミティブに関してのみ操作する。三角形のプリミティブを表すデータは、少なくとも一つの頂点に関する物体のx,y,z画素座標、少なくとも一つの頂点の物体の色のRGB値、少な

くとも一つの頂点に対応するテクスチャ・マップの部分のS,T座標、および三角形の平面方程式を含む。テクスチャ・マッピング・チップ46は、物体の画素のz座標および物体の色のR,G,B値を無視する。チップ46は、x,y画素座標を補間し、プリミティブを表すそれぞれのx,y画面表示画素に対応するSおよびT座標を補間する。それぞれの画素について、テクスチャ・マッピング・チップは、キャッシュ・メモリから対応するテクスチャMIPマップの部分にアクセスし、複数のテクセル(texel)の加重平均を含みうる画素について結果のテクスチャ・データを計算する。

【００２０】

一つの典型的な実施例において、キャッシュは64ブロックの256x256テクセルを格納する。先の技術システムのテクスチャ・マッピング・ハードウェアに用いられているローカル・メモリとは異なり、本発明のキャッシュ・メモリは、大きいテクスチャのような、レンダリングされるプリミティブにマップするテクスチャMIPマップの連続全体を格納しなくてもよい。むしろキャッシュ・メモリは、任意のある時間に、現在レンダリングしているプリミティブに実際に使用されるMIPマップの連続の個別の部分のみ格納する。従って多くの応用について、レンダリングされる画像の完全なテクスチャ・データの１部分だけが、任意の時間にキャッシュ・メモリに格納される。

【００２１】

それぞれのテクスチャの完全なMIPマップの連続は、ホスト・コンピュータ15の主メモリ17に配置および格納される。レンダリングされるプリミティブのそれぞれの画素について、テクスチャ・マッピング・チップ46はキャッシュ・メモリ48のディレクトリにアクセスし、テクスチャMIPマップの対応する１つまたは複数のテクセルがキャッシュに存在するかどうか判断する。アクセスした時点で対応するテクセルがキャッシュ・メモリに格納されていれば、キャッシュのヒットが生じ、テクスチャ・マッピング・チップ46によってテクセルがキャッシュからフェッチおよび操作され、フレーム・バッファ基板に渡される結果のテクスチャ・データを計算する。しかしテクスチャ・マッピング・チップ46がアクセスする時にプリミティブの画素についてキャッシュ・メモリの中に対応するテクセルが格納されていない場合、キャッシュ・ミスが生じる。キャッシュ・ミスが生じたとき、プリミティブをレンダリングするために必要なテクスチャMIPマップ・データの部分がホスト・コンピュータ15の主メモリ17からキャッシュ・メモリ48にダウンロードされ、場合によっては前に格納されていた一部のデータを置き換える。しかし、レンダリングされるあらゆるプリミティブについてMIPマップの連続の全体をダウンロードする従来のテクスチャ・マッピング・システムとは異なって、本発明は、プリミティブを現在レンダリングするために実際に必要なMIPマップの連続の部分、または現在レンダリングされたプリミティブの部分だけをダウンロードする。より詳細に後述するが、キャッシュ・ミスが起きたときは、テクスチャ・マッピング・チップ46によって割込み制御信号が発信され、ホスト・コンピュータ15のテクスチャ割込みマネージャーを起動させる。割込み制御信号はライン94を通して分配器チップ30に供給され、そしてライン95を通してホスト・コンピュータに割込み信号を供給する。

【００２２】

要求されるテクスチャ・データは、ホスト・コンピュータによってその主メモリから読み出され、３Ｄプリミティブをレンダリングするフロント・エンド基板を通るテクスチャ・マッピング・チップへのパイプラインを迂回して、バス24を通してテクスチャ・マッピング基板48にダウンロードされる。このように、キャッシュ・ミスの割込みが生じたとき、フロント・エンド基板は３Ｄプリミティブを操作して、バス18を通してテクスチャ・マッピング・チップおよびフレーム・バッファ基板に出力プリミティブ・データを供給し続ける一方で、キャッシュ・ミスを起こしたプリミティブに関連するテクスチャ・データが主メモリ17からダウンロードされる。従来のテクスチャ・マッピング・システムとは対照的に、テクスチャ・マッピング・ハードウェアにテクスチャ・データをダウンロードするのに、３Ｄプリミティブのパイプラインをフラッシュ(flush)する必要がなく、従ってシステムの帯域幅と性能を増加させる。それぞれの画素に関する結果のテクスチャ・データは

、テクスチャ・マッピング・チップ46によって５つのバス28を通してフレーム・バッファ基板に供給される。５つのバス28はそれぞれフレーム・バッファ基板に備えられた５つのフレーム・バッファ制御器チップ50A、50B、50C、50Dおよび50Eに結合され、結果のテクスチャ・データをフレーム・バッファ制御器チップに平行に供給する。フレーム・バッファ制御器チップ50AないしEはそれぞれ、関連するVRAM(ビデオ・ランダム・アクセス・メモリ)チップ51AないしEに結合される。フレーム・バッファ基板はさらに、４つのビデオ形式チップ52A、52B、52Cおよび52DならびにRAM DAC(ランダム・アクセス・メモリ・デジタル-アナログ・コンバータ)54を含む。フレーム・バッファ制御器チップは、表示画面の別々の、重複しないセグメントを制御する。それぞれのフレーム・バッファ制御器チップは、フロント・エンド基板からバス18を通してプリミティブ・データを受け取り、テクスチャ・マッピング基板からバス28を通して結果のテクスチャ・マッピング・データを受け取る。フレーム・バッファ制御器チップはプリミティブ・データを補間して、プリミティブを表すそれぞれのセグメントの画面表示の画素の座標、およびそれぞれの画素の座標について対応する物体R,G,B色の値を計算する。テクスチャ・マッピング基板から結果のテクスチャ・データが供給されるプリミティブ(すなわち三角形)について、フレーム・バッファ制御器チップは画素ごとに物体の色の値と結果のテクスチャ・データを結合して、表示画面上に表示されるそれぞれの画素の最終的なR,G,B値を生成する。

【００２３】

物体およびテクスチャの色の値を結合する方法は、多くの異なった方法で制御することができる。例えば置き換えモードでは、画素をレンダリングする際にテクスチャの色の値だけが使用されるように、物体の色の値をテクスチャの色の値と簡単に置き換えることができる。代替として変調モードでは、物体およびテクスチャの色の値を一緒に倍増させて、画素の最終的なR,G,B値を生成することができる。さらに、対応するテクスチャの色の値を物体の色の値と結合する方法を規定する比率を明示する色制御ワードを、それぞれのテクセルについて格納することができる。それぞれの画素に対応する結果のテクセル・データについて結果の色制御ワードを決定し、それをフレーム・バッファ制御器チップにバス28を通して供給することができ、その結果、制御器チップは対応する結果の制御ワードによって示される比率を使用してそれぞれの画素について最終的なR,G,B値を決定することができる。

【００２４】

フレーム・バッファ制御器チップ50AないしEによって生成された、それぞれの画素のR,G,B値を含む結果の画像ビデオ・データは、対応するVRAMチップ51AないしEに格納される。VRAMチップ51AないしEのそれぞれのグループは８つのVRAMチップを含み、従って、40のVRAMチップがフレーム・バッファ基板上に位置する。それぞれのビデオ形式チップ52AないしDはVRAMチップの異なるセットに接続され、そこからデータを受け取る。ビデオ・データはVRAMチップから連続的にシフトされ、64ビットのバス58A、58B、58Cおよび58Dを通って33MHzの速度で４つのビデオ形式チップ52A、52B、52Cおよび52Dにそれぞれ供給される。ビデオ形式チップは、RAMDACによって処理できるようにビデオ・データをフォーマットし、フォーマットされたデータを32ビットのバス60A、60B、60Cおよび60Dを通して33MHzの速度でRAMDAC54に供給する。それからRAMDAC54は、デジタルの色データをアナログのR,G,B色制御信号に変換し、それぞれの画素のR,G,B制御信号を、R,G,B制御ライン29に沿って画面ディスプレイ(図示しない)に供給する。

【００２５】

本発明のある実施例では、テクスチャ・マッピング基板12およびフレーム・バッファ基板14上のハードウェアが複製されるため、あるプリミティブをレンダリングするタスクを複数のプリミティブに関して平行して実行することができ、結果としてシステムの帯域幅が増える。本発明のこのような代替実施例が図２に示されており、複製されたハードウェアを有する本発明のコンピュータ・グラフィックス・システムのブロック図を示している。図２のシステムは、４つの３Ｄ加速器チップ32A、32B、32Cおよび32Dと、キャッシュ・メモリ48Aおよび48Bにそれぞれ関連する２つのテクスチャ・マッピング・チップ46Aおよび4

6Bと、VRAMチップのグループにそれぞれ関連する10のフレーム・バッファ・チップ50Aないし50Jを含む。図２のシステムの動作は、上述された図１のシステムの動作と同様である。図２の実施例におけるハードウェアの複製は、あるプリミティブをレンダリングする操作を複数のプリミティブについて平行して実行することができるので、システムの帯域幅の増加を可能にする。

【００２６】

加速器チップ32A、32Bおよび32Cを表わす簡略されたブロック図が、図３に示されている。ホスト・コンピュータ15からのプリミティブ・データは入力FIFO70を通って、頂点RAM(バッファ0)71および頂点RAM(バッファ0)72を含む二重バッファの頂点RAMに供給される。加速器はパイプライン構成において２つの個別のプロセッサを含む。左のスタック78は、クリップ・コード・ロジック82、レジスタ・ファイル73、マルチプレクサ74、浮動小数点ALU75および浮動小数点乗算器76を含む。左のスタック78の動作は、アドレス解読ロジック、変換エンジンおよび分解エンジンを含む左のスタック制御ユニット77によって制御される。左のスタック78からの結果は、プリミティブRAM(バッファ0)80およびプリミティブRAM(バッファ1)81を含む二重バッファのプリミティブRAMを通って右のスタック79に供給される。右のスタックは、色クランパ/比較器85、レジスタ・ファイル180、マルチプレクサ182、浮動小数点ALU184、浮動小数点乗算器186および浮動小数点除算/平方根ユニット188を含む。右のスタック79からの結果は、出力FIFO91を通ってテクスチャ・マッピング基板12およびフレーム・バッファ基板14(図1)に供給される。右のスタック79の動作は、右のスタック制御ユニット83によって制御される。右のスタックはライティング(lighting)、クリッピングおよび平面方程式の計算を実行する。クリッピングおよびライティングのパラメータは、ホスト・コンピュータ15によってクリッピング/ライティングRAM84に供給される。

【００２７】

加速器は、変換、四辺形の分解、三角形およびベクトルに関する部分的な勾配計算、ライティング(またはシェーディング(shading))、クリッピング、およびランダムに向きを定められ、定義された四辺形(クワッド)、三角形およびベクトルの平面方程式(または勾配)の計算を実行する。出力は、画素データにレンダリングするためにフレーム・バッファ基板14のスキャン・コンバータに供給され、また画素ごとのテクスチャの色の値を生成するためにテクスチャ・マッピング基板12に供給される。

【００２８】

図３の例では、サイズおよびコストを制限するため、右のスタック79だけが除算器を含んでいる。右のスタックは除算器を含み、左のスタックは除算器を含まないので、左のスタックが変換、部分的な勾配計算、クリップ検査、分解およびライティングのための前計算を実行して、加速器の機能を区分する。右のスタックは、ライティング、クリッピングおよび平面方程式の計算を実行する。根本的な操作の流れは、プリミティブ・データがホスト・コンピュータから入力FIFO70に入力することである。プリミティブ・データは頂点RAMバッファ71および72のうちの一方に入る。そして左のスタック78は、変換、分解および部分的な勾配計算を実行し、結果をプリミティブRAMバッファ80および81のうちの一方に置く。左のスタックが１つのプリミティブに関してその操作を完了すると、右のスタック78にそのプリミティブについて操作を開始するように通知する。そして左のスタックは、次のプリミティブに関して操作を始めることができる。同時に右のスタックは、プリミティブRAMバッファにあるプリミティブに関してライティング、(必要であれば)クリッピング、および平面方程式の計算を行う。

【００２９】

代替の実施例では、左と右のスタックが共に、除算器を含む同一のハードウェアを含む。この構成では、それぞれのスタックが異なるプリミティブに同じ操作を実行するように機能を区分することができる。

【００３０】

図４は、平面方程式の生成を実行する典型的な回路素子のブロック図である。ALU(算術論

理ユニット)184、乗算器186および除算器188は、操作上、マルチプレクサ/データ・フォーマッタ182を通してレジスタ・ファイル180およびRAM(ランダム・アクセス・メモリ)196に接続される。図４に示されるような一つの実施例において、RAM196は２つのRAMセグメント、RAM196AおよびRAM196Bから構成される。RAM196Aは図３の二重バッファのプリミティブRAM80および81に対応し、RAM196Bは図３のクリッピング/ライティングRAM84に対応する。好ましい実施例では、ALU184、乗算器186および除算器188はすべて浮動小数点回路であるが、発明の利点を実現するために必ずしも必要ではない。ALU184、乗算器186および除算器/平方根計算機188の組み合わせを、処理ユニット185と呼ぶことができる。RAM196は、三角形および四辺形に関する頂点データを格納する。ALU184は加算および減算のような数値演算を実行し、乗算器186および除算器188はそれぞれ乗算および除算を実行する。レジスタ・ファイル180およびマルチプレクサ/データ・フォーマッタ182は記憶域およびデータ・ルーチングを提供し、一般にデータ・フロー回路181と呼ぶことができる。入力は、LS_leftバス183およびLS_rightバス189で受け取ることができる。出力はLS_OUTバス187上に供給することができ、それは充填スキャン・ステッパがステッピングに使用する情報を受け取ることができるFIFOに接続されていることが好ましい。図３の右のスタック制御器83の一部として実行されうる状態(state)機械190は、図４の他の素子を制御して、より詳しく後述するように平面方程式を生成する。

【００３１】

図４でRAM196は、図１のホスト・コンピュータ15および上流の処理回路から、四辺形のデータ定義(典型的には頂点)を受け取る。図１で示したシステムに関して、好ましい実施例において図４の回路素子は３Ｄ加速器チップ32AないしCのそれぞれの中に属している。RAM196は、変換および辺の勾配計算を行う加速器の中の上流回路からデータを受け取る。上流回路によってRAM196に書き込まれたデータは、後述するように平面方程式の計算および頂点のソーティングを行う図４の回路によってアクセスされる。RAM196は高速に操作するために二重バッファであることが好ましい。

【００３２】

一般に、三角形に基づく充填スキャン・コンバータをもつ３Ｄコンピュータ・グラフィックス・システムは、外部ソースから四辺形の定義を受け取り、この四辺形の定義を第１および第２の三角形の定義に分割し、これらの定義を平面方程式ジェネレータの入力に与える。それからシステムは、第１の三角形について平面方程式を生成し、充填スキャン・コンバータにこれらの平面方程式を与える。図１の中で、充填スキャン・コンバータはフレーム・バッファ基板14内に存在する。充填スキャン・コンバータは、平面方程式に基づくそれぞれの画素のそれぞれのパラメータの値を決定することによって、充填された三角形をレンダリングする。続いてシステムは、第２の三角形について平面方程式を生成し、それらを充填スキャン・コンバータに供給することができる。

【００３３】

発明の一つの見地によれば、グラフィックス・システムは四辺形を２つの三角形に分割するため、第１の三角形の平面方程式の生成による結果を使用することにより、第２の三角形の平面方程式をより効率的に生成することができる。発明のこの見地を分かりやすくするために、平面方程式、および平面方程式で用いられる値の微分によって充填スキャン・コンバータの機能を述べる。

【００３４】

図５を参照すると、三角形７は一連のスパン8AないしNに分解することができる。それぞれのスパンは、充填される三角形７の中の１行の画素を表している。そして充填スキャン・コンバータは、これらのスパンに沿ってステップし、三角形の中のすべての画素についてパラメータのそれぞれの値を決定することによって、画素データを生成する。これを行うために、充填スキャン・コンバータはそれぞれのスパンについて、スパンの位置Yと、スパンの始点および終点のX座標と、赤、緑、青およびZの始点値と、勾配(dz/dx、dred/dx、dgreen/dxおよびdblue/dx)という情報を与えられる。

【００３５】

それぞれの頂点(V0,V1,V2)のパラメータを示す値の間の補間を行うことによって、それぞれのパラメータの始点値がそれぞれのスパンについて決定される。始点値および表示パラメータ(赤、緑、青または奥行きを含む)の勾配(dp/dx)が分かると、連続した画素のパラメータは以下の方程式で補間される。

【００３６】

【数１】

$$p1 = p0 + dp/dx$$

【００３７】

上記の方程式で、p1は次の画素の１つのパラメータであり、p0は現在の画素の１つのパラメータであり、dp/dxはこのパラメータの偏導関数である。例えばパラメータが赤の場合、次の画素(p1)の赤の強度は、X軸上の連続する画素の間で赤の強度が変化する量(dp/dx)に、現在の画素(p0)の赤の強度を加えることによって決定される。

【００３８】

充填スキャン・コンバータは、次のスパン8Bのパラメータの始点値を決定するために、三角形の辺に沿ってそれぞれのパラメータについて同様の補間を行う。これらのパラメータpはそれぞれXおよびYの線形関数であるから、三角形の任意の点のパラメータの値を以下の形式の平面方程式として表すことができる。

【００３９】

【数２】

$$p = ax + by + c$$

【００４０】

上記の方程式で、pは赤、緑、青またはz座標のいずれかである。上記のような平面方程式は、Computer Graphics, Principles and Practice (Foley et al, Addison Wesley, 1990)の中でより詳しく説明されている。具体的には、「a」、「b」および「c」は、それぞれユニークなxy位置のpの値を定義する定数である。ここで、「平面方程式の生成」とはこれらの定数の値を生成することに言及し、また以下で述べるが、代替の定数の値の決定もまた平面方程式を説明する。

【００４１】

第１または開始のスパン8Aの第１の画素の値p0は、平面方程式ジェネレータによって与えられる。充填スキャン・コンバータは、説明のために最小のYスパンとするこの開始スパンに基づいて、他のスパン8Bないし8Nの始点のパラメータ値を計算することができる。慣例によって、表示画面上で、Xは左から右に増加し、Yは上から下に増加する。充填スキャン・コンバータは、辺に沿ってY方向にステッピングするために以下の方程式を用いることができる。辺は、$x=edge_slope^{*}y+b$(ここでedge_slope=dx/dy(Yに関するXの変化))によって表すことができる。

【００４２】

【数３】

$$p0 = ax0 + by0 + c$$
$$p1 = ax1 + by1 + c$$

【００４３】

ここで、p0は辺を表す一つの画素のあるパラメータ値であり、p1は次のスパンの始点の画素を表す画素の同じパラメータの値である。また(x0, y0)および(x1, y1)は、辺上の連続する点を定義する。しかし、x1=x0＋dx、および辺に沿ったYのステップに関してy1=y0＋1であるので、以下の方程式が成り立つ。

【００４４】

【数４】

$$p1=a(x0+dx)+b(y0+1)+c$$
$$=(ax0+by0+c)+adx+b$$
$$=p0+adx+b$$

【００４５】
上記の方程式で、「a」の値は、三角形全体に関するの定数である。一定のdyの一般的な線において、dxの値は定数であるが典型的には整数でない。しかし画素は離散的な位置にあるので、充填スキャン・コンバータは整数のX値にそれぞれの画素を当てはめ、つまりdxを整数値Δxに置き換えることができる。整数値Δxは実際の線のdxと選ばれた画素との間で誤差が増えるに従って変化する。しかし選ばれた整数Δxは常に、切り捨てのedge_slopeまたは切り上げのedge_slopeのどちらかである。従って上記の方程式は次のようになる。
【００４６】
【数５】
p1 = p0 + b + 切り捨て(edge_slope) + 調整
【００４７】
切り捨て(A)とは、最も近い整数に切り捨てされた「A」値をいう。[b+切り捨て(edge_slope)]の項は、前もって計算された定数であり、
【００４８】
【数６】
調整 = a [(絶対値(x1 - x0) ＞ 絶対値(edge_slope)(Δxが切り上げのedge_slope)の場合]
調整 = 0 [絶対値(x1 - x0) =＜ 絶対値(edge_slope)(Δxが切り捨てのedge_slope)の場合]
【００４９】
p(I)がスパン8上のx=Iのパラメータpの値である場合、画素p(I)を決定するために次式が与えられる。
【数７】
p(I)=p(I- 1) + b + [切り捨て(edge_slope))] + 調整
調整 = a [絶対値(x(I) - x(I-1)) ＞ 絶対値(edge_slope)の場合]
調整 = 0 [絶対値(x(I) - x(I-1)) =＜ 絶対値(edge_slope)の場合]
【００５０】
edge_slopeがX(I-1)に加えられてX(I)になるとき、少数位置からの桁上げ(carry)があるときに限ってΔxは切り上げたedge_slopeと等しくなることに留意されたい。上記の方程式は、如何なる乗算の実行も必要としない。こうしてスパン8Aの始点の画素が与えられ、スパン8Aの初期パラメータが与えられると、次のスパン8Bの第１の画素の始点パラメータは単純な加算機および計数器を用いて生成することができるので、充填スキャン・コンバータの構造が単純化される。
【００５１】
スパン8Aないし8Nのすべてのスパンは、典型的にXスパンとして生成されるので、三角形の辺は典型的にY方向にステップされる。図6aおよび図6bは、充填スキャン・コンバータが充填する三角形のタイプIおよびタイプIIをそれぞれ示している。この指定に関して、タイプIの三角形９は、三角形９の中でY方向に最も長い左側の１辺を含み、タイプIIの三角形10は、三角形10の中でY方向に最も長い右側の１辺を含む。この指定に関して、充填スキャン・コンバータを辺１として識別される辺から常にステップするように設計することができ、その結果充填スキャン・コンバータの構造を単純化することができる。さらに、充填スキャン・コンバータを三角形の上から下に(Yを増加させて)ステップするように設計することができ、その結果また充填スキャン・コンバータの構造を単純化することができる。最長の辺を含む三角形のタイプに依存して、スパンは左から右(図6aのタイプIの

三角形9)または右から左(図6bのタイプIIの三角形10)のどちらかの向きに補間される。完全に水平な１辺をもつ三角形は、どちらのタイプとしても扱うことができる。

【００５２】

図７を参照して、充填スキャン・コンバータはx'0、y0およびm2として示される値を使って図示された三角形を充填する。この例では、x'0は第１のスパンのY値と交差する辺１上の点のX座標値であり、整数である必要はない。値y0は、第１のスパンのY座標であり、整数である。値m2は、辺１の勾配(yの変化で割ったxの変化)である。始点の画素のX値の選択は、X方向にステップする次のX値に基づく。ステップするX方向は、m2の符号によって決定することができる。スパンの終点を決定するために、対向する辺(辺２または辺３のいずれも)もまた同様の方法でステップされる。

【００５３】

以上のことをまとめてみると、充填スキャン・コンバータは以下の値を使って、平面方程式の一つの変形を表す。

【００５４】

１．平面方程式の一般的な形式からのそれぞれのパラメータの「a」値。この値はdp/dxであり、X方向に関するパラメータ「p」の導関数に等しい。dP_dX(ここでP=[R, G, B, Z])と表す。

【００５５】

２．平面方程式の一般的な形式からのそれぞれのパラメータの「b」値。この値はdp/dyであり、Y方向に関するパラメータ「p」の導関数に等しい。この値は充填スキャン・コンバータに渡される前に、「a」が上記の通りの場合、[切り捨て(edge1_slope)]の値と結合される。「切り捨て(edge1_slope)」は辺１の勾配dx/dyの整数部分である。このように、(b+切り捨て(edge1_slope))の量が使用される。この値はdP_dE(ここでP=[R, G, B, Z])と表す。

【００５６】

３．p0(生成される第１の画素)のパラメータの始点値。これらはP_Start(ここでP=[R, G, B, Z])と表す。

【００５７】

さらに、充填スキャン・コンバータは以下のX,Y辺情報を使用して、三角形の辺をステップし、三角形のスパンを定義する。

【００５８】

４．三角形の全ての３辺の勾配dx/dy。これらはedgeN_slope(ここでN=[1, 2, 3])と表す。

【００５９】

５．辺１の始点および終点のY値。充填スキャン・コンバータはこれらの値を丸めるので、辺１の始点と終点のY値は、頂点0と頂点2の実際のy軸の値と等しい。図6aおよび図6bに示されるように、辺１が常に(Yに関して)長い辺になるように、頂点が典型的にソートされる。これらを、edge1_y_startおよびedge1_y_stopと表す。

【００６０】

６．辺３の始点のY値。この値は特定のY交差に合わせて調整する必要はない。充填スキャン・コンバータは、典型的にこの値を内部に(internally)切り詰める(truncate)。最初に辺1および辺2がスパンを定義したあとに、この値が使用される。しかし、Yスパンがedge3_y_startより大きいかまたは等しくなると、辺１および辺３がスパンを定義する。

【００６１】

７．それぞれの辺の始点のX値。その辺の整数の始点のY値に合わせて調整される。(X値自体は、偶然を除いて通常整数ではない。)これらは、edgeN_x_start(ここでN=[1,2,3])と表す。

【００６２】

８．三角形が図6aおよび図6bに示されるようなタイプIの三角形であるかまたはタイプIIの三角形であるかを指示するフラグ(X_increment)。図6aに示されるタイプIの三角形に関

して、辺１と辺２または辺１と辺３の間のすべてのスパンは、X方向で増加する。図6bに示されるタイプIIの三角形に関して、辺１と辺２または辺１と辺３の間のすべてのスパンは、X方向で減少する。

【００６３】

三角形の3つの頂点から平面方程式を生成するために、すべての3つの頂点が形式(X,Y, Z, 赤、緑、青)の座標を持つと仮定することができる。言い換えると、それらはすべて、XおよびY座標とともに奥行きおよび色の情報を持つ。ここで示された概念はすべて、頂点がXおよびY座標のみ、または上記のパラメータの任意の組み合わせをもつ例についても等しく適用することができる。

【００６４】

上述したように、平面方程式ジェネレータは、四辺形が2つの三角形に分割された後の三角形の定義を受け取り、これらの三角形の定義に基づいて平面方程式を生成することができる。これは、図８で６つのステップ120、122、124、126、128、130に分割されたプロセスに従って実行される。これらのステップの順序は、図８に示される順序で実行される必要はなく、ある範囲で同時にまたは平行に実行されることができる。ここで、これらのステップについて詳細に述べる。説明では、頂点はすでにv0.y＜=v1.y＜v2.yというようにソートされていると仮定する。言い換えると、頂点V0のY軸の値は、頂点V1のY軸の値よりも小さいかあるいは等しく、また図6aおよび図6bに示すように頂点V2のY軸の値よりも小さい。

【００６５】

平面方程式ジェネレータは、最初に辺のデルタ（変化量）を計算する(ステップ120)。１つのパラメータの辺のデルタ（変化量）は、そのパラメータがある辺に沿って変化する量である。Z軸(奥行き)および色のパラメータについては２つの辺のデルタのみ必要とされる。なぜなら、偏導関数を決定するにはこのような２つの値だけで十分だからである。また以下の方程式で示されるように、辺のデルタは１方向に関してそれぞれに定義されるため、三角形のまわりで一貫する方向を持つ。これは四辺形を分解するのに有用である。しかし、平面方程式を生成するために、一般に外積が２辺の間で決定され、２辺の両方に隣接する頂点が固定される。実例のため、固定される頂点は、ここで頂点V0として示す。しかしその他のどの頂点も外積の固定される頂点として選ぶことができ、その場合、以下の方程式は異なる形式で表される。以下の方程式で、変数「n」は変数が一貫した辺方向に関する正規化座標であることを示し、「mn」は変数が一貫した辺方向に関する正規化座標の負であることを示す。例えば、nx1は頂点0から頂点1(辺2)へのX軸の正規化された変化である。また、mnx2はが辺1についてX軸に沿った正規化の変化の負であり、辺1のdxと呼ばれる。以下に示すように、mnx2は、頂点0(v0.x)のX値から頂点2(v2.x)のX値を引いた差に等しい。辺の指定に関しては、図6a、図6bまたは図７を参照されたい。最後に、dxe3(辺3のdx)は、辺３すなわち頂点V2から頂点V1へのX軸に沿ったX軸の変化である。同様の変数が、他のパラメータについても使用される。

【００６６】

【数８】

```
mnx2 = v0.x - v2.x
nx1  = v1.x - v0.x
dxe3 = v2.x - v1.x
mny2 = v0.y - v2.y
ny1  = v1.y - v0.y
dye3 = v2.y - v1.y
mnz2 = v0.z - v2.z
nz1  = v1.z - v0.z
mnred2 = v0.red - v2.red
nred1  = v1.red - v0.red
mngreen2 = v0.green - v2.green
```

ngreen1 = v1.green - v0.green
mnblue2 = v0.blue - v2.blue
nblue1 = v1.blue - v0.blue
【００６７】
それから平面方程式ジェネレータは、XおよびYに関して辺の勾配を計算する(ステップ122)。上記で計算されたいくつかの値を用いて、例えば辺1の勾配(edge1_slope)は辺1のYの変化(上記mny2)で割った辺1のXの変化(上記mnx2)に等しく、他の辺についても同様の計算が行われる。
【００６８】
【数９】
edge1_slope = mnx2/mny2
edge2_slope = nx1/ny1
edge3_slope = dxe3/dye3
【００６９】
そして平面方程式ジェネレータは、辺の始点/終点のY値を初期化する(ステップ124)。これらは、辺1の始点のY値(edge1_y_start)、辺1の終点のY値(edge1_y_stop)、および辺3の始点のY値(edge3_y_start)を含む。これらは調整された２次的な画素ではないので、値は必ずしも整数ではなく、頂点のY軸値から直接決定することができる。
【００７０】
【数１０】
edge1_y_start = v0.y
edge1_y_stop = v2.y
edge3_y_start = v1.y
【００７１】
次に、平面方程式ジェネレータは、最も近い頂点(図7の頂点V0)から第１のYスパン交点まで辺(図7の辺1)に沿って調整した、辺の始点のXを計算する(ステップ126)。これらの値(edge1_x_start、edge2_y_startおよびedge3_y_start)を生成するために、最初に2つの端数が決定される。端数frac_y0は、頂点V0からY軸に沿って次に大きい整数Y(すなわち第1のYスパン)までの量であり、frac_y1は頂点V1のY値と次に大きい整数Yとの間の差である。それぞれの始点Xは、以下の通り、辺の勾配の一つと端数の一つの積に頂点の一つのX値を加えることによって決定することができる。
【００７２】
【数１１】
frac_y0 = 切り上げ(v0.y) - v0.y
frac_y1 = 切り上げ(v1.y) - v1.y
edge1_x_start = v0.x + edge1_slope*frac_y0
edge2_x_start = v0.x + edge2_slope*frac_y0
edge3_x_start = v1.x + edge3_slope*frac_y1
【００７３】
好ましい実施例で、切り上げ操作は、通常最も近くにある最も大きい整数に丸めることである。例えば(-1.3)は(-1.0)に丸められる。同様に切り捨ては、好ましくは通常次に小さい整数に丸めることである。そして平面方程式ジェネレータは、Zおよび色ならびに三角形のタイプ(IまたはII)に関して偏導関数を計算する(ステップ128)。最初に、三角形がタイプIまたはタイプIIのどちらであるか示す、偏導関数の分母(one_over_cd)を決定する。偏導関数の計算はすべて頂点0から頂点2および頂点0から頂点1の２つのベクトルの外積に基づいている。しかし、１つの三角形には１つの平面だけが定義されうるので、いかなる平面についてもその平面の法線は１つしか存在せず(符号および大きさは無視する)、従って外積はどの頂点(0,1または2)に基づいてもよい。
【００７４】
【数１２】

$$one_over_cd = \frac{1.0}{mny2*nx1 - mnx2*ny1}$$

$$x_increment = [1\ (one_over_cd < 0.0\ (sign(one_over_cd)) \quad [0\ (one_over_cd \geqq 0.0\ (sign(one_over_cd))$$

$$el_slope = truncate(edgel_slope)$$

$$dz/dx = (mny2*nz1 - mnz2*ny1)*one_over_cd$$
$$dz/dy = (mxz2*nx1 - mnx2*nx1)*one_over_cd$$
$$dz/del = dz/dy + dz/dx*el_slope$$

$$dred/dx = (mny2*nred1 - mnred2*ny1)*one_over_cd$$
$$dred/dy = (mnred2*nx1 - mnx2*nred1)*one_over_cd$$
$$dred/del = dred/dy + dred/dx*el_slope$$

$$dgreen/dx =(mny2*ngreen1 - mngreen2*ny1)*one_over_cd$$
$$dgreen/dy =(mngreen2*nx1 - mnx2*ngreen1)*one_over_cd$$
$$dgreen/del =dgreen/dy + dgreen/dx*el_slope$$

$$dblue/dx =(mny2*nblue1 - mnblue2*ny1)*one_over_cd$$
$$dblue/dy =(mnblue2*nx1 - mnx2*nblue1)*one_over_cd$$
$$dblue/del = dblue/dy + dblue/dx*el_slope$$

【００７５】
この中で「truncate」操作とは、0値により近い次の整数に丸めることをいう。次に平面方程式ジェネレータは、頂点から第１の画素まで調整した、Zおよび色の始点値を計算する(ステップ130)。
【００７６】
【数１３】

```
frac_x0 = [切り上げ(edge1_x_start) - v0.x (x_increment = 1の場合)]
          [切り捨て(edge1_x_start) - v0.x (x_increment = 0の場合)]

z_start     = v0.z     + dz/dy*frac_y0     + dz/dx*frac_x0
red_start   = v0.red   + dred/dy*frac_y0   + dred/dx*frac_x0
green_start = v0.green + dgreen/dy*frac_y0 + dgreen/dx*frac_x0
blue_start  = v0.blue  + dblue/dy*frac_y0  + dblue/dx*frac_x0
```

【００７７】
one_over_cdの符号は、外積が面(page)の外または中のどちらを指しているのか決定する。このように、この符号は三角形がタイプIまたはタイプIIのどちらの三角形であるか決定するために使用することができる。
【００７８】
始点/終点の値のうちY以外はすべて、充填スキャン・コンバータによる使用のために頂点から離れて調整される。edgeN_x_startは、頂点xから、辺と頂点の下の第１のスパンとの交点に調整される。Z、赤、緑、青の始点はすべて、充填スキャン・コンバータによって頂点V0からレンダリングされる第１の画素に調整される。充填スキャン・コンバータによって計算される最初の画素は、図７に示されるように、V0のすぐ下のスパンの左端にある１画素であり、辺1の上またはその内側にある。
【００７９】
上述したように、四辺形は２つの個々の三角形に分割することができ、それぞれの三角形は適当な平面方程式を生成する平面方程式ジェネレータに個別に入力される。図9aは２つの三角形に分割できる四辺形を示し、図9bは２つの三角形に分割できる凹形四辺形を示す。図9aおよび図9bから分かるように、結果は１つの共有される辺をもつ２つの三角形である。
【００８０】
発明の一つの見地によれば、共有される辺および２つの共有される頂点は、詳しくは第２の三角形に関して上述の平面方程式の計算の一部を省くために利用される。第１の三角形ついて平面方程式が生成される一方で、第２の三角形について一部の計算を始めることによって、付加的なパフォーマンスを得ることができる。
【００８１】
図10aは、図9aに示される四辺形の辺および頂点について命名規則を示し、図10bは、図9bに示される四辺形の辺および頂点の命名規則を示す。例えば図10aにおいて、第１の三角形101は、3つの頂点V1.0、V1.1およびV1.2、ならびに辺1.1、辺1.2および辺1.3をもつ。第２の三角形102は、頂点V2.0、V2.1およびV2.2、ならびに辺2.1、辺2.2および辺2.3をもつ。これらの図で、最初の数字は三角形を識別し、２番目の数字は三角形の頂点または辺を表す。例えば、V2.1は第２の三角形102の頂点1である。これらの命名規則は、同一の辺1(三角形102の辺1.1および三角形101の辺2.1)および同一の頂点0(V1.0およびV2.0)および頂点2(V1.2およびV2.2)を共有する、２つの三角形の生成を容易にする。同様の共有が、図10Bの第１の三角形103と第２の三角形104の間にも存在する。
【００８２】
辺1は四辺形の２つの三角形の間で共有されているので、第２の三角形の平面方程式の生成は、第１の三角形の平面方程式による結果を用いて実行することができ、その結果処理効率が改善される。例えば、変数mnx2は辺1に関するX軸の正規化された変化であり、辺1のdxと呼ぶ。mnx2は図10aの三角形101についてすでに計算されており、指定によって三角形101および三角形102は辺1を同一にするので、mnx2は三角形102に関して再度計算する必要がない。如何に処理効率を改善することができるかという方法について、より詳細に以

下で説明する。

【００８３】

再び図４を参照して、レジスタ180は、RAM196よりも速く、頂点データのようなデータについてALU184、乗算器186および除算器188にアクセスする。従って、RAM196に代わってレジスタ180からパラメータをアクセスできれば、動作はより短い時間で実行される。発明の一つの実施例では計算を実行する前に、通常、パラメータがRAM196からフェッチされ、レジスタ180に入れられる。そしてレジスタ180から同じパラメータが直接アクセスされる場合、アクセス時間が減少する。しかし第２の三角形の平面方程式を生成するとき、四辺形の２つの三角形に共有される辺および共有される頂点に関連する任意のデータおよびその他の中間のパラメータに関して、RAM196からレジスタ180へのフェッチを省くことができる。この説明の中で「フェッチ」という語は、RAM196からレジスタ・ファイル180へ数値を移す操作を説明するために用いられる。

【００８４】

第２の三角形に関して省略することができるRAM196からレジスタ180へのフェッチは、v0.x、v0.y、v0.z、v2.x、v2.y、v2.z、mnx2およびmny2を含む。なぜならこれらの値は、第１の三角形について実行された計算の結果としてレジスタ180にすでに格納されているからである。これらの値は頂点0、頂点2または辺1と関連するのですでに格納されており、それらの値ははすべて、図10aおよび図10bに示される指定の結果として両方の三角形に同一である。

【００８５】

その上、mnz2、mnred2、mngreen2、mnblue2およびedge1_slopeの値の計算は、第２の三角形に関して省略することができる。これらの計算を省略できるのは、これらの値が第１の三角形について実行された計算の結果としてすでに計算されており、図10aおよび図10bに示すように共有される辺1と関連しているからである。

【００８６】

図11は、本発明の一つの実施例に従って、四辺形に関する平面方程式を計算するプロセスを示している。ステップ170では、四辺形の定義を受け取る。ステップ171で、四辺形を第１の三角形と第２の三角形に分割する。先に述べたように、このステップは、それぞれの三角形について２つの共有される頂点と一つの共有される辺が存在するように、四辺形を分割することを含むことができる。さらに、分割するステップは、それぞれの三角形の共有される頂点と共有される辺が同じ指定をもつように三角形の辺および頂点を指定することを含むことができる。ステップ172で、第１の三角形の平面方程式が生成される。第１の三角形の平面方程式は充填スキャン・コンバータに与えられる(ステップ174)。ステップ176で、第１の三角形の平面方程式を生成した結果を用いて第２の三角形の平面方程式を生成する。例えば、ステップ176は、レジスタにすでに格納されている値のような、第１の三角形の平面方程式を生成した中間の結果を使用することを含むことができる。さらにその結果は、mnz2等のような平面方程式の定数を含むことができる。従って、ステップ176は、ステップ174が完了する前に始めることができる。図４に示した回路は個別のALU184、乗算器186、および除算器188を含むので、いくつかの操作を平行して実行することができる。ステップ178で、第２の三角形の平面方程式を充填スキャン・コンバータに与える。

【００８７】

図12は、本発明の一つの実施例で実行されるフェッチおよび計算の操作を表にしたものであり、上で述べた方法で三角形を指定し、フェッチおよび計算のような重複した動作を省くことによって達成される利点を示している。フェッチおよび計算の一部は、二回実行してもよいことに注意されたい。例えば一つの実施例では、図12に示されるように、edge1_slopeは第１の三角形と同様に第２の三角形についても生成される。さらにv0.x、v2.xおよびv2.yのフェッチは、第２の三角形について省略されなかった。平面方程式の生成を実行するために用いられる異なった構造に基づいて、これは速度に影響を与えうる。しかし回路の構造を単純化するために、edge1_slopeの計算を代わって２回実行することは有

利でありうる。このような実施例では、おそらく２クロック・サイクルが加わることになる。

【００８８】

さらに図４のような配置において、乗算器186には多くの場合ALU操作よりも多くの乗算/除算の操作があるので、しばしば処理のスループットに関する障害となる。また図４に示した実施例で、乗算器186および除算器188は、各々の入力および出力について効果的に時分割多重送信(time multiplex)する共通のバスを共有する。その結果、上の表で示したフェッチのようにいくつかのステップを省略できたとしても、乗算器186それ自身のスループットを増やさない限り、ALU184および除算器188によるレジスタ・ファイル180へのアクセスがスループットを減少させることはできないので、スループットに影響を与えることはできない。

【００８９】

しかし、代替の実施例では、edge1_slopeの計算およびv0.x、v2.x、v2.yのフェッチが、両方とも第２の三角形に関して省略される。省略するフェッチや計算の組合せは本発明の範囲内である。

【００９０】

上で示したように、図４は、三角形の生成および平面方程式の生成を実行するシステムの好ましい実施例を示す。しかしこれらの機能は、図示のハードウェアまたはソフトウェアのどちらでも、またはハードウェアおよびソフトウェアの組み合わせとして、実行することができる。ソフトウェアの実施例で、ソフトウェアは、汎用コンピュータまたは必要な機能を実行するために最適化されたコンピュータで実行することができる。このようなコンピュータは、一般にCPU、ハード・ドライブまたはEPROMのようなプログラム・メモリ、ランダム・アクセス・メモリのようなデータ・メモリ、およびこれらの素子の間のインターフェースへの処理のバスを含む。また、通信デバイスを充填スキャン・コンバータへのインターフェースに使用することもできる。

【００９１】

発明の別の見地は、頂点のソーティングに関連する。充填スキャン・コンバータは、典型的には、表示画面の最上部のスパンを横切ってステップし、それからより下のスパンに進み(Y値を増やす)、より低いスパンのステッピングを実行する(図５参照)。最上部のスパンを最初に実行するというこの命令は、頂点V0を最上部の頂点として定義することによって充填スキャン・コンバータに指示することができる。従ってグラフィックス・システムは、３つの頂点V0、V1、V2に関するデータのような三角形の情報を受け取る頂点ソーターを含むことができる。このような頂点ソーターは頂点に関するデータをソートして、平面方程式ジェネレータにソート順にデータを供給する。図6aおよび図6bに関して述べたように辺および頂点の指定を保持するために、この頂点のソーティングによって辺のソーティングも行われる。平面方程式ジェネレータは、例えば充填スキャン・コンバータに供給される平面方程式を生成するという図８に示したステップを実行する。

【００９２】

しかし、図８で計算する平面方程式のある値だけが頂点のソートに依存する。こうして、ソーティングは図８に示すプロセスのうちの最初の数ステップと同時に行うことができ、その結果プロセスはより効率的になり、必要な処理資源が削減される。

【００９３】

頂点のソーティングと平面方程式の生成は、上述のように、図４および図15に示す処理ユニットによって同時に実行することができる。

【００９４】

典型的に、頂点のソーティングは、最大のものに対する最小のものを見つけるためにそれぞれの頂点の対のY座標値を比較することを含む。これらのY座標値を比較するプロセスには計算サイクルが余分にかかるが、代わりにその時間を使って平面方程式を計算し始めることができる。その代わり、Yをソーティングする命令は、図８で先に述べた平面方程式の一部として計算される辺のデルタYでコード化されることに注意されたい。辺のデルタ

は頂点に関して一貫して定義されるので(図6aおよび図6bを参照)、辺のデルタYの符号を使ってソーティングの結果を決定することができる。

【００９５】

図13は、異なったソートの結果になる三角形のいくつかの例を示す。例えば、三角形201は、辺と頂点の表記が図6aおよび図6bの指定に従っているのでソーティングを必要としない。しかし三角形203については、頂点V1が、最小のY座標値をもつので、三角形201のような指定を得るために頂点V0と交換されなければならない。

【００９６】

図14は、図13のそれぞれの例に関して頂点と辺の適当な交換を行うための正確な表を示している。図14に示すように、ソートされた頂点の順序およびソートされた辺の順序は、それぞれの辺に関する辺のデルタdyの符号から決定することができる。「辺3のDY符号」、「辺2のDY符号」および「辺1のDY符号」の「0」は、辺のデルタdyの符号が正であることを示し、「1」は辺のデルタdyの符号が負である。例えば、三角形202について、辺1のdy(頂点V0から頂点V2)が正であり、辺2のdy(頂点V1から頂点V0)は負であり、辺3のdy(頂点V2から頂点V1)が正である。「ソート後の頂点の順序」とは、オリジナルの頂点に関するソート後の頂点の順序をいう。例えば(b)の場合、ソート後の頂点の順序は(2,0,1)であり、ソート後の頂点V2はオリジナルの頂点V0と等しく、ソート後の頂点V0はオリジナルの頂点V1と等しく、ソート後の頂点V1はオリジナルの頂点V2と等しいことを意味する。同様に、ソート後の辺の順序は、オリジナルの辺に関するソート後の辺の順序を示す。

【００９７】

図８および平面方程式を生成するステップに関して上述したように、辺のデルタはそれぞれ、三角形のまわりで一貫した方向を持つように１方向で定義される。三角形202についてソーティングは、頂点および辺の指定を時計回りに効果的に回転させることによって達成される。そのようにして、図14はソート後の頂点の順序およびソート後の辺の順序を示している。図14は、図13に示す三角形201ないし206のそれぞれの結果の頂点の順序と辺の順序を含む。また図14は、図13に示さなかった２つの付加のケースも示している。具体的には、ケース(g)はそれぞれの辺が正の辺のデルタをもつ場合を示し、ケース(h)はそれぞれの辺が負の辺のデルタをもつ場合を示す。先に述べたような辺の定義では、このようなケースは通常起こらない。しかし、例えば三角形が縮小されて有効に点に変形した場合、すべての辺のデルタが0となることができ、それぞれの辺のデルタの符号は正または負である。三角形が変形して水平線になる場合にも同様の状況が存在しうる。従って、(g)および(h)に示されたケースは存在しうる。これらのケースでは、頂点が全て効果的に等しいので、実際のソーティングは必要でない。従ってこれらのケースにおいては、ソート後の頂点の順序およびソート後の辺の順序は、任意の頂点のソーティングの前の順序になりうる。

【００９８】

図15は、頂点をソーティングするための回路の一つの実施例を示し、図４に示した平面方程式の計算回路の一部として実行することができる。回路は使用されうるロジックを表わしており、代替の実現を効果的に行って、同様の目的を達成することもできる。回路は、ram_sign信号229上で受け取られる３辺のデルタYの符号を保持する入力レジスタ220A、220Bおよび220Cを含む。論理回路222の出力は出力レジスタ224A、224Bおよび224Cに供給され、ソートされた信号を供給し、それらの信号は図17のアドレス・マップに示される方法で図４のレジスタ・ファイル180をアドレスする際に使用される。

【００９９】

操作上、それぞれの入力レジスタ220A、220Bおよび220Cは、辺のデルタのそれぞれの符号を示す時分割多重送信されたデータを運ぶram_sign信号229を受け取る。ライン226上の信号set_e1_dyは、ram_sign信号229が辺1のデルタYの符号を保有することを示し、ライン227上の信号set_e2_dyは、ram_sign信号229が辺２のデルタYの符号を保有することを示し、ライン228上の信号set_e3_dyは、ram_sign信号229が辺３のデルタYの符号を保有することを示す。辺のデルタYの符号信号は、例えば辺のデルタのそれぞれを計算する図８のステ

ップ120から受け取られる。論理回路222は、ソート後の頂点の順序を決定するために図14に示されるロジックに従ってロジックの操作を行い、レジスタ224Aないし224Cに出力を与える。例えば出力レジスタ224Aは、ソート後の頂点V0を表わす２ビットの値を保有し、それを出力信号sorted_v0としてライン231上に供給する。出力レジスタ224Bはソート後の頂点V2を表わす２ビットの値を保有し、それを出力信号sorted_v2としてライン232上に供給する。出力レジスタ224Cは、ソート後の頂点V1に関する値を保有し、出力信号sorted_v1としてライン233上に供給する。ライン230上の信号set_sort_regsは、ソート後の頂点の値を保有するために、出力レジスタ224Aないし224Cをラッチ(latch)するために使用することができる。さらに論理回路222は、ライン234上に信号sorted_edges_same_as_verticesと指定される別の出力を供給し、ソーティングの結果として三角形のタイプが入れ替わらなかったことを示すために用いられる。(三角形のタイプIおよびIIについて図6aおよび図6bを参照)。図15に示す回路が、ソート後の辺の番号を供給する必要はない。ソート後の辺の番号は、ライン231ないし233のソート後の頂点の信号およびsorted_edges_same_as_vertices信号234から解読することができる。

【０１００】

図15に示す回路からの出力が与えられれば、図４に示す回路を使って一つの典型的な実施例におけるソートされた平面方程式を生成することができる。一つの実施例では、図15からの出力はレジスタ・ファイル180の頂点データへのポインタとして使用されるため、ソーティングの結果としての実際上のデータ交換を回避する。図17は、この実施例で使用されるレジスタ・ファイル180のためのメモリ・マップを示す。図17でラベル256は、頂点V0、頂点V1または頂点V2のいずれの頂点に関連してレジスタ・ファイル180に格納される値を示す。アドレス252は十進アドレスであり、アドレス254はアドレス252に一致する２進アドレスである。領域256はそれぞれのアドレスの内容を定義する。例として、X、Y、Z、R、G、Bの値が頂点のそれぞれと関連付けられている。これらの値の一つにアクセスするために、適当なアドレスが与えられる。二進アドレスの最初の２つの数字は、どの頂点が選択されるかを示し(頂点V0の場合01、頂点V1の場合10、頂点V2の場合11)、最後の３つの二進数字はその頂点の個々の値を示す。このように、最初の２つの数字はソート後の頂点を表すことができる。これらの２つの数字は図15の信号224AないしCからの出力であるので、頂点データを再配置する必要はない。また図17の頂点データは、辺情報、詳しくは３辺のそれぞれについての辺のdxおよび辺のdyを含むことに注意されたい。図17に示す配列によって、これらの値はまた、図15に示す回路の出力から参照することできる。

【０１０１】

上に示したように、図４の回路を使って四辺形の迅速な生成を行うことができる。ソートされた平面方程式の生成を行う際、ALU184、乗算器186および除算器188によって速くアクセスできるように、頂点および辺のデータを、RAM196からフェッチてレジスタ・ファイル180に格納することができる。レジスタ・ファイル180は、２ビットのソートされた頂点の信号224AないしCがレジスタ・ファイル180から所望のソート後の頂点を選ぶことができるように、頂点データと辺をレジスタ・ファイルに格納する。さらに、３ビットのフィールドを使って、所望のソートされた頂点の適当なパラメータ(例えばX、Y、Z、赤、緑、青)を指定することができる。従って、sorted.v0.xを実行される計算への入力として要求する場合、レジスタ180に２ビットの頂点指定および３ビットのパラメータ指定を含むアドレスを与えることによって、データをレジスタ180から読み取ることができる。

【０１０２】

図16および以下の方程式は、ソート後のパラメータ値を使って平面方程式を計算する際、図４の回路によって実行される操作を示している。操作は、図16に示された順序で実行される必要はなく、重複させたり、同時に実行することも可能である。以下の方程式で、変数名にある「sorted」の語は、ソーティングの後の値を示す。従って、例えばオリジナルの頂点が、v0、v1およびv2(v1.y＜v0.y＜v2.y)である場合、sorted.v0 = v1、sorted.v1 = v0およびsorted.v2 = v2である。

【０１０３】

ステップ132で、辺のデルタが図８のステップ120と同様の方法で計算される。従って、mnx2、nx1、dxe3、mny2、ny1、dye3、mnz2、nz1、mnred2、nred1、mngreen2、ngreen2、mnnblue2およびnblue1の値のそれぞれは、ソーティングの前またはソーティングの間に計算することができる。

【０１０４】

【数１４】

```
mnx2 = v0.x - v2.x
nx1 = v1.x - v0.x
dxe3 = v2.x - v1.x
mny2 = v0.y - v2.y
ny1 = v1.y - v0.y
dye3 = v2.y - v1.y
mnz2 = v0.z - v2.z
nz1 = v1.z - v0.z
mnred2 = v0.red - v2.red
nred1 = v1.red - v0.red
mngreen2 = v0.green - v2.green
ngreen1 = v1.green - v0.green
mnblue2 = v0.blue - v2.blue
nblue = v1.blue - v0.blue
```

【０１０５】

ステップ134で、ソート後の辺の勾配が計算される。これは、辺の勾配がソーティングなしで計算される図８のステップ122と区別される。図８のプロセスは、先にソートされた頂点データに依存するからである。従って、ソートされた辺の勾配は、出力レジスタ224AないしCからの出力を使って適当なソートされた頂点を決定し、以下に示すように計算することができる。アスタリスク(*)を伴う下の値は、ソーティングとは独立した値であるから、ソーティングの前または間に計算することができる。

【０１０６】

【数１５】

```
edge1_slope* = mnx2 / mny2
edge2_slope* = nx1 / ny1
edge3_slope* = dxe3 / dye3
sorted.e1_slope = edge1_slope または edge3_slope (ソートによる)
sorted.e2_slope = edge1_slope または edge2_slope (ソートによる)
sorted.e3_slope = edge2_slope または edge3_slope (ソートによる)
```

【０１０７】

例えば、sorted_edges_same_as_vertices信号が有効である場合、sorted.e1_slopeは、ソート後のV0のdxをソート後のV0のdyで割ったものとして計算される。そうでなければ、sorted.e1_slopeはソート後のV2のdxをソート後のV2のdyで割ったものとして計算される。(ソート後の頂点の指定について図17を参照。例えば「V0のdx」はedge1_dxをいう)。同様に、信号sorted_edges_same_as_verticesが有効である場合、sorted.e2_slopeはソート後のV1のdxをソート後のV1のdyで割ったものとして計算される。そうでない場合、ソート後のV0のdxをソート後のV0のdyで割ったものとして計算される。また、信号sorted_edges_same_as_verticesが有効である場合、sorted.e3_slopeはソート後のV2のdxをソート後のV2のdyで割ったものとして計算され。そうでない場合、ソート後のV1のdxをソート後のV1のdxで割ったものとして計算される。

【０１０８】

ステップ136で、ソート後の辺の始点/終点のY値が決定される。ソート後の頂点は分かっているので、始点/終点の値はソート後の頂点から直接決定することができる。

【数１６】

edge1_y_start = sorted.v0.y
edge1_y_stop = sorted.v2.y
edge3_y_start = sorted.v1.y

【０１０９】

それから図７のステップ126と同様の方程式を用いるが、代わりにソート後の値を使用して、辺の始点のXを計算することができる(ステップ138)。

【数１７】

frac_y0 = 切り上げ(sorted.v0.y) - sorted.v0.y
frac_y1 = 切り上げ(sorted.v1.y) - sorted.v1.y
edge1_x_start = sorted.v0.x + sorted.e1_slope* frac_y0
edge2_x_start = sorted.v0.x + sorted.e2_slope* frac_y0
edges_x_start = sorted.v1.x + sorted.e3_slope* frac_y1

【０１１０】

奥行きおよび/または色が要求される場合、ステップ140でZおよび任意の色に関する偏導関数が決定される。以下の方程式のいくつかは、三角形のソーティングには依存せず、ソーティング回路がソートを実行した時点の任意の順序で実行することができることに留意されたい。例えばすべての偏導関数は、辺に関するものを除いてソートに依存しない。x_increment値は上述のようにソートに依存する。x_incrementによって表される外積の符号は、三角形のタイプを変更した(タイプIからII、またはタイプIIからI)ソーティングとは反対になるからである。従ってsorted_edges_same_as_vertices信号234がアクティブであれば、x_incrementは前に述べたとおりである。しかしsorted_edges_same_as_vertices信号234がアクティブでない場合、これはタイプの入れ替えを示しており、x_incrementの信号は反対になる。

【０１１１】

【数１８】

$$one_over_cd^* = \frac{1.0}{mny2^*nx1 - mnx2^*ny1}$$

x_increment = [sign(one_over_cd) (sorted_edges_same_as_verticesの場合)]
[-sign(one_over_cd) (その他の場合)]

el_slope = truncate(sorted.el_slope)

dz/dx* = (mny2*nz1 - mnz2*ny1)*one_over_cd
dz/dy* = (mnz2*nx1 - mnx2*nz1)*one_over_cd
dz/del = dz/dy + dz/dx*el_slope

dred/dx* = (mny2*nred1 - mnred2*ny1)*one_over_cd
dred/dy* = (mnred2*nx1 - mnx2*nred1)*one_over_cd
dred/del = dred/dy + dred/dx*el_slope

dgreen/dx* = (mny2*ngreen1 - mngreen2*ny1)*one_over_cd
dgreen/dy* = (mngreen2*nx1 - mnx2*ngreen1)*one_over_ cd
dgreen/del =dgreen/dy + dgreen/dx*el_slope

dblue/dx* = (mny2*nblue1 - mnblue2*ny1)*one_over_cd
dblue/dy* = (mnblue2*nx1 - mnx2*nblue1)*one_over_cd
dblue/del = dblue/dy + dblue/dx*el_slope

【０１１２】
次に、ソート後のZおよび色の始点値を、頂点から第１の画素へ調整して計算する(ステップ142)。
【０１１３】
【数１９】
frac_x0 = [切り上げ(edge1_x_start)-sorted.v0.x (x_increment=1の場合)]
[切り捨て(edge1_x_start)-sorted.v0.x (x_increment=0の場合)]

z_start = sorted.v0.z + dz/dy*frac_yo + dz/dx*frac_x0
red_start = sorted.v0.red + dred/dy*frac_y0 + dred/dx*frac_x0
green_start = sorted.v0.green + dgreen/dy*frac_y0 + dgreen/dx*frac_x0
blue_start = sorted.v0.blue + dblue/dy*frac_y0 + dblue/dx*frac_x0

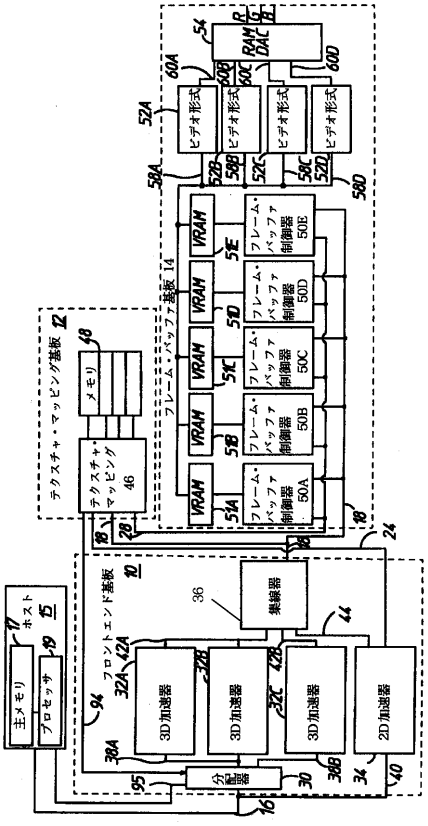
【０１１４】
この結果、頂点のソーティングは、別の方法で必要とされるよりも速い時間で少ない処理資源を用いて実行することができる。例えば辺のデルタYは、平面方程式の生成のためにすでに算出されていることになっているので、それぞれの頂点のY座標値の比較を行う代わりに、これらのデルタを比較の基準に使用して、頂点のソーティングを実行することができる。それぞれの辺のデルタYは、一貫した辺の指定に従ってY軸がその辺に関して増加しているかを示す符号をもっているので、論理回路を使って頂点を速くソートしてソート後の頂点を与えることができる。
【０１１５】
さらに、ソーティングのステップ全体は、平面方程式の生成に関して効果的に明白である。ソーティング操作を実行し、そのソーティング操作の結果に基づいて平面方程式を計算する代わりに、平面方程式を一つのステップの中で効果的に生成することができる。
【０１１６】
同時に実行されるソーティングは、図４に示すようなハードウェア、ソフトウェア、またはハードウェアとソフトウェアの組み合わせのいずれにおいても実行することができる。
【０１１７】
本発明は例として次の実施態様を含む。
【０１１８】
（１）コンピュータ・グラフィックス・システムで、三角形を表わす画素データを生成する装置であって、
三角形を表す頂点データ(e1，e2，e3)に応じて平面方程式データの第１の部分を生成する手段と、
ソートされた頂点データを供給するために、上記平面方程式データの上記第１の部分に応じて、上記平面方程式の計算を容易にするソーティング規準に従って上記頂点データをソートする手段(220，222，224)と、
上記ソートされた頂点データに応じて、上記頂点データのソーティングに従って上記平面方程式データの第２の部分を生成する手段と、
を含む、三角形を表わす頂点データに応じて平面方程式データを生成する処理回路(185，181，190)と、
上記平面方程式データの第１および第２の部分に応じて上記三角形を表わす画素データを生成する充填スキャン・コンバータ(14)と、
を備える、上記装置。
【０１１９】
（２）充填スキャン・コンバータ(14)が第１の軸および第２の軸を含むディスプレイの画素データを生成し、
上記平面方程式データの上記第１の部分を生成する手段が、三角形の第１の辺上の第１の軸の座標値の差を表す、第１の符号をもつ第１の辺のデルタを生成する手段と、三角形の第２の辺上の第１の軸の座標値の差を表す、第２の符号をもつ第２の辺のデルタを生成する手段と、三角形の第３の辺上の第１の軸の座標値の差を表す、第３の符号をもつ第３の辺のデルタを生成する手段とを含む、
上記(1)の装置。
（３）ソートする手段が、第１の符号、第２の符号および第３の符号に基づいて上記ソートされた頂点データを決定する論理回路(222)を含む、上記(2)の装置。
（４）ソートする手段が、第１の符号(226)、第２の符号(227)および第３の符号(228)に基づいて上記ソートされた頂点データを決定する手段(222)を含み、
処理回路(185，181，190)が、頂点データ(e1，e2，e3)を格納するレジスタ・ファイル(180)含み、上記ソートされたデータを決定する手段(222)が、ソートされた頂点データを供給するために、レジスタ・ファイル(180)に格納された頂点データ(e1，e2，e3)を参照するソートされたアドレスを与える手段を含む、
上記(2)の装置。

【０１２０】
（５）平面方程式データに応じて三角形を表わす画素データを生成する充填スキャン・コンバータ(14)を含むコンピュータ・グラフィックス・システムで、平面方程式データを生成する方法であって、
三角形を示す頂点データ(e1，e2，e3)を受け取るステップと、
頂点データがソートされているかどうかに依存しない平面方程式データの第１の部分を生成するステップと、
ソートされた頂点データを生成するために頂点データをソートするステップと、
ソートされた頂点データを使用して平面方程式データの第２の部分を生成するステップと、
平面方程式データの第１の部分および第２の部分を充填スキャン・コンバータ(14)に供給するステップと、
を含む、上記方法。
【０１２１】
（６）充填スキャン・コンバータ(14)が第１の軸および第２の軸を含むディスプレイの画素データを生成し、平面方程式データの第１の部分を生成するステップが、
三角形の第１の辺上の第１の軸の座標値の差を表す、第１の符号をもつ第１の辺のデルタを生成するステップと、
三角形の第２の辺上の第１の軸の座標値の差を表す、第２の符号をもつ第２の辺のデルタを生成するステップと、
三角形の第３の辺上の第１の軸の座標値の差を表し、第３の符号をもつ第３の辺のデルタを生成するステップと
を含む、上記(5)の方法。
【０１２２】
（７）ソートするステップが、第１の符号、第２の符号および第３の符号に基づいて、ソートされた頂点データを決定することを含み、
頂点データを受け取るステップが、頂点データ(e1，e2，e3)をレジスタ・ファイルに格納することを含み、
ソートするステップが、ソートされた頂点データを供給するために、レジスタ・ファイルに格納された頂点データ(e1，e2，e3)を参照するソートされたアドレス・データを生成することを含む、
上記(6)の方法。
【０１２３】
（８）平面方程式データに応じて三角形を表わす画素データを生成する充填スキャン・コンバータ(14)を含むコンピュータ・グラフィックス・システムで、平面方程式データを生成する装置(181，185，190)であって、
三角形を示す頂点データを受け取る手段と、
頂点データがソートされているかどうかに依存しない平面方程式データの第１の部分を生成する手段と、
ソートされた頂点データを生成するために頂点データをソートする手段と、
ソートされた頂点データを使用して平面方程式データの第２の部分を生成する手段と、
平面方程式データの第１の部分および第２の部分を充填スキャン・コンバータ(14)に供給する手段と、
を備える上記装置。
【０１２４】
（９）充填スキャン・コンバータ(14)が第１の軸および第２の軸を含むディスプレイの画素データを生成し、平面方程式データの第１の部分を生成する手段が、
三角形の第１の辺上の第１の軸の座標値の差を表す、第１の符号をもつ第１の辺のデルタを生成する手段と、
三角形の第２の辺上の第１の軸の座標値の差を表す、第２の符号をもつ第２の辺のデルタ

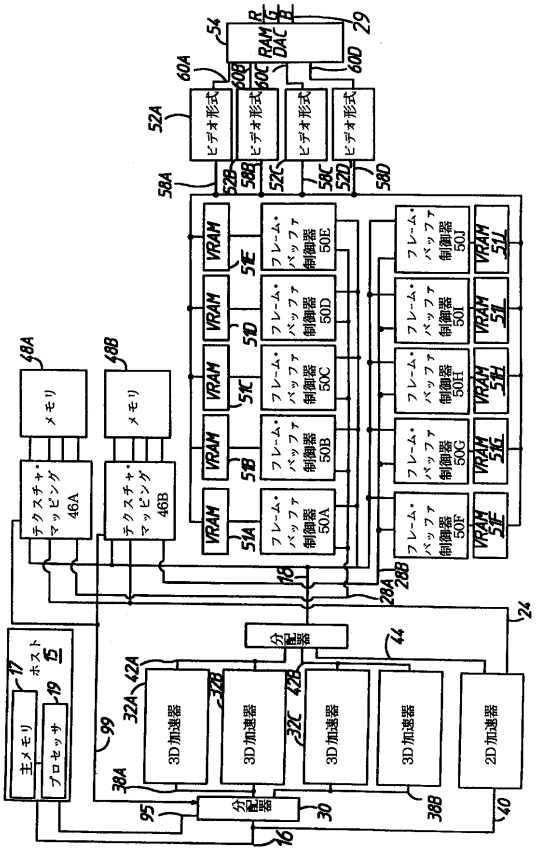
を生成する手段と、
三角形の第３の辺上の第１の軸の座標値の差を表す、第３の符号をもつ第３の辺のデルタを生成する手段と
含む、上記(8)の装置。
【０１２５】
（１０）ソートする手段が、第１の符号、第２の符号および第３の符号に基づいてソートされた頂点を決定する手段を含み、
頂点データを受け取る手段が、レジスタ・ファイル(180)に頂点データ(e1，e2，e3)を格納する手段を含み、
ソートする手段が、ソートされたデータを供給するために、レジスタ・ファイル(180)に格納された頂点データを参照するソートされたアドレスを生成する手段を含む、
上記(9)の装置。
【０１２６】
【発明の効果】
本発明によれば、コンピュータ・グラフィックス・システムで三角形を表す画素データを生成するプロセスにおいて、平面方程式の生成と頂点のソーティングを同時に実行することにより、そのプロセスがより効率的になり、必要な処理資源が削減される。
【図面の簡単な説明】
【図１】本発明を具体化したコンピュータ・グラフィックス・システムの第１の実施例のブロック図である。
【図２】本発明を具体化したコンピュータ・グラフィックス・システムの第２の実施例のブロック図である。
【図３】本発明の実施例を具体化した加速器のブロック図である。
【図４】図１および図３のそれぞれの加速器の一部分のブロック図であり、 平面方程式のデータを生成する処理回路を示す。
【図５】充填スキャン・コンバータによって実行される時に充填される三角形のスパンの生成を示す。
【図６】図6aは、最も長い辺が三角形の左側にある三角形の第１のタイプを示す。図6bは、最も長い辺が三角形の右側にある三角形の第２のタイプを示す。
【図７】三角形を充填する充填スキャン・コンバータによって用いられる画素および関連するパラメータの詳細を示す。
【図８】平面方程式の連続的な生成に関して図４の回路の操作を示すフローチャートである。
【図９】図9aは、２つの三角形に分割することができる四辺形を示す。
図9bは、２つの三角形に分割することができる凹形の四辺形を示す。
【図１０】図10aは、図9aに示される四辺形の辺および頂点の命名規則を示す。図10bは、図9bに示される四辺形の辺および頂点の命名規則を示す。
【図１１】四辺形を作る三角形の平面方程式を生成する別の方法に関して、図４に示される回路の操作を示すフローチャートである。
【図１２】図11のステップに関連するフェッチおよび計算の操作を示す図表であり、一部のフェッチおよび計算の操作の削除を示す。
【図１３】頂点をソートする前の三角形のいくつかの例を示す。
【図１４】図13に示す三角形で行われる頂点のソーティングを示す。
【図１５】図14に従って頂点のソーティングを実行する回路の一例を示す。
【図１６】頂点のソーティングが平面方程式の生成と同時に実行されるプロセスを示す。
【図１７】図４のレジスタ・ファイル内の頂点データのメモリ・マップである。
【図１８】いくつかの多角形およびその多角形を充填するためにアクティブにされた画素を示す。
【符号の説明】
１０　　フロント・エンド基板

１２　テクスチャ・マッピング基板
１４　フレーム・バッファ基板
１５　ホスト・コンピュータ
３２　３Ｄ加速器
３４　２Ｄ加速器
８０,８１　プリミティブRAMバッファ
８４　クリッピング/ライティングRAM
１８０　レジスタ・ファイル
１８２　マルチプレクサ/データ・フォーマッタ
１８４　加算および乗算のALU
１８６　乗算器
１８８　除算器

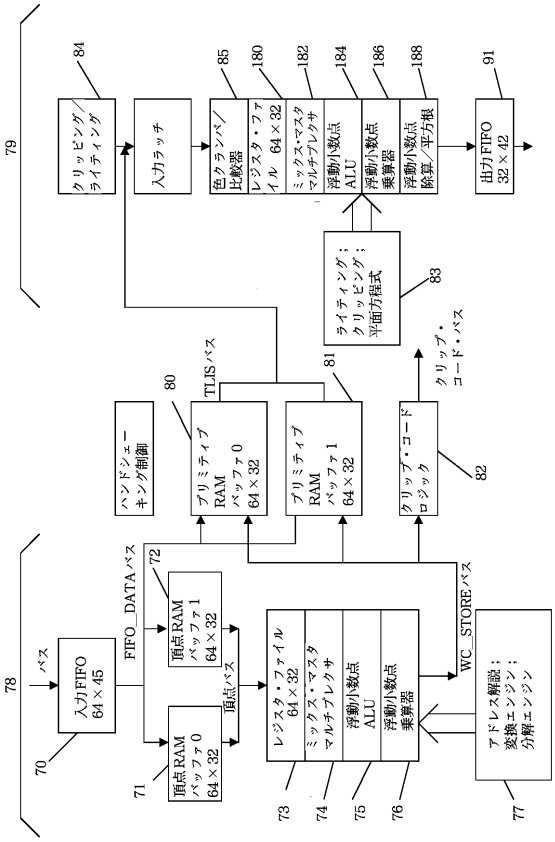
【図１】



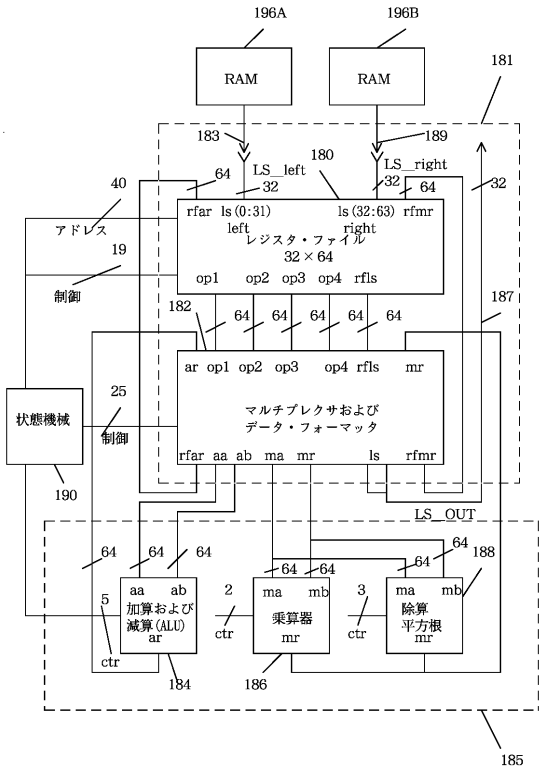
【図２】



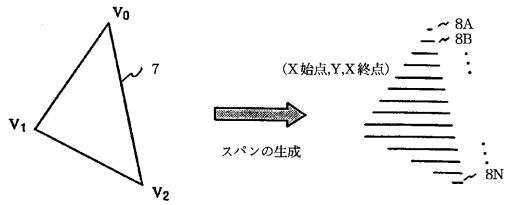
【図３】



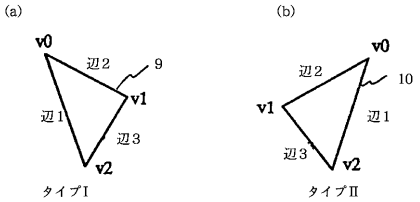
【図４】



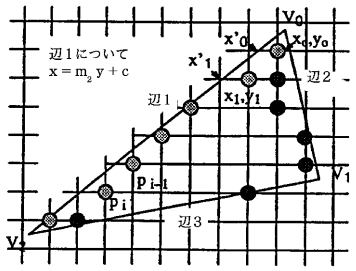
【図５】



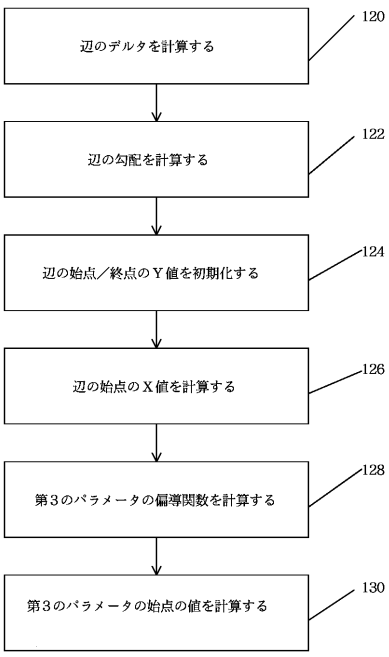
【図６】



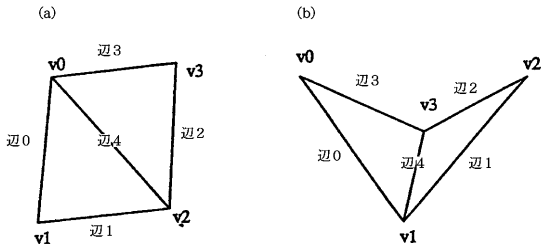
【図７】



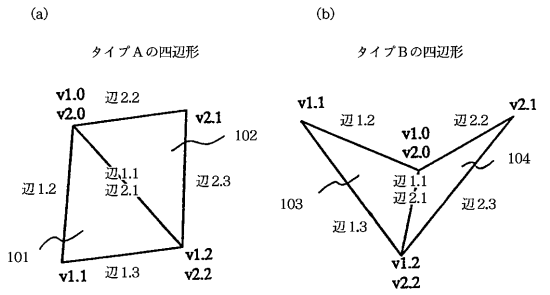
【図８】



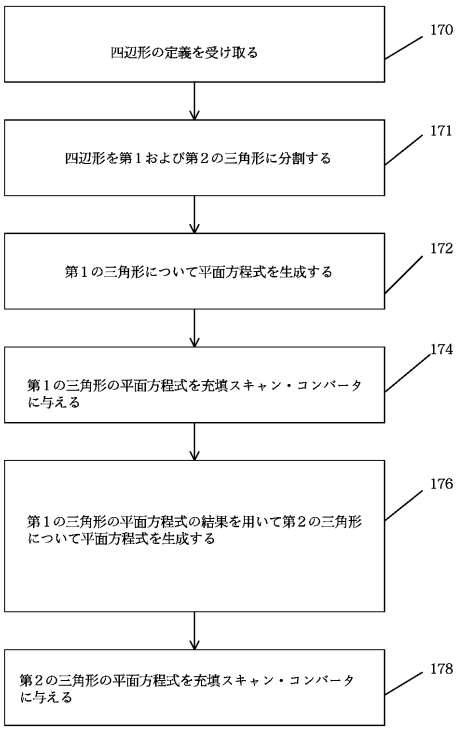
【図９】



【図１０】



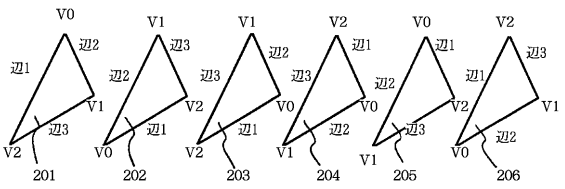
【図１１】



【図１２】

第１の三角形		第２の三角形	
フェッチ	計算	フェッチ	計算
ny1	v0.red − v2.red	dxe3	1.0/(temp−temp2), v0.z−v2.z
mnx2	v1.red − v0.red	v1.y	nz1 * mny2, temp3−temp4
mny2	ny1 * mnred2, v0.green−v2.green		temp3=nred1*mny2,
nx1	ny1 * mnx2, v1.green−v0.green		temp4=ny1*mnred2
v0.z	ny1 * mngreen2, v0.blue−v2.blue		
v1.z	nx1 * mny2, v1.blue−v0.blue		
v2.z	nred1 * mny2		
v0.y	ngreen1 * mny2, temp−temp2		
	temp=nx1*mny2, temp2=ny1*mnx2		
dye3	ny1 * mnblue2, v1.z − v0.z		
dxe3	1.0/(temp−temp2), v0.z−v2.z		
v1.y	nz1 * mny2, temp3−temp4		
	temp3=nred1*mny2,		
	temp4=ny1*mnred2		
平面方程式の残り		平面方程式の残り	
○		○	
○		○	
○		○	

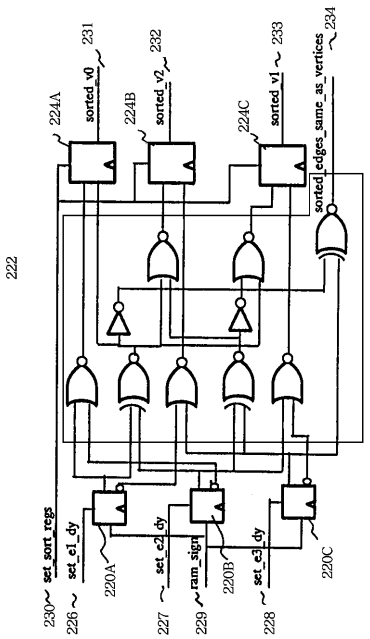
【図１３】



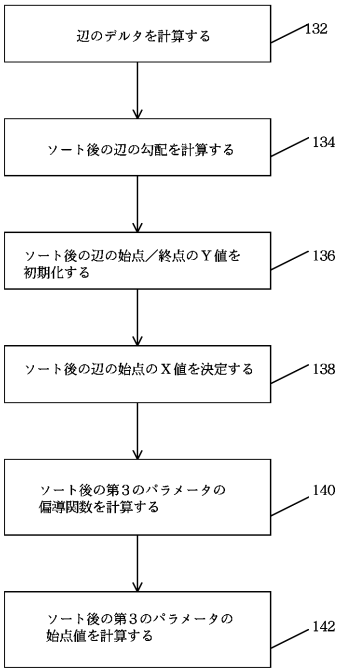
【図１４】

ケース	三角形（図13）	辺3のDYの符号	辺2のDYの符号	辺1のDYの符号	ソート後の頂点の順序	ソート後の辺の順序	注釈
(a)	201	0	0	1	0,1,2	1,2,3	無効ソート
(b)	202	0	1	0	2,0,1	3,1,2	時計回り
(c)	203	0	1	1	1,0,2	3,2,1	0,1交換
(d)	204	1	0	0	1,2,0	2,3,1	反時計回り
(e)	205	1	0	1	0,2,1	2,1,3	1,2交換
(f)	206	1	1	0	2,1,0	1,3,2	0,2交換
(g)	—	0	0	0	XXX	XXX	*
(h)	—	1	1	1	XXX	XXX	*

【図１５】



【図１６】

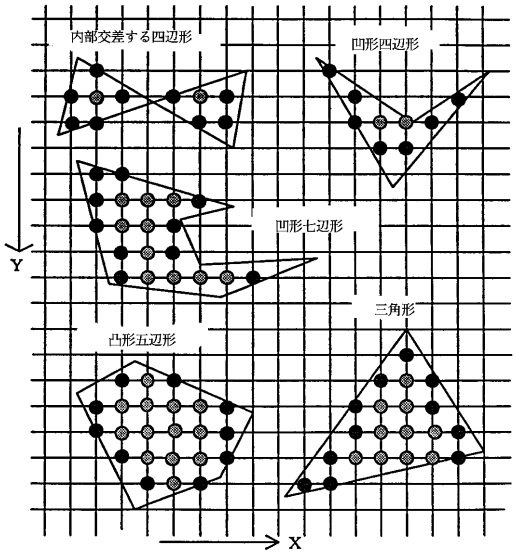


【図１７】

250

ラベル	アドレス (252)	アドレス（二進） (254)	内容 (256)
頂点 V0	8	01 000	X
	9	01 001	Y
	10	01 010	Z
	11	01 011	R
	12	01 100	G
	13	01 101	B
	14	01 110	edge1 − dx
	15	01 111	edge1 − dy
頂点 V1	16	10 000	X
	17	10 001	Y
	18	10 010	Z
	19	10 011	R
	20	10 100	G
	21	10 101	B
	22	10 110	edge2 − dx
	23	10 111	edge2 − dy
頂点 V2	24	11 000	X
	25	11 001	Y
	26	11 010	Z
	27	11 011	R
	28	11 100	G
	29	11 101	B
	30	11 110	edge3 − dx
	31	11 111	edge3 − dy

【図１８】



フロントページの続き

(56)参考文献　特開平０７－１０５４０５（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06T 11/00 - 11/40
G06T 15/00 - 15/60