

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第4887508号
(P4887508)

(45) 発行日 平成24年2月29日 (2012. 2. 29)

(24) 登録日 平成23年12月22日 (2011. 12. 22)

(51) Int. Cl.	F I
G09G 5/24 (2006.01)	G09G 5/24 620L
G06T 3/40 (2006.01)	G06T 3/40 P
G09G 5/28 (2006.01)	G09G 5/28 Z
G09G 5/36 (2006.01)	G09G 5/36 520C

請求項の数 22 (全 40 頁)

(21) 出願番号	特願2001-514385 (P2001-514385)	(73) 特許権者	502034291
(86) (22) 出願日	平成12年7月31日 (2000. 7. 31)		ビットストリーム インコーポレーティッド
(65) 公表番号	特表2003-506729 (P2003-506729A)		アメリカ合衆国, 02142, マサチューセッツ州 ケンブリッジ, ファースト ストリート 245, 17階
(43) 公表日	平成15年2月18日 (2003. 2. 18)	(74) 代理人	100094145
(86) 国際出願番号	PCT/US2000/021559		弁理士 小野 由己男
(87) 国際公開番号	W02001/009842	(74) 代理人	100094167
(87) 国際公開日	平成13年2月8日 (2001. 2. 8)		弁理士 宮川 良夫
審査請求日	平成19年7月27日 (2007. 7. 27)	(72) 発明者	カーシラ, サンボ, ジェイ.
(31) 優先権主張番号	09/363, 513		アメリカ合衆国, ニューハンプシャー州 03865, プレイストウ, スクウォラルラン ロード 6
(32) 優先日	平成11年7月29日 (1999. 7. 29)		
(33) 優先権主張国	米国 (US)		

最終頁に続く

(54) 【発明の名称】 人間による読み取りが可能な表示に使用するための2次元のピクセル画像において、ピクセル被カバ値を設定するためのコンピュータ化された方法

(57) 【特許請求の範囲】

【請求項 1】

人間による読み取りが可能な表示に使用するための2次元のピクセル画像において、ピクセル被カバ値を設定するためのコンピュータ化された方法であって、

前記ピクセル画像は、前記ピクセル画像の解像度よりも微細な解像度で定義される1又は複数の形状の高解像度の2次元表示を表わしており、

前記ピクセル画像は、複数のピクセルで形成されており、

各ピクセルは、前記高解像度の2次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、前記サンプリング領域が前記形状のうちの1つによってどの程度覆われているかを表わすピクセル被カバ値を有しており、

前記複数のピクセルの各々について、

前記ピクセルに対応するサンプリング領域内を異なる非平行な方向に延びている少なくとも2つのサンプリング線の各々について、前記サンプリング線が前記サンプリング領域内で前記形状のいずれかによってどの程度覆われているかの関数として、線被カバ値を決定するステップと、

前記2つのサンプリング線について決定した前記線被カバ値の非線形関数として、前記ピクセルのピクセル被カバ値を決定するステップと、
を含んでいるコンピュータ化された方法。

【請求項 2】

前記方法によって生成される前記2つのサンプリング線についての前記線被カバ値の

組み合わせの大部分において、

前記ピクセル被カバ値の変化の割合は、前記 2 つのサンプリング線のうちで前記線被カバ値が前記形状によってサンプリング線の 2 分の 1 が覆われた場合の値に近い方のサンプリング線の前記線被カバ値の変化に対して、前記 2 つのサンプリング線の方の前記線被カバ値の変化に対するよりも、大きい、請求項 1 に記載のコンピュータ化された方法。

【請求項 3】

前記方法によって生成される前記 2 つのサンプリング線についての前記線被カバ値の組み合わせの大部分において、

前記ピクセル被カバ値の変化の割合は、前記 2 つのサンプリング線のうちで前記線被カバ値が前記形状によってサンプリング線の 2 分の 1 が覆われた場合の値に近い方のサンプリング線の前記線被カバ値にのみ対応して変化する、請求項 2 に記載のコンピュータ化された方法。

10

【請求項 4】

前記方法によって生成される前記 2 つのサンプリング線についての前記線被カバ値の組み合わせの大部分において、

前記ピクセル被カバ値の変化の割合は、前記 2 つのサンプリング線の両方の線の前記線被カバ値の変化に対応して変化する、請求項 2 に記載のコンピュータ化された方法。

【請求項 5】

前記非線形関数は、参照テーブルにおいて前記少なくとも 2 つのサンプリング線の線被カバ値の関数としてアドレス指定される位置の値を参照することによって、前記ピクセル被カバ値を決定する、請求項 1 に記載のコンピュータ化された方法。

20

【請求項 6】

前記参照テーブルにおいて、前記 2 つのサンプリング線の線被カバ値の所与の組み合わせによってアドレス指定される位置の値が、前記ピクセルが線被カバ値の該当の組み合わせを有している状況において前もって計算した複数のピクセル被カバ値から導出されている、請求項 5 に記載のコンピュータ化された方法。

【請求項 7】

前記形状が、1 つ以上のフォント・セット内の文字の形状であり、

前記方法は、異なるフォント・セットからの文字を描画するときに、前記ピクセル被カバ値を決定するために異なる参照テーブルを使用することを含んでいる、請求項 5 に記載のコンピュータ化された方法。

30

【請求項 8】

前記非線形関数は、前記 2 つのサンプリング線の線被カバ値の関数としてピクセル被カバ値を決定し、

前記 2 つのサンプリング線の線被カバ値は、前記 2 つの線被カバ値の重み付け和を含んでおり、前記重み付け和における前記 2 つの線被カバ値のそれぞれの寄与は、前記線被カバ値の各々が中間の線被カバ値までどれだけ近いかの関数である、請求項 1 に記載のコンピュータ化された方法。

【請求項 9】

前記 2 つのサンプリング線が互いに直交している、請求項 1 に記載のコンピュータ化された方法。

40

【請求項 10】

前記ピクセル画像が、行および列に配置されたピクセルで構成され、

前記線被カバ値が、各ピクセルの 2 つのサンプリング線のみによって決定され、一方のサンプリング線が実質的に各ピクセル行の中央を延びており、一方のサンプリング線が実質的に各ピクセル列の中央を延びている、請求項 9 に記載のコンピュータ化された方法。

【請求項 11】

前記形状が、前記形状を前記ピクセル画像のピクセル解像度よりも高い解像度で定義し

50

ているアウトラインによって記述されており、

前記線被カバー値の決定が、線被カバー値を前記形状のアウトラインと前記サンプリング線との間の交点の間の距離の関数として決定する、請求項 1 に記載のコンピュータ化された方法。

【請求項 1 2】

人間による読み取りが可能な表示に使用するための 2 次元のピクセル画像を生成するためのコンピュータ化された方法であって、

前記ピクセル画像は、前記ピクセル画像の解像度よりも微細な解像度で定義される 1 又は複数の形状の高解像度の 2 次元表示を表わしており、

前記ピクセル画像は、複数のピクセルで形成されており、

各ピクセルは、高解像度の 2 次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、前記サンプリング領域が前記形状のうちの 1 つによってどの程度覆われているかを表わすピクセル被カバー値を有しており、

前記方法が、

前記形状と、前記高解像度の表示において第 1 の方向に延びている第 1 組の平行なサンプリング線との交点を計算するステップと、

前記形状と、前記高解像度の表示において前記第 1 の方向とは異なる前記第 1 の方向と平行でない第 2 の方向に延びている第 2 組の平行なサンプリング線との交点を計算するステップと、

前記ピクセルの前記サンプリング領域の前記第 1 組の 1 つ以上のサンプリング線の線被カバー値を、当該サンプリング線が前記サンプリング領域内で前記形状のいずれかによってどの程度覆われているのかの関数として決定し、次いで前記ピクセルの前記ピクセル被カバー値を、当該線被カバー値の関数として決定することによって、各ピクセルについて前記ピクセル被カバー値を計算することを含んでいる、第 1 のピクセル設定パスを実行するステップと、

前記第 1 のピクセル設定パスを実行した後に、前記形状と第 2 組のサンプリング線との間の 1 つ以上の前記交点が計算されているサンプリング領域を表わしているピクセルについてのみ、前記第 1 のパスにおいて設定された前記ピクセル被カバー値を変更する第 2 のピクセル設定パスを実行するステップと、

を含んでおり、

前記第 2 のピクセル設定パスが、前記交点が計算されているピクセルのピクセル被カバー値を、

前記ピクセルの前記サンプリング領域の前記第 2 組の 1 つ以上のサンプリング線の線被カバー値を、当該サンプリング線が前記サンプリング領域内で前記形状のいずれかによってどの程度覆われているのかの関数として決定するステップと、

次いで前記ピクセルの前記ピクセル被カバー値を、前記第 1 のピクセル設定パスにおいて前記ピクセルについて計算された前記線被カバー値および前記第 2 のピクセル設定パスにおいて前記ピクセルについて計算された前記線被カバー値の関数として決定するステップと、

によって変更する、コンピュータ化された方法。

【請求項 1 3】

前記ピクセル画像が、順次のアドレスにてメモリに保存された一連のピクセル行で構成され、各ピクセル行が、より密な間隔の順次のアドレスにてメモリに保存された一連の前記ピクセル被カバー値を含んでおり、

第 1 のピクセル設定パスが、前記ピクセル行の方向に延びているサンプリング線について実行される、

請求項 1 2 に記載のコンピュータ化された方法。

【請求項 1 4】

前記ピクセル画像が、前記ピクセルの 2 次元アレイで構成されており、前記第 1 および第 2 の方向が、該アレイの 2 つの次元に一致している、請求項 1 2 に記載のコンピュータ

10

20

30

40

50

化された方法。

【請求項 15】

前記第2のピクセル設定パスが、前記ピクセルについての前記ピクセル被カバー値を、前記第1および第2の方向に延びるサンプリング線の線被カバー値の非線形関数として決定させ、所与のサンプリング線の線被カバー値の変化の割合の関数として変化するピクセル被カバー値の割合が、前記線被カバー値の関数として変化する、請求項12に記載のコンピュータ化された方法。

【請求項 16】

人間による読み取りが可能な表示に使用するための2次元のピクセル画像を生成するためのコンピュータ化された方法であって、

前記ピクセル画像は、前記ピクセル画像の解像度よりも微細な解像度であるアウトラインによって定義される文字フォント形状の高解像度の2次元表示を表わしており、

前記ピクセル画像は、行および列に配置された複数のピクセルで形成されており、各ピクセルが、前記高解像度の2次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、前記サンプリング領域が前記形状のうちの1つによってどの程度覆われているかを表わすピクセル被カバー値を有しており、

前記方法は、

前記高解像度の2次元表示において、前記形状のアウトラインと前記ピクセル行に沿って延びる平行な行サンプリング線の組との間の交点、および前記形状のアウトラインと前記ピクセル列に沿って延びる平行な列サンプリング線の組との間の交点を計算するステップと、

第1のピクセル設定パスを実行するステップと、

第1のピクセル設定パスの実行後に、第2のピクセル設定パスを実行するステップと、を含んでおり、

前記計算は、

文字フォント形状の各アウトラインに沿って生じるそのような各交点を見つけるステップと、

所与のサンプリング線に関する各交点を、該サンプリング線に関する順序付けた交点リストの中に配置するステップであり、前記配置においては、該サンプリング線において生じる他の交点と比べて、前記交点の該サンプリング線の開始端までの距離を示す、順序付けられた位置にある、ステップと、

を含んでおり、

前記第1のピクセル設定パスは、

各ピクセル行において、

該当の行サンプリング線の開始端に最も近いピクセルから出発して、行の各ピクセルについて、

行サンプリング線の関連の交点リストにピクセルのサンプリング領域において生じる交点が存在していないならば、ピクセルをサンプリング線の現在の線の被カバー状態に対応するピクセル被カバー値に設定し、

そうでない場合に、

サンプリング線の現在の線の被カバー状態を、交点リストのピクセルのサンプリング領域の次の各交点を反映すべく変更し、

行の線の被カバー値を、ピクセルのサンプリング領域内の行サンプリング線の文字フォント形状によって覆われている部分の関数として計算し、

ピクセルを、ピクセルについて計算された行の線の被カバー値の関数として決定されるピクセル被カバー値に設定するステップ

を含んでおり、

前記第2のピクセル設定パスは、

自身の関連の交点リストに交点を有している各ピクセル列において、

該当の列サンプリング線の開始端に最も近いピクセルから出発し、該当の交点リストに

10

20

30

40

50

においてピクセルのサンプリング領域に交点が存在している列の各ピクセルについて、
列サンプリング線の現在の線の被カバー状態を、交点リストのピクセルのサンプリング領域の次の各交点を反映すべく変更し、

列の線の被カバー値を、ピクセルのサンプリング領域内の列サンプリング線の文字フォント形状によって覆われている部分の関数として計算し、

ピクセルのピクセル被カバー値を、第1のピクセル設定パスにおいてピクセルについて計算された行の線の被カバー値および第2のピクセル設定パスにおいてピクセルについて計算された列の線の被カバー値の関数として設定するステップを含んでいる
コンピュータ化された方法。

10

【請求項17】

前記第2のピクセル設定パスにおいて前記ピクセル被カバー値を前記ピクセルの前記行および前記列の前記線被カバー値の関数として設定するために使用される関数が、非線形関数であって、

前記行および前記列の前記線被カバー値の組み合わせの大部分において、前記ピクセル被カバー値の変化の割合は、前記ピクセルの前記行または前記列のサンプリング線のうちで線被カバー値が前記形状によって前記ピクセル内のサンプリング線の2分の1が覆われた場合の値に近い方のサンプリング線の線被カバー値の変化に対して、前記2つのサンプリング線の他方の線被カバー値の変化に対するよりも、大きい、請求項16に記載のコンピュータ化された方法。

20

【請求項18】

人間による読み取りが可能な表示に使用するための2次元のピクセル画像において、ピクセル被カバー値を設定するためのコンピュータ・プログラミングを含んでいるコンピュータで読み取ることができるメモリであって、

前記ピクセル画像は、前記ピクセル画像の解像度よりも微細な解像度で定義される1つ以上の形状の高解像度の2次元表示を表わしており、

前記ピクセル画像は、複数のピクセルで形成されており、

各ピクセルは、高解像度の2次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、前記サンプリング領域が前記形状のうちの1つによってどの程度覆われているかを表わすピクセル被カバー値を有しており、

30

前記プログラミングが、

前記ピクセルに対応するサンプリング領域内を異なる非平行な方向に延びている少なくとも2つのサンプリング線の各々について、前記サンプリング線が前記サンプリング領域内で前記形状のいずれかによってどの程度覆われているかの関数として、線被カバー値を決定するステップと、

前記2つのサンプリング線について決定した前記線被カバー値の非線形関数として、前記ピクセルの前記ピクセル被カバー値を決定するための命令を含んでいる、コンピュータで読み取ることができるメモリ。

【請求項19】

前記方法によって生成される前記2つのサンプリング線についての前記線被カバー値の組み合わせの大部分において、

40

前記ピクセル被カバー値の変化の割合は、前記2つのサンプリング線のうちで前記線被カバー値が前記形状によってサンプリング線の2分の1が覆われた場合の値に近い方のサンプリング線の被カバー値の変化に対して、前記2つのサンプリング線の他方の線被カバー値の変化に対するよりも、大きい、請求項18に記載のコンピュータで読み取ることができるメモリ。

【請求項20】

人間による読み取りが可能な表示に使用するための2次元のピクセル画像についてピクセル被カバー値を設定するためのコンピュータ・システムであって、

前記ピクセル画像は、前記ピクセル画像の解像度よりも微細な解像度で定義される1又

50

は複数の形状の高解像度の２次元表示を表わしており、

前記ピクセル画像は、複数のピクセルで形成されており、

各ピクセルは、前記高解像度の２次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、前記サンプリング領域が前記形状のうちの１つによってどの程度覆われているかを表わすピクセル被カバー値を有しており、

前記コンピュータ・システムは、

コンピュータ・メモリから読み出したプログラム命令を実行可能な１又は複数の処理装置と、

前記１又は複数の処理装置をプログラムするためのプログラム命令を保存する１又は複数のコンピュータが読み取り可能なメモリとを有しており、

前記プログラム命令が、

前記ピクセルに対応するサンプリング領域内を異なる非平行な方向に延びている少なくとも２つのサンプリング線の各々について、各サンプリング線が前記サンプリング領域内で前記形状のいずれかによってどの程度覆われているかの関数として、線被カバー値を決定するための演算ロジック、および

前記ピクセルの前記ピクセル被カバー値を、前記２つのサンプリング線について決定した前記線被カバー値の非線形関数として、決定するための演算ロジック

のための命令を含んでいるコンピュータ・システム。

【請求項 ２ １】

前記方法によって生成される前記２つのサンプリング線についての前記線被カバー値の組み合わせの大部分において、

前記ピクセル被カバー値の変化の割合は、前記２つのサンプリング線のうちで前記線被カバー値が前記形状によってサンプリング線の２分の１が覆われた場合の値に近い方のサンプリング線の前記線被カバー値の変化に対して、前記２つのサンプリング線の他方の線被カバー値の変化に対するよりも、大きい、請求項 ２ ０ に記載のコンピュータ・システム

【請求項 ２ ２】

表示に用いられる２次元のピクセル画像においてピクセル被カバー値を設定するためのコンピュータ化された方法であって、

前記ピクセル画像は、前記ピクセル画像の解像度よりも微細な解像度で定義される１又は複数の形状の高解像度の２次元表示を表わしており、

前記ピクセル画像は、複数のピクセルで形成されており、

各ピクセルは、前記高解像度の２次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、前記サンプリング領域が前記１又は複数の形状によってどの程度覆われているかを表わすピクセル被カバー値を有しており、

前記複数のピクセルの各々について、

前記ピクセルに対応するサンプリング領域内を非平行な方向に延びている少なくとも２つのサンプリング線の各々について、線被カバー値を決定するステップであり、前記線被カバー値は、前記サンプリング線が前記サンプリング領域内で前記１又は複数の形状によってどの程度覆われているかを示している、ステップと、

前記決定された線被カバー値の非線形関数を用いて、前記ピクセル被カバー値を決定するステップと、

を含んでいる方法。

【発明の詳細な説明】

【発明の属する分野】

【 ０ ０ ０ １】

本発明は、アンチエイリアシングを伴うアウトラインフォントおよび他のグラフィック要素のスキャン変換またはピクセル・レンダリングに関する。

【従来の技術】

【 ０ ０ ０ ２】

10

20

30

40

50

今日では、情報の伝達に使用される画像の多くはピクセル画像であり、すなわち、個々のピクセルまたは画素のアレイで構成された画像である。コンピュータの画面上の表示は、通常は、ピクセル画像を使用して生成されている。コンピュータ用の表示装置において、ピクセルとは、完全な色値を表示するために使用することができる表示画面の最小単位を表わしている。モノクロの表示装置においては、通常はピクセルが、表示装置が点灯または消灯を行うことができる最小の面積である。カラー表示装置においては、ピクセルは、表示装置が複合色の出力を生じさせることができる画面上の最小の面積であることが多い。通常は、複合色は、赤色、緑色、および青色のサブピクセルの組など、1つのフルカラー・ピクセルを定めるために組み合わせられるサブピクセルから形成される。

【0003】

10

多くの印刷画像も、やはりピクセル画像として生成される。ツートンの印刷画像においては、多くの場合、各ピクセルが、プリンタによって紙片上に形成できる最小のマークを表わす。グレースケールの印刷画像においては、多くの場合、ピクセルは、プリンタによってグレースケール画像の生成に使用される種々のグレースケール値の任意の1つを有するドット・パターンの全体が生成される最小の領域を表わす。カラーの印刷画像においては、フルカラーのピクセルが、シアン、マゼンタ、イエロー、およびブラックなどの基本色の組のそれぞれの個々のグレースケールのサブピクセルを組み合わせることによって生成される。

【0004】

20

多くのビットマップ画像は、表示しようとする各ピクセルに関連付けられた色値（ここで、色値は、白または黒、グレースケール、あるいは複合色値を表わすことができる）を有するビットマップ形式で保存された情報の単なる表示である。そのようなビットマップの表示に関して必要となる演算は、ビットマップからコンピュータ・メモリのスクリーン画像の表示に使用される部位へのピクセル値の移動と大差がない。スクリーン上への表示が所与のビットマップの解像度よりも高い解像度または低い解像度である場合、ビットマップを所望のサイズへと拡大縮小すること、および得られた画像のうちで部分的にのみカバーされているピクセルの色値を、拡大縮小後のビットマップによってカバーされている程度に比例して調節することが、必要とされるすべてである。

【0005】

30

他方で、コンピュータの画面または印刷出力に表示される多くのビットマップは、描画されるべき形状の記述から生み出され、そのような形状は、表示すべきビットマップ画像のピクセルのサイズよりもかなり高い解像度で記述されている。そのような場合、そのような高解像度の記述を、それらが記述している形状の適切なピクセル画像へと変換するために、或るアルゴリズムが必要になる。これは、とくには、いわゆるスケーラブル・フォントにおいて一般的である。スケーラブル・フォントは、文字のフォントであって、その形状が、直線および曲線に関して高い解像度で精密に記述されている。このため、フォントの所与の文字の形状を、単にその文字の精密な記述の所与のピクセル画像への投影を拡大または縮小することで、幅広くさまざまなサイズの範囲において表示することが可能である。

【0006】

40

本明細書および後続の特許請求の範囲において、ピクセル画像の解像度よりも高いまたは微細な解像度で定められた形状とは、そのような形状の境界をピクセル画像の解像度よりも高い解像度で特定できる定義を有している形状を指す。高い解像度とは、定められた形状の解像度であり、必ずしもそのような形状を定める式または記述に使用される点または数の解像度ではない。例えば、文字の曲線または線セグメントを記述するために使用される式または記述において端点または制御点として使用されるすべての点が、所与のピクセル画像のピクセル格子の角に生じるアウトラインフォントを定めることができる。そのような場合、フォントの形状を記述するために使用される点が、ピクセル画像と同じ解像度を有することになるが、文字の形状が曲線または斜めの線を含んでいる場合、そのような線によって記述される形状の解像度は、ピクセル画像の解像度よりもはるかに高くなる

50

。

【発明の実施の形態】

【0007】

図1は、本発明の譲受人の名前でもある単語「B i t s t r e a m」100を示しており、文字102の形状が高解像度のアウトラインで記述されているため、それらの輪郭が滑らかに見える。

【0008】

図2は、図1の文字102の形状が、アウトラインの記述の解像度よりも低いピクセル解像度で表示されるときに、どのようによりギザギザに見えるのかを示している。

【0009】

図3は、図2のピクセル画像の破線の囲み108で示された部分の拡大である。図3においては、ピクセル画像の個々のピクセル110が示されている。図3に示した画像のピクセルは、ビデオ表示の場合に一般的であるとおり、直交する行112および列114に配置されている。

【0010】

図3において、大文字「B」が、3つのアウトライン113A、113B、および113Cを有して示されており、小文字「i」が、2つのアウトライン113Dおよび113Eを有して示されている。

【0011】

図4は、アウトライン113A～113Eを、それらアウトラインによって定められる形状に対応するピクセル画像の上に重ねられた破線で示している。このピクセル画像は、図2の破線の囲み108内に示された「B i」の画像に相当している。

【0012】

図4から見て取れるように、例えばアウトライン113A～113Eのような高解像度のアウトラインをより低い解像度のピクセル画像へと変換するプロセスは、例えば図4に示したジグザグの縁118など、ジグザグの縁を有する画像を生み出すことが多い。

【0013】

従来技術において、アンチエイリアシングとして知られるプロセスを使用することによって、ピクセル画像の縁を観察者にとってより滑らかに見えるようにできることが、よく知られている。アンチエイリアシングは、図5に示したグレースケールのピクセル120によって示されているように、描画されるより高解像度の形状によって部分的に覆われるピクセルに、中間の被カバー値を持たせるプロセスである。

【0014】

図6は、個々のピクセル110の格子を示すことなく図5のピクセル画像を示している点を除き、図5と同一である。

【0015】

アンチエイリアシングは、一般に、部分的に覆われるピクセルへと、当該ピクセルのうちの1つ以上の高解像度の形状によって覆われている割合に比例するカラー値またはグレースケール値としても知られる被カバー値を割り当てようとする。これが、ピクセル110が形状114によって部分的に覆われて示されている図7に表示されている。

【0016】

図7に示したピクセル110について、形状によって覆われている正確な割合を計算するため、アウトラインフォントの記述によってもたらされる形状の幾何学的定義を取り上げて、幾何学的方法を使用することが可能である。ひとたびこれが実行されると、その割合に対応する被カバー値を、アンチエイリアシングの目的のためにピクセルへと割り当てることができる。残念ながら、そのような演算はかなりの量の処理を必要とし、多数の文字をそのような正確に計算されたアンチエイリアシングにて描画することは、望ましくないほどに遅くなる可能性がある。したがって、部分的に覆われているピクセルの被カバー値を近似を使用して計算することが一般的になっている。

【0017】

図 8 は、そのような近似方法の 1 つを示している。この方法では、描画される形状 1 1 4 のアウトラインの記述の曲線が、相応の直線セグメント 1 2 2 の連続によって近似されている。このような近似は、形状 1 1 4 の正確な面積を計算しようと試みるよりも、大幅に高速である。

【 0 0 1 8 】

図 9 は、他の従来技術の近似方法を示している。この方法は、ピクセル 1 1 0 内に位置する点 1 2 4 のアレイのそれぞれについて、その点が形状 1 1 4 の範囲に入るか否かを判断する。次いで、ピクセル内で形状 1 1 4 の範囲に入る点の、形状 1 1 4 の範囲に入らない点に対する割合の関数である被カバー値を、ピクセルへと割り当てる。

【 0 0 1 9 】

図 1 0 は、本件出願の発明者によってこれまで使用されてきた近似方法を示している。この方法によれば、各ピクセルに、1 ~ 5 本の水平サンプリング線 1 2 6 および 1 ~ 5 本の垂直サンプリング線 1 2 8 が組み合わされている。各サンプリング線について、その線の何パーセントが形状 1 1 4 によって覆われているかが判断される。次いで、4 つのサンプリング線のそれぞれについての被カバー値の平均に等しい被カバー値が、ピクセル 1 1 0 に割り当てられる。

【 0 0 2 0 】

この先の方法に使用されるプログラムは、使用の許諾を受けた O E M が、水平および垂直の両方のスキャン線の数に 1 ~ 5 の間の任意の値に設定できるようにしている。しかしながら、発明者は、図 1 0 に示した両方向において 3 本未満のサンプリング線を使用すると、不十分な結果が生まれがちであることに気付いており、そのような被許諾者らに、各方向について少なくとも 3 本のサンプリング線を使用するように推奨している。

【 0 0 2 1 】

上述の方法はすべて、ピクセル画像における使用のために認容できるアンチエイリアシングを生み出している。図 8、9、および 1 0 に関して上述した近似は、通常は、アンチエイリアシング画像の描画の速度についてかなりの改善を生み出している。しかしながら、部分的に覆われているピクセルに対してピクセル被カバー値を割り当てるための近似方法であって、さらに高速な近似方法を生み出すことが望ましい。

【発明の概要】

【 0 0 2 2 】

本発明の目的は、アンチエイリアス処理されたビットマップ画像のピクセルにピクセル被カバー値をより迅速に決定するための装置、方法、および媒体を提供することにある。

【 0 0 2 3 】

本発明のさらに他の目的は、そのような装置、方法、および媒体であって、可能にする演算の軽減に鑑みて、そのようなピクセル被カバー値の決定がかなり正確である装置、方法、および媒体を提供することにある。

【 0 0 2 4 】

本発明の第 1 の態様によれば、人間による読み取りが可能な表示に使用するための 2 次元のピクセル画像において、ピクセル被カバー値を設定するためのコンピュータ化された方法が提供される。2 次元のピクセル画像は、フォント化された文字またはグラフィック・デザインの形状など、ピクセル画像の解像度よりも微細な解像度で定義される 1 つ以上の形状の高解像度の 2 次元表示を表わしている。ピクセル画像は、複数のピクセルで形成されている。各ピクセルは、高解像度の 2 次元表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、サンプリング領域が形状のうちの 1 つによってどの程度覆われているかを表わすピクセル被カバー値を有している。

【 0 0 2 5 】

本発明のこの第 1 の態様によれば、この方法が、画像の複数のピクセルの各々について以下のステップを含んでいる。すなわち、ピクセルに対応するサンプリング領域内を異なる方向に延びている少なくとも 2 つのサンプリング線の各々について、サンプリング線がサンプリング領域内で形状のいずれかによってどの程度覆われているかの関数として、線

10

20

30

40

50

被カバー値を決定するステップと、2つのサンプリング線について決定した線被カバー値の非線形関数として、ピクセルのピクセル被カバー値を決定するステップである。

【0026】

本発明のこの第1の態様の多数の実施形態においては、当該方法によって生成される異なる方向に延びている2つのサンプリング線についての線被カバー値の組み合わせの大部分において、ピクセル被カバー値の変化の割合は、2つのサンプリング線のうちで線被カバー値がそのような形状によってサンプリング線の2分の1が覆われた場合の値に近い方のサンプリング線の線被カバー値の変化に対して、2つのサンプリング線の他方の線被カバー値の変化に対するよりも、大きい。

【0027】

いくつかのそのような実施形態においては、2つのサンプリング線についての線被カバー値の組み合わせの大部分において、ピクセル被カバー値の変化の割合は、2つのサンプリング線のうちで描こうとする形状によって2分の1が覆われた状態に近い方のサンプリング線の線被カバー値にのみ対応して変化する。

【0028】

他のそのような実施形態においては、2つのサンプリング線についての線被カバー値の組み合わせの大部分において、ピクセル被カバー値の変化の割合は、2つのサンプリング線の両方の線被カバー値の変化にตอบสนองして変化する。

【0029】

本発明のこの第1の態様のいくつかの実施形態においては、ピクセル被カバー値の決定に使用される非線形の関数が、参照テーブルにおいて2つのサンプリング線の線被カバー値の関数としてアドレス指定される位置の値を参照することによって、それらの値を決定する。線被カバー値の所与の組み合わせについての参照テーブル内の値は、ピクセルが線被カバー値の該当の組み合わせを有している状況において前もって行われる複数のピクセル被カバー値の計算から、導出することができる。

【0030】

描こうとする形状が、異なるフォント・セット内の文字である場合には、異なるフォント・セットからの文字を描画するときに、異なる参照テーブルを使用してピクセル被カバー値を決定することができる。

【0031】

本発明のこの第1の態様のいくつかの実施形態においては、非線形の関数が、異なる方向に非平行に延びる2つのサンプリング線の線被カバー値の重み付け和を含む関数として、ピクセル被カバー値を決定する。この重み付け和において、2つの線被カバー値のそれぞれの寄与は、そのような線被カバー値の各々が中間的の線被カバー値までどれだけ近いかの関数である。

【0032】

本発明のこの第1の態様のいくつかの実施形態においては、異なる方向に延びる2つのサンプリング線が、互いに直交している。そのようないくつかの実施形態においては、ピクセル画像が、行および列に配置されたピクセルで構成され、被カバー値が、各ピクセルのわずか2つのサンプリング線について決定され、一方のサンプリング線が実質的に各ピクセル行の中央を延びており、一方のサンプリング線が実質的に各ピクセル列の中央を延びている。

【0033】

本発明のこの第1の態様の多くの実施形態においては、描こうとする形状が、形状をピクセル画像のピクセル解像度よりも高い解像度で定義しているアウトラインによって記述されており、線被カバー値が、形状のアウトラインとサンプリング線との交点の間の距離の関数として決定される。

【0034】

本発明の第2の態様によれば、人間による読み取りが可能な表示に使用するための2次元のピクセル画像を生成するためのコンピュータ化された方法であって、2つのピクセル

10

20

30

40

50

設定パスを使用する方法が提供される。ピクセル画像は、ピクセル画像の解像度よりも微細な解像度で定義される1つ以上の形状の高解像度の2次元表示を表わしている。ピクセル画像は、複数のピクセルで形成されている。各ピクセルは、高解像度の表示のうちの当該ピクセルに対応するサンプリング領域を表わすとともに、サンプリング領域が形状のうちの1つによってどの程度覆われているかを表わすピクセル被カバー値を有している。

【0035】

本発明のこの第2の2パスの態様は、以下を含んでいる。すなわち、形状と2次元高解像度の表示において第1の方向に延びている第1組の平行なサンプリング線との交点を計算すること、形状と2次元高解像度の表示において第1の方向とは異なる第2の方向に延びている第2組の平行なサンプリング線との交点を計算すること、ならびに第1および第2のピクセル設定パスを実行すること、を含んでいる。

10

【0036】

第1のピクセル設定パスは、各ピクセルのピクセル被カバー値を、前記ピクセルのサンプリング領域の第1組の1つ以上のサンプリング線の線被カバー値を、そのようなサンプリング線がサンプリング領域内で形状のいずれかによってどの程度覆われているのかの関数として決定し、次いでそのような線被カバー値の関数としてピクセルのピクセル被カバー値を決定することによって、計算することを含んでいる。

【0037】

第2のピクセル設定パスは、第1のピクセル設定パスを実行した後に実行され、形状と第2組のサンプリング線との間の1つ以上の交点が計算されているサンプリング領域を呈しているピクセルについてのみ、第1のパスにおいて設定されたピクセル被カバー値を変更する。第2のピクセル設定パスは、そのような交点が計算されているピクセルのピクセル被カバー値を、ピクセルのサンプリング領域の第2組の1つ以上のサンプリング線の線被カバー値を、そのようなサンプリング線がサンプリング領域内で前記形状のいずれかによってどの程度覆われているのかの関数として決定し、次いでピクセルのピクセル被カバー値を、第1のピクセル設定パスにおいてピクセルについて計算された線被カバー値および第2のピクセル設定パスにおいてピクセルについて計算された線被カバー値の関数として決定することによって、変更する。

20

おり、第1および第2の方向が、アレイの2つの次元に一致している。

【0038】

30

本発明のこの第2の2パスのいくつかの実施形態においては、ピクセル画像が、順次のアドレスにてメモリに保存された一連のピクセル行で構成され、各ピクセル行が、より密な間隔の順次のアドレスにてメモリに保存された一連のピクセル被カバー値を含んでいる。そのような実施形態においては、第1のピクセル設定パスが、そのようなピクセル行の方向に延びているサンプリング線について実行される。多くのそのような実施形態においては、ピクセル画像が、ピクセルの2次元アレイで構成されており、第1および第2の方向が、アレイの2つの次元に一致している。

【0039】

本発明のこの2パスの態様のいくつかの実施形態においては、第2のピクセル設定パスが、ピクセルのピクセル被カバー値を、第1および第2の方向に延びるサンプリング線の線被カバー値の非線形関数として決定させる。この非線形の関数においては、ピクセル被カバー値が所与の線被カバー値の変化の割合の関数として変化する割合が、線被カバー値の関数として変化する。

40

【0040】

本発明の第3の態様によれば、2次元のピクセル画像を生成するための2パスの方法であって、本発明の第2の態様の2パスの方法に類似した方法が提供される。本発明のこの第3の態様によれば、描こうとする形状が、ピクセル画像の解像度よりも微細な解像度であるアウトラインによって定義される文字フォント形状であり、画像のピクセルが、行および列に配置されている。第1組の線が、ピクセル行に沿って延びており、第2組の線が、ピクセル列に沿って延びている。この方法は、描こうとする文字フォント形状の各アウ

50

トラインに沿って、このアウトラインと行および列の線との間の各交点を見つける。発見されたそのような交点の各々が、交点が生じたサンプリング線に関する順序付けられた交点リストにおいて、リストにおける交点の順序に、リストの関連のサンプリング線に沿った交点の順序を反映させつつ、配置される。

【0041】

本発明の第3の態様の第1のピクセル設定パスは、各ピクセル行について実行されるループを含んでいる。このループにおいては、行の各ピクセルについて、その行のサンプリング線に関するリストにピクセルのサンプリング領域内で生じる交点が存在しない場合、ピクセルがサンプリング線の現在の被カバー状態（完全に覆われているか、全く覆われていないかである）に対応するピクセル被カバー値に設定される。そうでない場合、以下のステップ、すなわち1) サンプリング線の現在の被カバー状態を、ピクセルのサンプリング領域において生じる交点リスト内の1つ以上の次の交点のそれぞれを反映すべく変更するステップ、2) 行の線被カバー値を、ピクセルのサンプリング領域内の行サンプリング線の文字フォント形状によって覆われている割合の関数として計算するステップ、および3) ピクセルを、ピクセルについて計算された行の線被カバー値の関数として決定されるピクセル被カバー値に設定するステップ、が行われる。

【0042】

第2のピクセル設定パスは、本発明の第3の態様においては、第1のピクセル設定パスの後で実行され、自身の関連の交点リストに交点を有している各ピクセル列について実行されるループを含んでいる。このループにおいては、列のうちで自身の交点リストにピクセルのサンプリング領域において生じる交点を有している各ピクセルについて、以下のステップ、すなわち1) 列サンプリング線の現在の線被カバー状態を、交点リストのピクセルのサンプリング領域内の次の各交点を反映すべく変更するステップ、2) 列の線被カバー値を、ピクセルのサンプリング領域内の列サンプリング線の文字フォント形状によって覆われている割合の関数として計算するステップ、および3) ピクセルのピクセル被カバー値を、第1のピクセル設定パスにおいてピクセルについて計算された行の線被カバー値および第2のピクセル設定パスにおいてピクセルについて計算された列の線被カバー値の関数として設定するステップ、が実行される。

【0043】

本発明のこの第3の態様の多数の実施形態においては、第2のピクセル設定パスにおいてピクセル被カバー値を設定するために使用される関数が、非線形の関数であって、行および列の線被カバー値の組み合わせの大部分において、ピクセル被カバー値の変化の割合が、ピクセルの行または列のサンプリング線のうちで描こうとする形状によって2分の1が覆われた状態に近い方のサンプリング線の線被カバー値の変化に対して、他のサンプリング線の線被カバー値の変化に対してより、大きい。

【0044】

本発明のさらに他の態様によれば、コンピュータ・システムおよびコンピュータで読み取ることができるメモリに保存されたコンピュータ・プログラミングであって、上述の形式の方法を実行するコンピュータ・システムおよびコンピュータ・プログラミングが提供される。

【0045】

本発明のこれらの態様および他の態様が、好ましい実施形態についての以下の説明を添付の図面と連動させて検討することによって、さらに明らかになるであろう。

【好ましい実施形態の詳細な説明】

【0046】

図11は、本発明を具現化するコンピュータ・システムについて多数考えられる種類のうちの1つであるコンピュータ・システム130の概要を提示している。

【0047】

システム130は、ランダム・アクセス・メモリ（すなわち、「RAM」）136に保存されたインストラクションを実行するためのCPU134を有しているコンピュータ1

10

20

30

40

50

32を備えている。さらに、ランダム・アクセス・メモリ136は、CPU134によって使用されるデータ値を保存している。コンピュータは、CPUとキーボード140およびマウス142などのI/Oデバイスとの間のやりとりのために設計された入力/出力(すなわち、「I/O」)インターフェイス138をさらに備えている。ビデオ・インターフェイス144が、ビデオ・モニタ146の画面上に画像を生成するために必要とされる電子的出力を供給する。ハードディスク・コントローラ148が、CPUとハードディスク150との間の仲立ちを行う。ハードディスク150は、オペレーティングシステム・プログラム152および1つ以上のアプリケーション・プログラム(図11に示したアプリケーション・プログラム154など)など、コンピュータによって使用されるプログラムおよびデータを保存している。

10

【0048】

オペレーティングシステムは、フォント化されたスケーラブル・フォントの形状など、高解像度の形状をラスターライズするためのスキャン変換コード156を含んでいる。システム130のラスターライジング・コード156は、従来技術の多くのアンチエイリアシング・ラスターライゼーション・コードよりも少ない演算で、したがってより高速にアンチエイリアシングのラスターライゼーションを可能にする本発明の態様を含んでいる。

【0049】

さらに、オペレーティングシステムは、上述の一般的な形式のスケーラブル・フォントのアウトライン形状を定めたフォント158を保存している。

【0050】

20

さらに、コンピュータ132は、CPU134とCD ROMドライブ162との間の仲立ちを行うCD ROMコントローラ160を備えている。CD ROMドライブは、図11に示したCD ROM164などのCD ROMが、図11に破線164Aで示されているようにCD ROMドライブへと挿入されたときに、そのようなCD ROMを読み取ることができる。

【0051】

バス135が、CPU134をRAM136、I/Oコントローラ138、ビデオ・コントローラ144、ハードディスク・コントローラ148、およびCD ROMコントローラ160へと接続し、これらすべてのデバイスの相互の通信を可能にしている。

【0052】

30

本発明の態様を具現化するスキャン変換コード156を、CD ROM164ならびに他の実質的に任意の大容量記憶装置(例えば、フロッピーディスク、リムーバブル・ハードドライブ、およびデジタル・フラッシュROM)など、任意の種類のメモリ・デバイスに記録することができる。ひとたびそのようなプログラミング・コードがコンピュータ132へと入力されると、図11に示されているように、演算システムによって使用されるハードディスクまたは他の大容量記憶装置へと従来どおりに保存される。プログラミングは、ひとたびコンピュータの大容量記憶装置に保存されると、さらに実際に使用されるときに、そのインストラクションをCPU134によって実行して本発明によるスキャン変換を行うことができるよう、ランダム・アクセス・メモリ136へとロードされる。

【0053】

40

図13~32は、図13~32のピクセルが、3つの水平サンプリング線126および3つの垂直サンプリング線128を有するのではなく、ただ1つの水平サンプリング線164およびただ1つの垂直サンプリング線166を有する点を除き、図10に示したピクセル110と同様のピクセル110を示している。

【0054】

これらの図は、わずか2つのサンプリング線を使用して、図10に関して上述したシステムにおいて行われるように、ピクセルの被カバー率をサンプリング線の被カバーの割合の平均へと設定しようとした場合に、どのようにしてかなりの不正確さが生じうるのかを説明している。

【0055】

50

例えば、図 1 3 ~ 1 6 を参照すると、水平な上縁 1 6 8 を有している形状 1 6 6 によって覆われているピクセルの一部が、上方へと垂直方向に移動する場合、2 つのサンプリング線 1 6 4 および 1 6 6 の被カバーの割合の平均が、形状の上縁 1 6 8 が水平サンプリング線 1 6 4 を横切るときに大きく跳ね上がる。これは、上縁 1 6 8 が線 1 6 4 を横切る時点において、ピクセルの被カバー部分の割合がごくわずかに増加するだけで、線 1 6 4 の全体が、全くカバーされていない状態から完全にカバーされている状態へと移行するためである。図 1 0 に関して説明したように、ピクセルの被カバー値が、線の被カバー値の平均として計算される場合、ピクセルの被カバー値は、図 1 4 におけるほぼ 4 分の 1 から図 1 5 における 4 分の 3 超へと跳ね上がることになり、実際のピクセルの被カバー率がわずかなパーセントしか変化していないにもかかわらず、ほぼ 5 0 % にもなる変化である。

10

【 0 0 5 6 】

この不正確さは、ピクセル当たりには 2 つのサンプリング線しか存在しないときに人間の目にとって滑らかに見える縁を生成するという仕事について、アンチエイリアシングが比較的乏しい仕事しかしないという結果を引き起こすことになる。

【 0 0 5 7 】

図 1 7 ~ 2 0 は、ピクセルを横切って水平方向に移動する垂直縁 1 7 2 を有する形状 1 7 0 を示している点を除き、図 1 3 ~ 1 6 と同様である。

【 0 0 5 8 】

図 1 8 および 1 9 は、垂直縁 1 7 2 が図 1 8 および 1 9 に示した位置の間を比較的わずかに動くだけで、どのようにして水平および垂直のサンプリング線 1 6 4 および 1 6 6 の被カバー率の平均に大きな跳ね上がりが生じるのかを示している。

20

【 0 0 5 9 】

この問題に対する解決策を考え出す試みにおいて、本発明の発明者は、ピクセルの被カバー値を、水平および垂直のサンプリング線 1 6 4 および 1 6 6 の被カバー値の非線形関数として計算することで、被カバー値のより正確な見積もりを生成できることを発見した。それは、ピクセル被カバー値を 2 つのサンプリング値の線形関数（例えば、それらの値の平均）として計算した場合にピクセルの被カバー率のわずかな変化によって引き起こされる上述のピクセル被カバー値の大きな不連続を、排除できるからである。これは、個々のピクセルに比べて比較的大きな造作を有する形状をレンダリングするときに生じる部分的なピクセルの覆いの種類の大部分において、種々の方向に伸びているサンプリング線のうちでより中間的な被カバー値を有しているサンプリング線が、通常は、別の方向に伸びてより極端な被カバー値を有しているサンプリング線よりも、ピクセル全体の被カバー値により近い被カバー値を有するためである。非線形関数によれば、種々の線の被カバー値の寄与を、それらの線の被カバー値そのものの関数として変化させることができる。

30

【 0 0 6 0 】

図 1 2 は、図 1 1 に示したスキャン変換コード 1 5 6 内のアルゴリズム 1 9 0 の高度に概略化した擬似コードの記述である。このアルゴリズムは、所与のピクセルの被カバー値を、所与のピクセル内を種々の方向に延びているサンプリング線の非線形関数として計算する。このアルゴリズムは、描画すべき画像内の部分的に覆われたピクセルのそれぞれについて、ステップ 1 9 2 ~ 1 9 6 の実行を生じさせる。

40

【 0 0 6 1 】

ステップ 1 9 2 は、所与の境界ピクセルにおいて第 1 の方向（水平方向など）に延びている 1 つ以上のサンプリング線について、被カバーの程度を計算する。次いで、ステップ 1 9 4 が、ピクセルにおいて第 2 の方向（垂直方向など）に延びている 1 つ以上のサンプリング線について、被カバーの程度を計算する。次いで、ステップ 1 9 6 が、ピクセルの被カバー値、またはグレースケール値を、異なる方向に延びている線の被カバー値の非線形関数として計算する。

【 0 0 6 2 】

図 1 2 に示した一般的なアルゴリズムの多数の実施形態においては、ステップ 1 9 6 において使用される非線形関数が、ピクセルへと割り当てられる被カバー値を、被カバー線

50

のうちで最も中間的な被カバー値を有する被カバー線の被カバー値とともにより迅速に変化させる非線形関数である。

【0063】

例えば図13において、垂直サンプリング線166の被カバー値が、水平サンプリング線164よりも、ピクセル110の被カバーの割合を良好に表わしていることを見て取ることができる。この場合、サンプリング線166が、全く覆われていないサンプリング線164よりも、2分の1の被カバーを表わすことにより近い被カバー値を有している、同じことが、図14および15においても真である。

【0064】

図15においては、ピクセルが、半分が覆われるよりもわずかに進んでおり、その被カバー値は、垂直線166の被カバー値と同じである。水平サンプリング線164は、完全に覆われており、したがってその被カバー値は、全体としてのピクセルの被カバー値を正確に表わしてはいない。この場合、完全に覆われている水平サンプリング線の被カバー値よりも、1/2が覆われている状態により近い水平線の被カバー値により大きな重みを与えることで、ピクセルの被カバーについてのより正確な評価がもたらされる。同じことが、程度は低い図16の場合にも真である。

10

【0065】

図13～16について説明したように最も中間的な線の被カバーにより大きな重みを与える理由は、図17～20にも同様に当てはまるが、これらの場合においては、水平線164が、より中間的な被カバー値を有しており、全ピクセルの被カバー値について最良な推定をもたらす。

20

【0066】

図21～24は、垂直方向に延びる比較的長細い矩形の形状174を、ピクセル110に対する種々の位置に示している。

【0067】

形状174は、図21に示した位置を有するときは、2つのサンプリング線164および166のいずれも覆っていない。したがって、このピクセルについて計算される被カバー値は0になる。これは、この場合にはピクセルのほぼ4分の位置が実際には覆われているため、約25パーセントの誤りであるが、本発明の発明者は、このような誤差が、形状がピクセルの1つの角にのみ広がって居る場合には、通常は人間の目にとって顕著には知覚されないことを発見した。

30

【0068】

図22に示した場面では、形状174の上部が水平線164をわずかに横切って広がり、この線を2分の1よりもわずかに少なく覆っている。垂直線166を横切って広がってはいない。この場合には、水平線のより中間的な被カバー値に対して、垂直サンプリング線のゼロのカバー値よりも多くの重みを与えることが、通常は正確である。なぜならば、形状174がピクセル110に比べて大きいと仮定すると、ピクセルの実際のカバー値が、より中間的なカバー値により近い可能性が高いためである。

【0069】

図23は、長細い形状174が、垂直線166の実質的にすべておよび水平線164のほぼ2分の1のみを覆っている状況を示している。この場合、水平線164のより中間的な被カバー値が、全ピクセルの被カバーについて、水平サンプリング線166のより極端な被カバー値よりも正確な評価をもたらす。

40

【0070】

図24は、より中間的な被カバー値が、より極端なゼロの被カバー値よりも、ピクセル全体の実際の被カバーの割合をよりよく表わす他の場面を示している。

【0071】

図25～28は、45度の角度を有する線178を持つ形状176を、ピクセル110に対する種々の位置に示している。この場合、2つのサンプリング線、すなわち水平サンプリング線164および垂直サンプリング線166の値が、通常はほぼ正確に同じ値を有

50

することを、見て取ることができる。これが真であるとき、それらの被カバー値は等しく中間的であり、ピクセル全体の被カバー値の決定に等しく寄与すべきである。

【0072】

図29～32は、ピクセル110を、急峻な角度を有する縁182を持つ形状180が当該ピクセルに対して種々の位置に位置するものとして示している。

【0073】

図29においては、水平および垂直のサンプリング線のどちらも、形状によって少しも覆われていない。この場合、ピクセル110に被カバー値0が割り当てられる。この場合において、誤差が図21よりもはるかに小さいことに注意すべきである。

【0074】

図30および31においては、水平サンプリング線164が、最も中間的な被カバー値を有しており、ピクセルの被カバー値の決定においては、その値により大きな重みを与えられる。これは、水平線164の被カバー値がピクセル110の実際の被カバーに等しいため、適切である。

【0075】

図32においては、やはり水平サンプリング線164が、最も中間的な被カバー値を有しており、やはりピクセル全体の被カバー値の決定においては、より中間的な被カバー値を有するサンプリング線により大きな重みを与えることが適切である。

【0076】

図33～40は、図11に示したスキャン変換コード156について考えられる多数の実施形態のうちの1つの高度に概略化した擬似コードの表現である。

【0077】

図33～40の擬似コードは、コンピュータ画面上に描画され、あるいは何らかの印刷媒体上に印刷される文字のビットマップ表現のピクセル被カバー値を計算するための2パスのプログラムを示している。このプログラムの最高レベルは、図33に示したdrawTextルーチン200によって表わされている。このルーチンは、文字の形状が図1～5に関して上述した形式の高解像度のアウトラインによって記述されているテキストの1つ以上の線から、ピクセル画像を生成する。

【0078】

図33に示されているように、drawTextルーチン200は、描画しようとするビットマップの各文字について実行されるループ202を含んでいる。図33のdrawTextルーチンの2パスの手順は、ループ202の繰り返しのそれぞれにおいて実行される。この手順は、第1のxパスを実行し、次いで第2のyパスを実行する。大きくは図33のループ206によって実行されるxパスにおいて、被カバー値が、図35のXLinePassルーチンによってビットマップ250の各ピクセルに割り当てられる。このピクセル被カバー値は、単純に、ピクセルを通過して延びるxサンプリング線の被カバー値である。yパスにおいては、第1のパスにおいて設定された被カバー値が、yの交点が生じているピクセルについてのみ変更され、このようなパスでの演算を大きく減らしている。yの交点を有するピクセルについて、ピクセルの被カバー値が、ピクセルのxおよびyの両方のサンプリング線のカバー値の関数として決定される。

【0079】

この2パス・ループ202は、ステップ204から222を含んでいる。

【0080】

ステップ204が、ループ202の現在の文字について、図34に示したcharSetUpルーチン224を呼び出す。このcharSetUpルーチンは、描画しようとする所与の文字について、プログラムの2パス手順のxパスおよびyパスを実行する前に必要な準備のステップを実行する。

【0081】

charSetUpルーチンは、文字のアウトライン113のそれぞれについてループ226を実行する。これが図41に示されており、この図に示されている大文字「B」が

10

20

30

40

50

、3つのアウトライン113A、113B、および113Cを有している。このようなアウトラインのそれぞれについて、ループ230が、当該アウトラインの処理が完了するまで実行される。

【0082】

ループ230は、ステップ232、234、および236を有している。ステップ232は、ループ230の現在のアウトラインを巡って、図41に示されている当該アウトラインと水平または垂直の中央サンプリング線164または166のそれぞれとの間の次の交点241または243に達するまで前進する。本発明の発明者によって現在使用されている本発明の実施形態においては、ステップ232において、アウトラインの曲線部分は、短い直線セグメントの連続で近似されている。アウトラインと中央線との間の交点は、中央線と曲線の近似に使用されている直線との間の交点として発見される。本発明の他の実施形態においては、別の近似法を使用することが可能であり、あるいは追加の演算を行って、アウトラインの曲線と中央線との間の正確な交点を発見することができる。

10

【0083】

交点241または243に達するたびに、ステップ234が、交点がxまたはyのどちらの中央線であるかを判断する。次いで、ステップ236が、交点の表現242または244を、交点のxまたはyの線のそれぞれに関連付けられたリンク・リスト内の適切な位置に配置する。これが図41に示されており、符号X0～X8が付されたリンク・リスト238が、水平な中央サンプリング線164に関連付けられ、符号Y0～Y7が付されたリンク・リスト240が、垂直な中央サンプリング線166に関連付けられている。

20

【0084】

交点の表現242または244のそれぞれは、その水平または垂直のサンプリング線のそれぞれとの交点のxまたはy位置を含んでいる。また、そのような表現のそれぞれは、正またはマイナス1の値を有する交点のedge Valueを含んでいる。ワインディングカウント(winding count)を使用する従来技術のスキャン変換システムにおいて行われているように、各交点のエッジ値は、アウトラインとの交点が生じている方向から見たとき、アウトラインを巡る経路が右または左のどちらへと移動しているように見えるかによって決定される。

【0085】

ここで説明される本発明の実施形態においては、所与のxまたはyのサンプリング線に沿ったxおよびy位置が、32ビットの精度、サンプリング線に沿ったピクセルの数を表わしている26のビット、および所与のピクセル内のサンプリング線に沿った位置を表わしている最下位の6ビットで保存される。これらの6つのビットを、ゼロから63まで、ピクセル内の長さの64のグラデーションを定めるために使用することができる。被カバー値が、これら64のグラデーションから計算され、2で乗算され、したがって被カバー値を、ゼロから126まで偶数によって変化するものとして見ることができる。

30

【0086】

図41を参照することによって、ひとたびループ230のプロセスが文字のアウトラインのそれぞれについて実行されると、その文字に係するサンプリング線のそれぞれが、当該サンプリング線と文字のアウトラインとの間の各交点の表現を含んでいるリンク・リストであって、各リンク・リスト内の交点がサンプリング線に沿って正しい順序で位置しているリンク・リストを有することになることを、見て取ることができる。

40

【0087】

図34を参照すると、ひとたびループ320が文字のアウトライン113のそれぞれについて実行されると、ループ226が完了し、次いでステップ246および248が実行される。ステップ246は、xおよびyのサンプリング線164および166が通過するピクセルのそれぞれについて、ピクセル被カバー値を保持するために十分に大きいグレースケール・ビットマップを形成する。

【0088】

図42が、そのようなビットマップ250を示している。このビットマップは、列25

50

4に配置された複数の個々の画素252で構成されている。各列254は、表現される文字内の所与のxサンプリング線に関連付けられたすべてのピクセルを表わしている。ここに説明される本発明の実施形態においては、各ピクセルの被カバー値が、ゼロ~126の間の数によって表わされ、各ピクセルの1バイトでの表現を可能にしている。

【0089】

ひとたび図34のステップ246が実行されると、ステップ248が、ビットマップのすべての要素の値をゼロに設定する。この時点で、所与の文字について、charSetUpルーチンの動作が完了する。

【0090】

再び図33に目を向けると、ひとたびステップ204でのcharSetUpの呼び出しがdrawTextルーチンへと戻ると、ステップ206が、ループ202によって描画される文字に関連付けられたそれぞれのx線すなわち水平線164についてループを実行する。ループ206は、ステップ208~212を含んでいる。

【0091】

ステップ208は、ポインタfillStartを、現在のx線のリンク・リスト238の最初の交点の表現242へと向ける。例えば、ループ206が、図41のX2と付されたリンク・リストを有するx線について実行される場合には、fillStartが、図41に示されている交点242Aへと向けられる。

【0092】

次いで、ステップ210が、ポインタfillEndを、現在のx線のリンク・リストの第2の交点の表現へと向ける。これにより、ループ206が図41のリンク・リストX2を有するx線について実行される場合には、fillEndポインタがノード242Bへと向けられる。

【0093】

ひとたびこれら2つのステップが実行されると、ステップ212が、ルーチンXLinePass254を呼び出す。

【0094】

図35が、XLinePassルーチンを示している。XLinePassルーチンは、windingCount変数をゼロにするステップ256を有している。文字のラストライジングの当業者であれば理解できるとおり、ワインディングカウントは、描画される文字に関連付けられた領域の或る部分が文字の内側または外側のどちらであるかを把握するために使用される。

【0095】

次に、図35においてXLineLoopと呼ばれているループ258が、ルーチンXLinePassが呼び出しされたxサンプリング線についての処理が完了するまで実行される。XLineLoopは、ステップ260~304で構成されている。

【0096】

ステップ260は、変数currentPixel#を、fillStartが関連付けられている交点が位置しているピクセルの番号に設定する。例えば、XLinePassが、図41において符号X2が付されたリンク・リストを有する線について呼び出された場合には、ステップ260が、currentPixel#を、図41および43に示されリンク・リスト238の第1の要素242Aに対応する交点241Aを有しているピクセル110Aの番号に設定する。次いで、ステップ262が、fillStartおよびfillEndの交点が向けられた交点241のedgeValueを、windingCount変数へと加える。

【0097】

fillStartおよびfillEndが、リンク・リストX2の交点の表現242Aおよび242Bに向けられているとき、交点の表現242Aは、形状の外側から形状の内側への変化をマークする交点を表わしているため、交点の表現242AのedgeValueは1である。このedgeValueを、それまではゼロであったwinding

10

20

30

40

50

C o u n t へと加えることで、w i n d i n g C o u n t が 1 に設定される。

【 0 0 9 8 】

交点の表現 2 4 2 B は、形状の内側から形状の外側へと移行する交点を表わしているため、交点の表現 2 4 2 B の e d g e V a l u e はマイナス 1 である。この e d g e V a l u e を加えた後、w i n d i n g C o u n t はゼロとなり、f i l l S t a r t および f i l l E n d によって表わされる交点の間の距離が、中央線のうちで描画すべき形状によって覆われる長さである。

【 0 0 9 9 】

ひとたびステップ 2 6 2 が完了すると、ステップ 2 6 4 が、ワインディングカウントが 0 でない間、ループを実行する。上述の場合には、このループは、w i n d i n g C o u n t がすでにゼロであるため、いかなるステップも実行しない。しかしながら、形状の外側から再び形状の外側までの移行を完了する前に、中央線が複数組の関連のプラス 1 およびマイナス 1 の交点を通過できるよう、w i n d i n g C o u n t の計算を実行するとき、文字の形状の重なり合いの異なる部分を有することも可能である。図 4 1 および 4 3 に示した大文字「B」などのように単純な文字形状の場合には、そのような重なり合いの部分は存在しない。

【 0 1 0 0 】

重なり合っている形状を有する文字においては、ループ 2 6 4 は、現在のリンク・リストの交点に沿って進み、ステップ 2 6 6 が、f i l l E n d の交点を現在のリンク・リストの次の交点に設定し、ステップ 2 6 8 が、f i l l E n d の対応する e d g e V a l u e を w i n d i n g C o u n t がゼロに達するまで w i n d i n g C o u n t へと加える。

【 0 1 0 1 】

ひとたびプログラムがステップ 2 7 0 に達すると、現在の f i l l S t a r t および f i l l E n d の間の距離が、現在の x 線において、描画される文字によって覆われ、そのように覆われてはいない部分によって境界付けられている連続部分を表わしている。

【 0 1 0 2 】

この時点で、ステップ 2 7 0 が、f i l l E n d の位置が c u r r e n t P i x e l # を有するピクセルの内側であるか否かを確認すべくテストを行う。そのようである場合、ステップ 2 7 2 が、f i l l E n d および f i l l S t a r t の交点の間の距離を、現在のピクセル、すなわち図 4 2 に示したビットマップ 2 5 0 中で c u r r e n t P i x e l # に含まれたピクセル番号を有しているピクセルへと加える。これは、このような場合においては、f i l l E n d および f i l l S t a r t の交点の間の距離を現在のピクセルのピクセル被カバー値へと加えるべきであるからである。

【 0 1 0 3 】

ステップ 2 7 0 のテストが、f i l l E n d の交点が現在のピクセルに位置していないことを発見した場合、e l s e 命令 2 7 4 が、ステップ 2 7 6 ~ 2 9 4 の実行を生じさせる。

【 0 1 0 4 】

ステップ 2 7 6 は、現在のピクセルの被カバー値に、f i l l S t a r t の位置と現在のピクセルの右縁との間の距離を加える。これは、ステップ 2 7 4 のテストが満足されない場合に、f i l l S t a r t および f i l l E n d の位置の間の距離が 2 つ以上のピクセルにまたがって延びており、そのカバー距離がまたがっている個々のピクセルのそれぞれの個々の x 線の被カバー値への寄与が必要とされるために行われる、

【 0 1 0 5 】

ステップ 2 7 6 が、f i l l S t a r t の交点と現在のピクセルの右縁との間の距離を図 4 2 に示したビットマップ 2 5 0 の対応するビットマップ要素へと加えた後、ステップ 2 7 8 が c u r r e n t P i x e l # を増加させ、X L i n e P a s s 2 5 4 が実行されている現在の x 線に沿った次のピクセルへと向ける。

【 0 1 0 6 】

ひとたびこれらがなされると、ステップ280が、fillEndの交点がcurrentPixel#よりも大きいピクセル番号を有するピクセルに位置しているか否かを確認すべくテストを行う。そのようである場合、ステップ282～292が実行される。

【0107】

ステップ282は、現在のピクセルのアドレスが奇数であるか否かを確認すべくテストを行う。その場合、ステップ284および286が実行される。ステップ284は、現在のピクセルを値126へと設定し、現在のピクセルが完全に覆われていることを知らせる。次いで、ステップ286が、currentPixel#を1つだけ増加させる。

【0108】

ステップ282～286が実行された後、現在のピクセルは、偶数番号のアドレスを有する。次いで、ステップ290および292で構成されるループ288が、currentPixel#がfillEndの交点が生じるピクセル番号よりも小さい間、実行される。ステップ290は、126の2つの連続する値をcurrentPixel#のピクセルで始まるビットマップへと書き込むために、2バイトの書き込みインストラクションを使用する。次いで、ステップ292が、currentPixel#を2だけ増加させる。

10

【0109】

ループ288は、currentPixel#がfillEndの交点が生じるピクセルの番号と等しくなり、あるいはfillEndの交点が生じるピクセルの番号よりも1つだけ大きくなるまで続けられる。ひとたびこれらがなされると、ステップ294が実行される。ステップ294は、fillEndのピクセルに対応するピクセルを、fillEndのピクセルの左縁とfillEndの交点の位置との間の距離に等しい被カバー値に設定する。

20

【0110】

ひとたびこれらがなされると、ステップ296が、現在のx線のリンク・リストにおいてfillEndによって示される交点の後ろにさらなる交点が存在するか否かを確認すべく、テストを行う。存在する場合、ステップ298および300が、fillStartおよびfillEndをそれぞれ、現在のリンク・リストの次の2つの交点へと向ける。これは、プログラムを、fillStartおよびfillEndの新しい値によって示される交点にて出発して、XLineLoop258のさらなる繰り返しを実行するように準備させる。

30

【0111】

しかしながら、テスト296によって、現在のx線のリンク・リストに表わされている交点がもはや存在しないと判断される場合には、ステップ302が、ステップ304にXLineLoop258からの離脱およびXLinePass254の処理の終息を実行させる。これは、現在のx線のリンク・リストにそれ以上の交点が存在しない場合には、2パスのプロセスのうちのxパスについて、リスト上のすべての交点が処理済みであり、当該x線のすべての被カバー部分が図42に示したビットマップ250の対応するピクセルに対して適切な寄与を済ませているために行われる。

40

【0112】

図33に戻ると、ひとたびステップ212におけるXLinePassの呼び出しが戻されると、ステップ214が、drawtextルーチン200において使用される2パスのプロセスのうちの第2のパスを開始させる。この第2のパスは、ループ202の繰り返しによって描画される文字に関して図41に示したyサンプリング線166のそれぞれについて実行されるループ214を含んでいる。ループ214は、ステップ216～220を含んでいる。

【0113】

ステップ216が、fillStartポイントを、ループ214の実行の対象のy線のリンク・リスト240の最初の交点へと向ける。例えば、図41のリンク・リストY3を有するy線が現在の線であるならば、fillStartは、図41に示されている最

50

初の y 交点 2 4 4 A となるように向けられる。

【 0 1 1 4 】

ひとたびこれになされると、ステップ 2 1 8 が、fill End のポイントを、y 線のリンク・リストの第 2 の交点へと向ける。上述の場合には、これは、図 4 1 に示されている y 交点 2 4 4 B である。

【 0 1 1 5 】

ひとたび fill Start および fill End が、現在の y 線のリンク・リストの最初の 2 つの交点を指すと、ステップ 2 2 0 が、この現在の y 線について Y Line Pass ルーチン 3 0 6 を呼び出す。

【 0 1 1 6 】

図 3 6 が、Y Line Pass ルーチンを示している。このルーチンは、y サンプリング線と描画しようとする文字のアウトラインとの間に 1 つ以上の交点が存在している図 4 2 のビットマップ 2 5 0 のピクセル 2 5 2 のみが増え変わる点で、図 3 5 に示した X Line Pass 2 5 4 と異なっている。これは、第 2 のパスにおけるピクセル被力値のより複雑な計算を、文字の形状の縁に生じるピクセルのみに限定するため、とくには文字が比較的大きなサイズで描画される場合に演算を節約する。

【 0 1 1 7 】

Y Line Pass 3 0 6 は、図 3 5 のステップ 2 5 6 に相当するステップ 3 0 8 で開始する。このステップは、winding Count 変数をゼロにする。次いで、ステップ 3 0 9 が、ピクセルについて現在計算されている y 線の被力値を表わしている変数 CY をゼロにする。次に、ステップ 3 1 0 が、pending Pixel # 変数 2 8 0 を NULL 値、すなわち pending Pixel # が図 4 2 のビットマップ 2 5 0 の要素を現在表わしていないことを示す数へと設定する。次に、Y Line Loop 3 1 2 が、現在の y サンプリング線のリンク・リスト上のすべての交点 2 4 4 が処理されるまで実行される。Y Line Loop は、ステップ 3 1 4 ~ 3 6 2 で構成されている。

【 0 1 1 8 】

ステップ 3 1 4 は、fill Start および fill End の交点の edge Value を、winding Count へと加える。これは、図 3 5 に関して上述したステップ 2 6 2 と同等である。多くの場合、Y Line Loop 3 1 2 の最初の繰り返しにおいては、fill Start が、y 線のリンク・リストの最初の交点を指しており、fill End の交点が、このリストの第 2 の交点を指している。例えば、Y Line Loop 3 1 2 へと最初に進んだときに図 4 1 に示したリンク・リスト Y 3 を有している y 線が処理される場合には、fill Start および fill End が、それぞれ交点 2 4 4 A および 2 4 4 B を指す。

【 0 1 1 9 】

fill Start の交点が、y 線において形状によって覆われる連続部分を開始させ、fill End の交点が、この被力部分を終了させる場合、これら 2 つの交点の正およびマイナス 1 の edge Value がステップ 3 1 4 において winding Count へと加えられると、winding Count はゼロに等しい。形状が重なり合っている部分形状で構成されている場合を除き、これが常に当てはまる。

【 0 1 2 0 】

winding Count がゼロでない場合、ステップ 3 1 6 が、ステップ 3 1 8 において fill End を現在の y 線のリンク・リストの次の交点に向けようにし、ステップ 3 2 0 においてこの新しい交点の edge Value を winding Count へと加えるようにする。ステップ 3 1 8 および 3 2 0 は、winding Count がゼロに等しくなるまで繰り返され、その時点で、fill Start および fill End の交点の間の距離が、現在の y 線の単離された連続的な被力部分を表わしている。

【 0 1 2 1 】

ひとたび winding Count がゼロに等しくなると、ステップ 3 2 2 が、pending Pixel # がビットマップ内のピクセルを現在特定しているか否かを確認する

10

20

30

40

50

ためにテストを行う。pending Pixel # が或るピクセルを表わしており、fill Start の交点がこの係属中のピクセルの外側にある場合、ステップ 324 ~ 330 が実行される。ステップ 324 が、係属中のピクセルの x 線の被カバーを表わす変数 CX を、図 33 のループ 202 の 2 パスの手順の x パスにおいてすでに設定された係属中のピクセルの被カバー値またはグレースケール値へと設定する。

【0122】

次いで、ステップ 326 が、係属中のピクセルの被カバー値を、変数 CX および CY の両者の非線形関数として計算し、ここで CY は、ピクセル内の y 線の被カバー率を表わしている。これは、図 37 ~ 40 に示した式など、いくつかある種々の式のいずれかによって実行できる。図 37 ~ 39 の式はすべて、計算されるピクセル被カバー値が、現在のピクセルにおいて 50 パーセントの被カバーの表示に近い被カバー値を有している方のサンプリング線 (x または y のサンプリング線) の線の被カバー値により強く依存するようにしている。これらの式において、ピクセルおよび線の被カバー値は、すべてゼロ ~ 126 の範囲にある。すなわち、これらの式に示される 63 および 64 という値が、ほぼ 2 分の 1 のカバー値を表わす。

【0123】

これらの式の中で最も単純な 1 つは、図 39 に示した式である。この式は、ピクセルの被カバー値をその x または y サンプリング線の被カバー値に設定する 2 択の式であり、それら 2 つの値のうちで 50 % の被カバーをより近く表わしている方の値を選択される。x および y の被カバー値が等しいピクセルについては、ピクセルの被カバー値として CX が選択されるが、そのような場合には CX が CY に等しいため、実際には、被カバー値が CX および CY の値に等しくなるように設定されるということである。本発明の他のいくつかの実施形態においては、図 39 に示した 2 択の式は変形され、CX および CY が中間値 63 の反対側にあるが中間値 63 からの距離が同じまたは実質的に同じである場合に、ピクセルの被カバー値が、2 つの被カバー値のいずれかの値にではなく、中間値 63 に設定される。

【0124】

ひとたびステップ 326 が現在のピクセルの被カバー値の計算のために非線形関数を使用すると、ステップ 328 が、計算された値を係属中のピクセルへと書き込む。

【0125】

ひとたびステップ 328 が実行されると、ステップ 330 が、将来の計算に備えて CY をゼロに設定する。次いで、ステップ 332 が、fill Start および fill End の交点が同じピクセル内にあるか否かを確認すべくテストを行う。そのようである場合には、ステップ 333 および 336 が実行される。ステップ 334 が、CY を、CY の先の値に fill Start および fill End の交点の間の距離を加えた値に等しく設定する。次いで、ステップ 336 が、pending Pixel # を fill End のピクセルのピクセル番号に設定する。

【0126】

ステップ 332 において条件に合致しない場合、else 命令 338 が、ステップ 340 ~ 352 の実行を生じさせる。ステップ 340 は、変数 CY を、CY の先の値に fill Start の交点と y 線がその交点が生じているピクセルの終わりに達する場所との間の距離を加えた値に等しく設定する。ステップ 342 は、pending Pixel # を fill Start が位置しているピクセルの番号に等しく設定する。次いで、ステップ 344 ~ 348 が実行されるが、これらは上述したステップ 324 ~ 328 と同一である。ステップ 342 が、CX を、システムの 2 パスのプロセスの x パスにおいて設定された係属中のピクセルの先のピクセル被カバー値に等しく設定する。次いで、ステップ 346 が、係属中のピクセルの被カバー値を CY および CX の関数として計算し、ステップ 348 が、計算された被カバー値を係属中のピクセルに書き込む。

【0127】

ひとたびステップ 344 ~ 348 が完了すると、ステップ 350 が、CY を、現在の y

10

20

30

40

50

線が `fillEnd` のピクセルの開始に出会う場所と `fillEnd` の交点の位置との間の距離に等しく設定する。最後に、ステップ 352 が `pendingPixel#` を `fillEnd` のピクセルの番号に設定する。

【0128】

図 36 の `YLineLoop` を検討することによって、図 36 のステップ 332 の条件が、 y 線の独立した連続被カバー部分を開始および終了させている交点と同じピクセル内で生じている場合に満足されることを見て取ることができる。そのような場合が、図 43 に示した交点 243C および 243D によって説明される。これら 2 つの y 交点が生じているピクセルにおいて、 CY は、図 43 に示した距離 380 に等しい。このピクセルにおいて他に y 線の被カバー部分は存在しないため、 CY のこの値が、このピクセルについて 2 パスのプロセスの x パスにおいて計算した先の被カバー値に相当する値 CX と連動させてピクセルの被カバー値を計算するために、後の `YLineLoop312` の繰り返しのにおいて、ステップ 322 ~ 330 によって使用される。

【0129】

図 43 に示した 2 つの y 線の交点 243E および 243F は、図 36 の `else` ステップ 338 が、ステップ 340 ~ 352 の実行を生じさせる場合を説明している。この場合には、ステップ 340 が、`fillStart` 交点 243E の位置とこの交点のピクセル 110D の終わりとの間の距離 382 が、このピクセルの被カバー値を計算する目的において CY として使用されるようにする。

【0130】

ステップ 350 は、`fillEnd` の交点 243F が生じているピクセル 110F の被カバー値を計算するために使用される CY を、図 43 に示されており、`fillEnd` の交点とピクセル 110F の開始との間を延びている距離 384 に等しくさせる。

【0131】

`fillStart` および `fillEnd` の交点の所与の組について、ステップ 322 ~ 352 が完了すると、ステップ 354 が、現在の y 線のリンク・リストにおいて現在の `fillEnd` によって示されている交点の後ろにさらなる交点が存在するか否かを確認すべくテストを行う。存在する場合、ステップ 356 および 358 が実行される。ステップ 356 が、`fillStart` を現在の y 線のリンク・リストの次の交点へと向け、ステップ 358 が、`fillEnd` をそのリストの次の交点へと向ける。ステップ 354 のテストによって、現在の y 線のリンク・リストに後続の交点が存在しないと判断される場合には、`else` 命令 360 が、ステップ 362 に `YLineLoop312` からの離脱を実行させる。

【0132】

ステップ 360 が `YLineLoop` からの離脱を行うとき、ステップ 364 が、現時点において `pendingPixel#` が非 `NULL` で或るか否か、すなわち描画される文字のビットマップのピクセルに関連付けられた数を有しているか否かを確認すべくテストを行う。`pendingPixel#` がそのようなピクセルに関連付けられている場合、ステップ 364 は、ステップ 366 ~ 370 の実行を生じさせる。

【0133】

ステップ 366 は、変数 CX を、係属中のピクセルの先の被カバー値またはグレースケール値へと設定する。次いで、ステップ 368 が、係属中のピクセルの新たな被カバー値を、図 37 ~ 40 に示したような非線形関数を使用して、 CY および CX の両者の関数として計算する。次のステップ 370 が、計算されたピクセル被カバー値を係属中のピクセルへと書き込む。

【0134】

図 44 ~ 51 が、図 41 および 43 に示した大文字「B」のピクセル被カバー値の計算のために図 39 の 2 択のアルゴリズムが使用される場合について、図 33 ~ 36 に関して上述した 2 パスの手順の動作を説明するうえで役に立つ。

【0135】

10

20

30

40

50

ここで説明される本発明の実施形態において、ピクセルおよび線の被カバー値はすべて、ゼロ～126の間の範囲にある。したがって、図44、46、48、および50において、値126は、完全にカバーされていることを表わし、値ゼロは、カバーされていないことを表わしている。この実施形態において、126よりも小さい値は半透明を表わし、ゼロが完全な透明を表わしている。これは、文字の間隔が狭い場合に被カバー値の組み合わせが可能であるよう、隣接する文字について計算されたビットマップの縁の相互の重ね合わせを可能にしている。また、文字の画像を背景色または背景画像の上へと描くことを可能にしている。

【0136】

図44は、大文字「B」について、図33に示したループ202の最初のパス、すなわちxパスによって計算されたピクセル被カバー値を示している。これらのピクセル被カバー値は、各ピクセルのx線の被カバー値に等しくなる。

10

【0137】

図45は、図44に示した数値的なビットマップ値を近似するために種々のグレースケールの濃淡を使用している。これは、xパスの後のビットマップに含まれている情報について、より視覚的に理解しやすい表現を提供するために行われている。

【0138】

図46は、図43に示され、y線の交点を有している各ピクセルに関連付けられたy線の被カバー値を表わしている。2パスの手順の第2のパス、すなわちyパスにおいては、このようなy交点が存在するピクセルについてのみ、y線の被カバー値が計算される。これは、演算を少なくし、描画プロセスを高速にするために行われる。現在の実施形態においては、各ピクセルに示されているデータが、そのピクセルの処理の間のみ保存され、ストレージの要件を軽くしている。したがって、この図に示したデータが、何らかのデータ構造に一度に存在するわけではない。

20

【0139】

図47は、図46に示した数値データについて、グレースケールの濃淡を使用して図式的表示を提供している。

【0140】

図48は、図46に示したy線の被カバー値のうちで、より中間的な値を有しているもの、すなわち図44に示した同じピクセルのx線の被カバー値よりも63に近いy線の被カバー値を示している。図39に示した式によれば、これらのより中間的なy線の被カバー値のみが、描画される文字のピクセルの被カバー値としてビットマップに書き込まれる。

30

【0141】

図49は、この数値データの図式的表現である。

【0142】

図50は、2パス・プロセスの完了時の、図48に示したより中間的なy線の被カバー値が書き込まれた後の図44のビットマップを示している。

【0143】

図51は、図50に示した数値データの図式的表示であり、2パス・プロセスの完了時の大文字「B」のアンチエイリアス済みのビットマップを示している。

40

【0144】

図52は、考えられるxおよびyの被カバー値CxおよびCyの範囲に鑑み、所与のピクセルについて図39に示した2択の式に従って計算されるピクセル被カバー値を表わしている表である。

【0145】

この表においては、x線の被カバー値Cxが、表の上部を横切って挙げられており、y線の被カバー値Cyが、表の左側を下方へと延びる列に挙げられている。この表においては、被カバー値の一部のみが使用されている。本発明の現在の実施形態においては、x線、y線、およびピクセルの被カバー値が、0～126の間の偶数に限定されているが、中

50

間的な被カバー値を表示するという目的のため、63という被カバー値がこの表に示されている。当然ながら、他の実施形態においては、他の数字付けの仕組みも使用可能である。

【0146】

図53は、そのエントリ値がスプレッド・シートを使用することによって図52の表から導出された表である。図53の表は、図52の表と同じ行および列の定義を有しているが、図53の表のエントリは、2つの部分比の間の全体の比を表わしている。全体の比の分子である2つの部分比のうちの第1は、ピクセルの被カバー値の変化の割合 A （ここで、「 A 」はカバーされている面積を表わしている）を、 x 線の被カバー値の変化の割合 CX で除算したものである。全体の比の分母である第2の部分比は、ピクセルの被カバー値の変化の割合 A を、 y 線の被カバー値の変化の割合 CY で除算したものである。

10

【0147】

この表から見て取ることができるように、その領域の大部分において、ピクセルの被カバー値は、サンプリング線の被カバー値のうちで最も中間的な値を有しており、すなわち表において CX および CY の値がどちらも63であるエントリにより近いサンプリング線の被カバー値（ x または y 線の被カバー値）の変化に伴って、より大きく変化する。表の符号602が付された部分では、 y 線の値 CY が、 x 線の値 CX よりもより中間的な値を有している。表のこの領域においては、図39の2択の式によれば、 CX （この領域においては、 CY よりも極端である）が変化しても被カバー値 A が全く変化しないため、全体の比は0である。したがって、全体の比の分子がゼロである。

20

【0148】

同様に、図53に示した領域604においては、表の計算に使用したスプレッド・シートが生成する「#DIV0!」というエラー・メッセージによって示されているように、全体の比が無限大の値を有する。これは、この領域においては CX が CY よりも中間的であるがゆえ、この領域においては CY の値が変化してもピクセルの被カバー値 A が全く変化しないため、この領域604においては全体の比の分母がゼロであるためである。

【0149】

図53と同様の表が、図37および38に示した式について形成されるならば、それらも、それらの関連の表の領域の大部分を覆う領域であって、ピクセルの被カバー値がより中間的な値を有する線の被カバー値 CX または CY に関連してより迅速に変化する領域602および604とおおむね同様の形状を有する領域を有するであろう。同じことが、図40の方法によって使用される参照テーブルによって決定されるピクセル被カバー値の変化の割合について計算される同様の表についてもおおむね真である。

30

【0150】

図54～58は、本発明において使用することができるサンプリング線の他の多数の配置のうちのいくつかを示している。

【0151】

図54においては、ピクセルに、2つの水平な x サンプリング線 x_1 および x_2 、ならびに2つの垂直な y サンプリング線 y_1 および y_2 が組み合わされている。この配置においては、サンプリング線が、各ピクセルの境界において生じている。

40

【0152】

図55では、ピクセルが、2つの水平な x サンプリング線 x_1 および x_2 、ならびに2つの垂直な y サンプリング線 y_1 および y_2 を有している。しかしながら、この場合には、サンプリング線が、それぞれ水平および垂直方向にピクセルを横切る途中のほぼ $1/4$ および $3/4$ の位置に生じている。

【0153】

図56においては、図13～32に関して検討した x および y のサンプリング線と同様に、ピクセル当たり2つのサンプリング線しか存在していないが、図56においては、それぞれのサンプリング線がピクセルの境界に対して45度回転させられている。結果として、+45と印されている1つのサンプリング線が、プラス1の傾斜を有しており、-4

50

5 と印されている 1 つのサンプリング線が、マイナス 1 の傾斜を有している。

【 0 1 5 4 】

図 5 7 においては、ピクセルが 4 つのサンプリング線を有しており、そのうちの + 4 5 A および + 4 5 B と印されている 2 つが、プラス 1 の傾斜を有しており、 - 4 5 A および - 4 5 B と印されている 2 つが、マイナス 1 の傾斜を有している。

【 0 1 5 5 】

最後に、図 5 8 は、各ピクセルが 4 つのサンプリング線を有している本発明の実施形態を示している。すなわち、x と印されている水平な中央 x 線、y と印されている垂直な中央 y 線、ならびに 1 つが + 4 5 と印されて + 1 の傾斜を有しており、1 つが - 4 5 と印されて - 1 の傾斜を有している 2 つの中央線である。この 4 つの中央線からなる組み合わせは、図 1 3 ~ 3 2 に示したようにただ 1 つの中央の x および y 線を使用するよりも、より正確なピクセルの被カバー情報をもたらすと考えられる。しかしながら、本発明の発明者は、図 1 3 ~ 3 2 のようにただ 1 つの水平な中央線およびただ 1 つの垂直な中央線を使用することで、通常は充分に良好な結果がもたらされ、そのようにわずか 2 つの線を使用することで、演算が大幅に少なくなることを確認済みである。

【 0 1 5 6 】

図 5 9 は、図 5 4 ~ 5 8 に示したサンプリング線の配置のすべてにおいて使用することができる式 3 8 と同様の式を示している。この式においては、2 つまたは 4 つのサンプリング線のそれぞれに、式中の番号 L 1 ~ L 4 が与えられる。そのような線のそれぞれについて計算される被カバー値は、それぞれ C L 1 ~ C L 4 と称される。サンプリング線が 2 本しか存在しない場合には、L 3 および L 4 に関する項は削除される。図 1 3 ~ 3 2 に示したピクセルの場合には、ひとたび L 3 および L 4 に関する項が削除されると、図 5 9 の式は図 3 8 の式に一致する。

【 0 1 5 7 】

図 3 7 に示した式など、ピクセルの被カバー値をピクセルの種々のサンプリング線の被カバー値の非線形関数として計算するための他の式も、図 5 4 ~ 5 8 に示した形式のサンプリング線を有するピクセルを取り扱うように調整できることを、当業者であれば理解できるであろう。

【 0 1 5 8 】

図 4 0 は、参照テーブルからピクセルの被カバー値を決定するための手順 6 5 0 を説明している。この手順は、C X および C Y の組み合わせから参照アドレスを形成するステップ 6 5 2 を含んでいる。次いで、ステップ 6 5 4 が、その参照アドレスを使用して参照テーブルから該当するピクセル被カバー値を読み出す。

【 0 1 5 9 】

そのような表を保存するために必要とされるメモリを少なくするため、C X および C Y について考えられるすべての値の範囲、すなわち 0 ~ 1 2 6 の値を、それぞれが 0 ~ 7 の範囲にわたる 3 ビットの 2 進数によって表わされる 8 つの実質的に等間隔の領域へと写像することができる。表中の各エントリのアドレスを、2 つのそのような 3 ビット数 (1 つが C X の値についてであって、1 つが C Y の値についてである) の連続として表わすことができる。

【 0 1 6 0 】

他の実施形態においては、他のサイズの参照テーブルおよび他の形式のアドレス指定の仕組みを使用することができる。例えば、図 5 4、5 5、5 7、および 5 8 のように各ピクセルが 4 つのサンプリング線を有している場合には、所与のピクセル被カバー値を表わすために使用されるアドレスを、それぞれがピクセルの 4 つのサンプリング線の個々の 1 つの被カバー値を表わしている 4 つのそのような 3 ビット数から連結させることができる。

【 0 1 6 1 】

図 6 0 は、そのような参照テーブルの学習のための方法 7 0 0 を示している。

【 0 1 6 2 】

10

20

30

40

50

方法 700 は、参照テーブルの学習に使用される多数の異なる文字のそれぞれのピクセル・レンダリングまたはスキャン変換のためのループ 702 の実行を含んでいる。このループ 702 において、ピクセル画像へとスキャン変換されるそれぞれの文字について、ステップ 704 が、現在の文字のビットマップの各ピクセル行についてループを実行する。ループ 704 は、ループ 704 の現在のピクセル行の各ピクセルについて実行される内側ループ 706 を含んでいる。

【0163】

内側ループ 706 は、ステップ 708 ~ 716 を実行する。ステップ 708 は、図 7 または 8 に関して上述した方法などを使用して、現在のピクセルの被カバー値について比較的正確な計算を行う。次いで、ステップ 710 が、ピクセルが境界ピクセルであるか否か、すなわち完全な被カバーまたは非カバーを表わす被カバー値以外の被カバー値を有しているか否かを確認するため、テストを行う。

10

【0164】

現在のピクセルが境界ピクセルである場合、ステップ 712 ~ 716 が実行される。ステップ 712 は、現在のピクセルについてサンプリング線の被カバー値 C_X および C_Y を計算する。次いで、ステップ 714 が、ステップ 708 で計算された正確なピクセル被カバー値を、学習参照テーブルにおいて、 C_X および C_Y の上記値によって表わされる数の連結によって形成されるアドレスの要素へと加える。最後に、ステップ 716 が、実行済みの学習事例の数を記録するために、`sampleCount` 変数を増加させる。

【0165】

20

ひとたび学習対象のすべての文字のスキャン変換についてステップ 702 のループが実行されると、ステップ 718 が、学習テーブルの各要素を `sampleCount` で除算し、テーブルに保存されているピクセル被カバー値を正規化する。ひとたびこれがなされると、参照テーブルは、図 40 の式によって使用できる状態である。

【0166】

図 61 は、図 54、55、57、および 58 に関して上述したようなピクセル当たり 3 つ以上のサンプリング線を有しているスキャン変換の仕組みにおいて使用するための参照テーブルをどのように学習するのかを示している。とくに、図 59 が、図 58 に示した 4 つのサンプリング線からなる星形の配置において使用するための参照テーブルの学習のための方法 700A を示している。

30

【0167】

図 61 の方法 700A は、図 60 のステップ 712 および 714 の代わりに、ステップ 712A および 714A を使用する点を除き、図 60 の方法 700 と同一である。ステップ 712A は、図 60 のステップ 712 において行われるように x および y のサンプリング線についてだけでなく、図 58 に示した +45 および -45 度の線についても、ピクセルの被カバー値を計算する。ステップ 714A は、 C_X および C_Y を表わす数字だけでなく、+45 および -45 の線の被カバー値を表わす数字を含めた連結にて、学習テーブルのアドレスを指定する点で、図 60 のステップ 714 と相違している。

【0168】

そのような学習テーブルが、事実上あらゆる種類のサンプリング線の仕組みについて容易に生成可能であることを、当業者であれば理解できるであろう。

40

【0169】

図 62 および 63 は、ピクセルの被カバー値を見つけるために参照テーブルを使用する他の方法であって、異なるフォント・セットのために別個の参照テーブルが使用される方法に関係している。

【0170】

図 63 は、図 60 の学習方法 700 に類似する学習方法 700B を示している。わずかな相違は、新たなステップ 720 および 722 の追加、ならびにステップ 714B、716B、および 718B の示される変更にある。これらの相違はすべて、異なるフォント・セットのための異なる参照テーブルの学習を可能にするように設計されている。

50

【 0 1 7 1 】

新たなステップ 7 2 0 は、方法 7 0 0 B の開始において実行される。ステップ 7 2 0 は、異なる参照テーブルを生成すべき異なるフォント・セットを定義する。このステップは、多くの場合、人間の選択を用いることによって実行されると考えられる。この方法の別の実施形態においては、フォント・セットを異なって定めることができる。いくつかの実施形態においては、異なるフォント・セットが、異なるフォントのサイズを代表している。他には、異なるフォント・セットが、異なるフォントの様式、または異なるフォントの様式およびサイズの組み合わせを代表している。

【 0 1 7 2 】

ひとたび異なるフォント・セットが定められると、方法 7 0 0 B は、ステップ 7 1 4 B 、 7 1 6 B 、 7 2 2 、 および 7 1 8 B が実行されるまで、方法 7 0 0 の動作と同様の様相で進行する。

【 0 1 7 3 】

ステップ 7 1 4 B は、現在のピクセルについて正確に計算されたピクセル被カバー値を、現在の文字のフォントが属するフォント・セットについて学習を行っている特定の参照テーブルの該当の要素へと加える点を除き、図 6 0 のステップ 7 1 4 に類似している。

【 0 1 7 4 】

ステップ 7 1 6 B は、現在の文字のフォント・セットに関連付けられている別個の `sampleCount` 変数を増加させる点を除き、図 6 0 のステップ 7 1 6 に類似している。

【 0 1 7 5 】

ひとたびステップ 7 0 2 の学習ループのすべてが方法 7 0 0 B によって実行されると、ステップ 7 2 2 が、それぞれのフォント・セットについてループを実行する。そのような繰り返しのそれぞれにおいて、ステップ 7 1 8 B が、そのループの該当のフォント・セットについて生成される個々の学習テーブルについて実行される。ステップ 7 1 8 B は、学習テーブルのすべての要素を `sampleCount` で除算するという機能を、現在のフォント・セットに関連する学習テーブルについて現在のフォント・セットに関連する `sampleCount` にて行う点を除き、図 6 0 のステップ 7 1 8 に類似している。

【 0 1 7 6 】

図 6 3 は、ピクセルの被カバー値を決定するために、異なるフォント・セットについて生成された別個の参照テーブルのグループを使用する点を除き、図 4 0 の手順 6 5 0 と同様の手順 6 5 0 B を示している。そのステップ 6 5 2 B は、現在のピクセルのサンプリング線のカバー値を表わしている数字から、現在の参照アドレスを形成する。次いで、ステップ 6 5 4 B が、描画されようとしている現在の文字のフォント・セットに関連付けられた参照テーブルにおいて、上記参照アドレスに保存されているピクセル被カバー値を取得する。

【 0 1 7 7 】

本発明についての上述の説明および図面が、あくまで本発明の説明および例示のために提示されたものであり、添付の特許請求の範囲の解釈によるものを除き、本発明が上述の説明および図面に限定されるわけではないことを、理解すべきである。以上の開示を目にした当業者であれば、特許請求の範囲の技術的範囲から離れることなく、それらに変更または変形を加えることが可能である。

【 0 1 7 8 】

とくには、この出願が、多くの特許出願において一般的であるよりもさらに詳しく本発明を説明しており、本発明の発明者が、本発明についてさらに詳細な教示を公衆に提示したことで、特許請求の範囲がそのようなさらに詳細な教示へと限定されてしまうことによって不等に不利益を被ることがないと確信していることに、注意すべきである。発明者が何を自身の発明であると考えているかについて正確な記載がもたらされるよう、以下の特許請求の範囲の表現には、かなりの考慮が払われており、そのような表現が、どのような細部がまさに特定の請求項に記載されている発明の一部分であると考えられ、どのような

10

20

30

40

50

細部がまさに特定の請求項に記載されている発明の一部分であると考えられないかについて、最も正確な記載として受け取られるように希望される。

【0179】

コンピュータの技術分野の当業者であれば理解できるとおり、ソフトウェアにて実行されるものとして上述された機能の多くは、ハードウェアにて実行することが可能である。例えば、上述され特許請求の範囲に後述されるスキャン変換機能の多く、あるいはすべてを実行するために、ASICまたはマイクロプロセッサなど、専用目的のハードウェアを集積回路に配置することが可能である。

【0180】

同様に、コンピュータの技術分野の当業者であれば、ソフトウェアがきわめて柔軟な技術形式であって、大部分の機能が事実上無数の方法で実行可能であることを、理解できるであろう。したがって、本発明の発明者は、特許請求の範囲を、上述した特定のソフトウェアの構成、順序、および構造に限定しようとするものではない。なぜならば、当業者であれば、きわだって異なる構成、順序、および構造を有するプログラミングを使用して、特許請求の範囲の教示を容易に実行できるからである。同様に、すでに示したように、上述したもの以外の多数の種類のサンプリング線の配置ならびに多数の種類のピクセルの被カバーの計算を、本発明の多数の態様において使用することが可能である。

【0181】

以上の検討の大部分は、ピクセルが直交する行および列に並べられているピクセル画像の描画に集中しているが、本発明は、印刷画像の描画においてときに使用されるような非直交の行および列においても使用可能である。

【0182】

また、図13および54～58に示したサンプリング線の配置が、あくまで本発明において使用できると考えられるサンプリング線の考えられる多数の配置のうちのいくつかを表わしているに過ぎないことを、理解すべきである。そのような可能性のあくまで一例として、通常はアンチエイリアシングの精度を低下させてしまうと考えられるが、図54に示したサンプリング線の配置に類似した配置であって、2つの水平なサンプリング線および2つの垂直なサンプリング線が、それらの助けによって被カバー値が計算されようとしているピクセルによって表わされる実際の領域のわずかに外側に位置している配置が可能である。そのような場合、そのようなサンプリング線は、特許請求の範囲の目的においては、そのピクセルの「サンプリング領域」内にあると考えられる。

【0183】

また、本発明の他の実施形態において、ピクセルの被カバー値を、ピクセルのサンプリング線の被カバー値だけでない他の情報の関数として計算することが可能であることを、理解すべきである。例えば、図62および63に関して上述したように、文字のフォントを、そのサイズも含め、ピクセルの被カバー値の決定のもととなる追加の情報として使用することが可能である。他の実施形態においては、隣接ピクセルの被カバー値および/またはピクセル内でのサンプリング線の被カバーの位置など、他の多数の種類の情報を使用することが可能である。

【0184】

さらに、ピクセル内を種々の方向に延びている線の被カバー値の関数としてピクセル値を決定するために使用される非線形関数が、シグモイド関数ならびにニューラル・ネットワークまたは他の自動学習の技法によって学習される関数など、他の種類の非線形関数であってよいことを、理解すべきである。

【0185】

また、本発明の特許請求の範囲が、2次元の形状の描画に限定されないことを、理解すべきである。上述した特定の方法を、3次元以上で記述されている高解像度の形状の描画を包含するように拡張できることを、当業者であれば理解できるであろう。

【0186】

さらに、本発明を、線または領域の所与の部分が描画しようとする形状の内側または外

10

20

30

40

50

側のどちらであるかを把握するためにワインディングカウント以外の技法を使用するスキャン変換の仕組みにおいても使用できることを、理解すべきである。同様に、リンク・リスト以外の他のプログラミング構造も、所与のサンプリング線に関する交点の表現を保存するために使用することができる。

【0187】

本発明は、図11に示したコンピュータのほかにも、多数のさまざまな形式のコンピュータにて実行可能である。いくつか挙げるならば、複数のCPUを有するコンピュータや、スキャン変換プロセスの演算の実行を助けるために使用できる別途のグラフィック、ビデオ、またはDSPプロセッサを備えているコンピュータにて、本発明を使用することが可能である。いわゆるネットワーク・コンピュータまたはセットトップ・コンピュータなど、ハードディスクを持たないコンピュータにおいて使用することも可能である。これらに限られるわけではないが電子ブック、パーム・コンピュータ、ウェアラブル・コンピュータ、および電子広告板など、より高解像度の形状のピクセル画像を描画するために使用される任意の形式のコンピュータにおいて使用可能である。

【図面の簡単な説明】

【0188】

【図1】 この場合にはアウトラインフォントによって定められるテキストの一部分の形状であるが、一連の形状の比較的高解像度の描写を示している。

【図2】 より低い解像度のピクセル画像で描いた場合の同じ形状を示している。

【図3】 図2に示したビットマップの一部の拡大であり、このビットマップの個々のピクセルを、それが描く文字のアウトラインをピクセル上に重ねつつ示している。

【図4】 文字のアウトラインが破線で示され、それらアウトラインがビットマップの実際の画像に重ねられている点を除き、図3と同様である。

【図5】 アンチエイリアス処理の後のビットマップ画像を示している点を除き、図4と同様である。

【図6】 アンチエイリアス処理後のビットマップ画像を図5に示したピクセル格子の妨げなく示している点を除き、図5と同様である。

【図7】 高解像度の形状によって部分的に覆われているビットマップの個々のピクセルを示している。

【図8～10】 図7のピクセルなどのピクセルについて、描こうとしている1つ以上の形状に覆われている割合の計算を高速化するために従来技術において使用されている近似方法の説明を容易にするために使用される。

【図11】 本発明を実行できる多数の考えられるコンピュータ・システムのうちの1つを示している。

【図12】 本発明においてピクセル画像へとピクセル被カバー値を割り当てる方法の1つについて、高度に概略化した擬似コードの記述である。

【図13～32】 本発明の一態様に従ってピクセルの被カバー値を計算するために、中央の水平サンプリング線および中央の垂直サンプリング線の被カバー値をどのように使用できるのかを説明するために使用される。

【図33～36】 個々の文字の形状がアウトラインフォントを使用して定められているテキストのピクセル画像をレンダリングする2パスの方法について、高度に概略化した擬似コードの記述である。

【図37～40】 ピクセル画像の適切なピクセル被カバー値をピクセルの垂直および水平の両方のサンプリング線の被カバー値の非線形関数として決定するために、図33～36の2パスの方法および他のピクセル設定の仕組みによって使用できる式または方法を示している。

【図41】 文字について生成されるビットマップ画像のそれぞれの水平および垂直サンプリング線について、図35のアルゴリズムによってどのように交点リストが用意されるのかを説明するうえで役に立つ。

【図42】 文字について生成されたビットマップ画像をどのようにメモリに保存できる

10

20

30

40

50

のかを説明している。

【図 4 3】 ピクセルの y 線の被カバー値を図 3 6 のアルゴリズムに従ってどのように計算できるのかを説明するうえで役に立つ。

【図 4 4 ~ 5 1】 図 3 9 の 2 択のピクセル設定の式が使用される場合について、図 3 3 ~ 3 6 の 2 パスの方法の動作を説明するうえで役に立つ。

【図 5 2】 種々の x および y 線の被カバー値を有するピクセルについて、図 3 9 の式に従って計算されたピクセル被カバー値の表である。

【図 5 3】 図 3 9 において使用される式の領域の大部分において、ピクセル被カバー値が、2 つのサンプリング線 x または y のうちのもっとも中間的なサンプリング値を有している方の値とともにより高速に変化することを説明するために使用される表である。

10

【図 5 4 ~ 5 8】 本発明において使用することができる他の考えられる多数のサンプリング線の配置のうちのいくつかを示している。

【図 5 9】 図 5 4 ~ 5 8 に示したようなサンプリング線の配置からピクセル被カバー値を計算するために使用できる多数の非線形関数のうちの 1 つを示している。

【図 6 0 ~ 6 2】 ピクセル被カバー値の決定において使用するための参照テーブルの学習のための方法を説明している。

【図 6 3】 図 6 2 の方法によって学習された参照テーブルにおいて使用できるピクセル被カバー値の決定方法を説明している。

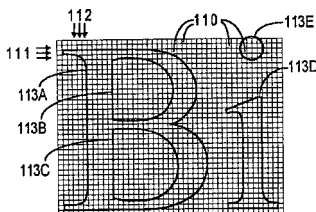
【図 1】

Bitstream¹⁰² ← 100

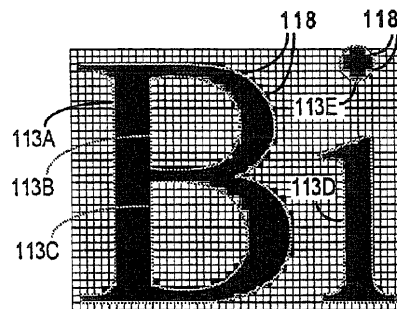
【図 2】

Bitstream¹⁰² ← 100

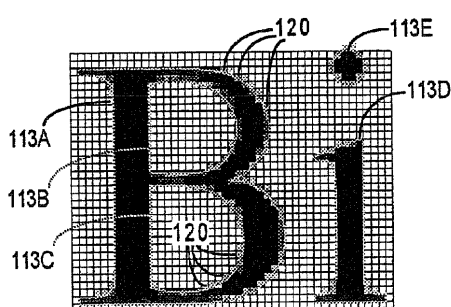
【図 3】



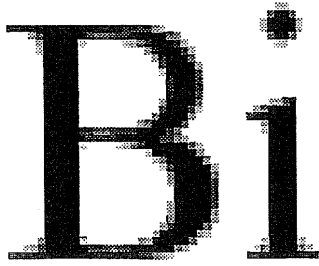
【図 4】



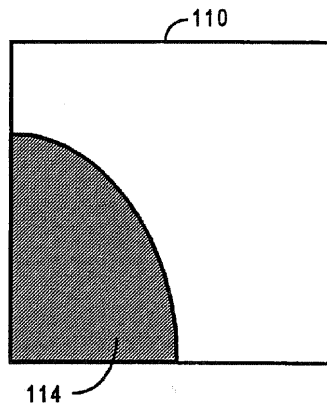
【図 5】



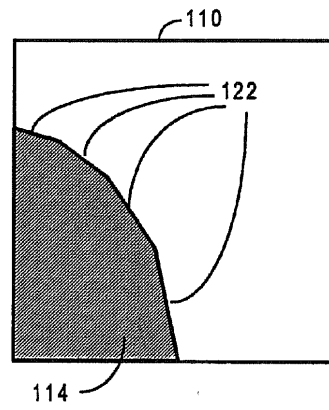
【図 6】



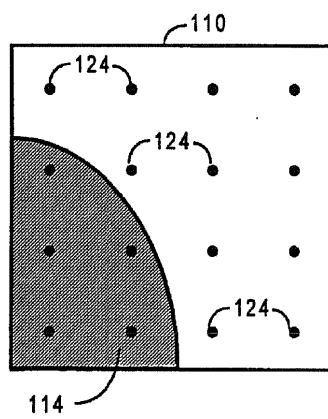
【図 7】



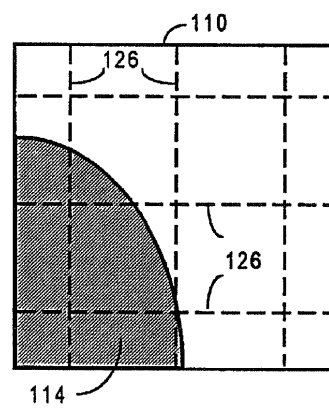
【図 8】



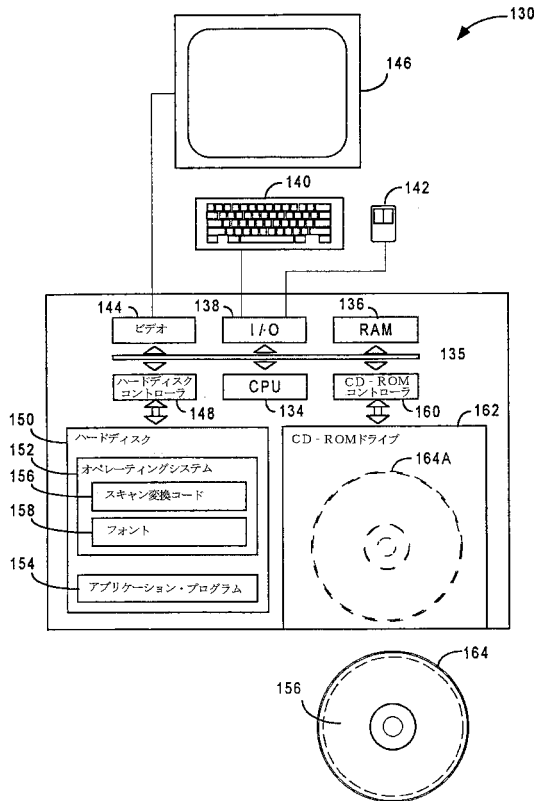
【図 9】



【図 10】



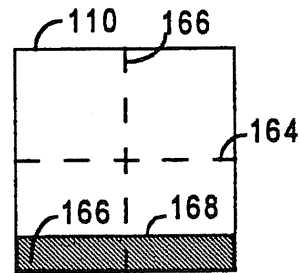
【図 1 1】



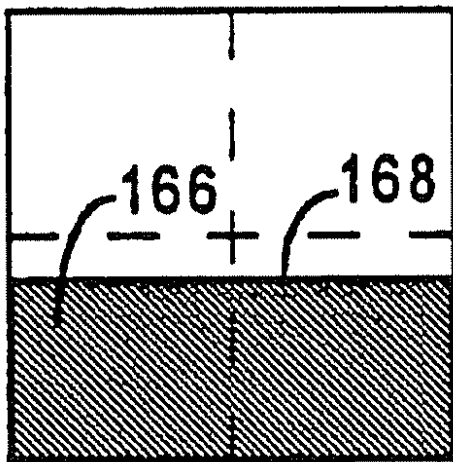
【図 1 2】

- ピクセル被カバー値の計算方法～188
- 各境界ピクセルについて～190
- ピクセルの第1の方向の1つ以上の線についてカバーの程度を計算～192
- ピクセルの第2の方向の1つ以上の線についてカバーの程度を計算～194
- 各線のカバーの程度の非線形関数として、ピクセルのグレースケールを計算（計算によって得られるグレースケール値への所号の線の被カバー値の影響は、被カバー値の関数として変化する）～196

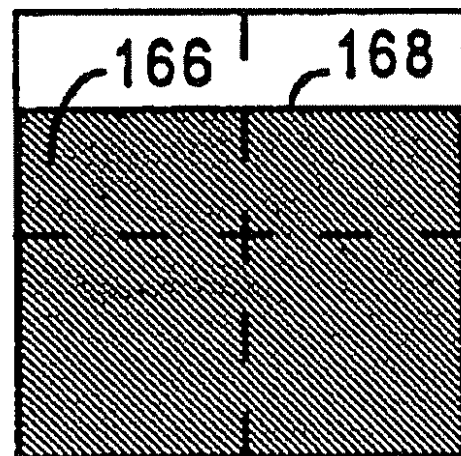
【図 1 3】



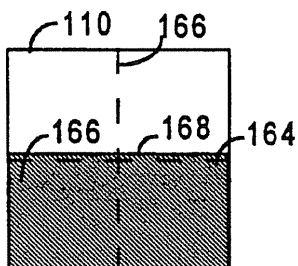
【図 1 4】



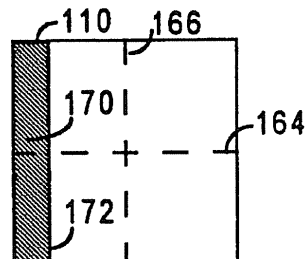
【図 1 6】



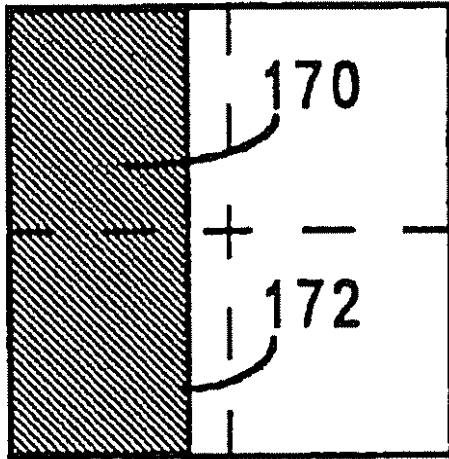
【図 1 5】



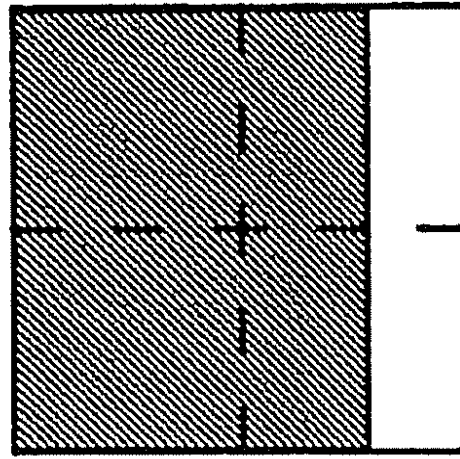
【図 1 7】



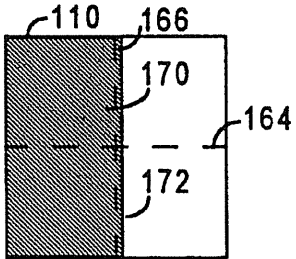
【図 18】



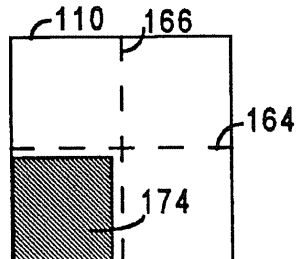
【図 20】



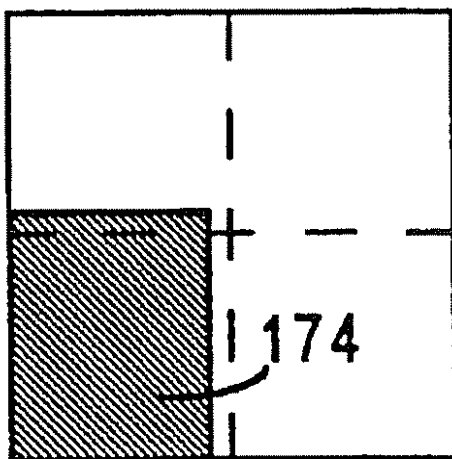
【図 19】



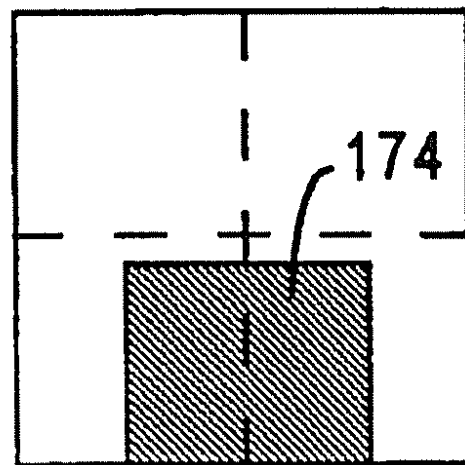
【図 21】



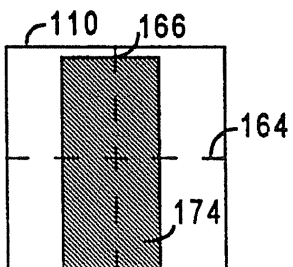
【図 22】



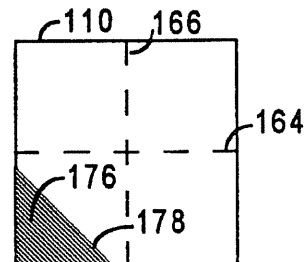
【図 24】



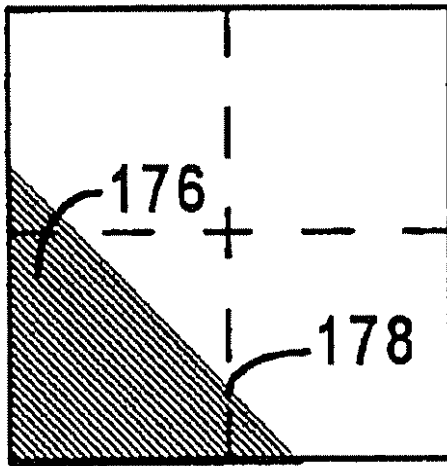
【図 23】



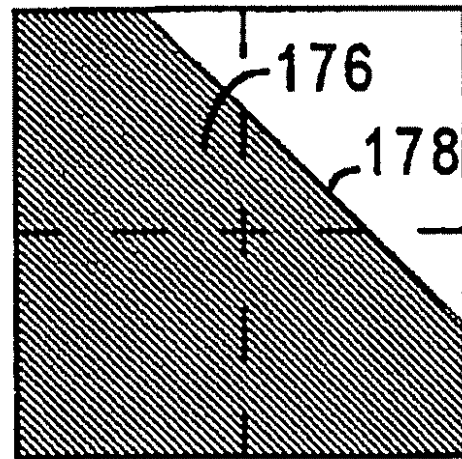
【図 25】



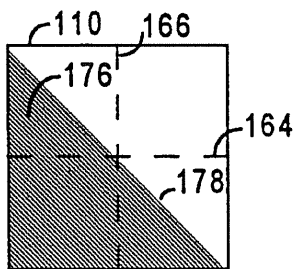
【図 26】



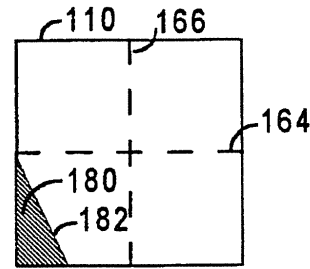
【図 28】



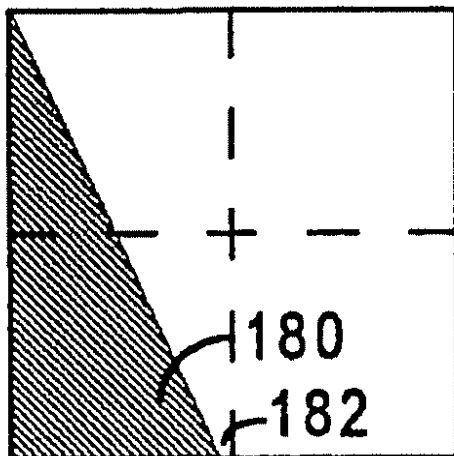
【図 27】



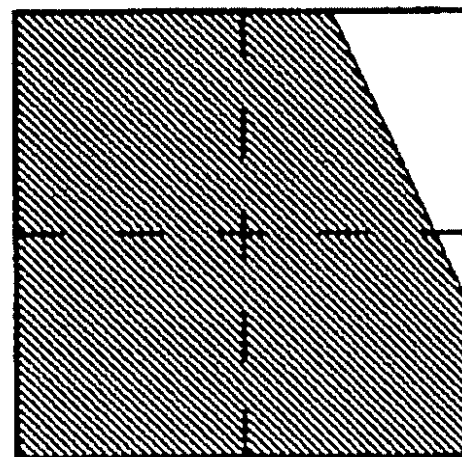
【図 29】



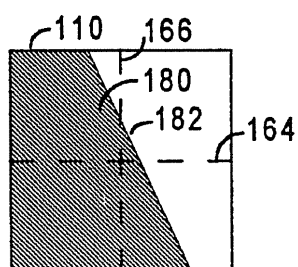
【図 30】



【図 32】



【図 31】



【図 33】

- drawText～200
 - 描こうとするビットマップの各文字について、～202
 - CharSetUpを呼び出す～204
 - 文字の各XLineについて～206
 - fillStartをXLineのリンク・リストの最初の交点に向ける～208
 - fillEndをXLineのリンク・リストの2番目の交点に向ける～210
 - XLinePassを呼び出す～212
 - 文字の各YLineについて～214
 - fillStartをYLineのリンク・リストの最初の交点に向ける～216
 - fillEndをYLineのリンク・リストの2番目の交点に向ける～218
 - YLinePassを呼び出す～220
 - 文字のビットマップ値を対応するスクリーン位置に加え、さらに/またはそれをビットマップ・キャッシュに保存する～222

【図 3 4】

- charSetUp～224
 - 文字のアウトラインのそれぞれについて～226
 - アウトラインを完走するまで～230
 - 中央線との次の交点に達するまで、アウトラインを巡って前進し～232
 - 交点がXまたはYのどちらの線にあるかを判断し～234
 - 交点の表示を、その線のリンク・リストの適切な位置に配置し～236
 - 文字のX線およびY線を保持するために充分に大きいグレースケール・ビットマップを形成する～246
 - ビットマップをゼロにする～248

【図 3 5】

- XLinePass～254
 - winding countをゼロに～256
 - XLineループ～258
 - currentPixel #をfillStartの位置が位置しているピクセルの番号に設定～260
 - fillStartおよびfillEndのedgeValueをwindingCountに加算～262
 - windingCountがゼロでないとき～264
 - fillEndをリンク・リストの次の交点に設定～266
 - 新しいfillEndのedgeValueをwindingCountに加算～268
 - fillEndの位置がcurrentPixel #にある場合～270
 - fillEndおよびfillStartの位置の間の距離を、currentPixel #のビットマップ要素へと加算～272
 - そうでないならば～274
 - currentPixel #のビットマップ要素へと、fillStartの位置とcurrentPixel #のピクセルの右縁との間の距離を加算～276
 - currentPixel #を増やす～278
 - fillEndの位置がcurrentPixel #よりも大きいピクセル番号を有する場合～280
 - ビットマップのcurrentPixel #の要素のアドレスが奇数である場合～282
 - currentPixel #のビットマップ要素を126に設定～284
 - currentPixel #を1だけ増加～286
 - currentPixel #がfillEndのピクセル番号よりも小さい間～288
 - currentPixel #のビットマップ要素から出発し、126という2つの連続値をビットマップへと書き込むために、2バイト・ワードの書き込みを使用～290
 - currentPixel #を2だけ増加～292
 - fillEndのピクセルのビットマップ要素を、fillEndのピクセルの左縁とfillEndの位置との間の距離に設定～294
 - XLineのリンク・リストにおいてfillEndの後方に交点が存在する場合～296
 - fillStartを次の交点へと向け～298
 - fillEndをリンク・リストの次の交点へと向け～300
 - そうでないならば～302
 - XLineループから離脱～304

【図 3 6】

- YLinePass～306
 - winding countをゼロに～308
 - CYをゼロに～309
 - pendingPixel #をNULL値に設定～310
 - YLineループ～312
 - fillStart交点およびfillEnd交点のedgeValueをwindingCountに加算～314
 - windingCountがゼロでないとき～316
 - fillEnd交点をリンク・リストの次の交点に向け～318
 - fillEnd交点のedgeValueをwindingCountに加算～320
 - pendingPixel #が非NULLであり、fillStart交点の位置がpendingPixel #を有していない場合～322
 - CXをpendingPixel #のビットマップ要素の先のグレースケールに等しく設定～324
 - グレースケールをCYおよびCXの関数として計算～326
 - 計算したグレースケールをpendingPixel #のビットマップ要素へと書き込み～328
 - CYをゼロに設定～330
 - fillStart交点およびfillEnd交点の位置が同じピクセルにある場合～332
 - CY=CY+fillStart交点およびfillEnd交点の位置の間の距離～334
 - pendingPixel #をfillEnd交点のピクセル番号に設定～336
 - そうでない場合～338
 - CY=CY+fillStart交点の位置とfillStart交点のピクセルの端部との間の距離～340
 - pendingPixel #をfillStart交点のピクセル番号に等しく設定～342
 - CX=pendingPixel #のビットマップ要素の先のグレースケールに設定～344
 - グレースケールをCYおよびCXの関数として計算～346
 - 計算したグレースケールをpendingPixel #のビットマップ要素へと書き込み～348
 - CY=fillEnd交点のピクセルの始点とfillEnd交点の位置との間の距離～350
 - pendingPixel #をfillEnd交点のピクセル番号に設定～352
 - YLineのリンク・リストにおいてfillEnd交点の後に交点が存在する場合～354
 - fillStart交点を次の交点へと向け～356
 - fillEnd交点をリンク・リストの次の交点へと向け～358
 - そうでないならば～360
 - YLineループから離脱～362
 - pendingPixel #が非NULLである場合～364
 - CX=pendingPixel #のビットマップ要素の先のグレースケールに設定～366
 - グレースケールをCYおよびCXの関数として計算～368
 - 計算したグレースケールをpendingPixel #のビットマップ要素に書き込み～370

【図 3 7】

$$- (((3970-(63-CX)^2)*CX) + ((3970-(63-CY)^2)*CY)) / (((3970-(63-CX)^2) + (3970-(63-CY)^2))$$

【図 3 8】

$$- ((64-ABS(63-CX))*CX + (64-ABS(63-CY))*CY) / (64-ABS(63-CX) + 64-ABS(63-CY))$$

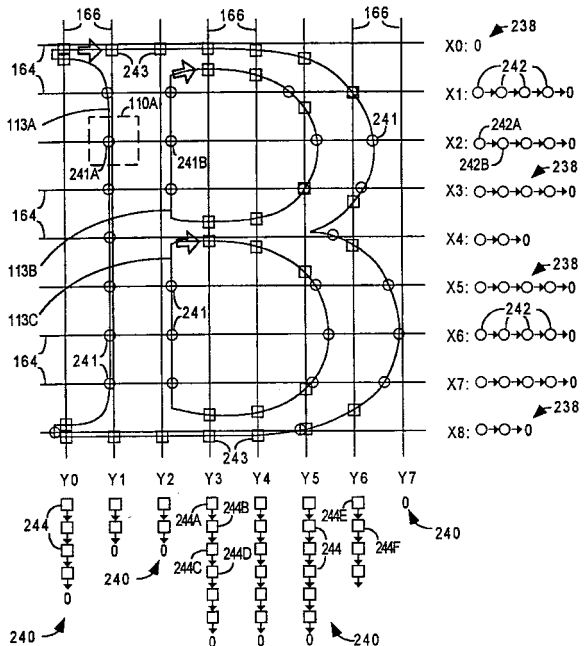
【図 3 9】

- binaryPixelCoverageValueAlgorithm
 - (CX=CY) ならば、
 - CX
 - そうでない場合、(ABS(63-CX)) > ABS(63-CY) ならば、
 - CY
 - そうでないならば、CX

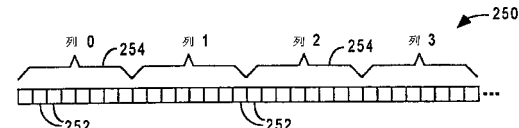
【図 4 0】

- LookUpPixelCoverageAlgorithm～650
- 現在のピクセルのCXおよびCYの値から、現在の参照アドレスを形成～652
- 現在の参照アドレスにある参照テーブルのエントリから現在のピクセルの被カバー値を取得～654

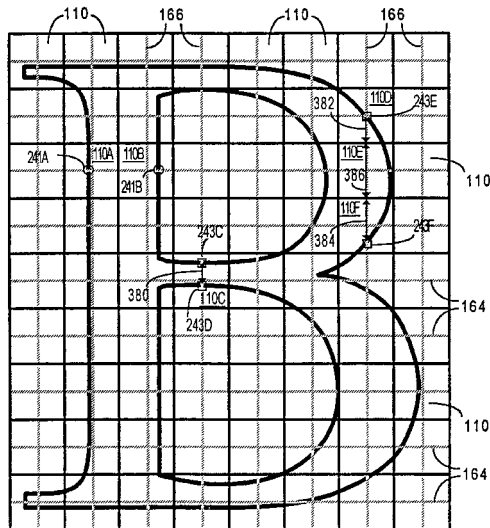
【図 4 1】



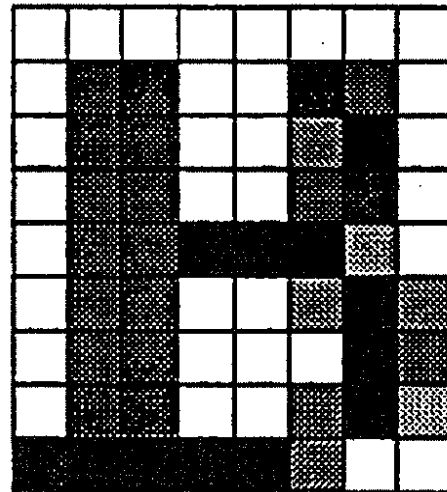
【図 4 2】



【 図 4 3 】



【 図 4 5 】



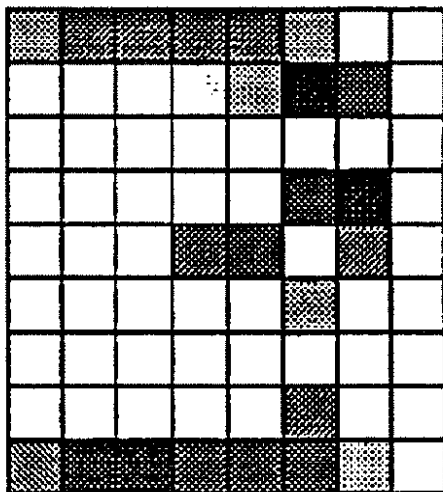
【 図 4 4 】

	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X0	0	0	0	0	0	0	0	0
X1	0	70	92	0	0	96	62	0
X2	0	70	92	0	0	32	120	0
X3	0	70	92	0	0	66	92	0
X4	0	70	126	126	126	126	24	0
X5	0	70	92	0	0	36	126	32
X6	0	70	92	0	0	0	126	58
X7	0	74	92	0	0	48	126	20
X8	96	126	126	126	126	42	0	0

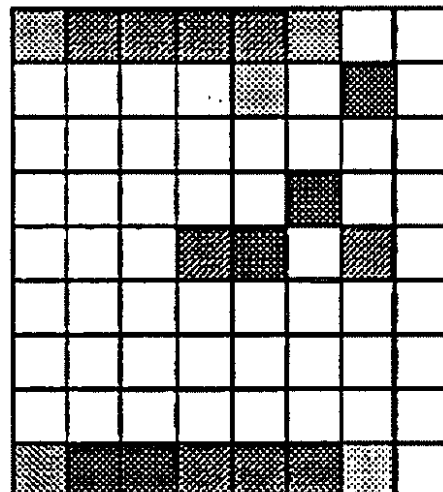
【 図 4 6 】

	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X0	26	52	52	52	46	26		
X1				4	20	102	64	
X2								
X3						64	102	
X4				54	70		46	
X5						28		
X6								
X7						46		
X8	38	76	76	48	58	58	12	

【 図 4 7 】



【 図 4 9 】



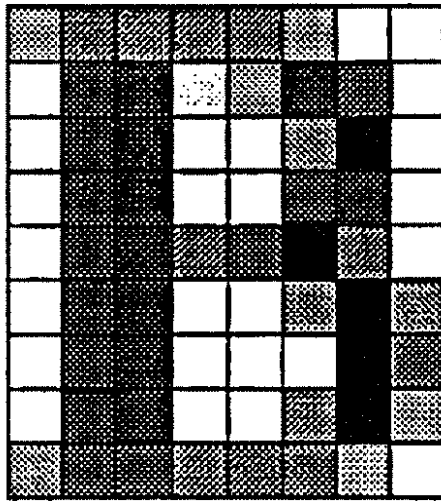
【 図 4 8 】

	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X0	26	52	52	52	46	26		
X1				4	20		64	
X2								
X3						64		
X4				54	70		46	
X5								
X6								
X7								
X8	38	76	76	48	58	58	12	

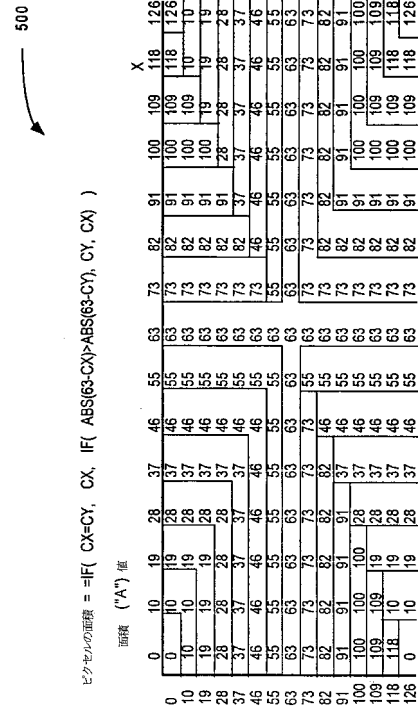
【 図 5 0 】

	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X0	26	52	52	52	46	26	0	0
X1	0	70	92	4	20	96	64	0
X2	0	70	92	0	0	32	120	0
X3	0	70	92	0	0	64	92	0
X4	0	70	126	54	70	126	46	0
X5	0	70	92	0	0	36	126	32
X6	0	70	92	0	0	0	126	58
X7	0	74	92	0	0	48	126	20
X8	38	76	76	48	58	58	12	0

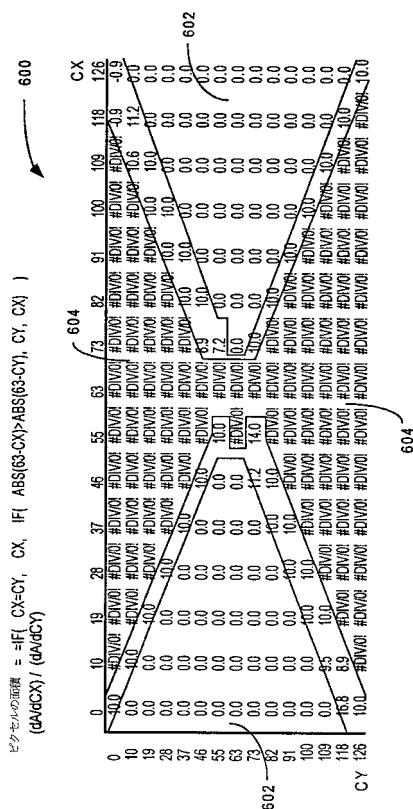
【図 5 1】



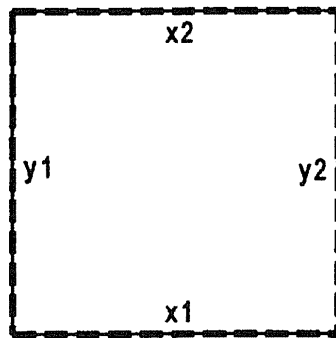
【図 5 2】



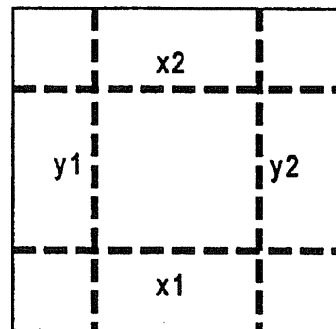
【図 5 3】



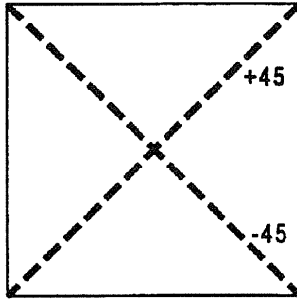
【図 5 4】



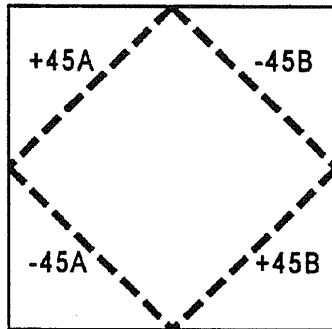
【図 5 5】



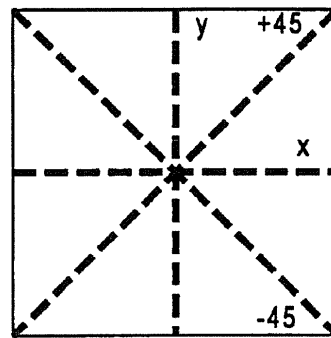
【図 56】



【図 57】



【図 58】



【図 59】

$$A = \frac{(64-ABS(63-CL1))*CL1 + (64-ABS(63-CL2))*CL2 + (64-ABS(63-CL3))*CL3 + (64-ABS(63-CL4))*CL4}{(64-ABS(63-CL1)) + (64-ABS(63-CL2)) + (64-ABS(63-CL3)) + (64-ABS(63-CL4))}$$

【図 60】

- trainLookUpTable～700
- スキャン変換すべき複数の文字のそれぞれについて～702
- スキャン変換の各行について～704
- 行の各ピクセルについて～706
- ピクセルのグレースケールの正確な計算を実行～708
- ピクセルが完全に内側または完全に外側ではない場合～710
- ピクセルのCXおよびCYの値を計算～712
- ピクセルのグレースケールを、CXおよびCYの連結によってアドレス指定される学習テーブルの要素へと追加～714
- sampleCountを増加～716
- 学習テーブルの各要素をsampleCountで除算～718

【図 61】

- trainLookUpTablesForStarPixels～700A
- スキャン変換すべき複数の文字のそれぞれについて～702
- スキャン変換の各行について～704
- 行の各ピクセルについて～706
- ピクセルのグレースケールの正確な計算を実行～708
- ピクセルが完全に内側または完全に外側ではない場合～710
- ピクセルのCX、CY、C+45、およびC-45の値を計算～712A
- ピクセルのグレースケールを、CX、CY、C+45、およびC-45の連結によってアドレス指定される学習テーブルの要素へと追加～714A
- sampleCountを増加～716A
- 学習テーブルの各要素をsampleCountで除算～718A

【図 62】

- trainLookUpTablesForMultipleFontSet～700B
- フォント・セットを定義～720
- スキャン変換すべき複数の文字のそれぞれについて～702
- スキャン変換の各行について～704
- 行の各ピクセルについて～706
- ピクセルのグレースケールの正確な計算を実行～708
- ピクセルが完全に内側または完全に外側ではない場合～710
- ピクセルのCXおよびCYの値を計算～712
- ピクセルのグレースケールを、CXおよびCYの連結によってアドレス指定される現在の文字のフォント・セットのための学習テーブルの要素へと追加～714B
- 現在の文字のフォント・セットについてのsampleCountを増加～716B
- 各フォント・セットについて～722
- そのフォント・セットの学習テーブルの各要素を、そのフォント・セットのsampleCountで除算～718B

【図 63】

- look-up table algorithm～650B
- 現在のピクセルのCXおよびCYの値から、現在の参照アドレスを形成～652B
- 現在の文字のためのフォント・セットに対応する参照テーブルにおいて、現在の参照アドレスにある参照テーブルのエントリから現在のピクセルの域カバー値を取得～654B

フロントページの続き

審査官 福永 健司

(56)参考文献 特開平 0 9 - 1 1 4 4 4 1 (J P , A)

特開平 0 7 - 0 2 0 8 4 5 (J P , A)

特開平 0 5 - 0 4 0 4 6 3 (J P , A)

特開平 0 2 - 2 7 5 4 9 4 (J P , A)

NAKAMAE E , IEEE COMPUTER GRAPHICS AND APPLICATIONS , 米国 , 1 9 8 9 年 3 月 1 日 , V9 N
2 , P21-29

(58)調査した分野(Int.Cl. , D B 名)

G09G5/00-5/42