

(19)日本国特許庁 (J P)

(12) 特 許 公 報 (B 2)

(11)特許番号

第2612260号

(45)発行日 平成9年(1997)5月21日

(24)登録日 平成9年(1997)2月27日

(51)Int.Cl. <sup>6</sup>	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 T 15/00		9365-5H	G 0 6 F 15/72	4 5 0 A

発明の数1(全 8 頁)

(21)出願番号	特願昭61-225140	(73)特許権者	999999999 ダイキン工業株式会社 大阪府大阪市北区中崎西2丁目4番12号 梅田センタービル
(22)出願日	昭和61年(1986)9月24日	(72)発明者	上田 智章 草津市岡本町字大谷1000番地の2 ダイ キン工業株式会社滋賀製作所内
(65)公開番号	特開昭63-80375		
(43)公開日	昭和63年(1988)4月11日		
審判番号	平7-9359	合議体	
		審判長	木屋野 忠
		審判官	菅野 嘉昭
		審判官	内藤 二郎
		(56)参考文献	A. S. Glassner 著「図説 コンピュータ・グラフィックス」P. 100-105 (1985. 9. 5:アスキー)

(54)【発明の名称】 テクスチャマッピング装置

1

(57)【特許請求の範囲】

【請求項1】 テクスチャ平面の所望の領域の画素情報を、ディスプレイ平面の所望の領域の図形データ上を投影するテクスチャマッピング装置において、ディスプレイ平面上の所望の領域を構成するディスプレイ側単位多角形の、最も短い辺に隣合う2辺を互に対向する2辺として検出する第1の辺検出手段と、テクスチャ平面の所望の領域を構成するテクスチャ側単位多角形領域の、上記2辺に対応する2辺を検出する第2の辺検出手段と、ディスプレイ側単位多角形の上記互に対向する2辺を直線補間する第1の辺補間手段と、テクスチャ側単位多角形領域の上記2辺を直線補間する第2の辺補間手段と、第1の辺補間手段から出力される補間データを入力とする第1の線分補間手段と、第2の辺補間手段から出力される補間データを入力とし、かつ上記第1の線分補

2

間手段と同期して補間演算を行なう第2の線分補間手段と、第2の線分補間手段から出力されるアドレスデータを入力として対応するアドレスのデータが読出されるマッピングメモリとを有することを特徴とするテクスチャマッピング装置。

【請求項2】 第1の辺補間手段が、互に対向する2辺のうち、長い方の辺の2本の座標軸方向の長さの和により定まる分割数に基いて両辺の直線補間を行なうものである上記特許請求の範囲第1項記載のテクスチャマッピング装置。

【請求項3】 第2の辺補間手段が、第1の辺補間手段における分割数に基いて両辺の直線補間を行なうものである上記特許請求の範囲第1項記載のテクスチャマッピング装置。

【請求項4】 第2の線分補間手段が、第1の線分補間手

10

段により描画される線分の画素数により定まる分割数に基いて線分の補間を行なうものである上記特許請求の範囲第 1 項記載のテクスチャマッピング装置。

【請求項 5】第 2 の線分補間手段が、分割数により定まる所定数の画素の平滑化を施された図形データに基いて線分の補間を行なうものである上記特許請求の範囲第 4 項記載のテクスチャマッピング装置。

【発明の詳細な説明】

<産業上の利用分野>

この発明はテクスチャ平面の所望の領域の画素情報を、ディスプレイ平面の所望の領域の図形データ上に投影するテクスチャマッピング装置に関する。

<従来の技術>

従来から、予め画像入力装置により取込まれた図形データ、或は図形描画装置により描画された図形データ等の所望の領域を、所望の立体図形の表面に投影することにより、デザイン設計、映像効果の確認等を行なう要求が強く、このような要求を満足させるために、所望の 2 次元図形を所望の 3 次元図形上に投影して表示するテクスチャマッピング装置が提供されている。

従来から提供されているテクスチャマッピング装置としては、2 次元のテクスチャ原図をスキャンライン方向の線分に分解し、ディスプレイ面においてスキャンライン方向に走査しながら逆透視変換を各画素単位で行なうようにしたもの（「テクスチャマッピングについて〔1〕」柴本猛 小林誠 講演論文集〔III〕（社）情報処理学会 昭和 60 年 9 月 9 日発行）が提供されていた。

<発明が解決しようとする問題点>

上記の構成のものにおいては、1 画素単位にマトリクス演算を行なう必要があるため、処理速度が遅くなるという問題がある。また、テクスチャ原図を構成する単位多角形と、ディスプレイ面における図形を構成する単位多角形との組合せによっては、正確なマッピングを行なうことができず、ディスプレイ面において表示される図形データの品質が低下してしまうことになるという問題もある。

さらに、ディスプレイ面上の図形を構成する多角形がバタフライ面になっている場合がある。したがって、多角形の 3 頂点と他の頂点とが同一平面上に存在しているか否かを予め判別することにより、多角形がバタフライ面であるか否かを識別する必要があり、処理速度が一層遅くなってしまいう問題があるのみならず、正確なテクスチャマッピングを行なうことができなくなってしまうという問題がある。

具体的に説明すると、互に対向する辺の頂点がスプライン曲線上にある場合には、四角形状の単位多角形に分解することにより、必然的にバタフライ面が生成されるのである。そして、このバタフライ面を 2 つの三角形に分解し、各三角形をそれぞれぬりつぶすことにより、テ

クスチャマッピングを行なえば、バタフライ面の性質上、ぬりつぶしてはいけない部分をぬりつぶしてしまうことになってしまうのである。

<発明の目的>

この発明は上記の問題点に鑑みてなされたものであり、余分なメモリを必要とせず、かつ多角形がバタフライ面であるか否かに拘わらず、正確に、かつ高速にテクスチャ原図のマッピングを行なうことができるテクスチャマッピング装置を提供することを目的としている。

<問題点を解決するための手段>

上記の目的を達成するための、この発明のテクスチャマッピング装置は、第 1、および第 2 の辺検出手段と、第 1、および第 2 の辺補間手段と、第 1、および第 2 の線分補間手段と、マッピングメモリとを有するものである。

上記第 1 の辺検出手段は、ディスプレイ平面上の所望の領域を構成するディスプレイ側単位多角形領域の、最も短い辺に隣合う 2 辺を互に対向する 2 辺としてを検出するものであり、上記第 2 の辺検出手段は、テクスチャ平面の所望の領域を構成するテクスチャ側単位多角形領域の、上記 2 辺に対応する 2 辺を検出するものである。

上記第 1 の辺補間手段は、ディスプレイ側単位多角形領域の互に対向する 2 辺を直線補間するものであり、上記第 2 の辺補間手段は、テクスチャ側単位多角形領域の 2 辺を直線補間するものである。

上記第 1 の線分補間手段は、第 1 の辺補間手段から出力される補間データを入力とするものであり、上記第 2 の線分補間手段は、第 2 の辺補間手段から出力される補間データを入力とするものであり、かつ第 1 の線分の線分補間手段と同期して補間演算を行なうものである。

上記マッピングメモリは、第 2 の線分補間手段から出力されるアドレスデータを入力として対応するアドレスのデータが読出されるものである。

但し、上記第 2 の辺検出手段としては、上記 2 辺に対応するテクスチャ平面上の 2 辺を検出するものであることが好ましい。

また、上記第 1 の辺補間手段としては、互に対向する 2 辺のうち、長い方の辺の 2 本の座標軸方向の長さの和により定まる分割数に基いて両辺の直線補間を行なうものであることが好ましく、しかも、上記第 2 の辺補間手段としては、第 1 の辺補間手段における分割数に基いて両辺の直線補間を行なうものであることが好ましい。

さらに、上記第 2 の線分補間手段としては、第 1 の線分補間手段により補間される線分の画素数により定まる分割数に基いて線分の補間を行なうものであることが好ましく、さらには、分割数により定まる所定数の画素の平滑化を施された図形データに基いて補間を行なうものであることが好ましい。

<作用>

以上の構成のテクスチャマッピング装置であれば、第

1の辺検出手段によりディスプレイ平面上の所望の領域を構成するディスプレイ側単位多角形領域の互に対向する2辺を検出するとともに、第2の辺検出手段によりテクスチャ平面の所望の領域を構成するテクスチャ側単位多角形領域の、上記2辺に対応する2辺を検出することができる。

そして、上記1対ずつの辺を、それぞれ第1の辺補間手段、第2の辺補間手段により直線補間する。

その後、第1の辺補間手段から出力される補間データを入力として第1の線分補間手段により画素データを発生させ、第2の辺補間手段から出力される補間データを入力として第2の線分補間手段によりアドレスデータを発生させる。

そして、第2の線分補間手段により出力させられたアドレスデータをマッピングメモリに供給することにより、対応するアドレスのデータを読み出すことができる。

即ち、マッピングメモリから読み出されたデータをディスプレイの対応画素データとして表示することにより、テクスチャマッピングを行なうことができる。

そして、上記第1の辺検出手段が、最も短い辺に隣合う2辺を互に対向する2辺として検出するものであるから、描画する画素数を少なくすることができる。この結果、描画する画素数が少なくなること起因してテクスチャマッピングの所要時間を短縮することができる(テクスチャマッピングの速度を早くすることができる)。

また、上記第1の辺補間手段が、互に対向する2辺のうち、長い方の辺の2本の座標軸方向の長さの和により定まる分割数に基いて両辺の直線補間を行なうものである場合には、ディスプレイ平面上の描画要素を構成する全ての画素についてテクスチャマッピングを行なうことができる。そして、上記第2の辺補間手段が、第1の辺補間手段における分割数に基いて両辺の直線補間を行なうものである場合には、テクスチャ平面とディスプレイ平面との間における対応関係を確保することができる。

さらに、上記第2の線分補間手段が、第1の線分補間手段により描画される線分の画素数により定まる分割数に基いて線分の補間を行なうものである場合には、ディスプレイ平面上において描画される線分を構成する画素の脱落がなくなる。

さらには、分割数により定まる所定数の画素の平滑化を施された図形データに基いて線分の補間を行なうものである場合には、テクスチャ平面における図形が大きく、ディスプレイ平面における図形投影領域が小さくても、テクスチャ平面における図形に対する忠実度を高く維持した状態でテクスチャマッピングを行なうことができる。

さらに詳細に説明する。

この発明においては、テクスチャマッピングの処理対象を、三角形、凸四角形、および4本の線分により囲まれた非平面に限定されることを前提としている。尚、こ

のような限定を施すことができるのは、コンピュータグラフィックで用いられる全ての図形が、多角形の集合体と、4本の線分によって囲まれた非平面の集合体として表現されるのであり、しかも、上記多角形は、三角形と凸四角形に分解することができるためである。

そして、テクスチャ平面をu-v平面、ディスプレイ平面をx-y平面と仮定する。

さらに、ディスプレイ平面におけるマッピング領域を規定する2本の辺a,bの端点の座標を(x<sub>as</sub>,y<sub>as</sub>)(x<sub>ae</sub>,y<sub>ae</sub>)(x<sub>bs</sub>,y<sub>bs</sub>)(x<sub>be</sub>,y<sub>be</sub>)、テクスチャ平面における上記2辺に対応する2辺a',b'の端点の座標を(u<sub>as</sub>,v<sub>as</sub>)(u<sub>ae</sub>,v<sub>ae</sub>)(u<sub>bs</sub>,v<sub>bs</sub>)(u<sub>be</sub>,v<sub>be</sub>)で表現する。

そして、上記辺a,bで囲まれる領域を(L+1)本の線分に分解するために、上記辺a,b,a',b'を(L+1)等分して直線補間演算を行なう。尚、この直線補間演算は必要に応じてz,l値についても行なう。

したがって、上記の直線補間演算を行なうことにより得られる線分jの端点の各座標値は、

$$\begin{aligned} x_{aj} &= x_{as} + j(x_{ae} - x_{as}) / (L + 1) \\ y_{aj} &= y_{as} + j(y_{ae} - y_{as}) / (L + 1) \\ u_{aj} &= u_{as} + j(u_{ae} - u_{as}) / (L + 1) \\ v_{aj} &= v_{as} + j(v_{ae} - v_{as}) / (L + 1) \\ x_{bj} &= x_{bs} + j(x_{be} - x_{bs}) / (L + 1) \\ y_{bj} &= y_{bs} + j(y_{be} - y_{bs}) / (L + 1) \\ u_{bj} &= u_{bs} + j(u_{be} - u_{bs}) / (L + 1) \\ v_{bj} &= v_{bs} + j(v_{be} - v_{bs}) / (L + 1) \end{aligned}$$

{但し、0 ≤ j ≤ L+1であり、好ましくは、L = x + y (xはディスプレイ平面上におけるx座標軸方向の長さであり、yはy軸方向の長さである)}  
で表わされる。

次いで、上記のようにして得られたディスプレイ平面の線分とテクスチャ平面上における線分とをM等分して直線補間演算を行なう。

したがって、上記の直線補間演算を行なうことにより得られる各点の座標値は、

$$\begin{aligned} x_{jk} &= x_{aj} + k(x_{bj} - x_{aj}) / M \\ y_{jk} &= y_{aj} + k(y_{bj} - y_{aj}) / M \\ u_{jk} &= u_{aj} + k(u_{bj} - u_{aj}) / M \\ v_{jk} &= v_{aj} + k(v_{bj} - v_{aj}) / M \end{aligned}$$

(但し、0 ≤ k ≤ Mであり、好ましくは、Mディスプレイ平面における描画される線分の画素数である)  
で表される。

そして、上記(u<sub>jk</sub>,v<sub>jk</sub>)に対応するマッピングメモリのデータを(x<sub>jk</sub>,y<sub>jk</sub>)のアドレスに表示させることにより、ディスプレイ平面上にテクスチャマッピングが施された図形を表示することができる。また、以上の場合において、ディスプレイ平面におけるマッピング領域を規定する2本の辺a,bとして、最も短い辺に隣合う2本の辺が選択されているのであるから、上述のよう

に得られた各点のうち、例えば整数化されることにより互いに重複する点の数を少なくすることができ、この結果、最も短い辺を選択した場合と比較して、テクスチャマッピングの所要時間を短縮することができる。

#### <実施例>

以下、実施例を示す添付図面によって詳細に説明する。

第 1 図はこの発明のテクスチャマッピング装置の一実施例を示すブロック図であり、2 辺の  $x, y, z$  値（ディスプレイ平面における 3 次元座標データ）、 $l$ （輝度データ）、 $u, v$  値（テクスチャ平面における 2 次元座標データ）に対応する辺補間回路（11）（12）...（16）（21）（22）...（26）と、上記辺補間回路（11）（12）（21）（22）から出力される  $x, y$  値を入力する線分補間回路（31）と、上記辺補間回路から出力される  $z, l, u, v$  値とそれぞれ入力とする線分補間回路（32）（33）（34）（35）と、上記線分補間回路（34）（35）から出力される  $u, v$  値を格納するマッピングメモリ（41）と、マッピングメモリ（41）から読出される  $R, G, B$  データ、（色コードデータ）および上記線分補間回路（33）から出力される  $I$  データを入力としてシェーディング補正等を行なう乗算回路（51）と、描画コマンドデータを取込むための I/O インターフェース（61）と、辺選択処理等を行なうプロセッサ（62）と、メモリ（63）とから構成されている。尚、上記各辺補間回路、線分補間回路は、それぞれ除算回路と、除算結果を累積加算する加算回路とから構成され、各補間処理を並行させて遂行することができるようにしている。

また、上記プロセッサ（62）は、図示しない上位プロセッサから伝送された頂点データに基いて、辺補間動作を行なうべき 2 辺を選択するものであり、上記線分補間回路（31）（32）から出力される  $x, y, z$  値データ、およびマッピングメモリ（41）から読出される色コードデータ（ $R, G, B$  データ）を図示しないフレームメモリに供給するようにしている。

以上の構成のテクスチャマッピング装置の動作は次のとおりである。

まず、プロセッサ（62）において、伝送されてきた複数個の頂点データに基いて、辺補間を行なうべき 2 辺の始点、終点に対応する頂点データを選択し、各辺毎に 1 対ずつの頂点データをそれぞれ辺補間回路（11）（12）...（14）、辺補間回路（21）（22）...（24）に供給するとともに、テクスチャ平面における図形の、上記頂点データに対応する頂点データを選択し、各辺毎に 1 対ずつの頂点データをそれぞれ辺補間回路（15）（16）、辺補間回路（25）（26）に供給する。また、1 対の頂点データにより定まる辺の長さに基づいて辺補間を行なうべき分割数データを算出して（例えば、2 頂点間の  $x$  方向ピクセル数と  $y$  方向ピクセル数とを加算した値に 1 を加算した数として算出し）、上記辺補間回路（11）（12）...

（16）、辺補間回路（21）（22）...（26）に供給する。

次いで、上記両データが供給された各辺補間回路においては、上記辺の長さ（両頂点に対応する各値の差）を分割数データにより除算し、一方の頂点データに対して上記除算値を順次累積的に加算することにより、辺補間データを得、対応する線分補間回路に供給する。

そして、上記線分補間回路（31）においては、一対の辺補間データに基いて、線分の長さを算出するとともに、線分の長さに基づいて線分補間を行なうべき分割数データを算出し（例えば、2 点間の  $x$  方向ピクセル数と  $y$  方向ピクセル数とを大小比較し、大なる方のピクセル数を選択し）、上記線分の各座標軸方向の始終点の差を分割数データにより除算し、一方の端点データに対して上記除算値を順次累積的に加算することにより、線分補間データを得、図示しないフレームメモリに供給する。

また、残余の線分補間回路においては、端点に対応する値の差を、上記線分補間回路（31）において算出された分割数データにより除算し、一方の端点データに対して上記除算値を順次累積的に加算することにより、線分補間データを得る。そして、線分補間回路（32）からの線分補間データを  $z$  値としてフレームメモリに供給する。また、線分補間回路（34）（35）からの線分補間データをマッピングメモリ（41）に供給し、マッピングメモリ（41）から読出された色コードデータ（ $R, G, B$  データ）、および線分補間回路（33）からの線分補間データ（ $I$  値）を乗算回路（51）に供給することにより、シェーディング処理を施し、シェーディング処理が施された状態で色コードデータ（ $R, G, B$  データ）としてフレームメモリに供給する。

即ち、線分補間回路（31）（32）により得られたディスプレイ平面上の各画素座標に、マッピングメモリ（41）の図形データを投影することにより、テクスチャマッピング処理が施された図形に対応するデータをフレームメモリに格納することができ、フレームメモリの内容に基いてディスプレイ上における表示を行なわせることにより、テクスチャマッピング処理が施された図形を可視的に表示することができる。

即ち、2 辺の直線補間を行なうとともに、得られた線分を直線補間することによりテクスチャ原図とディスプレイ平面における図形との対応をとっているため、ディスプレイ面上におけるマッピング領域がバタフライ面であっても、テクスチャ原図を確実に投影することができる。

また、上記の説明から明らかなように、テクスチャ面上においてオーバーラップさせて描画を行なう部分が発生する可能性がかなり高いのであるが、直線補間動作は 1 回除算を行なった後は単純に累積加算を行なわせるのみであるから、補間回路をハードウェアにより構成することにより、マトリクス変換を行なう従来例と比較して、全体としての処理時間を短縮することができる。

第2図は上記の実施例において描画ドット数を減少させることにより全体としての処理時間を短縮する動作を説明する図であり、ディスプレイ平面における4頂点のx,y座標データが与えられた場合に(3頂点の場合にも同様に適用することができるが、詳細は省略する)、最も短い辺H1を得(例えば、対応する2頂点同士のx座標値の差とy座標値の差とを加算した値が最も小さい辺を最も短い辺とし)、上記辺に隣合う辺H2,H3を2辺として選択する。また、テクスチャ平面においては、上記2辺に対応する2辺を選択する。

このように2辺を選択することにより、描画する画素数を少なくすることができる。さらに詳細に説明すれば、例えば、第3図Aに示すように、四角形のデータ構造を有していながら直角二等辺三角形に近似される形状(最も長い辺が2Lドット、上記辺、および上記辺から最も離れた頂点までがLドット)を有している場合を考えれば、最も短い辺、および対向とする辺とを選択した場合には、第3図Bに示すように、上記図形が2L本の線分に分解される。逆に他の2辺を選択した場合にも、第3図Cに示すように、2L本の線分に分解される。しかし、第3図Bの場合には、2L本の線分の長さがL~2Lドットの範囲であり、描画ドット数が $3L^2$ となるのに対して、第3図Cの場合には、全ての線分の長さがLドットであり、描画ドット数が $L^2$ となるので、描画ドット数を $2L^2$ だけ減少させることができる。

上記のようにして選択された2辺に基いて、直線補間を行なうための分割数を決定する。

この動作は次のとおりである。

第4図Aに示すように2辺が選択された場合において、各辺の端点間に存在するx方向のドット数  $x_1$ ,  $x_2$ , y方向のドット数  $y_1$ ,  $y_2$ を算出して、各辺のドット数( $|x_1| + |y_1| + 1$ と $|x_2| + |y_2| + 1$ )を比較し、多い方のドット数を直線補間のための分割数として選択する。

上記のように分割数を選択すれば、第4図Bに示すように、長い方の辺については、辺が通過する全ての画素が分解される線分の端点になり、逆に短い方の辺については、第4図Cに示すように、辺が通過する全ての画素が、分解される1本以上の線分の端点になる。したがって、分解されて生ずる全ての線分は互に1画素以上離れることのない状態となり、マッピング欠落画素の発生を確実に防止することができる。

また、テクスチャ平面における2辺の直線補間を行なう場合の分割数として上記分割数をそのまま使用するので、テクスチャ原図のサイズによっては分解されて生ずる全ての線分のうち少なくとも一部が、互に2画素以上離れる状態になる可能性があるが、最終的には可視的表示が行なわれるのはディスプレイ平面上の図形であるから、特に不都合はない。

上記のようにして対向する2辺の直線補間が行なわれ

た後は、直線補間データに基いて定まる線分に対する直線補間を行なう。

この場合には、ディスプレイ平面における線分の長さに基づいて分割数を算出し、この分割数に基づいてディスプレイ平面上の線分、およびテクスチャ平面上の線分の直線補間を行なう。

したがって、ディスプレイ平面上における直線補間により得られる画素データは、ドットの欠落が全くない状態になる。他方、テクスチャ平面上における直線補間により得られる画素データは、線分の長さによってはドットの欠落が発生した状態になることが考えられるのであるが、可視的に表示されるのはディスプレイ平面における直線補間により得られる画素のみであるから、上記ドットの欠落が発生しても特に不都合はない。

但し、直線のような図形については、点線状に表示されるという不都合が生ずるので、このような不都合をも解消する場合には、第5図に示すように、直線補間により得られた画素を中心とする所定範囲の画素の平滑化処理を行なった後、ディスプレイ平面上に投影すればよい。

第6図は他の実施例を示すブロック図であり、第1図に示す実施例と異なる点は、辺補間回路(11)(12)...(16)(21)(22)...(26)を省略して、2辺の直線補間動作をプロセッサ(62)により行なわせるようにした点のみである。

したがって、この実施例の場合には、2辺の直線補間動作を辺補間回路により行なう場合と比較して、所要時間が長くなると思われる。

しかし、テクスチャマッピングにおいては、マッピングメモリからのデータの読出し、またはフレームメモリへのデータの書き込みの時間によりマッピング処理速度が制限されるので、全体としてみれば、所要時間は余り長くなり、上記第1図の実施例とほぼ同程度のテクスチャマッピング処理速度を達成することができる。

さらに詳細に説明すると、通常のぬりつぶし処理をスキャンライン方向に行なう場合には、非常に高速で描画を行なうことができるのであるが、上記テクスチャマッピングにおいてはスキャンラインに対して傾斜した線分に沿ってぬりつぶしを行なうのであるから、描画速度が遅くなり(例えば、スキャンライン方向の描画速度の約1/8)、辺の直線補間を行なうのに必要なプロセッサ(62)の処理時間を、描画速度に余り影響を与えずに確保することができる。この結果、プロセッサ(62)による処理が多少遅くなっても、テクスチャマッピング処理全体としてみれば、同程度の処理速度を達成することができるのである。

尚、この発明は上記の実施例に限定されるものではなく、例えば輝度I、z値等をディスプレイ平面上における直線補間演算と同期させて行ない、乗算回路(51)により乗算処理を行なわせることにより、シェーディング

処理、隠面処理、セクショニング処理等をも併せて行なうことが可能である他、乗算回路(51)を省略してテクスチャマッピング処理のみを行なわせることが可能であり、その他この発明の要旨を変えない範囲内において、種々の設計変更を施すことが可能である。

<発明の効果>

以上のようにこの発明は、ディスプレイ平面における多角形の種類に拘わらず正確に、かつ描画する画素数の増加を大幅に抑制して高速にテクスチャ原図のマッピングを行なうことができるとともに、余分なメモリを必要とせず、構成を簡素化することができるという特有の効果奏する。

\*【図面の簡単な説明】

第1図はこの発明のテクスチャマッピング装置の一実施例を示すブロック図、

第2図から第5図はそれぞれテクスチャマッピング動作の一部を説明する図、

第6図は他の実施例を示すブロック図。

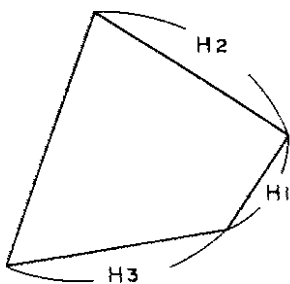
(11) (12) ... (16) (21) (22) ... (26) .....辺補間回路、

(31) (32) ... (35) .....線分補間回路、

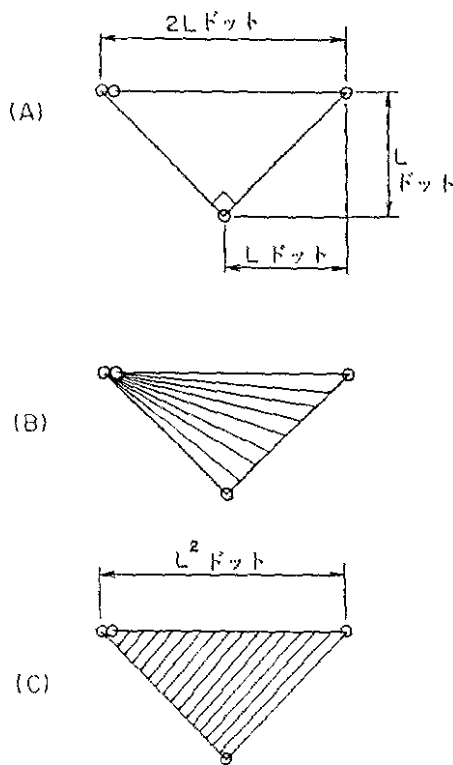
(41) .....マッピングメモリ、

(62) .....プロセッサ

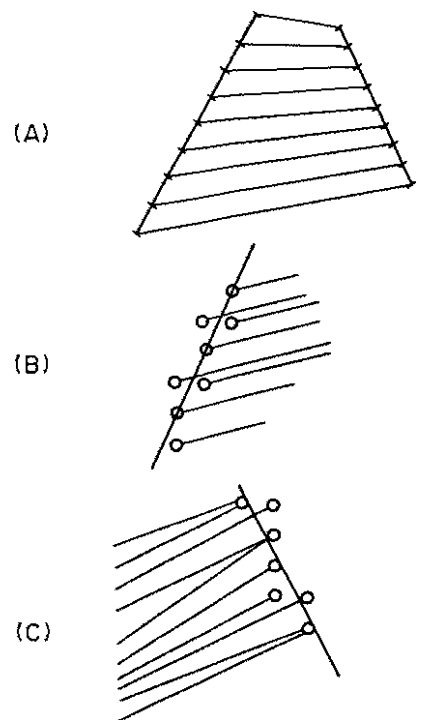
【第2図】



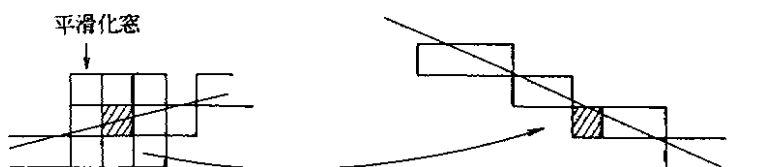
【第3図】



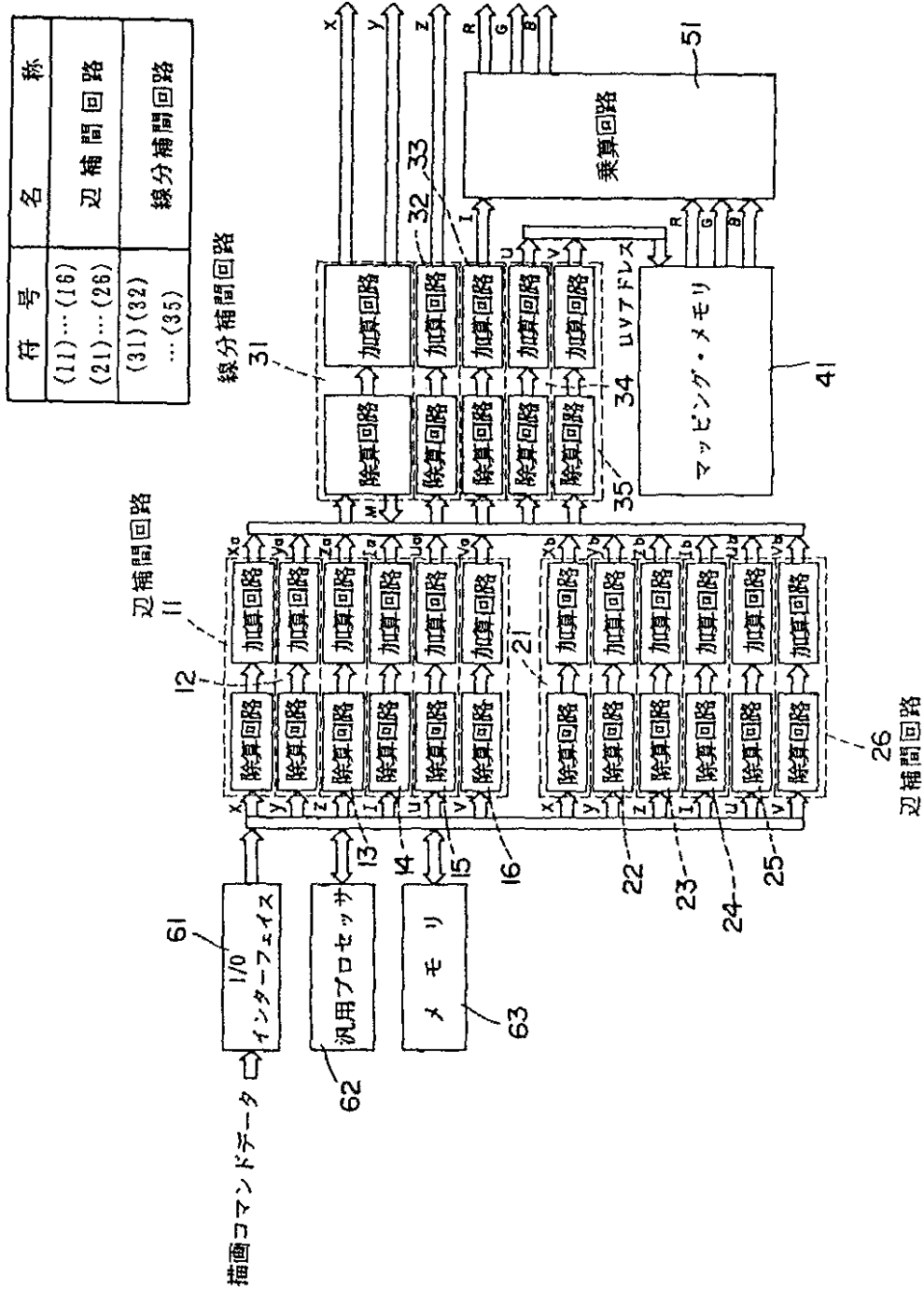
【第4図】



【第5図】



【第1図】



符号	名称
(11) ... (16)	辺補間回路
(21) ... (26)	線分補間回路
(31) (32)	線分補間回路
... (35)	線分補間回路

【第6図】

