

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号  
特許第5058575号  
(P5058575)

(45) 発行日 平成24年10月24日(2012.10.24)

(24) 登録日 平成24年8月10日(2012.8.10)

(51) Int.Cl.

G 0 6 T 9/20 (2006.01)

F I

G O 6 T 9/20

請求項の数 12 (全 50 頁)

(21) 出願番号	特願2006-335070 (P2006-335070)	(73) 特許権者	000001007
(22) 出願日	平成18年12月12日 (2006.12.12)		キヤノン株式会社
(65) 公開番号	特開2008-146496 (P2008-146496A)		東京都大田区下丸子3丁目30番2号
(43) 公開日	平成20年6月26日 (2008.6.26)	(74) 代理人	100076428
審査請求日	平成21年12月9日 (2009.12.9)		弁理士 大塚 康德
		(74) 代理人	100112508
			弁理士 高柳 司郎
		(74) 代理人	100115071
			弁理士 大塚 康弘
		(74) 代理人	100116894
			弁理士 木村 秀二
		(72) 発明者	石田 良弘
			東京都大田区下丸子3丁目30番2号 キ
			ヤノン株式会社内

最終頁に続く

(54) 【発明の名称】 画像処理装置及びその制御方法、プログラム

(57) 【特許請求の範囲】

【請求項 1】

画像をベクトル化するベクトル処理を実行する画像処理装置であって、  
前記画像を構成する各画素の属性情報に基づいて、前記画像を複数の領域に分割する分割手段と、  
前記分割手段により分割した各領域の境界線に関する領域境界情報を抽出する抽出手段と、  
前記抽出手段により抽出した領域境界情報に基づいて、前記境界線の交点を同定する交点情報生成手段と、  
前記交点情報生成手段で同定した交点を端点として、前記境界線における前記端点間の部分毎に、関数近似処理を実行することにより、当該関数近似された端点間毎の境界線を示す端点間ベクトルデータを生成する端点間ベクトル生成手段と、  
前記端点間ベクトル生成手段により生成した端点間ベクトルデータを用いて、前記分割手段により分割された各領域を囲む領域別のベクトルデータを生成する領域別ベクトル生成手段と  
を備え、  
前記端点間ベクトル生成手段で前記関数近似処理により生成される前記端点間ベクトルデータの端点の位置は、前記関数近似処理の実行前の前記端点の位置から変化しないことを特徴とする画像処理装置。

【請求項 2】

前記領域別ベクトル生成手段は、

前記分割手段により分割した領域の属性情報に基づいて、前記端点間ベクトル生成手段により生成した端点間ベクトルデータから、前記分割手段により分割した各領域について当該各領域を囲む 1 つ以上の端点間ベクトルデータを同定する同定手段と、

前記同定手段で前記各領域について同定した端点間ベクトルデータに対して、前記各領域に対応する属性情報を付与する付与手段とを備え、

前記同定手段で前記各領域について同定した前記各領域を囲む 1 つ以上の端点間ベクトルデータと前記付与手段で付与した前記各領域の属性情報からなるベクトルデータを、前記領域別のベクトルデータとして生成する

ことを特徴とする請求項 1 に記載の画像処理装置。

10

【請求項 3】

前記領域境界情報は、異なる領域同士が隣接している領域の境界間にある画素に関する情報である

ことを特徴とする請求項 1 又は 2 に記載の画像処理装置。

【請求項 4】

前記交点情報生成手段は、前記抽出手段により抽出した領域境界情報に基づいて、前記境界線の交点を同定し、当該同定した交点を示す交点情報を生成する

ことを特徴とする請求項 1 乃至 3 のいずれか 1 項に記載の画像処理装置。

【請求項 5】

前記領域別ベクトル生成手段は、前記交点情報生成手段により生成した交点情報と前記端点間ベクトル生成手段により生成した端点間ベクトルデータとを用いて、前記領域別のベクトルデータを生成する

20

ことを特徴とする請求項 4 に記載の画像処理装置。

【請求項 6】

前記分割手段は、前記画像中の画素同士の特徴の乖離度に基づいて、前記画像中の画素をクラスタリングし、当該クラスタリングで得られたクラスタに基づいて、前記画像を複数の領域に分割する

ことを特徴とする請求項 1 乃至 5 のいずれか 1 項に記載の画像処理装置。

【請求項 7】

前記端点間ベクトル生成手段は、前記抽出手段により抽出した領域境界情報に基づいて生成される端点間の境界線の画像を用いて、前記端点間ベクトルデータを生成する

30

ことを特徴とする請求項 1 に記載の画像処理装置。

【請求項 8】

前記端点間ベクトル生成手段は、前記抽出手段により抽出した領域境界情報に基づき全ての境界線の画像を生成し、当該生成した境界線の画像と前記端点とに基づいて前記端点間ベクトルデータを生成する

ことを特徴とする請求項 1 に記載の画像処理装置。

【請求項 9】

前記画像中の処理対象領域を指定する指定手段を更に備え、

当該指定手段で指定された処理対象領域に対して、前記分割手段と前記抽出手段と前記交点情報生成手段と前記端点間ベクトル生成手段と前記領域別ベクトル生成手段とによる各処理を実行する

40

ことを特徴とする請求項 1 に記載の画像処理装置。

【請求項 10】

前記画像の属性情報は、前記画像の色情報であり、

前記分割手段は、前記画像の色情報に基づいて、同色と判断した領域毎に分割する

ことを特徴とする請求項 1 に記載の画像処理装置。

【請求項 11】

画像をベクトル化するベクトル処理を実行する画像処理装置の制御方法であって、

分割手段が、前記画像を構成する各画素の属性情報に基づいて、前記画像を複数の領域

50

に分割する分割工程と、

抽出手段が、前記分割工程により分割した各領域の境界線に関する領域境界情報を抽出する抽出工程と、

交点情報生成手段が、前記抽出工程により抽出した領域境界情報に基づいて、前記境界線の交点を同定する交点情報生成工程と、

端点間ベクトル生成手段が、前記交点情報生成工程で同定した交点を端点として、前記境界線における前記端点間の部分毎に、関数近似処理を実行することにより、当該関数近似された端点間毎の境界線を示す端点間ベクトルデータを生成する端点間ベクトル生成工程と

領域別ベクトル生成手段が、前記端点間ベクトル生成工程により生成した端点間ベクトルデータを用いて、前記分割工程で分割された各領域を囲む領域別のベクトルデータを生成する領域別ベクトル生成工程と

を備え、

前記端点間ベクトル生成工程で前記関数近似処理により生成される前記端点間ベクトルデータの端点の位置は、前記関数近似処理の実行前の前記端点の位置から変化しない

ことを特徴とする画像処理装置の制御方法。

【請求項 12】

コンピュータを、

画像を構成する各画素の属性情報に基づいて、前記画像を複数の領域に分割する分割手段、

前記分割手段により分割した各領域の境界線に関する領域境界情報を抽出する抽出手段

と、  
前記抽出手段により抽出した領域境界情報に基づいて、前記境界線の交点を同定する交点情報生成手段と、

前記交点情報生成手段で同定した交点を端点として、前記境界線における前記端点間の部分毎に、関数近似処理を実行することにより、当該関数近似された端点間毎の境界線を示す端点間ベクトルデータを生成する端点間ベクトル生成手段、

前記端点間ベクトル生成手段により生成した端点間ベクトルデータを用いて、前記分割手段により分割された各領域を囲む領域別のベクトルデータを生成する領域別ベクトル生成手段、

として機能させるためのプログラムであって、

前記端点間ベクトル生成手段で前記関数近似処理により生成される前記端点間ベクトルデータの端点の位置は、前記関数近似処理の実行前の前記端点の位置から変化しない

ことを特徴とするプログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、カラー画像をベクトル化するベクトル処理を実行する画像処理装置及びその制御方法、プログラムに関するものである。

【背景技術】

【0002】

従来、画像のベクトル化に関連する技術としては、ワードプロセッサ等で用いられるフォント（文字書体）を対象とした処理技術が広く一般的に知られている。フォント（文字書体）を対象としたベクトル化の技術は、入力としては、予め美しくデザインされた文字のみを対象としている。具体例としては、アナログ的に作成された1文字を512×512や1024×1024ピクセル等の比較的大きなサイズの2値画像としてデジタル化したものをベースに、輪郭をベクトル化する。ベクトル化されたデータは、基本的には、各種の所望のサイズに拡大してもスムーズな輪郭表現が可能なことから、形状の品質が良いという特徴がある。さらに、各種のサイズの文字を、1つのデータをもとに生成可能であるという特徴があり、簡便さや、データ量の削減効果を有する。

10

20

30

40

50

## 【 0 0 0 3 】

このため、フォント（文字書体）のみならず、一般的な 2 値画像に対しても、これらの特徴を生かした効果を狙って、ベクトル処理を施す技術（例えば、特許文献 1）が提案されている。

## 【 0 0 0 4 】

加えて、ベクトル化されたデータは、基本的には座標値を基本とした数値データであり、コンピュータでの編集処理等も容易なため、ラインアート（線画像）にベクトル処理を施す技術（例えば、特許文献 2）も提案されている。また、昨今では、フルカラーの部分を含む読み取り画像に対するベクトル化に関する提案もなされてきている（例えば、特許文献 3）。

10

## 【 0 0 0 5 】

尚、上記特許文献 1 の前提となる、2 値画像のベクトル化に関しては、本出願人は、特許文献 4 を提案している。

## 【 0 0 0 6 】

この特許文献 4 において、『注目画素と、その近傍画素の状態により、予め定められた位置を輪郭線を構成する点とし、近傍画素の状態により該輪郭線を構成する点の接続方向を決定する工程とを備える。また、前記輪郭線を構成する点と、該輪郭線を構成する他の点との接続状態を判断する工程と、前記注目画素の位置をラスト走査順に画像データ上を更新し、注目画素毎に近傍画素の状態に基づいて前記工程を実行する。これにより、輪郭点を抽出する。』

20

## 【 0 0 0 7 】

以上の構成において、画像データにおける注目画素と、その近傍画素の状態を保持し、この注目画素をラスト走査順に取り出し、その注目画素とその近傍画素の状態に基づいて、水平方向及び垂直方向の画素間ベクトルを検出する。そして、これら画素間ベクトル同士の接続状態を判別して、この判別された画素間ベクトルの接続状態をもとに、画像データの輪郭を抽出するように動作する。』という手法を提案している。

## 【 0 0 0 8 】

この手法は、画像の中の全ての輪郭線を 1 回のラスト走査順だけで抽出でき、かつ、全画像データを記憶するための画像メモリを必要としないため、メモリの容量を少なくできる効果を有している。また、入力画像の画素の中心位置ではなく、画素の縁単位に輪郭を抽出することによって、1 画素幅の細線に対しても有為な幅を有する輪郭線抽出法である。

30

## 【 0 0 0 9 】

さらに、出願人は既に特許文献 5 を提案している。この特許文献 5 においては、2 値画像の輪郭情報を直線のみならず、2 次や 3 次のベジェ曲線近似をすることで、より少ないデータ量で高画質な変倍画像を表現する輪郭情報を関数近似する処理手法と処理装置を開示している。2 値画像からアウトラインベクトルを抽出し、そのアウトラインベクトル表現の状態で、特許文献 1 による手法で滑らかに関数近似されたアウトラインベクトルを用いても、所望の倍率（任意）で変倍された高画質のデジタル 2 値画像を得ることが可能である。ここで、2 値画像からアウトラインベクトルを抽出する方法としては、特許文献 4 で開示される方法を用いて構成することができる。

40

【特許文献 1】特登録 0 3 0 4 9 6 7 2 号公報

【特許文献 2】特願 2 0 0 6 - 1 0 0 3 8 1

【特許文献 3】特登録 0 2 8 4 5 1 0 7 号公報

【特許文献 4】特登録 0 3 0 2 6 5 9 2 号公報

【特許文献 5】特開 2 0 0 5 - 3 4 6 1 3 7 号公報

【発明の開示】

【発明が解決しようとする課題】

## 【 0 0 1 0 】

しかしながら、従来より知られるカラー画像のベクトル化方法においては、色空間上で

50

既定される色領域に属する画素からなる領域毎に輪郭形状をベクトル化するのが基本である。そのため、それぞれの領域の輪郭形状をベジェ関数近似や平滑化処理すると、読み取り画像上で隣接する色領域間の輪郭が必ずしも同一の状態の輪郭形状に近似や平滑化がされずに、隙間や重なりが生じてしまうという問題があった。この様子を、図57と図58に示した。図57は、入力画像から色空間上で既定される色領域に属する画素からなる領域毎に輪郭形状を抽出した例を示している。ここで、背景の白地を数えないものとする、入力画像上では、隣接する（重なる）色領域が3つ存在している。

【0011】

図58は、色領域に属する画素からなる領域毎に輪郭形状をベクトル化し、平滑化（関数近似）した場合の例を示しており、各領域間に隙間や重なりが生じてしまっている。

10

【0012】

本発明は上記の課題を解決するためになされたものであり、複数の色領域から構成されるカラー画像から、より適切に各色領域をベクトル化することができる画像処理装置及びその制御方法、プログラムを提供することを目的とする。

【課題を解決するための手段】

【0013】

上記の目的を達成するための本発明による画像処理装置は以下の構成を備える。即ち、前記画像を構成する各画素の属性情報に基づいて、前記画像を複数の領域に分割する分割手段と、

前記分割手段により分割した各領域の境界線に関する領域境界情報を抽出する抽出手段と、

20

前記抽出手段により抽出した領域境界情報に基づいて、前記境界線の交点を同定する交点情報生成手段と、

前記交点情報生成手段で同定した交点を端点として、前記境界線における前記端点間の部分毎に、関数近似処理を実行することにより、当該関数近似された端点間毎の境界線を示す端点間ベクトルデータを生成する端点間ベクトル生成手段と、

前記端点間ベクトル生成手段により生成した端点間ベクトルデータを用いて、前記分割手段により分割された各領域を囲む領域別のベクトルデータを生成する領域別ベクトル生成手段と

を備え、

30

前記端点間ベクトル生成手段で前記関数近似処理により生成される前記端点間ベクトルデータの端点の位置は、前記関数近似処理の実行前の前記端点の位置から変化しない。

【発明の効果】

【0024】

本発明によれば、複数の色領域から構成されるカラー画像から、より適切に各色領域をベクトル化することができる画像処理装置及びその制御方法、プログラムを提供できる。

【発明を実施するための最良の形態】

【0025】

以下、本発明の実施の形態について図面を用いて詳細に説明する。

【0026】

40

本発明は、画像上で隣接する色領域間の輪郭形状に関数近似や平滑化を施しても、隙間や重なりが発生がなく、良好なベクトル化（ベクトル処理）が可能な技術について提供する。本発明では、カラー画像、とりわけ、デジタルカメラ等により撮影された自然画像に比べて色数の少ないイラスト画像のベクトル化について提供する。

【0027】

尚、各実施形態で説明するベクトルとは、ビットマップ画像データを作成する際に用いられる数式等の描画コマンドで表現されたデータ（ベクトルデータ）を意味するものである。また、ベクタ画像データをビットマップ画像データに変換することを、ラスタ化やラスタライズという。そのため、このビットマップ画像データは、ラスタデータと呼ばれることもある。

50

## 【 0 0 2 8 】

また、以下の説明では、ベクトルデータのことを、単にベクトルと略称する。

## 【 0 0 2 9 】

< 実施形態 1 >

図 1 は本発明の実施形態 1 の画像処理装置の機能構成を示すブロック図である。

## 【 0 0 3 0 】

同図において、11は、入力としての処理対象画像を獲得するカラー画像獲得部である。12は、カラー画像獲得部11で獲得された画像を、同色領域とみなせる画素の連結としての部分領域に分割する領域同定部である。ここで、同色領域とみなせるとは、画素の特徴量（例えば、色情報（例えば、RGB値））が所定範囲内に含まれるものを、同一の画素の特徴量を有している領域と判定することである。

10

## 【 0 0 3 1 】

この領域同定部12は、領域分割部121と領域テーブル生成部122を備える。領域分割部121は、色情報に基づき幾何学的に連結した同色領域とみなせる画素の集まりに画像を分割する。また、領域テーブル生成部122は、領域分割部121から得られる各領域とそれら各領域の色情報（属性情報（例えば、RGB値））の対応づけ情報を、例えば、図2に示すテーブル形式（領域テーブル201）で生成する。

## 【 0 0 3 2 】

20は、領域同定部12で同定された各領域間の領域境界を1画素幅の細線で構成した領域境界細線画像を生成する領域境界細線画像生成部である。領域境界細線画像生成部20は、領域境界情報抽出部210、交点情報生成部220及び境界線画像生成部230を備える。

20

## 【 0 0 3 3 】

領域境界情報抽出部210は、領域同定部12で同定された各領域間の領域境界となる境界画素とそれらの各境界画素が隔てる領域間を示す領域境界情報を抽出する。また、交点情報生成部220は、領域境界画素を4-連結化した領域境界線の分岐点となる領域境界線の交点を示す交点情報を生成する。換言すれば、互いに異なる領域間の境界において、互いに異なる境界の交点を示す交点情報を生成する。また、境界線画像生成部230は、抽出された領域境界情報と生成された交点情報とから、領域境界細線画像を生成する。

## 【 0 0 3 4 】

領域境界情報抽出部210は、水平方向境界点抽出部211、水平方向境界点テーブル生成部212、垂直方向境界点抽出部213、垂直方向境界点テーブル生成部214、斜め方向境界点抽出部215及び斜め方向境界点テーブル生成部216を備える。

30

## 【 0 0 3 5 】

水平方向境界点抽出部211は、例えば、図3に示すように、領域同定部12により同定された同色領域とみなせる画素の連結としての部分領域で構成される領域同定結果画像を、水平方向に走査したときの各走査線上での領域間の境界となる画素を抽出する。即ち、水平方向における、ある部分領域から他の部分領域への切り替わりになる画素を抽出する。

## 【 0 0 3 6 】

水平方向境界点テーブル生成部212は、水平方向境界点抽出部211で抽出された水平方向境界点の各点（画素）に関する情報を、図4に示す水平方向境界点テーブル401として生成する。

40

## 【 0 0 3 7 】

垂直方向境界点抽出部213は、例えば、図3に示すように、領域同定部12により同定された同色領域とみなせる画素の連結としての部分領域で構成される領域同定結果画像を、垂直方向に走査したときの各走査線上での領域間の境界となる画素を抽出する。即ち、垂直方向における、ある部分領域から他の部分領域への切り替わりになる画素を抽出する。

## 【 0 0 3 8 】

50

垂直方向境界点テーブル生成部 2 1 4 は、垂直方向境界点抽出部 2 1 3 で抽出された垂直方向境界点の各点（画素）に関する情報を、図 4 に示す垂直方向境界点テーブル 4 0 2 として生成する。

【 0 0 3 9 】

斜め方向境界点抽出部 2 1 5 は、例えば、図 3 に示すように、領域同定部 1 2 により同定された同色領域とみなせる画素の連結としての部分領域で構成される領域同定結果画像を、斜め方向に走査したときの斜め走査線上での領域間の境界となる画素を抽出する。即ち、斜め方向における、ある部分領域から他の部分領域への切り替わりになる画素を抽出する。

【 0 0 4 0 】

斜め方向境界点テーブル生成部 2 1 6 は、斜め方向境界点抽出部 2 1 5 で抽出された斜め方向境界点の各点（画素）に関する情報を、図 4 に示す斜め方向境界点テーブル 4 0 3 として生成する。

【 0 0 4 1 】

交点情報生成部 2 2 0 は、交点同定部 2 2 1 と交点テーブル生成部 2 2 2 を備える。

【 0 0 4 2 】

交点同定部 2 2 1 は、図 4 の水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2 及び斜め方向境界点テーブル 4 0 3 とから、交点情報を生成する。

【 0 0 4 3 】

具体的には、図 5 に示すように、3 つのテーブル 4 0 1 ~ 4 0 3 上の少なくとも 2 つ以上のテーブル上に存在する境界点の内、それが属する 2 つ以上のテーブル上でそれぞれの隣接する領域番号の組み合わせが完全には一致していない境界点を探索する。即ち、異なる領域番号の組み合わせが存在するような境界点を探索し、それを交点（交点情報）とする。図 5 の例では、C 及び E が交点となる。

【 0 0 4 4 】

交点テーブル生成部 2 2 2 は、交点同定部 2 2 1 で同定された交点について、その交点が属するテーブル上の隣接する領域番号の内の少なくともいずれかには表記されている領域番号を全て抜き出す。そして、これにより、その交点の隣接する領域の領域番号の全てを対応する交点とともに管理する、図 6 に示す交点テーブル 6 0 1 を生成する。

【 0 0 4 5 】

境界線画像生成部 2 3 0 は、境界線画像描画部 2 3 1 と境界線画像細線化部 2 3 2 を備える。

【 0 0 4 6 】

境界線画像描画部 2 3 1 は、生成された領域境界情報と生成された交点情報とから、領域境界画素を 4 - 連結化した境界線画像を生成する。また、境界線画像細線化部 2 3 2 は、得られた境界線画像に 4 - 連結の状態を保った細線化を行ない、4 - 連結の領域境界細線画像を生成する。

【 0 0 4 7 】

例えば、境界線画像細線化部 2 3 2 は、境界線画像描画部 2 3 1 にて生成される領域境界画素を 4 - 連結化した境界線画像（2 値画像）を、公知の 2 値画像細線化手法により、4 - 連結に細線化された 2 値画像としての 4 - 連結の領域境界細線画像を出力する。

【 0 0 4 8 】

ここで、公知の 2 値画像細線化手法としては、H i l d i t c h 法がある。これは、例えば、酒井幸子著「デジタル画像処理の基礎と応用」第 2 版，I S B N 4 - 7 8 9 8 - 3 7 0 7 - 6，C Q 出版社，2 0 0 4 年 2 月 1 日発行，P . 5 1 - P . 5 4，等）に記載されている。

【 0 0 4 9 】

一例として、図 7（a）に示される 2 値画像を境界線画像細線化部 2 3 2 に入力した場合には、境界線画像細線化部 2 3 2 から出力されることになる細線化済画像は、図 7（b）となる。

10

20

30

40

50

## 【 0 0 5 0 】

図 1 の説明に戻る。

## 【 0 0 5 1 】

3 0 は、領域境界細線画像生成部 2 0 で生成された 4 - 連結の領域境界細線画像を、交点から他の交点までを繋ぐ 4 - 連結の細線毎に、其々の交点を端点とした端点間ベクトルの集まりとしてベクトル化する端点間ベクトル生成部である。

## 【 0 0 5 2 】

端点間ベクトル生成部 3 0 は、端点間ベクトル抽出部 3 1 0 と端点間ベクトル関数近似（平滑）部 3 2 0 を備える。

## 【 0 0 5 3 】

端点間ベクトル抽出部 3 1 0 は、4 - 連結の領域境界細線画像を、交点から他の交点までを繋ぐ 4 - 連結の細線毎に、其々の交点をベクトルの端点として、端点間を繋ぐ各細線の画素の中心位置上を通過する線芯ベクトルを端点間ベクトルとして抽出する。

## 【 0 0 5 4 】

端点間ベクトル関数近似（平滑）部 3 2 0 は、抽出された端点間ベクトルを、端点から端点までの間が、端点部分が端点として保存された形で関数近似する。即ち、端点部分がベジェ曲線のアンカーポイントとなるようにベジェ関数近似し、関数近似（平滑化）済端点間ベクトルを生成する。

## 【 0 0 5 5 】

ここで、端点間ベクトル抽出部 3 1 0 は、図 8 に示すように、端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 と端点間毎線芯（周回）ベクトル生成部 3 1 2 0 とを備える。

## 【 0 0 5 6 】

端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 は、4 - 連結の状態を保って細線化されている領域境界細線画像を入力する。次に、1 回のラスト走査のみにより、処理対象の細線化済 2 値画像中の線図形を、それらを構成している端点や交点間をつなぐ独立した線や閉曲線毎のそれぞれに対応したベクトル列の集まり（以降、「端点間毎線素粗輪郭ベクトル」という）として抽出する。そして、その処理結果を出力する。

## 【 0 0 5 7 】

尚、出力される処理結果には、ベクトル列のそれぞれを構成する各ベクトルが、処理対象の線図形の端点（交点）位置に対応する部分から抽出されたものか否か等を表す情報（以降、「付与情報」ともいう）が含まれる。

## 【 0 0 5 8 】

また、端点間毎線芯（周回）ベクトル生成部 3 1 2 0 は、端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 より得られる付与情報と端点間毎線素粗輪郭ベクトル（以降、これらを合わせて「端点間毎輪郭（周回）ベクトル」という）を入力する。そして、付与情報を利用して、一画素幅の幅を持った輪郭（周回）ベクトルの各ベクトルの位置をそれぞれ半画素分ずつ相応する方向へ予め定める規則により微調整する。これによって、幅 0 に線芯化された端点間毎線芯（周回）化済ベクトルを生成する。

## 【 0 0 5 9 】

図 7（c）は、図 7（b）の画像を入力 2 値画像とした際の端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 から出力される端点間毎輪郭（周回）ベクトルを輪郭画像として描画して可視化したものである。また、図 7（d）は、図 7（c）の画像を入力した際の端点間毎線芯（周回）ベクトル生成部 3 1 2 0 から出力される端点間毎線芯（周回）化済ベクトルを線画像として描画して可視化したものである。

## 【 0 0 6 0 】

また、端点間ベクトル関数近似（平滑）部 3 2 0 は、端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0、端点間毎補助ベクトル入り（周回）ベクトル平滑化部 3 2 2 0、端点間毎平滑化済ベクトル同定（非周回化）部 3 2 3 0 を備える（図 8 参照）。

## 【 0 0 6 1 】

ここで、端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0 は、端点間毎線芯

10

20

30

40

50

(周回)ベクトル生成部 3 1 2 0 より得られる端点間毎線芯(周回)化済ベクトルを入力する。そして、次工程の処理をする端点間毎補助ベクトル入り(周回)ベクトル平滑部 3 2 2 0 により平滑化(関数近似)処理される際に、端点部分に相当するベクトルが、他のベクトルと統合されて端点位置が不明にならぬようにするための補助ベクトルを挿入する。つまり、端点部分に相当する部分が端点として保存される(ベジェ曲線のアンカーポイントとなる)ように補助ベクトルを挿入することにより、端点間毎補助ベクトル入り(周回)ベクトルを生成する。

#### 【0062】

加えて、端点間毎補助ベクトル入り(周回)ベクトル生成部 3 2 1 0 は、補助ベクトルを挿入された各端点間毎補助ベクトル入り(周回)ベクトルのそれぞれに対し、どのベクトルが端点部にあたるかを明示する始端点・終端点情報をも生成する。

10

#### 【0063】

図 7 (e) は、図 7 (d) の画像を入力した際の端点間毎補助ベクトル入り(周回)ベクトル生成部 3 2 1 0 から出力される端点間毎補助ベクトル入り(周回)ベクトルを線画像として描画して可視化したものである。

#### 【0064】

端点間毎補助ベクトル入り(周回)ベクトル平滑部 3 2 2 0 は、端点間毎補助ベクトル入り(周回)ベクトル生成部 3 2 1 0 より端点間毎補助ベクトル入り(周回)ベクトルを入力する。次に、各(周回)ベクトル毎に平滑化(関数近似)処理を行うことによって、平滑化された平滑化済端点間毎補助ベクトル入り(周回)ベクトルを生成する。そして、それを、次工程の処理をする端点間毎平滑化済ベクトル同定(非周回化)部 3 2 3 0 に出力する。

20

#### 【0065】

端点間毎平滑化済ベクトル同定(非周回化)部 3 2 3 0 は、端点間毎補助ベクトル入り(周回)ベクトル平滑部 3 2 2 0 より平滑化(関数近似)済端点間毎補助ベクトル入り(周回)ベクトルを入力する。また、端点間毎平滑化済ベクトル同定(非周回化)部 3 2 3 0 は、端点間毎補助ベクトル入り(周回)ベクトル生成部 3 2 1 0 より始端点・終端点情報を入力する。

#### 【0066】

そして、端点間毎平滑化済ベクトル同定(非周回化)部 3 2 3 0 は、各平滑化済の(周回)ベクトル毎にそれぞれ始端点と終端点となるベクトルを同定して、始端点と終端点の間のみの部分の非周回型のベクトル列を生成する。

30

#### 【0067】

図 7 (f) は、図 7 (e) の画像を入力した際の端点間毎平滑化済ベクトル同定(非周回化)部 3 2 3 0 から出力される平滑化済ベクトルを線画像として描画して可視化したものである。

#### 【0068】

図 1 の説明に戻る。

#### 【0069】

4 0 は、各領域毎にそれを取り囲む平滑化(関数近似)済端点間ベクトル列とその内部色とでなる領域ベクトルに変換する領域別ベクトル生成部である。この処理は、端点間ベクトル生成部 3 0 から得る平滑化(関数近似)済端点間ベクトルと、領域境界細線画像生成部 2 0 から得る各交点が隣接する領域の領域番号情報と、領域同定部 1 2 から得る各領域とそれらの色情報(属性情報)の対応づけ情報を用いて行う。

40

#### 【0070】

領域別ベクトル生成部 4 0 は、領域境界ベクトル列同定部 4 1 0 と、領域属性付与部 4 2 0 とを備える。

#### 【0071】

領域境界ベクトル列同定部 4 1 0 は、各領域毎にそれを取り囲む関数近似(平滑化)済端点間ベクトル列を同定する。この処理は、端点間ベクトル生成部 3 0 で生成された関数

50

近似（平滑化）済端点間ベクトルと、領域同定部 1 2 から得る各領域とそれらの色情報（属性情報）の対応づけ情報を用いて行われる。

【 0 0 7 2 】

領域属性付与部 4 2 0 は、各領域を取り囲む平滑化（関数近似）済端点間ベクトル列にその内部の色情報を付与して、各領域毎にそれを取り囲む平滑化（関数近似）済端点間ベクトル列とその色情報からなる領域ベクトルを生成する。この処理は、領域境界ベクトル列同定部 4 1 0 で同定された各領域毎の平滑化（関数近似）済端点間ベクトル列と、領域同定部 1 2 から得る各領域とそれらの色情報（属性情報）の対応づけ情報を用いて行う。

【 0 0 7 3 】

次に、図 1 の画像処理装置のハードウェア構成について、図 9 を用いて説明する。

10

【 0 0 7 4 】

図 9 は本発明の実施形態 1 の画像処理装置のハードウェア構成を示す図である。

【 0 0 7 5 】

同図において、5 1 8 は、図 9 に示す各種構成要素を相互に接続するためのバスである。1 はスキャナである。2 は画像メモリであり、3 は画像入出力 I / O である。画像メモリ 2 は画像入出力 I / O 3 を介して、スキャナ 1 で読み取られたカラー画像を保持する。図 1 のカラー画像獲得部 1 1 は、このスキャナ 1、画像入出力 I / O 3 及び画像メモリ 2 によって実現される。

【 0 0 7 6 】

また、画像メモリ 2 は、領域同定部 1 2 により生成される同色領域とみなせる画素の連結領域を部分領域として構成される領域同定結果画像を保持する。また、画像メモリ 2 は、領域境界細線画像生成部 2 0 の境界線画像生成部 2 3 0 で生成される境界線画像や、得られた境界線画像に対する、4 - 連結の状態での細線化によって生成される 4 - 連結の領域境界細線画像を保持する。

20

【 0 0 7 7 】

このように、画像メモリ 2 は、領域同定部 1 2 や領域境界細線画像生成部 2 0、端点間ベクトル生成部 3 0 を構成する一部としても機能する。

【 0 0 7 8 】

4 は通信 I / F であり、ネットワーク等を経由して、システムの外部と通信する。5 1 9 は CPU である。5 2 0 は RAM で、例えば、ワーキングメモリとして機能する。6 は ROM であり、後に説明する手順に沿って CPU 5 1 9 で実行されるプログラムや予め定められるパラメータやデータ等を格納している。

30

【 0 0 7 9 】

ここで、画像メモリ 2 と、CPU 5 1 9 や RAM 5 2 0、ROM 6 により、図 1 の領域同定部 1 2、領域境界細線画像生成部 2 0、端点間ベクトル生成部 3 0 が構成されるとともに、領域別ベクトル生成部 4 0 も実現される。

【 0 0 8 0 】

5 2 1 はハードディスク装置 5 2 2 とのディスク I / O（入出力回路）である。

【 0 0 8 1 】

以下、画像処理装置の動作について、図 1 0 を用いて説明する。

40

【 0 0 8 2 】

図 1 0 は本発明の実施形態 1 の画像処理装置の動作を示すフローチャートである。

【 0 0 8 3 】

尚、この処理は、例えば、CPU 5 1 9 が、ROM 6 に記憶されているプログラムを実行することによって、図 1 に示される各種機能構成によって実現される。

【 0 0 8 4 】

まず、ステップ S 1 1 で、カラー画像獲得部 1 1 が、入力としての処理対象カラー画像を入力する。次に、ステップ S 1 2 とステップ S 1 3 は、領域同定部 1 2 での処理内容を実行する。即ち、ステップ S 1 1 で入力した画像を領域分割処理し、領域分割処理結果を RAM 5 2 0 に保存する。

50

## 【 0 0 8 5 】

より具体的には、ステップ S 1 2 で、領域分割部 1 2 1 が、色情報に基づき幾何学的に連結した同色領域とみなすことができる画素の集まりに画像を分割する。そして、ステップ S 1 3 で、領域テーブル生成部 1 2 2 が、得られる各領域内の画素の色平均値をもって、その領域の色情報（属性情報）とし、図 2 に示す領域テーブル 2 0 1 を作成して、R A M 5 2 0 上に保存する。ステップ S 1 2 の領域分割処理の詳細については後述する。

## 【 0 0 8 6 】

次に、ステップ S 2 1 からステップ S 2 6 は、領域境界情報抽出部 2 1 0 での処理内容を実行する。即ち、ステップ S 1 2 で同定された各領域間の領域境界となる境界画素とそれらの各境界画素が隔てる領域間を示す領域境界情報を抽出する。

10

## 【 0 0 8 7 】

まず、ステップ S 2 1 で、水平方向境界点抽出部 2 1 1 が、同色領域とみなすことができる画素の連結としての部分領域で構成される領域同定結果画像を、水平方向に走査したときの各走査線上での領域間の境界となる画素を抽出する。即ち、ある領域から他の領域への切り替わりになる水平方向境界点の画素を抽出する。ステップ S 2 2 で、水平方向境界点テーブル生成部 2 1 2 が、抽出した水平方向境界点の各点に関する座標と隣接する領域番号を用いて、水平方向境界点テーブル 4 0 1（図 4）を生成する。

## 【 0 0 8 8 】

ステップ S 2 3 で、垂直方向境界点抽出部 2 1 3 が、同色領域とみなすことができる画素の連結としての部分領域で構成される領域同定結果画像を、垂直方向に走査したときの各走査線上での領域間の境界となる垂直方向境界点の画素を抽出する。ステップ S 2 4 で、垂直方向境界点テーブル生成部 2 1 4 が、抽出した垂直方向境界点の各点に関する座標と隣接する領域番号を用いて、垂直方向境界点テーブル 4 0 2（図 4）を生成する。

20

## 【 0 0 8 9 】

ステップ S 2 5 で、斜め方向境界点抽出部 2 1 5 が、同色領域とみなすことができる画素の連結としての部分領域で構成される領域同定結果画像を、斜め方向に走査したときの各走査線上での領域間の境界となる斜め方向境界点の画素を抽出する。ステップ S 2 6 で、斜め方向境界点テーブル生成部 2 1 6 が、抽出した斜め方向境界点の各点に関する座標と隣接する領域番号を用いて、斜め方向境界点テーブル 4 0 3（図 4）を生成する。

## 【 0 0 9 0 】

尚、ステップ S 2 1、ステップ S 2 3 とステップ S 2 5 の境界点抽出処理の詳細については後述する。

30

## 【 0 0 9 1 】

次に、ステップ S 3 0 と S 3 1 は、交点情報生成部 2 2 0 での処理内容を実行する。即ち、領域境界画素を 4 - 連結化した領域境界線の分岐点となる領域境界線の交点の情報を生成する。

## 【 0 0 9 2 】

まず、ステップ S 3 0 で、交点同定部 2 2 1 が、水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2 及び斜め方向境界点テーブル 4 0 3 とから、交点を抽出する。

## 【 0 0 9 3 】

具体的には、3つのテーブル 4 0 1 ~ 4 0 3 上で2つ以上のテーブルに存在する境界点の内、各テーブル 4 0 1 ~ 4 0 3 それぞれでの隣接する領域番号とから3つ以上の領域番号を持つ境界点を探索して、それを交点とする。

40

## 【 0 0 9 4 】

尚、3つのテーブル上で2つ以上のテーブルに存在する境界点とは、水平境界点かつ垂直方向境界点、水平方向境界点かつ斜め方向境界点、垂直方向境界点かつ斜め方向境界点、あるいは水平方向境界点かつ垂直方向境界点かつ斜め方向境界点のいずれかである。

## 【 0 0 9 5 】

ステップ S 3 1 で、交点テーブル生成部 2 2 2 が、ステップ S 3 0 で同定された交点に基づいて、交点テーブル 6 0 1 を生成する。

50

## 【 0 0 9 6 】

具体的には、水平方向境界点テーブル 4 0 1 での隣接する領域番号と垂直方向境界点テーブル 4 0 2 での隣接する領域番号と斜め方向境界点テーブル 4 0 3 での隣接する領域番号の少なくともいずれかには表記されている領域番号を全て抜き出す。これにより、その交点の隣接する領域の領域番号の全てとして、図 6 の交点テーブル 6 0 1 を生成する。

## 【 0 0 9 7 】

尚、ステップ S 3 0 の交点抽出処理の詳細については後述する。

## 【 0 0 9 8 】

ステップ 5 0 からステップ S 8 0 は、境界線画像生成部 2 3 0 と端点間ベクトル生成部 3 0 の処理内容を実行する。即ち、ステップ S 1 2 で生成された領域境界情報とステップ S 3 0 で獲得した交点情報とから、領域境界細線画像を生成し、各端点間ベクトルを生成する。

10

## 【 0 0 9 9 】

まず、ステップ S 5 0 で、境界線画像描画部 2 3 1 を実行する。即ち、領域境界情報から、領域境界画素を 4 - 連結化した境界線ビットマップ画像（境界線画像）を生成する。ステップ S 6 0 で、境界線画像細線化部 2 3 2 を実行する。即ち、4 - 連結化した境界線画像から領域境界細線画像を生成する。ステップ S 7 0 で、端点間ベクトル生成部 3 0 を実行する。即ち、領域境界細線画像を、交点と交点の間を繋ぐ境界線画像毎に、順次に端点間ベクトルを生成する。

## 【 0 1 0 0 】

20

ステップ S 8 0 で、全ての境界線について端点間ベクトル化の処理が終了したか否かを判定する。この処理では、ステップ S 3 1 で生成された交点テーブル 6 0 1 にあるすべての交点と交点の間を繋ぐ境界線画像の端点間ベクトル生成処理が終了していない場合（ステップ S 8 0 で N O ）、まだ未処理の境界線があると判定する。その場合、ステップ S 5 0 に戻り、以上の処理を繰り返し実行する。一方、全ての交点と交点の間を繋ぐ境界線画像の端点間ベクトル生成処理が終了している場合（ステップ S 8 0 で Y E S ）、未処理の境界線がないと判定し、ステップ S 1 0 0 へ進む。

## 【 0 1 0 1 】

ステップ S 5 0 の境界線ビットマップ生成処理では、図 4 の各境界点テーブル 4 0 1 ~ 4 0 3 に保持される水平方向境界点、垂直方向境界点、斜め方向境界点を所望とする領域境界毎に、これらテーブルの隣接する領域番号の等しいものを全て順次探し出す。そして、その探し出した境界点を用いて、ビットマップ画像を生成する。

30

## 【 0 1 0 2 】

即ち、後述する処理済チェックがなされていない境界点を、例えば、水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2、斜め方向境界点テーブル 4 0 3 の順にサーチする。そして、最初に保持される処理済のチェックがされていない境界点を探して 1 点を描画する。続いて、その点の隣接する領域番号の組み合わせと同じ領域番号の組み合わせを持つ点を水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2、斜め方向境界点テーブル 4 0 3 の全てにわたり、順次に探し出しては描画する。そして、条件に合う境界点を全て、その領域番号の組み合わせで示される領域間を分ける境界線として描画する。

40

## 【 0 1 0 3 】

尚、境界線として描画された境界点は、既に境界線ビットマップ生成処理が済んだものとして、各境界点テーブル 4 0 1 ~ 4 0 3 の境界点の其々に対応する処理済チェックフラグ欄（不図示）にチェックする（フラグセットする）。これにより、処理済であることを明示する。尚、この処理済チェックフラグがチェックされていない境界点が存在しない場合には、全ての境界点がビットマップとして生成済みであることを意味する。

## 【 0 1 0 4 】

尚、この処理で描画される境界点のビットマップ画像は、例えば、後述する図 5 0 ( e ) の、ハッチングパターンが付されている画素 ( 1 ) の部分に示されるような、ある交点

50

間を結ぶ 4 - 連結化した領域となる。しかしながら、必ずしも、一画素幅の線画像となるとは限らず、一部 2 画素幅以上になる場合がありえる。

#### 【 0 1 0 5 】

ステップ S 6 0 の境界線画像細線化処理では、ステップ S 5 0 で得られる境界線ビットマップ画像（2 値画像）を、公知の 2 値画像細線化手法で、4 - 連結に細線化された 2 値画像として出力する。

#### 【 0 1 0 6 】

次に、ステップ S 7 0 の端点間ベクトル生成処理について詳述する。

#### 【 0 1 0 7 】

上述したように、ステップ S 7 0 では、端点間ベクトル生成部 3 0 を実行する。即ち、領域境界細線画像を、交点と交点の間を繋ぐ境界線画像毎に、順次に端点間ベクトルを生成する。端点間ベクトル生成部 3 0 は、図 8 に示したように、端点間ベクトル抽出部 3 1 0 と端点間ベクトル関数近似（平滑）部 3 2 0 と備える。さらに、端点間ベクトル抽出部 3 1 0 は、端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 と端点間毎線芯（周回）ベクトル生成部 3 1 2 0 とを備える。また、端点間ベクトル関数近似（平滑）部 3 2 0 は、端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0、端点間毎補助ベクトル入り（周回）ベクトル平滑化部 3 2 2 0、端点間毎平滑化済ベクトル同定（非周回化）部 3 2 3 0 を備える。

#### 【 0 1 0 8 】

以下、主として、図 1 1 ~ 図 2 7 を用いて端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 での処理内容を説明する。

#### 【 0 1 0 9 】

実施形態 1 では、図 1 1 に示すように、2 値画像における注目画素 1 0 1 と、その近傍の 8 個の画素の状態を見て処理を進めるもので、注目画素をラスト走査し、1 画素毎にずらしながら画像全体の処理を逐次行う。

#### 【 0 1 1 0 】

図 1 1 において、× は注目画素 1 0 1 を示しており、「0」及び「2」で示された位置は、主走査方向に対し注目画素 1 0 1 と同じ位置にあり、副走査方向のそれぞれ 1 ラスタ前の画素（0）及び 1 ラスタ先の画素（2）を示す。「1」及び「3」で示された位置は、注目画素 1 0 1 と同一のラスタ上にあり、それぞれ 1 画素前の画素（3）及び 1 画素先の画素（1）を示している。

#### 【 0 1 1 1 】

更に、「A」及び「B」は、主走査方向に 1 画素先の位置にあり、それぞれ 1 ラスタ前及び 1 ラスタ先の位置にある画素を示し、「C」及び「D」は主走査方向の 1 画素前の位置にある画素で、それぞれ 1 ラスタ先及び 1 ラスタ前の位置にある画素を示している。

#### 【 0 1 1 2 】

次に注目画素 1 0 1 及びその近傍の 8 画素の状態に応じた、それぞれの場合の処理を説明する。

#### 【 0 1 1 3 】

注目画素 1 0 1 が白画素である場合は、その処理を終了してラスタ走査を 1 画素分進め、注目画素位置を更新する。

#### 【 0 1 1 4 】

注目画素 1 0 1 が黒画素の場合は、近傍の画素の状態によって、以下の処理を行う。ここで、図 1 2 ~ 図 2 7 に各状態における処理内容を示してある。図 1 2 ~ 図 2 7 において、斜めのチェッカーパターンで示される 印は黒画素を示し、点線の 印は白画素を示している。「d」は「do not care」、即ち、「d」の表示のある位置の画素は白画素でも黒画素でも構わないことを意味している。

#### 【 0 1 1 5 】

また、図 1 2 ~ 図 2 7 の各画素マトリクスにおいて、 印と 印は、横方向ベクトルの始点及び縦方向ベクトルの終点を示していて、 印と 印は縦方向ベクトルの始点及び横

10

20

30

40

50

方向ベクトルの終点を示している。これらのベクトルの始点・終点間の実線矢印は、始点及び終点共に定まった、矢印の向く方向の輪郭ベクトルを示している。一方、点線矢印は、始点もしくは終点のいずれか一方のみが定まった、矢印の向く方向の輪郭ベクトルを示している。これら輪郭ベクトルの始点及び終点を総称して、以降、輪郭点と呼ぶことにする。尚、図12、図17、図22の画素マトリクスパターンの場合は、輪郭ベクトルを抽出しない。

#### 【0116】

また、印と印は端点部から抽出されるベクトルの始点・終点で、それぞれ端点部から抽出される横方向ベクトルの始点及び縦方向ベクトルの終点、端点部から抽出される縦方向ベクトルの始点及び横方向ベクトルの終点であることを示している。

10

#### 【0117】

図12～図27の画素マトリクスに応じて抽出される輪郭点には、対応付けられる付与情報として、図28の一覧表に示した輪郭点情報のいずれかが付記されている。この輪郭点情報が、各輪郭点情報に付記されて抽出される付与情報となる。

#### 【0118】

図28の一覧表中のNo. 01～16に示される16種の輪郭点情報は、「xxx\_yy\_y\_zzz」または「xxx\_yyy」なる形で表記されている。いずれも、「xxx」の部分は、対象の輪郭点が終点となるベクトルの向きを示し、「yyy」の部分は、同輪郭点が始点となるベクトルの向きを示している。上向きをUP、下向きをDOWN、右向きをRIGHT、左向きをLEFTと表現する。この場合、例えば、No. 01～03はいずれも、上向きベクトルの始点かつ右向きベクトルの始点になっている輪郭点であることを意味している。

20

#### 【0119】

尚、「\_zzz」の部分は、この付与情報をもつ輪郭点が、端点部から抽出されたものであることを意味していて、図28の端点情報の欄に記載してある。ここで、\_TRは、線素の上端部の右側(Top Right)から抽出された端点であることを示している(図29(a)参照)。また、\_LTは、線素の左端部の上側(Left Top)から抽出された端点であることを示している(図29(b)参照)。

#### 【0120】

また、\_RBは、線素の右端部の下側(Right Bottom)から抽出された端点であることを示している(図29(c)参照)。また、\_BRは、線素の下端部の右側(Bottom Right)から抽出された端点であることを示している(図29(d)参照)。

30

#### 【0121】

また、\_LBは、線素の左端部の下側(Left Bottom)から抽出された端点であることを示している(図29(b)参照)。また、\_BLは、線素の下端部の左側(Bottom Left)から抽出された端点であることを示している(図29(d)参照)。

#### 【0122】

また、\_RTは、線素の上端部の右側(Right Top)から抽出された端点であることを示している(図29(c)参照)。\_TLは、線素の左端部の上側(Top Left)から抽出された端点であることを示している(図29(a)参照)。

40

#### 【0123】

線素の端点ではない部分(以降、非端点部という)から抽出される輪郭点には、この「\_zzz」部のない輪郭点情報が付与され、これらの例を図30(a)～(d)に示している。

#### 【0124】

但し、注目画素が黒画素の場合で、近傍の画素の状態によって、図12～図27に各状態における輪郭点とその付与情報を抽出する際の実際の処理では、図28の輪郭点情報は「値」の欄の対応する位置に表記される数値で抽出されるように構成するものとする。

50

## 【 0 1 2 5 】

図 3 0 において、斜めのチェッカーパターンで示される 印は黒画素を示し、点線の印は白画素を示している。

## 【 0 1 2 6 】

これら輪郭ベクトルの始点・終点位置は、主走査方向及び副走査方向共に、画素と画素の中間の位置にあるものとする。また、主走査方向及び副走査方向共に、画素のある位置は正整数で示され、画素位置を 2 次元の座標で表現をする。

## 【 0 1 2 7 】

但し、ここでは小数の表現を避けるために便宜上、以降の画素位置を偶数のみで表現することにし、始点、終点の位置を奇数の整数で表現することにする。即ち、 $m$  画素  $\times$   $n$  画素の画像は、 $2m \times 2n$  の正の偶数（整数）の座標表現で表わすものとする。このように、主走査方向を  $x$  座標、副走査方向を  $y$  座標とした 2 次元座標であらわすとき、これ以降、2 値画像は、それぞれが  $m$  画素よりなる  $n$  ラスタで構成される  $m \times n$  画素（ $m, n$  は正の整数）で表現するものとする。また、第  $j$  番目のラスタの第  $i$  番目の画素位置を  $(2i, 2j)$ （ $i, j$  は正の整数で、 $i \leq m, j \leq n$ ）で表現するものとする。また、主走査方向（ $x$  座標）は左から右に向かう方向が正の方向であり、副走査方向（ $y$  座標）は上から下に向かう向きが正の方向である。

## 【 0 1 2 8 】

以下、端点間毎輪郭（周回）ベクトル抽出部 3 1 1 0 の一連の動作を、図 3 1 A ~ 図 3 3 に従って説明を加える。

## 【 0 1 2 9 】

実施形態 1 は、特許文献 4 に比し、4 - 連結に細線化された 2 値画像を端点、交点を検出しつつ、端点や交点をつなぐ互いに独立した線素や閉曲線の集まりとして、それぞれの線成分に対応する独立したベクトル群を抽出する。加えて、各輪郭点に輪郭点情報をも付与して抽出する点で、その機能は大きく異なる。また、図 1 1 ~ 図 2 7 に示したように、処理対象として用いる  $3 \times 3$  の画素パターン群も大きく異なる。但し、 $3 \times 3$  の 9 画素で構成される走査窓を用いて、画像をラスタ走査し、逐次  $3 \times 3$  の画素パターンとして注目画素とその 8 近傍の画素の状態を判断しながら、輪郭点を抽出していく点は、特許文献 4 と共通する。相違点としては、各  $3 \times 3$  の画素パターンに応じた輪郭点と輪郭点情報を抽出する点である。

## 【 0 1 3 0 】

図 3 1 A は本発明の実施形態 1 の端点間輪郭（周回）ベクトル抽出処理全体を示すフローチャートである。

## 【 0 1 3 1 】

尚、この処理は、画像処理装置の CPU 5 1 9 の制御によって実現される。また、本処理は、特許文献 4 の図 2 2 で示されるフローチャートに準ずるものである。しかしながら、特許文献 4 では、輪郭点情報の概念はない。そのため、実施形態 1 では、この輪郭点情報を、抽出した輪郭点に付加された属性情報として考える。また、特許文献 4 で抽出した輪郭点の接続情報の処理をする際にも、それぞれの輪郭点の輪郭点情報も各輪郭点の情報の一部として付随したものとして一貫して扱うようにする。

## 【 0 1 3 2 】

これは、各輪郭点データのデータ形式の内部にと輪郭点の座標値のみならず、その、輪郭点データをもその一部として取り込んだデータ構造として扱えば、容易に実現できるので、詳述はしない。

## 【 0 1 3 3 】

まず、ステップ S 1 で、ベクトル列抽出処理を実行する。これは、2 値画像データからベクトル列を各輪郭点に付随させて輪郭点情報込みで抽出する。そして、各ベクトルの始点の座標及びこのベクトルに流入してくる（このベクトルの始点の座標が終点となっている）ベクトル、流出してゆく（このベクトルの終点の座標が始点となっている）ベクトルを、図 3 1 B 及び図 3 1 C に示すテーブル形式で出力する。

## 【 0 1 3 4 】

次に、ステップ S 2 で、図 3 1 B 及び図 3 1 C に示すテーブルから流入及び流出ベクトルの項目番号を辿る。これにより、図 3 2 に示すような画像中の総輪郭線数、各輪郭線毎の輪郭線の総点数、輪郭線中の各点の x 座標、y 座標と各点の輪郭点情報の組み合わせを記憶したベクトル列テーブル 3 2 0 1 を作成する。

## 【 0 1 3 5 】

次に、ステップ S 3 で、ディスク I / O 5 2 1 を介して、このベクトル列テーブル 3 2 0 1 の情報をファイル形式で、ハードディスク装置 5 2 2 に記憶して、一連の動作を終了する。つまり、このファイルによって、画像中の領域別のベクトルデータが構成される。

## 【 0 1 3 6 】

次に、図 3 1 A のステップ S 1 のベクトル列抽出処理の詳細について、図 3 3 を用いて説明する。

## 【 0 1 3 7 】

図 3 3 は本発明の実施形態 1 の図 3 1 A のステップ S 1 のベクトル列抽出処理の詳細を示すフローチャートである。

## 【 0 1 3 8 】

まず、ステップ S 1 1 0 で、注目画素が白画素か黒画素かを判定する。白画素の場合（ステップ S 1 1 0 で Y E S）、ステップ S 1 3 0 へ進む。一方、黒画素の場合（ステップ S 1 1 0 で N O）、ステップ S 1 2 0 へ進む。

## 【 0 1 3 9 】

ステップ S 1 2 0 で、注目画素の周囲 8 画素の状態を見て、その状態に応じた処理を実行する。

## 【 0 1 4 0 】

ステップ S 1 3 0 で、画素位置を更新する。そして、ステップ S 1 4 0 で、最終画素の処理が終了したか否かを判定する。終了していない場合（ステップ S 1 4 0 で N O）、ステップ S 1 1 0 へ戻り、次の注目画素も同様に処理を行う。一方、終了している場合（ステップ S 1 4 0 で Y E S）、元のルーチンにリターンする。

## 【 0 1 4 1 】

ここで、ステップ S 1 のベクトル列抽出処理の具体例として、図 1 2 に示す状態（ケース（c a s e）0）を処理する場合を説明する。ここで、注目画素（図 1 2 の中心画素）の座標は、 $(2i, 2j)$  であるとする。

## 【 0 1 4 2 】

まず、 $(2i - 1, 2j - 1)$  を水平ベクトルの始点として登録する。即ち、図 3 1 B の水平ベクトルカウンタ 1 2 3 0 が示す水平ベクトル始点の x 座標テーブルの欄（図 3 1 B の 1 2 3 1）に  $(2i - 1)$  を格納し、同じく水平ベクトル始点 y 座標テーブルの欄（図 3 1 B の 1 2 3 2）に  $(2j - 1)$  を格納する。そして、この水平ベクトルに流入してくる元の垂直ベクトルは、図 3 1 C の垂直ベクトルカウンタ 1 2 4 0 が、この時点で指している垂直ベクトルの始点座標テーブル内の位置の次の位置に格納されるベクトルである。また、この水平ベクトルが流出していく先の垂直ベクトルは、この時点で垂直ベクトルカウンタ 1 2 4 0 が指している垂直ベクトルの始点座標テーブルの位置にある。

## 【 0 1 4 3 】

即ち、図 3 1 B に示す水平ベクトルの流入ベクトル項目番号の欄において、水平ベクトルカウンタ 1 2 3 0 が示す位置（図 3 1 B の例では、1 2 3 3）に、垂直ベクトルカウンタ 1 2 4 0 の値に 1 加えた値を格納する。また、水平ベクトルの流出ベクトル項目番号の欄には、水平ベクトルカウンタ 1 2 3 0 が示す位置（図 3 1 B の例では、1 2 3 4）に、垂直ベクトルカウンタ 1 2 4 0 の値を格納する。

## 【 0 1 4 4 】

次に、同様に、 $(2i + 1, 2j - 1)$  を垂直ベクトルの始点として登録する。この垂直ベクトルに流入する水平ベクトルは、1 回目に登録した水平ベクトルであると登録する。また、この垂直ベクトルが流出する水平ベクトルは、水平ベクトルの始点座標テーブル

10

20

30

40

50

の 1 回目に登録された水平ベクトルの次の位置に格納される水平ベクトルであると登録する。

【 0 1 4 5 】

即ち、図 3 1 C を参照して説明すると、欄 2 4 1 には  $(2i + 1)$  が、欄 2 4 2 には  $(2i - 1)$  が、更に、欄 2 4 3 には、この時の水平ベクトルカウンタ 1 2 3 0 の値が、欄 2 4 3 にはこの時の水平ベクトルカウンタ 1 2 3 0 の値に + 1 した値が格納される。

【 0 1 4 6 】

次に、水平ベクトルカウンタ 1 2 3 0 の内容を + 1 する。次に、 $(2i + 1, 2j + 1)$  を水平ベクトルの始点座標として登録し、この水平ベクトルに流入する垂直ベクトルは、2 回目に登録したベクトルであると登録する。また、この水平ベクトルが流出する垂直ベクトルは、2 回目に登録された垂直ベクトルの始点座標テーブル内における次の位置に格納される垂直ベクトルであると登録する。

10

【 0 1 4 7 】

次に、垂直ベクトルカウンタ 1 2 4 0 の内容を + 1 する。次に、 $(2i - 1, 2j + 1)$  を垂直ベクトルの始点座標として登録し、この垂直ベクトルに流入する水平ベクトルは、3 回目に登録された水平ベクトルであると登録する。また、この垂直ベクトルが流出する水平ベクトルは、1 回目で登録した水平ベクトルであると登録する。こうして最後に、水平ベクトルカウンタ 1 2 3 0 及び垂直ベクトルカウンタ 1 2 4 0 を共に + 1 して処理を終了する。

【 0 1 4 8 】

20

次に、図 3 3 のステップ S 1 2 0 の処理の詳細について、図 3 4 を用いて説明する。

【 0 1 4 9 】

図 3 4 は本発明の実施形態 1 のステップ S 1 2 0 の処理の詳細を示すフローチャートである。

【 0 1 5 0 】

ステップ S 2 1 0 で、CPU 5 1 9 のレジスタを「0」にクリアする。次に、ステップ S 2 2 で、図 1 1 の「0」で示された位置の画素の状態（以下、 $f(0)$  で表現する）が黒画素（以降、「1」で表わす）であるか否かを判定する。つまり、 $f(0) = 1$  であるか否かを判定する。 $f(0) = 1$  である場合（ステップ S 2 2 0 で YES）、ステップ S 2 3 0 へ進み、レジスタの内容に 1 を加える。一方、 $f(0) = 0$ 、即ち、白画素（以降、「0」で表わす）である場合、ステップ S 2 4 0 へ進む。

30

【 0 1 5 1 】

次に、ステップ S 2 4 0 で、図 1 1 の「1」の位置の画素の状態が黒画素であるか否かを判定する。つまり、 $f(1) = 1$  であるか否かを判定する。 $f(1) = 1$  である場合（ステップ S 2 4 0 で YES）、ステップ S 2 5 0 へ進み、レジスタの内容に 2 を加える。一方、 $f(1) = 0$  である場合（ステップ S 2 4 0 で NO）、ステップ S 2 6 0 に進む。

【 0 1 5 2 】

次に、ステップ S 2 6 0 で、図 1 1 の「2」の位置の画素の状態が黒画素であるか否かを判定する。つまり、 $f(2) = 1$  であるか否かを判定する。 $f(2) = 1$  である場合（ステップ S 2 6 0 で YES）、ステップ S 2 7 0 へ進み、レジスタの内容に 4 を加える。一方、 $f(2) = 0$  である場合（ステップ S 2 6 0 で NO）、ステップ S 2 8 0 に進む。

40

【 0 1 5 3 】

次に、ステップ S 2 8 0 で、図 1 1 の「3」の位置の画素の状態が黒画素であるか否かを判定する。つまり、 $f(3) = 1$  であるか否かを判定する。 $f(3) = 1$  である場合（ステップ S 2 8 0 で YES）、ステップ S 2 9 0 へ進み、レジスタの内容に 8 を加える。一方、 $f(3) = 0$  である場合（ステップ S 2 8 0 で NO）、ステップ S 3 0 0 に進む。

【 0 1 5 4 】

ステップ S 3 0 0 で、レジスタの保持する値（処理番号）に応じた処理を実行する。

【 0 1 5 5 】

これにより、レジスタの内容は、図 1 1 に示した「0」、「1」、「2」、「3」の画

50

素位置の各画素の状態に応じて、0～15の値を取り得る。これらは、その値に応じてそれぞれ図12の(ケース(case)0)～図27(ケース15)に示されている。

【0156】

即ち、ステップS300で、図12～図27に示されるように定義された輪郭点とその輪郭点情報とをそれぞれのケースに応じて抽出する。

【0157】

このように、端点間毎輪郭(周回)ベクトル抽出部3110は、1回のラスト走査を行なう。これにより、入力される4-連結に細線化された2値画像(線図形)から、それらを構成する端点や交点間をつなぐ独立した線や閉曲線毎に対応するベクトル列の集まり(「端点間毎線素粗輪郭ベクトル」とその輪郭点情報を抽出し、その処理結果を出力する。

10

【0158】

これらは、例えば、図32に示す形式として出力される。同図における第一輪郭や第二輪郭、第a輪郭といったまとまりとして各線素から抽出されたベクトル列を示し、それらの各輪郭点毎に輪郭点情報が付与された形式となっている。

【0159】

ここで、図35に線図形の入力例と、これから抽出される端点間毎輪郭(周回)ベクトル、即ち、端点間毎線素粗輪郭ベクトル(太めに表現されたベクトル列)と輪郭点情報を示す。

【0160】

次に、端点間毎線芯(周回)ベクトル生成部3120の動作を、図36を用いて説明する。

20

【0161】

図36は本発明の実施形態1の端点間毎線芯(周回)ベクトル生成部の動作を示すフローチャートである。

【0162】

尚、この処理は、画像処理装置のCPU519の制御によって実現される。

【0163】

まず、ステップS1000で、端点間毎輪郭(周回)ベクトル抽出部3110より端点間毎輪郭(周回)ベクトルを入力する。ここで、端点間毎輪郭(周回)ベクトルは、線図形の端点(交点)位置に対応する部分から抽出されたものか否か等を表す情報(付与信息)を含む、端点や交点間をつなぐ独立した線や閉曲線毎のそれぞれに対応したベクトル列の集まりである。このベクトル列の集まりは、端点間毎線素粗輪郭ベクトルである。

30

【0164】

次に、ステップS1100で、ステップS1000で入力した図32に示す形式の端点間毎輪郭(周回)ベクトル群から、その中に含まれる1つのベクトル列(図32の第k輪郭(1 k a))を当面の処理対象として入力する。ここでは、ステップS1100が実行されるたびに順次昇順に定める。

【0165】

次に、ステップS1200で、ステップS1100で定めたベクトル列(総点数pとする)の中の1つのベクトルの輪郭点(第q点(1 q p))を当面の処理対象としてm1つのベクトルを入力する。ここでは、ステップS1200が実行されるたびに順次昇順に定める。

40

【0166】

次に、ステップS1300で、ステップS1200で定めた輪郭点の輪郭点情報を参照して、この輪郭点を終点とするベクトルか、もしくは、この輪郭点を始点とするベクトルのいずれかが左向き(LEFT)か否かを判定する。左向きである場合(ステップS1300でYES)、ステップS1400に進む。一方、左向きではない場合(ステップS1300でNO)、ステップS1500に進む。

【0167】

50

ステップS 1 4 0 0で、輪郭点(座標値( $x, y$ ) = ( $2 * i, 2 * j$ ))とする)の $y$ 座標値を1減じて(副走査方向に半画素分原点側にずらして)、つまり、 $y = 2 * i - 1$ を実行して、ステップS 1 6 0 0に進む。

【0 1 6 8】

ステップS 1 5 0 0で、輪郭点の $y$ 座標値を1増やして(副走査方向に半画素分原点とは反対側にずらして)、つまり、 $y = 2 * i + 1$ を実行して、ステップS 1 6 0 0に進む。

【0 1 6 9】

ステップS 1 6 0 0で、輪郭点の輪郭点情報を参照して、この輪郭点を終点とするベクトルか、もしくは、この輪郭点を始点とするベクトルのいずれかが上向き(UP)か否かを判定する。上向きである場合(ステップS 1 6 0 0でYES)、ステップS 1 7 0 0に進む。一方、上向きでない場合(ステップS 1 6 0 0でNO)、ステップS 1 8 0 0に進む。

【0 1 7 0】

ステップS 1 7 0 0で、輪郭点の $x$ 座標値を1増やして(主走査方向に半画素分原点とは反対側にずらして)、つまり、 $x = 2 * i + 1$ を実行して、ステップS 1 9 0 0に進む。

【0 1 7 1】

ステップS 1 8 0 0で、輪郭点の $x$ 座標値を1減じて(主走査方向に半画素分原点側にずらして)、つまり、 $x = 2 * i - 1$ を実行して、ステップS 1 9 0 0に進む。

【0 1 7 2】

ステップS 1 9 0 0で、ステップS 1 1 0 0で定めたベクトル列内の全ての輪郭点に対する処理が終了したか否かを判定する。終了している場合(ステップS 1 9 0 0でNO)、ステップS 2 0 0 0へ進む。一方、終了していない場合(ステップS 1 9 0 0でYES)、同ベクトル列内の次の輪郭点に、ステップS 1 3 0 0からステップS 1 9 0 0までの処理を施すべくステップS 1 2 0 0にもどる。

【0 1 7 3】

ステップS 2 0 0 0で、ステップS 1 0 0 0で入力した端点間毎線素粗輪郭ベクトル中の全てのベクトル列の処理が終了したか否かを判定する。終了している場合(ステップS 1 9 0 0でYES)、ステップS 2 1 0 0へ進む。一方、終了していない場合(ステップS 1 9 0 0でNO)、次のベクトル列に対して、ステップS 1 2 0 0からステップS 2 0 0 0までの処理を施すべくステップS 1 1 0 0へ戻る。

【0 1 7 4】

ステップS 2 1 0 0で、ステップS 1 0 0 0で入力した端点間毎線素粗輪郭ベクトルに対する処理結果を出力する。

【0 1 7 5】

尚、ステップS 1 3 0 0やステップS 1 6 0 0での判定は、上述の図28における「値」欄の数値を輪郭点情報としている。そのため、これらの数値を2進数で表現した際のLSBをbit 0、MSBをbit 7とすると、ステップ1 3 0 0ではbit 2(図28では「右向(0)左向(1)」欄)を調べて、0なら右向き、1なら左向きと判定することができる。ステップS 1 6 0 0では、bit 3(図28では「上向(0)下向(1)」欄)を調べて、0なら上向き、1なら下向きと判定することができる。

【0 1 7 6】

このようにして、端点間毎輪郭(周回)ベクトル抽出部3 1 1 0から端点間毎輪郭(周回)ベクトルを入力する。そして、それに含まれる付与情報を利用して、一画素幅の幅を持った輪郭(周回)ベクトルの各ベクトルの位置を、それぞれ半画素分ずつ相応する方向へ予め定める規則により微調整する。これにより、幅0に線芯化された端点間毎線芯(周回)化済ベクトルとする。これらは、端点間毎線芯(周回)化済ベクトルもまた、例えば、図32に示す形式として表現でき、実施形態1では、この形式で出力されるものとする。

10

20

30

40

50

## 【 0 1 7 7 】

ここで、上述の図 3 5 に、端点間毎線素粗輪郭ベクトル（太めに表現されたベクトル列）と輪郭点情報から幅 0 に線芯化された端点間毎線芯（周回）化済ベクトル（入力 2 値画像の画素上の位置に細めに描かれたベクトル列）の例を示す。

## 【 0 1 7 8 】

次に、端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0 の動作を説明する。

## 【 0 1 7 9 】

端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0 は、次工程である端点間毎補助ベクトル入り（周回）ベクトル平滑部 3 2 2 0 により平滑化（関数近似）処理される際に、補助ベクトルを挿入する。具体的には、端点部分に相当するベクトルが、他のベクトルと統合されて端点位置が不明にならぬように、その端点部分に相当する部分が端点として保存される（ベジェ曲線のアンカーポイントとなる）ように補助ベクトルを挿入する。

10

## 【 0 1 8 0 】

加えて、次々工程である端点間毎平滑化済ベクトル同定（非周回化）部 3 2 3 0 で、端点間毎補助ベクトル入り（周回）ベクトル平滑部 3 2 2 0 から得る平滑化（関数近似）済端点間毎補助ベクトル入り（周回）ベクトルから、始端点・終端点情報をも生成する。具体的には、各平滑化済の（周回）ベクトル毎にそれぞれの始端点と終端点となるベクトルを同定して、始端点と終端点の間のみの部分の非周回型のベクトル列を生成するために用いられる、各平滑化済の（周回）ベクトル毎の始端点・終端点情報をも生成する。

20

## 【 0 1 8 1 】

実施形態 1 では、端点間毎補助ベクトル入り（周回）ベクトル平滑部 3 2 2 0 は、特許文献 5 に開示される方法及び装置で構成されるものとする。このように構成することで、これまでの、必ずしも細線化されていない 2 値画像の輪郭（アウトライン）ベクトル用の平滑化手段を用いて、太さを意識せずに、細線のみで描かれている線画としてのベクトルの平滑化まで構成することが可能となる。

## 【 0 1 8 2 】

特許文献 5 によれば、ベクトル平滑化（関数近似）の流れは、以下の如きである。まず、粗輪郭データと称して、2 値のラスタ画像のデータより抽出された、水平ベクトル、垂直ベクトルが交互に並ぶ構成となる輪郭データを入力とする。そして、粗輪郭上の線分より接線線分を抽出する。抽出された接線線分よりアンカーポイントを抽出し、抽出されたアンカーポイント間の線分により構成されるグループを 2 次もしくは 3 次ベジェ曲線、及び直線をあてはめ、あるいは、ベジェ曲線近似を行い、3 次もしくは 2 次のベジェ曲線により置き換える。

30

## 【 0 1 8 3 】

ここで、アンカーポイントとして定めた点をもとに、アンカーポイント間にある輪郭点を 1 つのグループとして関数近似して方法は、関数近似法としての基本的な手法である。また、アンカーポイントは、その間が、2 次もしくは 3 次ベジェ曲線及び直線のいずれに関数近似されたとしても、アンカーポイント自体の位置（座標値）は、変化しないという特徴をもつ。

40

## 【 0 1 8 4 】

そこで、端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0 では、端点間毎線芯（周回）化済ベクトルの端点のそれぞれを、一方を始端点と他方を終端点とする。また、それらがともにアンカーポイントとなるように始端点と終端点間に補助ベクトル（補助輪郭点）を挿入していく処理をする。

## 【 0 1 8 5 】

特許文献 5 によれば、接線線分とするベクトルの抽出条件として、

- ( 1 ) 注目するベクトルの前後のベクトルの向きが互いに逆向きであるもの
- ( 2 ) 抽出済みの主接線線分に隣接し、ベクトルの長さ  $L_1$  が  $L_1$  <sub>4</sub> を満たすもの
- ( 3 ) ベクトルの長さ  $L_2$  が  $L_2$  <sub>5</sub> を満たすもの

50

等の条件が開示されている。

【 0 1 8 6 】

また、アンカーポイントの抽出に関して、抽出された接線線分上に新たな点を抽出し、それをアンカーポイントとする。アンカーポイントは、接線線分の端 2 つに対しそれぞれ抽出される。よって、1 つの接線線分に対し 2 つのアンカーポイントが抽出されるが、その 2 つのアンカーポイントが一致した場合には、1 つのアンカーポイントのみ抽出されることになる。

【 0 1 8 7 】

2 つのアンカーポイントが抽出される場合は、アンカーポイントに挟まれた部位は自動的にオブジェクト上の直線となる。接線線分上の 1 つの端点に対するアンカーポイント抽出方法としては、注目する接線線分であるベクトルに隣接するベクトルが接線線分であれば、この隣接する側の端点をアンカーポイントとする等が開示されている。

10

【 0 1 8 8 】

そこで、端点間毎補助ベクトル入り（周回）ベクトル生成部 3 2 1 0 は、点間毎線芯（周回）化済ベクトルの端点のそれぞれを、一方を始端点と他方を終端点とし、かつ、それらがともにアンカーポイントとなるように、補助点を挿入する。

【 0 1 8 9 】

ここで、図 3 7 ( a ) を用いて、始端点としたい端点  $P_0$  の直前に挿入する補助輪郭点  $P_{-1}$  の挿入方法を説明する。

【 0 1 9 0 】

20

補助輪郭点  $P_{-1}$  をこれを始点とする補助ベクトル  $V_{-1}$  が、始端点としたい端点  $P_0$  の直後の輪郭点  $P_1$  を始点とするベクトル  $V_1$  と逆向きになるように、補助輪郭点  $P_{-1}$  を定める。このようにすれば、上述の特許文献 5 の接線線分の抽出条件の ( 1 ) から、ベクトル  $V_0$  が接線線分となる。

【 0 1 9 1 】

また、補助輪郭点  $P_{-1}$  をこれを始点とする補助ベクトル  $V_{-1}$  が上述の特許文献 5 の接線線分の抽出条件の ( 3 ) を満たすように十分な長さを持つようにする。このようにすれば、補助ベクトル  $V_{-1}$  もまた接線線分となる。つまり、補助ベクトル  $V_{-1}$  とベクトル  $V_0$  とが共に接線線分となる。これにより、上述のアンカーポイント抽出方法の「注目する接線線分であるベクトルに隣接するベクトルが接線線分であれば、この隣接する側の端点をアンカーポイントとする」という条件から、始端点としたい端点  $P_0$  をアンカーポイントにできる。

30

【 0 1 9 2 】

次に、図 3 7 ( b ) を用いて、終端点としたい端点  $P_n$  の直後に挿入する補助輪郭点  $P_{n+1}$  の挿入方法を説明する。

【 0 1 9 3 】

補助輪郭点  $P_{n+1}$  をこれを終点とする補助ベクトル  $V_n$  が、終端点としたい端点  $P_n$  の直前の輪郭点  $P_{n-1}$  を終点とするベクトル  $V_{n-2}$  と逆向きになるように、補助輪郭点  $P_{n+1}$  を定める。このようにすれば、上述の特許文献 5 の接線線分の抽出条件の ( 1 ) から、ベクトル  $V_{n-1}$  が接線線分となる。また、補助輪郭点  $P_{n+1}$  をこれを終点とする補助ベクトル  $V_n$  が上述の特許文献 5 の接線線分の抽出条件の ( 3 ) を満たすように十分な長さを持つようにする。

40

【 0 1 9 4 】

このようにすれば、補助ベクトル  $V_n$  もまた接線線分となる。つまり、補助ベクトル  $V_{n-1}$  とベクトル  $V_n$  とが共に接線線分となる。これにより、上述のアンカーポイント抽出方法の「注目する接線線分であるベクトルに隣接するベクトルが接線線分であれば、この隣接する側の端点をアンカーポイントとする」という条件から、終端点としたい端点  $P_n$  をアンカーポイントにできる。

【 0 1 9 5 】

このように、始端点と終端点がアンカーポイントになるように、それぞれ、始端点の直

50

前への挿入点と終端点の直後への挿入点を定める。その後、これらの挿入点が水平ベクトルと垂直ベクトルの連続で連結され、始端点から終端点までの本来の輪郭点列と終端点の直後から始端点の直前までの挿入点をへて再び始端点につながる一連の輪郭点列を生成する。そして、この一連の輪郭点列の中で、どれが、始端点と終端点であるかを同定するための情報とを加えた始端点・終端点情報を生成する。

【0196】

実施形態1では、輪郭点列中の始端点の座標値と終端点の座標値そのものを改めて別途生成することで、始端点・終端点情報とする。

【0197】

次に、端点間毎補助ベクトル入り（周回）ベクトル生成部3210の動作について、図38を用いて説明する。

【0198】

図38は本発明の実施形態1の端点間毎補助ベクトル入り（周回）ベクトル生成部の動作を示すフローチャートである。

【0199】

尚、この処理は、画像処理装置のCPU519の制御によって実現される。

【0200】

まず、ステップS3000で、端点間毎線芯（周回）ベクトル生成部3120から幅0に線芯化された端点間毎線芯（周回）化済ベクトルを入力する。

【0201】

次に、ステップS3100で、ステップS3000で入力した図32に示す形式の端点間毎線芯（周回）化済ベクトルから、その中に含まれる1つのベクトル列（図32の第k輪郭（1 k a））を当面の処理対象として順に定める。

【0202】

次に、ステップS3200で、ステップS3100で定めたベクトル列（総点数pとする）の中の1つのベクトルの輪郭点（第q点（1 q p））として、1つのベクトルを入力する。そして、ステップS3300で、始端点が見つかるまでの間、ステップS3200を実行するたびに、第1点から第p点までを昇順に順次1点ずつ当面の処理対象として定める。

【0203】

次に、ステップS3300で、ステップS3200で定めた輪郭点の輪郭点情報を参照して、この輪郭点が始端点とすべきベクトルであるか否かを判定する。この輪郭点が始端点とすべきベクトルである場合（ステップS3300でYES）、ステップS3600に進む。一方、この輪郭点が始端点とすべきベクトルでない場合（ステップS3300でYES）、ステップS3400に進む。

【0204】

ここで、当該輪郭点が始端点とすべきベクトルであるか否かは、その輪郭点の輪郭点情報を参照して行う。具体的には、図28中のUP\_\_RIGHT\_\_LT（03H）、DOWN\_\_LEFT\_\_RB（0FH）、LEFT\_\_UP\_\_BL（17H）、RIGHT\_\_DOWN\_\_TR（1BH）のいずれかであるか否かをもちてその判定を行う。これらは、図28の「値」の数値のbit1とbit0が共に1であるか否かを調べることで実現することもできる。

【0205】

ステップS3400で、ステップS3100で入力した端点間毎線芯（周回）化済ベクトルの中に含まれる1つのベクトル列中の全てのベクトルの処理が終了したか否かを判定する。終了していない場合（ステップS3400でNO）、当該ベクトル列中の次のベクトルに対してステップS3200の判定を施すべくステップS3200へ戻る。一方、終了している場合（ステップS3400でYES）、ステップS3100で入力したベクトル列中には、始端点候補が存在しなかった場合であり、ステップS3500へ進む。

【0206】

10

20

30

40

50

ステップS3500で、当該ベクトル列は端点のない閉ループである旨を示す閉ループマーカを該ベクトル列に対して付与した上で、それに加えて、該ベクトル列中の一連の各ベクトルをそのまま出力する。

【0207】

この時のデータ形式を、図39(a)に示す。同図において、対象としたベクトル列は第s番目である場合で、そのベクトル列中にはv個のベクトルが含まれている場合の例である。尚、閉ループマーカとしては、端点座標としては、本来ありえない(-1, -1)の座標値を2点分、始端点・終端点情報として付与して出力している。ステップS3500の処理を終了すると、ステップS4200へ進む。

【0208】

一方、ステップS3600で、輪郭点を始端点として登録し、かつ、その輪郭点を始端点～終端点間の1ベクトルとして出力して、ステップS3700へ進む。

【0209】

ステップS3700で、ステップS3200と同様に、ステップS3100で定めたベクトル列(総点数pとする)の中の1つのベクトルの輪郭点(第q点(1 ≤ q ≤ p))とする。そして、ステップS3800で終端点が見つかるまでの間、ステップS3700を実行するたびにステップS3200で定められた輪郭点以降の点から第p点に向けて昇順に順次1点ずつ当面の処理対象として定め、ステップS3800に進む。

【0210】

次に、ステップS3800で、ステップS3700で定めた輪郭点の輪郭点情報を参照して、この輪郭点が終端点とすべきベクトルであるか否かを判定する。この輪郭点が終端点とすべきベクトルである場合(ステップS3800でYES)、ステップS4000に進む。一方、この輪郭点が終端点とすべきベクトルでない場合(ステップS3800でNO)、ステップS3900に進む。

【0211】

ここで、当該輪郭点が終端点とすべきベクトルか否かは、その輪郭点の輪郭点情報を参照して行う。具体的には、図28中のUP\_\_RIGHT\_\_TL(01H)、DOWN\_\_LEFT\_\_BR(0DH)、LEFT\_\_UP\_\_LB(15H)、RIGHT\_\_DOWN\_\_RT(19H)のいずれかであるか否かをもってその判定を行う。これらは、図28の「値」の数値のbit1とbit0がそれぞれ0と1であるか否かを調べることで実現することもできる。

【0212】

ステップS3900で、輪郭点を始端点～終端点間の1ベクトルとして出力して、ステップS3700へ戻る。

【0213】

ステップS4000で、輪郭点を終端点として登録し、かつ、その輪郭点を始端点～終端点間の1ベクトルとして出力して、ステップS4100へ進む。

【0214】

ステップS4100で、当該ベクトル列の始端点と終端点の両方が定まり、かつ、始端点～終端点間の一連の各ベクトルも出力された状態となっている。ここでは、終端点～始端点の間に挿入される一連の補助ベクトルを生成する。尚、ステップS4100の内容の詳細は、図40のフローチャートと図41及び図42を用いて追ってさらに説明を加える。ステップS4100の処理を終了すると、ステップS4200へ進む。

【0215】

ステップS4200で、ステップS3000で入力した端点間毎線芯(周回)化済ベクトル中の全てのベクトル列の処理が終了したか否かを判定する。終了している場合(ステップS4200でYES)、ステップS4300へ進む。一方、終了していない場合(ステップS4200でNO)、次のベクトル列に対してステップS3200からステップS4200までの処理を施すべくステップS3100へ戻る。

【0216】

10

20

30

40

50

ステップS 4 3 0 0で、ステップS 3 0 0 0で入力した端点間毎線芯（周回）化済ベクトルに対する処理結果を出力する。

【0 2 1 7】

図3 9（b）に、対象としたベクトル列に、始端点・終端点の両方が定まり、かつ、始端点～終端点間の一連の各ベクトルと終端点～始端点の間に挿入される一連の補助ベクトルを生成した場合に出力されるそのベクトル列に対する出力例を示す。

【0 2 1 8】

同図において、対象としたベクトル列は第u番目である場合で、そのベクトル列中には始端点と終端点自身も含めて始端点～終端点間の一連の各ベクトルがt個あり、かつ、終端点～始端点の間に挿入される一連の補助ベクトルがr個ある場合の例である。ここでは、第1点は始端点そのものであり、第t点は終端点そのものとなっている。

【0 2 1 9】

次に、図3 8のステップS 4 1 0 0の処理の詳細について、図4 0を用いて説明する。

【0 2 2 0】

図4 0は本発明の実施形態1のステップS 4 1 0 0の処理の詳細を示すフローチャートである。ここでは、当該ベクトル列の始端点と終端点の両方が定まり、かつ、始端点～終端点間の一連の各ベクトルも出力された状態となっており、終端点～始端点の間に挿入される一連の補助ベクトルを生成することが、ステップS 4 1 0 0の処理内容である。

【0 2 2 1】

まず、ステップS 4 1 1 0で、図3 7（a）を用いて説明したように、始端点としたい端点 $P_0$ の直前に挿入する補助輪郭点 $P_{-1}$ の座標値を算出する。即ち、補助輪郭点 $P_{-1}$ をこれを始点とする補助ベクトル $V_{-1}$ が、始端点としたい端点 $P_0$ の直後の輪郭点 $P_1$ を始点とするベクトル $V_1$ と逆向きになるように定める。また、補助輪郭点 $P_{-1}$ をこれを始点とする補助ベクトル $V_{-1}$ が、上述の特許文献5の接線線分の抽出条件の（3）を満たすように十分な長さ（例えば、10画素長以上）を持つように補助輪郭点 $P_{-1}$ を定める。

【0 2 2 2】

次に、ステップS 4 1 2 0で、図3 7（b）を用いて説明したように、終端点としたい端点 $P_n$ の直後に挿入する補助輪郭点 $P_{n+1}$ の座標値を算出する。即ち、補助輪郭点 $P_{n+1}$ をこれを終点とする補助ベクトル $V_n$ が、終端点としたい端点 $P_n$ の直前の輪郭点 $P_{n-1}$ を終点とするベクトル $V_{n-2}$ と逆向きになるように定める。また、補助輪郭点 $P_{n+1}$ をこれを終点とする補助ベクトル $V_n$ が、上述の特許文献5の接線線分の抽出条件の（3）を満たすように十分な長さ（例えば、10画素長以上）を持つように補助輪郭点 $P_{n+1}$ を定める。

【0 2 2 3】

ステップS 4 1 3 0で、算出した補助輪郭点 $P_{-1}$ を始点とする補助ベクトル $V_{-1}$ と、算出した補助輪郭点 $P_{n+1}$ を終点とする補助ベクトル $V_n$ が、共に水平ベクトルであるか否かを判定する。あるいは、共に垂直ベクトルであるか否かを判定する。いずれかの条件を満足する場合（ステップS 4 1 3 0でYES）、ステップS 4 1 4 0に進む。一方、いずれかの条件も満足しない場合（ステップS 4 1 3 0でNO）、ステップS 4 1 6 0に進む。

【0 2 2 4】

ステップS 4 1 4 0での処理では、 $V_{-1}$ と $V_n$ が、共に水平ベクトルあるいは共に垂直ベクトルである場合であり、この場合の例を図4 1に示す。ステップS 4 1 4 0では、終端点としたい端点 $P_n$ から2点目の補助輪郭点 $P_{n+2}$ を算出する。具体的には、 $P_n$ からみて $P_{-1}$ から遠くなる方向に、上述の特許文献5の接線線分の抽出条件の（3）を満たすように十分な長さ（例えば、10画素長以上）を持つように補助輪郭点 $P_{n+2}$ を定める。

【0 2 2 5】

ステップS 4 1 5 0の処理も、ステップS 4 1 4 0と同様に、 $V_{-1}$ と $V_n$ が、共に水平ベクトルあるいは共に垂直ベクトルである場合である。ステップS 4 1 5 0では、終端点としたい端点 $P_n$ から3点目の補助輪郭点 $P_{n+3}$ を算出する。具体的には、水平ベクトルと垂直ベクトルが交互につながる形式で、 $P_{n+2}$ と $P_{-1}$ を連結する様に、端点 $P_n$ の3点後の

10

20

30

40

50

補助輪郭点である点  $P_{n+3}$  を定める。ここで、補助輪郭点  $P_{n+3}$  は、始端点としたい端点  $P_0$  の 2 点前の補助輪郭点でもある。そして、一連の処理を終了すると、図 38 のフローチャートのステップ S 4 2 0 0 に戻る。

#### 【 0 2 2 6 】

一方、ステップ S 4 1 6 0 の処理では、 $V_{-1}$  と  $V_n$  の、どちらか一方が水平ベクトル、他方は垂直ベクトルである場合であり、この場合の例を図 4 2 に示す。ステップ S 4 1 6 0 では、終端点としたい端点  $P_n$  から 2 点目の補助輪郭点  $P_{n+2}$  を算出する。具体的には、 $P_{n+1}$  からみて  $P_{-1}$  から遠くなる方向に、特許文献 5 の接線線分の抽出条件の (3) を満たすように十分な長さ (例えば、10 画素長以上) を持つように補助輪郭点  $P_{n+2}$  を定める。

10

#### 【 0 2 2 7 】

ステップ S 4 1 7 0 の処理も、ステップ S 4 1 6 0 と同様に、 $V_{-1}$  と  $V_n$  が、どちらか一方が水平ベクトル、他方は垂直ベクトルである場合である。ステップ S 4 1 7 0 では、終端点としたい端点  $P_n$  から 3 点目の補助輪郭点  $P_{n+3}$  を算出する。具体的には、 $P_{n+2}$  からみて  $P_{-1}$  から遠くなる方向に、特許文献 5 の接線線分の抽出条件の (3) を満たすように十分な長さ (例えば、10 画素長以上) を持つように補助輪郭点  $P_{n+3}$  を定める。

#### 【 0 2 2 8 】

ステップ S 4 1 8 0 の処理も、ステップ S 4 1 6 0 及びステップ S 4 1 7 0 と同様に、 $V_{-1}$  と  $V_n$  が、どちらか一方が水平ベクトル、他方は垂直ベクトルである場合である。ステップ S 4 1 8 0 では、終端点としたい端点  $P_n$  から 4 点目の補助輪郭点  $P_{n+4}$  を算出する。具体的には、水平ベクトルと垂直ベクトルが交互につながる形式で、 $P_{n+3}$  と  $P_{-1}$  を連結する様に、終端点としたい端点  $P_n$  の 4 点後の補助輪郭点である点  $P_{n+4}$  を定める。ここで、補助輪郭点  $P_{n+4}$  は、始端点としたい端点  $P_0$  の 2 点前の補助輪郭点でもある。そして、一連の処理が終了すると、図 38 のフローチャートのステップ S 4 2 0 0 に戻る。

20

#### 【 0 2 2 9 】

このようにして、端点間毎補助ベクトル入り (周回) ベクトル生成部 3 2 1 0 は、端点間毎線芯 (周回) ベクトル生成部 3 2 1 0 より得られる端点間毎線芯 (周回) 化済ベクトルを入力する。そして、端点間毎補助ベクトル入り (周回) ベクトル平滑部 3 2 2 0 で処理される際に、端点部分に相当するベクトルが、他のベクトルと統合されて端点位置が不明にならぬように補助ベクトルを挿入する。つまり、端点部分に相当する部分が端点として保存される (ベジェ曲線のアンカーポイントとなる) ように補助ベクトルを挿入することにより、端点間毎補助ベクトル入り (周回) ベクトルを生成する。

30

#### 【 0 2 3 0 】

加えて、補助ベクトルを挿入された各端点間毎補助ベクトル入り (周回) ベクトルのそれぞれに対し、どのベクトルが端点部にあたるかを明示する始端点・終端点情報をも生成する。このようにして生成された処理結果は、例えば、図 3 2 に示す形式の各輪郭データ部分のそれぞれを図 3 9 (a) や図 3 9 (b) の各ベクトル列データに置き換えた形として表現でき、実施形態 1 では、この形式で出力されるものとする。

#### 【 0 2 3 1 】

尚、端点間毎補助ベクトル入り (周回) ベクトル平滑部 3 2 2 0 は、先述のように、特許文献 5 に開示される方法及び装置で構成できる。その出力は、端点部分に相当する部分が端点として保存された状態で各 (周回) ベクトル毎に平滑化 (関数近似) された平滑化 (関数近似) 済端点間毎補助ベクトル入り (周回) ベクトルとして生成される。そして、次工程の処理をする端点間毎平滑化済ベクトル同定 (非周回化) 部 3 2 3 0 に出力される。この際、端点間毎補助ベクトル入り (周回) ベクトル生成部 3 2 1 0 で生成された始端点・終端点情報もそのまま次工程の処理をする端点間毎平滑化済ベクトル同定 (非周回化) 部 3 2 3 0 に出力する。

40

#### 【 0 2 3 2 】

端点間毎平滑化済ベクトル同定 (非周回化) 部 3 2 3 0 は、端点間毎補助ベクトル入り (周回) ベクトル平滑部 3 2 2 0 より平滑化 (関数近似) 済端点間毎補助ベクトル入り (

50

周回)ベクトルを入力する。また、端点間毎補助ベクトル入り(周回)ベクトル生成部3210より始端点・終端点情報を入力する。更に、各平滑化済の(周回)ベクトル列毎に、そのベクトル列内の各ベクトルの座標値を、それぞれ始端点と終端点となる輪郭点の座標値と比較していくことにより同定する。そして、始端点と終端点の間のみの部分の非周回型のベクトル列を生成して端点間毎平滑化済(非周回)ベクトルとする。そして、次工程である領域別ベクトル生成部40に出力する。

【0233】

尚、生成された処理結果は、図32に示す形式に準じた形として表現することができる、但し、各輪郭点はそれまでの輪郭点情報の部分をアンカーポイントであるのか否かの属性情報としての付帯情報をもつ形をとる。ここで、同定された始端点を第1点とし、終端点

10

【0234】

即ち、図32の第一輪郭の部分が、第一線素のデータ部分となっており、第1点が第一線素の始端点で、第b点が第一線素の終端点となっているといった形式である。実施形態1では、このような形式で出力されるものとする。

【0235】

このようにして、図10のフローチャートにおける、ステップS50からステップS80のループでは、交点と交点の間を繋ぐ境界線画像毎に、順次に端点間ベクトルが生成されていく。そして、全ての境界線について端点間ベクトル化の処理を終了すると、ステップS100において、領域別のベクトルを生成する領域別ベクトル生成処理を実行する。

20

【0236】

即ち、図43に示すような交点と交点の間を繋ぐ境界線が存在するときに、図44の(a)~(h)に示されるように、1本ずつ、順次に交点と交点の間を繋ぐ境界線画像が生成されては、端点間ベクトルが生成されて行く。

【0237】

図45は、全ての境界線について端点間ベクトル化の処理を終了したものを、画像として再描画した形で模式的に示したものである。図46は、図45に画像として再描画した形で模式的に示した端点間毎平滑化済(非周回)ベクトルの全てを其々のベクトルの端点部を其々の元になった交点と交点の間を繋ぐ境界線画像の各交点を明示した形で画像として再描画し模式的に示したものである。

30

【0238】

ステップS30で抽出された交点やステップS50で生成された交点と交点の間を繋ぐ境界線画像の各交点は、ステップS50~ステップS80の処理を終了しても、端点間毎平滑化済(非周回)ベクトルの各端点としてその位置と位置関係が保たれている。この意味で、ステップS30で抽出された交点は、ステップS50~ステップS80の処理においては、端点として踏襲されて処理が進められていく。

【0239】

ステップS100で、領域別ベクトル生成部40の処理内容(領域別ベクトル生成処理)を実行する。即ち、端点間ベクトル情報と領域分割処理によって得られる各領域の色情報に基づき、各領域にそれを取り囲む関数近似(平滑化)済端点間ベクトル列とその内部色とでなる領域ベクトルに変換する。

40

【0240】

上述のように、領域別ベクトル生成部40は、領域境界ベクトル列同定部410と領域属性付与部420とで構成することが可能である。

【0241】

ここで、ステップS100の領域別ベクトル生成処理の詳細について、図47を用いて説明する。

【0242】

図47は本発明の実施形態1の図10のステップS100の領域別ベクトル生成処理の詳細を示すフローチャートである。

50

## 【 0 2 4 3 】

まず、ステップ S 1 0 1 0 で、注目領域番号を入力する。即ち、ステップ S 1 2 で得られた各領域の中のいずれか 1 つの領域の領域番号を入力する。これは、例えば、ステップ S 1 3 で、R A M 5 2 0 上に生成された、図 2 に示す各領域の領域番号とそれら各領域の色情報（属性情報）の領域テーブル 2 0 1 を参照して、各領域の領域番号を入力する。

## 【 0 2 4 4 】

次に、ステップ S 1 0 2 0 で、ステップ S 1 0 1 0 で入力した領域番号に関連する交点（端点：注目領域関連交点）を、ステップ S 3 1 で生成した図 6 の交点テーブル 6 0 1 を参照して入力する。例えば、ステップ S 1 0 1 0 で入力した領域番号が 2 の場合は、図 6 の交点テーブル 6 0 1 中の隣接領域番号欄に領域番号 2 を含む交点は、C : ( 2 2 , 4 0 )、E : ( 3 8 , 4 0 )、F : ( 4 4 , 2 4 ) の 3 点となる。

10

## 【 0 2 4 5 】

次に、ステップ S 1 0 3 0 で、ステップ S 7 0 で生成した端点間毎補助ベクトル入り（周回）ベクトルの内、ステップ S 1 0 2 0 で入力した交点を端点とするベクトル（領域境界ベクトル）を同定する。これは、図 3 2 に示す形式に準じた形としてのベクトルデータの始端点を表す第 1 点と終端点を表す最終点との組み合わせが、ステップ S 1 0 2 0 で抽出した交点の座標値と一致するものを選出することで実現することができる。

## 【 0 2 4 6 】

図 6 の交点テーブル 6 0 1 のデータの場合には、図 4 6 で表現されたベクトルデータの中で、3 本の関数近似（平滑化）済端点間ベクトルが、注目領域及び領域境界ベクトル列として同定される。具体的には、1 つ目は、F と C を結ぶベクトル（図 4 3 の（ 1 ）から得られる関数近似（平滑化）済端点間ベクトル）である。2 つ目は、C と E を結ぶベクトル（図 4 3 の（ 2 ）から得られる関数近似（平滑化）済端点間ベクトル）である。3 つ目は、E と F を結ぶベクトル（図 4 3 の（ 3 ）から得られる関数近似（平滑化）済端点間ベクトル）である。

20

## 【 0 2 4 7 】

次に、ステップ S 1 0 4 0 で、ステップ S 1 0 3 0 で同定した注目領域の領域境界ベクトル列に対して、対応する領域の色情報（属性情報）を付与する。この処理は、R A M 5 2 0 上に生成された、図 2 に示す各領域の領域番号とそれら各領域の色情報（属性情報）の領域テーブル 2 0 1 を参照する。そして、この参照により、ステップ S 1 0 1 0 で入力した領域番号に相当する領域の色情報（属性情報）を獲得して、その属性情報を注目領域の領域境界ベクトル列に対して付与することで実現される。

30

## 【 0 2 4 8 】

次に、ステップ S 1 0 5 0 で、全ての領域番号に対してステップ S 1 0 1 0 ～ステップ S 1 0 4 0 の処理が終了したか否かを判定する。終了していない場合（ステップ S 1 0 5 0 で N O ）、ステップ S 1 0 1 0 に戻る。一方、終了している場合（ステップ S 1 0 5 0 で Y E S ）、図 1 0 のフローチャートのステップ S 1 0 0 の領域別ベクトル生成処理を終了する。

## 【 0 2 4 9 】

尚、ステップ S 1 0 5 0 の判定は、例えば、図 2 の領域テーブル 2 0 1 における全ての領域番号に対する処理を終了したか否か（領域テーブル 2 0 1 の先頭から処理を始め、最後のデータまで処理をすすめたか否か）で判定可能である。

40

## 【 0 2 5 0 】

このようにして、実施形態 1 における、カラー画像、とりわけ、イラスト画像のベクトル化の一連の流れが構成される。

## 【 0 2 5 1 】

ここで、図 4 8 は、ステップ S 1 0 0 で得られる、各領域毎にそれを取り囲む関数近似（平滑化）済端点間ベクトル列とその内部色とでなる領域ベクトルを、これらを画像として再生した形で模式的に示したものである。

## 【 0 2 5 2 】

50

以下に、上記説明の途中で、後述するとした、領域分割処理、境界点抽出処理、交点抽出処理、交点抽出処理の詳細について説明する。

【0253】

領域分割処理

ステップS12の領域分割処理について、図49を用いて説明する。

【0254】

図49は本発明の実施形態1のステップS12の領域分割処理の詳細を示すフローチャートである。

【0255】

ステップS1201で、処理対象画像に対して、クラスタリング処理を行う。この処理では、まず、ラスタスキャンしたスタートの画素により、最初のクラスタを生成する。そして、次の画素に対して、全てのクラスタ間との乖離度を画素の特徴に基づいて算出する。乖離度が低いほど、画素とクラスタとの特徴が近いと考えられる。つまり、画素とクラスタは、同一とみなせる特徴（色）を持っていると考えられる。

10

【0256】

ここで、ある画素*i*の特徴とは、その画素の画素値（*R<sub>i</sub>*、*G<sub>i</sub>*、*B<sub>i</sub>*）である。また、あるクラスタ*c*の特徴とは、そのクラスタに属する全ての画素の画素値の平均値（*R<sub>i</sub>*、*G<sub>i</sub>*、*B<sub>i</sub>*）である。乖離度とは、例えば、画素*i*とクラスタ*c*の特徴値間の色空間上でのユークリッド距離  $\{(R_c - R_i)^2 + (G_c - G_i)^2 + (B_c - B_i)^2\}$  である。

【0257】

20

ここでは、乖離度の計算にRGB値を用いるが、これに限定されず、他のカラー空間の情報、あるいは、カラー以外の情報を、画素の特徴量として使うことも可能である。そして、乖離度の値が最小となるクラスタ番号を記録し、この乖離度を事前に設定された閾値と比較する。閾値未満である場合は、対象画素を記録されたクラスタに属させる。閾値以上である場合は、対象画素により新たなクラスタを生成する。この処理は全ての画素の処理が終わるまで繰り返し実行する。

【0258】

次に、ステップS1202で、クラスタリング処理結果に基づき、領域統合処理を行う。この領域処理では、まず、分離したい領域数の目標値を入力する。本発明では、この目標値は、何色くらいに分離するかの目安となる。そして、現在のクラスタの数を計数し、現在のクラスタの数を領域数の目標値と比較する。現在のクラスタ数が目標値以上である場合、クラスタの統合を行う。領域統合処理では、各クラスタ間の乖離度を計算し、その中から一番乖離度の低い2つのクラスタを1つに統合する。領域統合処理は、現在のクラスタ数は目標値未満になるまで繰り返し実行する。

30

【0259】

次に、ステップS1203で、領域統合処理結果に基づき、ノイズ領域判定処理を行う。このノイズ領域判定処理では、まず、領域分割処理結果をラベリングし、そして、各ラベル領域の面積を計算する。ラベル領域の面積は、対象領域内の画素数である。ラベル領域の面積は、ある閾値未満である場合は、この領域をノイズ領域と判定する。

【0260】

40

そして、ステップS1204で、このノイズ領域に含まれた各画素毎に、周囲に隣接する領域との乖離度を計算し、処理対象画素を一番乖離度の低い領域に属させるノイズ領域再処理を行う。ノイズ領域判定処理は、全てのラベル領域が処理済になるまで繰り返し実行する。文書画像の領域分離後のグラフィックス領域はより圧縮されて保存されるので、このノイズ領域判定処理とノイズ領域再処理は、圧縮の影響により発生したノイズへの対応策である。

【0261】

以上のようにして、カラー画像を同一とみなせる一貫した特徴（乖離度が閾値以上のもの）を有する、画像上で連結している其々の領域の集まりに分割することが可能となる。

【0262】

50

## 境界点抽出処理

ステップ S 2 1 の水平方向境界点抽出処理とステップ S 2 3 の垂直方向境界点抽出処理とステップ S 2 5 の斜め方向境界点抽出処理について、図 3 を参照して説明する。

## 【 0 2 6 3 】

まず、領域分割の結果を上下順、水平方向 ( ) に走査し、水平方向の各走査線上のある領域から他の領域への切り替わりになる画素を水平方向境界点として抽出する。図 3 では、画像は ( 背景 ) 領域 1、領域 2、領域 3、領域 4、そして、領域 5 に分割される例が示されている。

## 【 0 2 6 4 】

図 3 のように、画像の一番上から水平方向の走査により、第 1 0 行目の B 点で領域番号は 1 から 2 に変わるので、B を水平方向境界点として記録する。同様、下に行く内に、第 1 4 行目の A も水平境界点として記録する。第 2 2 行目の C、第 3 8 行目の E、第 4 4 行目の F、第 6 4 行目の H、I 等も順番に水平方向境界点として記録する。

## 【 0 2 6 5 】

そして、領域分割の結果を左右順、垂直方向 ( ) に走査し、垂直方向の各走査線上のある領域から他の領域への切り替わりになる画素を垂直方向境界点として抽出する。

## 【 0 2 6 6 】

図 3 のように、垂直方向の走査により、第 1 0 列目の D、第 1 5 列目の A、第 2 0 列目の B、第 2 4 列目の F、第 4 0 列目の E、C、第 5 8 列目の H、第 6 6 列目の I 等も順番に垂直方向境界点として記録する。

## 【 0 2 6 7 】

そして、領域分割の結果を上下順、斜め方向に走査し、斜め方向の各走査線上のある領域から他の領域への切り替わりになる画素を斜め方向境界点として抽出する。

## 【 0 2 6 8 】

図 3 のように、斜め方向の走査により、第 1 4 行目、第 1 5 列目の A 点で領域番号は 1 から 2 に変わるので、A 点を斜め方向境界点として記録する。同様に、B、J、C、K、E、G、D、H、I 等も斜め方向境界点として記録する。

## 【 0 2 6 9 】

ここでは、説明を簡単にするため、以上の幾つかの境界点のみを例に用いている。図 5 0 は、B 点と A 点の間の境界線を拡大して、この境界線の水平方向と垂直方向と斜め方向境界点の抽出例を示す図である。

## 【 0 2 7 0 】

図 5 0 ( a ) は A B 境界線付近の画素図であり、画素 ( 1 ) は領域 1 にある画素で、画素 ( 2 ) は領域 2 にある画素である。

## 【 0 2 7 1 】

図 5 0 ( b ) は水平方向境界点抽出処理により、最初に水平方向に領域番号が変わった所、即ち、ハッチングパターンが付されている画素 ( 1 ) は水平方向境界点として抽出された結果である。

## 【 0 2 7 2 】

図 5 0 ( c ) は垂直方向境界点抽出処理により、ハッチングパターンが付されている画素 ( 1 ) が垂直方向境界点として抽出された結果である。

## 【 0 2 7 3 】

図 5 0 ( d ) は斜め方向境界点抽出処理により、ハッチングパターンが付されている画素 ( 1 ) が斜め方向境界点として抽出された結果である。

## 【 0 2 7 4 】

図 5 0 ( e ) は水平方向境界点抽出、垂直方向境界点抽出、及び斜め方向境界点抽出処理の少なくともいずれか 1 つ以上により抽出された境界点が、ハッチングパターンが付されている画素 ( 1 ) で示されている。即ち、このハッチングパターンが付されている画素 ( 1 ) は、図 5 0 ( b ) と図 5 0 ( c ) と図 5 0 ( d ) のハッチングパターンが付されている画素 ( 1 ) を合わせたものとなっている。これらの境界点は、4 - 連結状態となって

10

20

30

40

50

いる。

【0275】

実施形態1における境界線画像の生成を明確にするため、さらに図51を用いて、水平方向境界点抽出処理と垂直方向境界点抽出処理と斜め方向境界点抽出処理の説明を加える。

【0276】

図51(a)は、下地領域中に四つの矩形領域からなる図の領域分割の結果である。図51(b)は、図51(a)の境界線周囲の画素をあらわすものである。

【0277】

図51(c)は、図51(b)からハッチングパターンが付加された画素を水平方向境界点として抽出した水平境界点抽出処理の結果である。

10

【0278】

図51(d)は図51(b)からハッチングパターンが付加された画素を垂直方向境界点として抽出した垂直方向境界点の結果である。

【0279】

図51(e)は図51(b)からハッチングパターンが付加された画素を斜め方向境界点として抽出した斜め方向境界点の結果である。

【0280】

図51(f)は以上の、水平方向、垂直方向、斜め方向の少なくともいずれか1つ以上の境界点抽出処理により抽出された全ての境界点である。これらの境界点は4-連結である。

20

【0281】

交点抽出処理

ステップS30の交点抽出処理について、図52を用いて説明する。

【0282】

図52は本発明の実施形態1のステップS30の交点抽出処理の詳細を示すフローチャートである。

【0283】

ステップS301で、ステップS22、ステップS24及びステップS26それぞれで生成された水平方向境界点テーブル401、垂直方向境界点テーブル402及び斜め方向境界点テーブル403を参照する。そして、その3つのテーブル401~403上で2つ以上のテーブルに存在する境界点の内、テーブル401~403それぞれでの隣接する領域番号とから3つ以上の領域番号を持つ境界点を探索して、それを交点とする。

30

【0284】

尚、3つのテーブル上で2つ以上のテーブルに存在する境界点とは、水平境界点かつ垂直方向境界点、水平方向境界点かつ斜め方向境界点、垂直方向境界点かつ斜め方向境界点、あるいは水平方向境界点かつ垂直方向境界点かつ斜め方向境界点のいずれかである。

【0285】

図5では、図3の領域分割の結果から抽出された境界点から構成された図4の水平方向境界点テーブル401、垂直方向境界点テーブル402及び斜め方向境界点テーブル403とから交点を抽出する例が示されている。

40

【0286】

境界点Cの水平方向の隣接領域番号は2、3で、垂直方向の隣接領域番号は1、2で、斜め方向の隣接領域番号は2、3であるので、C点は2つ以上の境界点テーブルに存在することが分かる。また、C点は3つ以上の隣接領域番号を持っているので、C点を交点として抽出する。

【0287】

同様に、E、F、Gも交点である。そのほか、A、B、D、H、I等は水平方向の隣接領域番号と垂直方向の隣接領域番号と斜め方向の隣接領域番号が同じなので、2つの領域の間にあることが分かり、交点ではないと判断する。

50

## 【 0 2 8 8 】

図 5 3 ( a ) では、図 5 1 ( a ) の領域分割結果からの交点抽出例が示されている。図 5 3 ( a ) は、各方向 ( 水平、垂直及び斜め ) の境界点抽出処理それぞれにより抽出された境界点に対する、各方向の境界点テーブル作成処理により生成された各境界点テーブル 4 0 1 ~ 4 0 3 である。そして、水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2 及び斜め方向境界点テーブル 4 0 3 の 3 つの境界点テーブル上で、少なくとも 2 つ以上の境界点テーブルにある境界点毎に線種別にマークしたものである。

## 【 0 2 8 9 】

図 5 3 ( b ) は図 5 3 ( a ) でマークされた境界点から、さらに 3 つ以上の互いに相異なる隣接領域番号を持つ境界点を交点として、これらを交点テーブル 6 0 1 としてまとめて記録した結果である。図 5 3 ( c ) は交点となった点を、矩形のグレーパターンで示したものである。

10

## 【 0 2 9 0 】

そして、ステップ S 3 0 2 で、すべての領域番号の其々が、ステップ S 3 0 1 で抽出された各交点の内のいずれかの交点の隣接領域番号に含まれているかどうかを判定する。全ての領域番号が含まれている場合 ( ステップ S 3 0 2 で Y E S )、全部の領域に交点が存在していると判定し、この交点抽出処理を終了する。

## 【 0 2 9 1 】

一方、いずれの交点の隣接領域番号にも含まれない領域番号がある場合 ( ステップ S 3 0 2 で N O )、この領域には隣接領域が複数存在することではなく、ステップ S 3 0 1 での処理では交点が見つからなかったと判定する。

20

## 【 0 2 9 2 】

図 3 の例では、ステップ S 3 0 1 で抽出された交点 C、E、F、G の隣接領域番号に 1 ~ 4 の領域番号が含まれているので、領域 1 ~ 4 の境界点には交点が存在する。しかし、領域番号 5 を隣接領域番号に含む交点には存在しないので、領域 5 は、他の領域との境界点に交点がない、独立した領域であると判断する。

## 【 0 2 9 3 】

ステップ S 3 0 3 で、ステップ S 3 0 2 で判定された交点のない領域に対し、ステップ S 2 2、ステップ S 2 4 及びステップ S 2 6 で生成された水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2 及び斜め方向境界点テーブル 4 0 3 を再び参照する。そして、交点のない領域番号を隣接領域番号に持つ境界点を任意に 2 点取り、この交点のなかった領域の交点にする。

30

## 【 0 2 9 4 】

例えば、図 4 の水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2 及び斜め方向境界点テーブル 4 0 3 とから隣接領域番号 5 を含み、かつ、2 つの隣接領域番号を持つ境界点 H と I を取り、領域 5 の交点にする。これらは、H と I の位置に限らず、例えば、図 4 の水平方向境界点テーブル 4 0 1、垂直方向境界点テーブル 4 0 2 及び斜め方向境界点テーブル 4 0 3 を各テーブルの先頭からサーチする。そして、最初に見つかる条件を満たす点 ( 即ち、隣接領域番号 5 を含む境界点 ) と、同条件を満たすその他の点の中で、最初に見つかった点から最も離れた点等としても良い。

40

## 【 0 2 9 5 】

以上説明したように、実施形態 1 によれば、画像を領域毎に分割してベクトル化する際には、画素間の特徴によって得られる乖離度に基づいて、画像を分割する。次に、これによって得られる領域に対して、隣接する領域間の境界に関する領域境界情報として、境界の交点を端点とした場合のその端点間を表現するベクトル ( 端点間ベクトル ) の向きを示す輪郭点情報を生成する。

## 【 0 2 9 6 】

そして、この輪郭点情報に基づいて、分割された領域の境界を構成する端点間ベクトルを同定する。より具体的には、この輪郭点情報によって、各端点間ベクトルの位置関係を把握した上で、分割された領域の境界を構成する端点間ベクトルを同定する。これにより

50

、領域を表現するベクトルデータを正確かつ適切に生成することが可能となる。つまり、従来技術のように、各領域間に隙間や重なりが生じるようなベクトルデータが生成されることを防止することができる。

【0297】

<実施形態2>

実施形態2では、図54に示すように、全ての境界線画像を全て含む細線画像を生成した後に、この画像から、ベクトル抽出時に交点を改めて検出しつつ、一括してベクトル化する構成をとる。この場合の処理を図55を用いて説明する。

【0298】

図55は本発明の実施形態2の画像処理装置の動作を示すフローチャートである。

10

【0299】

同図において、実施形態1の図10のフローチャートと異なる部分は、次のようになっている。つまり、図10のステップS70の端点間ベクトル生成処理が、ステップS50～ステップS80の処理ループ内にあるのに対し、図55では、ステップS50～ステップS80の処理ループ外のステップS90の端点間ベクトル生成処理として存在する点である。即ち、図54で示すように、全ての境界線画像を全て含む細線画像を生成した後に、この画像から、ベクトル抽出時に交点を改めて検出しつつ、一括してベクトル化する構成をとっている。

【0300】

ステップS90の端点間ベクトル生成処理（実施形態1の図10におけるステップS70の端点間ベクトル生成処理と同様な処理）時においても、実施形態1で説明したように、交点部を端点として抽出する。そのため、このように構成しても、実施形態1と同様に、図45や図46に画像として再描画した形で模式的に示したような、端点間毎平滑化済（非周回）ベクトルの集まりとして生成される。

20

【0301】

即ち、ステップS30で抽出された交点や、ステップS50で生成された交点と交点の間を繋ぐ境界線画像における各交点は、ステップS90の端点間ベクトル生成の処理において、端点間毎平滑化済（非周回）ベクトルの交点として同定される。そして、その交点と交点の間を繋ぐ端点間ベクトル群として一括して生成される。以降のステップS100の処理は、実施形態1の場合と全く同様である。

30

【0302】

以上説明したように、実施形態2によれば、実施形態1で説明した効果に加えて、境界線画像を全て含む細線画像に対して、それを構成する端点間ベクトル群を生成し、それに基づいて、領域別ベクトルを生成することができる。これにより、より効率的に領域別ベクトルを生成することができる。

【0303】

<実施形態3>

実施形態1及び2においては、ともに、入力画像の全ての分割領域をベクトル化することを前提に説明してきたが、本発明はこれに限定されるものではない。即ち、画像処理装置の通信I/F4等を経由して、あるいはポインティングデバイス（不図示）のユーザーインタフェース（以下、UIと記す）を用いて、ベクトル化対象とする分割領域を限定して、その領域のみをベクトル化するように構成してももちろん良い。

40

【0304】

この場合、図47のフローチャートにおけるステップS1010で、図2の領域テーブル201に記載されている注目領域番号を順番に自動入力するのではなく、通信I/F4やUI等を介して、ユーザの指示に基づき、ベクトル化対象とする分割領域を指定する。そして、この指定された分割領域に限定してベクトル化（領域別ベクトルを生成）するように構成してももちろん良い。例えば、図2～3のように5つの分割領域がある場合、ユーザが領域2、3、5を指定すると、この指定された領域2、3、5それぞれの領域別ベクトルが生成されることになる。

50

## 【0305】

この場合の処理を図56を用いて説明する。

## 【0306】

図56は本発明の実施形態3の画像処理装置の動作を示すフローチャートである。

## 【0307】

図56のフローチャートに示すように、ステップS40で、上述のベクトル化対象とする分割領域を限定するためのベクトル化対象領域情報（注目領域番号）を獲得する。そして、このステップS40以外は、実施形態2の図55のフローチャートと同様である。但し、ステップS50～ステップS100までの処理は、ステップS40で獲得したベクトル化対象領域情報が示す領域のみを、処理対象として一連の動作をする点が、実施形態3

10

## 【0308】

以上説明したように、実施形態3によれば、実施形態1で説明した効果に加えて、画像中の注目領域を処理対象として指定することが可能になるので、その注目領域に対してのみ、その領域の領域ベクトルを生成することができる。これにより、用途や目的に応じて、より効率的に必要なとする領域の領域ベクトルを生成することができる。

## 【0309】

以上、実施形態例を詳述したが、本発明は、例えば、システム、装置、方法、プログラムもしくは記憶媒体等としての実施態様をとることが可能である。具体的には、複数の機器から構成されるシステムに適用しても良いし、また、1つの機器からなる装置に適用し

20

## 【0310】

尚、本発明は、前述した実施形態の機能を実現するソフトウェアのプログラム（実施形態では図に示すフローチャートに対応したプログラム）を、システムあるいは装置に直接あるいは遠隔から供給する。そして、そのシステムあるいは装置のコンピュータが該供給されたプログラムコードを読み出して実行することによっても達成される場合を含む。

## 【0311】

従って、本発明の機能処理をコンピュータで実現するために、該コンピュータにインストールされるプログラムコード自体も本発明を実現するものである。つまり、本発明は、本発明の機能処理を実現するためのコンピュータプログラム自体も含まれる。

30

## 【0312】

その場合、プログラムの機能を有していれば、オブジェクトコード、インタプリタにより実行されるプログラム、OSに供給するスクリプトデータ等の形態であっても良い。

## 【0313】

プログラムを供給するための記録媒体としては、例えば、フロッピー（登録商標）ディスク、ハードディスク、光ディスクがある。また、更に、記録媒体としては、光磁気ディスク、MO、CD-ROM、CD-R、CD-RW、磁気テープ、不揮発性のメモリカード、ROM、DVD（DVD-ROM、DVD-R）などがある。

## 【0314】

その他、プログラムの供給方法としては、クライアントコンピュータのブラウザを用いてインターネットのホームページに接続する。そして、その接続先のホームページから本発明のコンピュータプログラムそのもの、もしくは圧縮され自動インストール機能を含むファイルをハードディスク等の記録媒体にダウンロードすることによっても供給できる。また、本発明のプログラムを構成するプログラムコードを複数のファイルに分割し、それぞれのファイルを異なるホームページからダウンロードすることによっても実現可能である。つまり、本発明の機能処理をコンピュータで実現するためのプログラムファイルを複数のユーザに対してダウンロードさせるWWWサーバも、本発明に含まれるものである。

40

## 【0315】

また、本発明のプログラムを暗号化してCD-ROM等の記憶媒体に格納してユーザに配布し、所定の条件をクリアしたユーザに対し、インターネットを介してホームページか

50

ら暗号化を解く鍵情報をダウンロードさせる。そして、その鍵情報を使用することにより暗号化されたプログラムを実行してコンピュータにインストールさせて実現することも可能である。

【0316】

また、コンピュータが、読み出したプログラムを実行することによって、前述した実施形態の機能が実現される。また、そのプログラムの指示に基づき、コンピュータ上で稼動しているOSなどが、実際の処理の一部または全部を行ない、その処理によっても前述した実施形態の機能が実現され得る。

【0317】

さらに、記録媒体から読み出されたプログラムが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれる。その後、そのプログラムの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行ない、その処理によっても前述した実施形態の機能が実現される。

【図面の簡単な説明】

【0318】

【図1】本発明の実施形態1の画像処理装置の機能構成を示すブロック図である。

【図2】本発明の実施形態1の領域番号と色情報（属性情報）の対応を示す領域テーブルの一例を示す図である。

【図3】本発明の実施形態1の境界点抽出処理を説明するための図である。

【図4】本発明の実施形態1の境界点テーブルの一例を示す図である。

【図5】本発明の実施形態1の境界線の交点の同定を説明するための図である。

【図6】本発明の実施形態1の交点テーブルの一例を示す図である。

【図7】本発明の実施形態1の線画像のベクトル処理の一連の処理経過の例を示す図である。

【図8】本発明の実施形態1の端点間ベクトル生成部の詳細機能構成を示すブロック図である。

【図9】本発明の実施形態1の画像処理装置のハードウェア構成を示す図である。

【図10】本発明の実施形態1の画像処理装置の動作を示すフローチャートである。

【図11】本発明の実施形態1の注目画素とその近傍画素を示す図である。

【図12】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図13】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図14】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図15】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図16】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図17】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図18】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図19】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図20】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図21】本発明の実施形態1の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

10

20

30

40

50

【図 2 2】本発明の実施形態 1 の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図 2 3】本発明の実施形態 1 の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図 2 4】本発明の実施形態 1 の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図 2 5】本発明の実施形態 1 の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図 2 6】本発明の実施形態 1 の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

10

【図 2 7】本発明の実施形態 1 の端点間毎輪郭（周回）ベクトル抽出における注目画素と近傍画素の状態に応じた輪郭点と輪郭点情報の抽出パターンを示す図である。

【図 2 8】本発明の実施形態 1 の輪郭点情報の一覧表を示す図である。

【図 2 9】本発明の実施形態 1 の端点部より検出される輪郭点例とそれらに付与される輪郭点情報を示す図である。

【図 3 0】本発明の実施形態 1 の端点部ではない部分より抽出される輪郭点例とそれらに付与される輪郭点情報を示す図である。

【図 3 1 A】本発明の実施形態 1 の端点間輪郭（周回）ベクトル抽出処理全体を示すフローチャートである。

【図 3 1 B】本発明の実施形態 1 のベクトル列抽出処理の具体例を説明するための図である。

20

【図 3 1 C】本発明の実施形態 1 のベクトル列抽出処理の具体例を説明するための図である。

【図 3 2】本発明の実施形態 1 の端点間每周回ベクトル列と輪郭点情報のデータ形式の例を示す図である。

【図 3 3】本発明の実施形態 1 の図 3 1 A のステップ S 1 のベクトル列抽出処理の詳細を示すフローチャートである。

【図 3 4】本発明の実施形態 1 のステップ S 1 2 0 の処理の詳細を示すフローチャートである。

【図 3 5】本発明の実施形態 1 の線素、端点間每周回ベクトル列と輪郭点情報、端点間毎線芯（周回）ベクトルを例示する図である。

30

【図 3 6】本発明の実施形態 1 の端点間毎線芯（周回）ベクトル生成部の動作を示すフローチャートである。

【図 3 7】本発明の実施形態 1 の始端点の直前と終端点の直後に挿入する補助輪郭点を説明するための図である。

【図 3 8】本発明の実施形態 1 の端点間毎補助ベクトル入り（周回）ベクトル生成部の動作を示すフローチャートである。

【図 3 9】本発明の実施形態 1 の端点間毎補助ベクトル入り（周回）ベクトル、及び、端点間毎平滑化済ベクトルのデータ形式の例を示す図である。

【図 4 0】本発明の実施形態 1 のステップ S 4 1 0 0 の処理の詳細を示すフローチャートである。

40

【図 4 1】本発明の実施形態 1 の終端点～始端点間へ挿入する補助ベクトルを説明するための図である。

【図 4 2】本発明の実施形態 1 の終端点～始端点間へ挿入する補助ベクトルを説明するための図である。

【図 4 3】本発明の実施形態 1 の交点と交点の間を繋ぐ境界線の例を示す図である。

【図 4 4】本発明の実施形態 1 の境界線画像の生成過程の一例を示す図である。

【図 4 5】本発明の実施形態 1 の図 4 4 の境界線の平滑化済端点間ベクトルの再描画例を示す図である。

【図 4 6】本発明の実施形態 1 の図 4 5 の再描画例を対応する交点を明示した形式で示す

50

図である。

【図 4 7】本発明の実施形態 1 の図 1 0 のステップ S 1 0 0 の領域別ベクトル生成処理の詳細を示すフローチャートである。

【図 4 8】本発明の実施形態 1 の領域属性が付与された領域ベクトルから再生した画像例を示す図である。

【図 4 9】本発明の実施形態 1 のステップ S 1 2 の領域分割処理の詳細を示すフローチャートである。

【図 5 0】本発明の実施形態 1 の境界線の抽出例を示す図である。

【図 5 1】本発明の実施形態 1 の境界線画像の生成を説明するための図である。

【図 5 2】本発明の実施形態 1 のステップ S 3 0 の交点抽出処理の詳細を示すフローチャートである。 10

【図 5 3】本発明の実施形態 1 の領域分割結果からの交点抽出例を示す図である。

【図 5 4】本発明の実施形態 1 の全ての境界線画像を含む細線画像を生成して得られる画像例を示す図である。

【図 5 5】本発明の実施形態 2 の画像処理装置の動作を示すフローチャートである。

【図 5 6】本発明の実施形態 3 の画像処理装置の動作を示すフローチャートである。

【図 5 7】処理対象の画像の一例を示す図である。

【図 5 8】従来技術における不具合を説明するための図である。

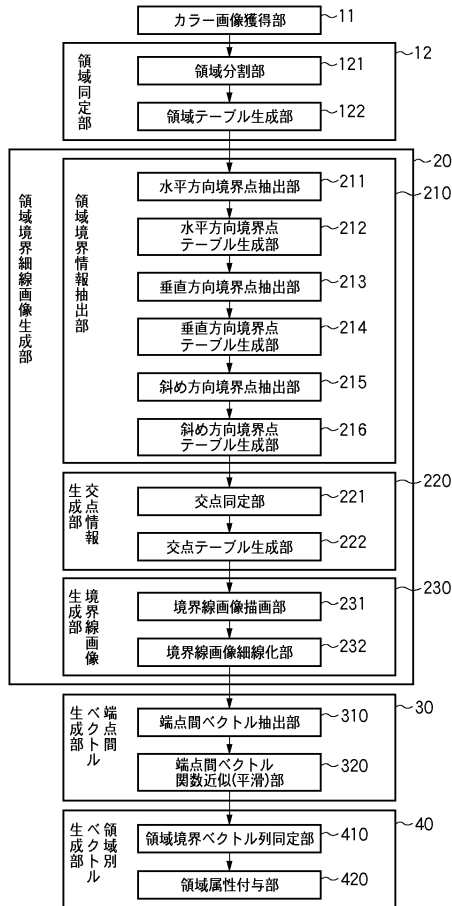
【符号の説明】

【 0 3 1 9 】 20

- 1        スキャナ
- 2        画像メモリ
- 3        画像入出力 I / O
- 4        通信 I / O
- 6        R O M
- 5 1 8    バス
- 5 1 9    C P U
- 5 2 0    R A M
- 5 2 1    I / O
- 5 2 2    H / D

30

【 図 1 】



【 図 4 】

水平方向境界点テーブル				401
水平境界点座標	隣接する領域番号			
⋮	⋮	⋮	⋮	
B : (10, 20)	1	2		
⋮	⋮	⋮	⋮	
A : (14, 15)	1	2		
⋮	⋮	⋮	⋮	
C : (22, 40)	2	3		
⋮	⋮	⋮	⋮	
E : (38, 40)	2	3		
⋮	⋮	⋮	⋮	
D : (54, 10)	1	4		
⋮	⋮	⋮	⋮	
H : (64, 58)	1	5		
⋮	⋮	⋮	⋮	
I : (64, 66)	1	5		
⋮	⋮	⋮	⋮	

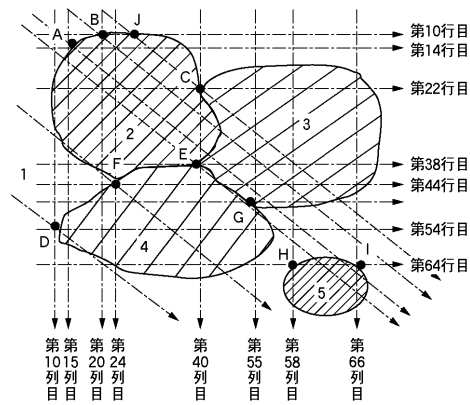
垂直方向境界点テーブル				402
垂直境界点座標	隣接する領域番号			
⋮	⋮	⋮	⋮	
D : (54, 10)	1	4		
⋮	⋮	⋮	⋮	
A : (14, 15)	1	2		
⋮	⋮	⋮	⋮	
B : (10, 20)	1	2		
⋮	⋮	⋮	⋮	
C : (22, 40)	1	2		
⋮	⋮	⋮	⋮	
E : (38, 40)	2	4		
⋮	⋮	⋮	⋮	
H : (64, 58)	1	5		
⋮	⋮	⋮	⋮	
I : (64, 66)	1	5		
⋮	⋮	⋮	⋮	

斜め方向境界点テーブル				403
斜め境界点座標	隣接する領域番号			
⋮	⋮	⋮	⋮	
A : (14, 15)	1	2		
⋮	⋮	⋮	⋮	
B : (10, 20)	1	2		
⋮	⋮	⋮	⋮	
J : (10, 28)	1	2		
⋮	⋮	⋮	⋮	
C : (22, 40)	1	2		
⋮	⋮	⋮	⋮	
K : (18, 15)	1	2		
⋮	⋮	⋮	⋮	
H : (64, 58)	1	5		
⋮	⋮	⋮	⋮	
I : (64, 66)	1	5		
⋮	⋮	⋮	⋮	

【圖 2】

領域番号	領域値
1	(R <sub>1</sub> , G <sub>1</sub> , B <sub>1</sub> )
2	(R <sub>2</sub> , G <sub>2</sub> , B <sub>2</sub> )
3	(R <sub>3</sub> , G <sub>3</sub> , B <sub>3</sub> )
4	(R <sub>4</sub> , G <sub>4</sub> , B <sub>4</sub> )
5	(R <sub>5</sub> , G <sub>5</sub> , B <sub>5</sub> )

【 図 3 】

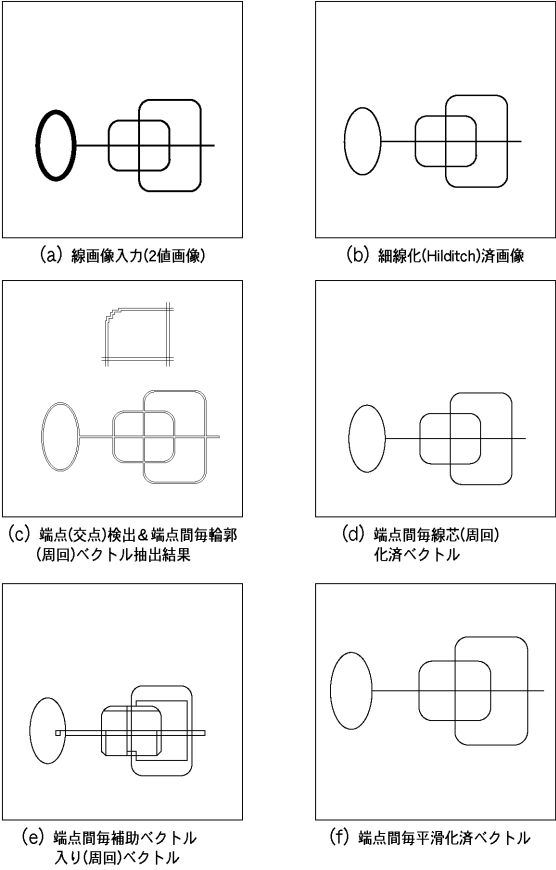


【 図 5 】

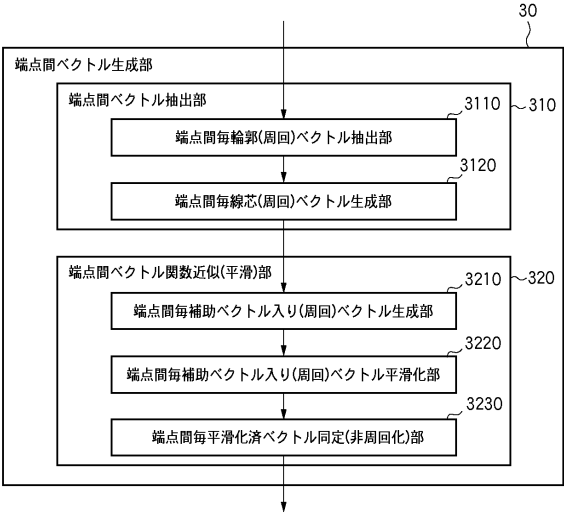
【図 6】

交点	隣接領域番号	601
C : (22, 40)	1, 2, 3	
E : (38, 40)	2, 3, 4	
F : (44, 24)	1, 2, 4	
G : (50, 56)	1, 3, 4	
H : (64, 58)	1, 5	
I : (64, 66)	1, 5	

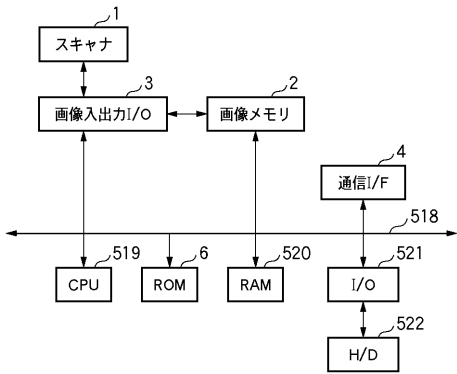
【図 7】



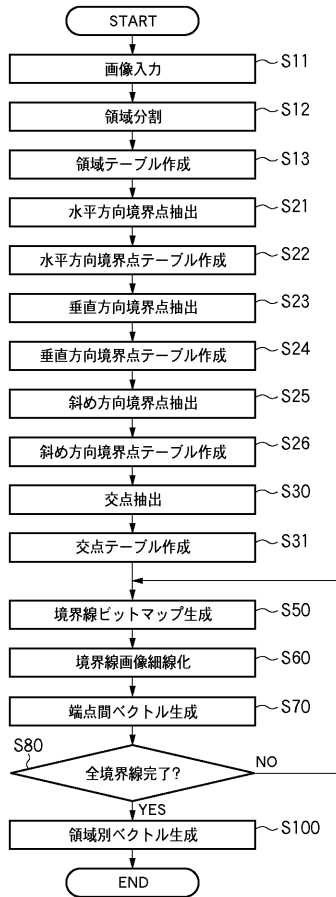
【図 8】



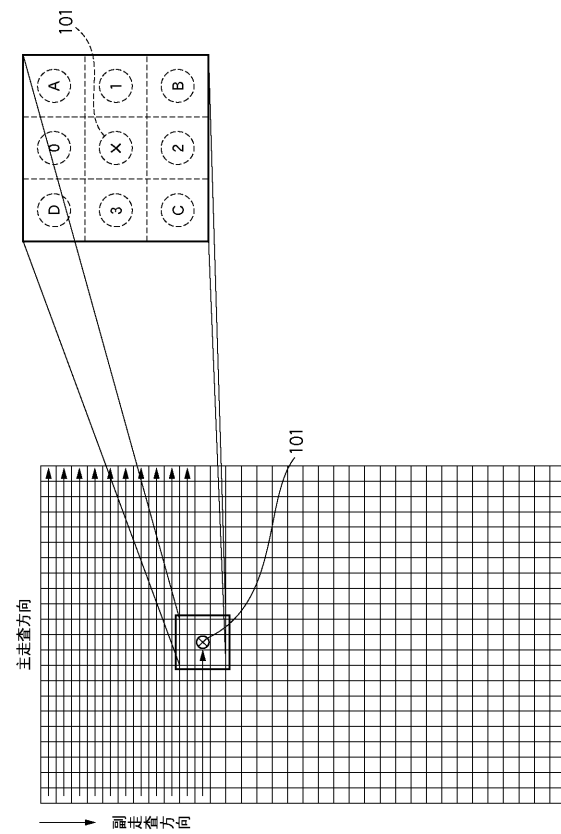
【図 9】



【図 10】

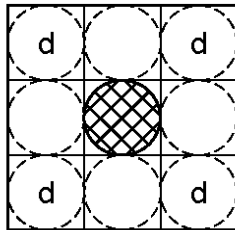


【図 11】



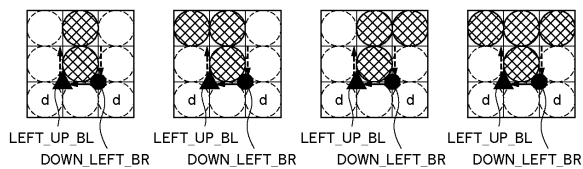
【図 12】

ケース0



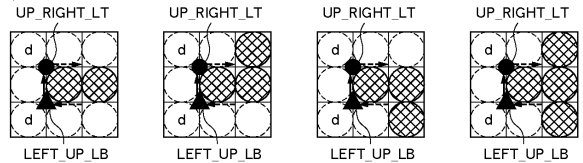
【図 13】

ケース1



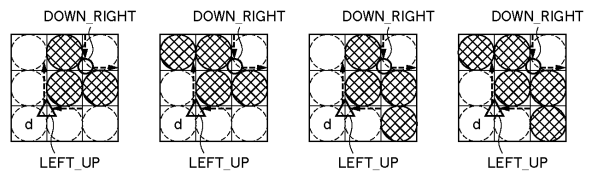
【図 14】

ケース2



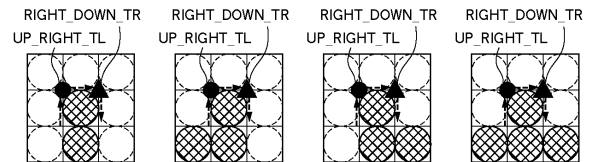
【図 15】

ケース3



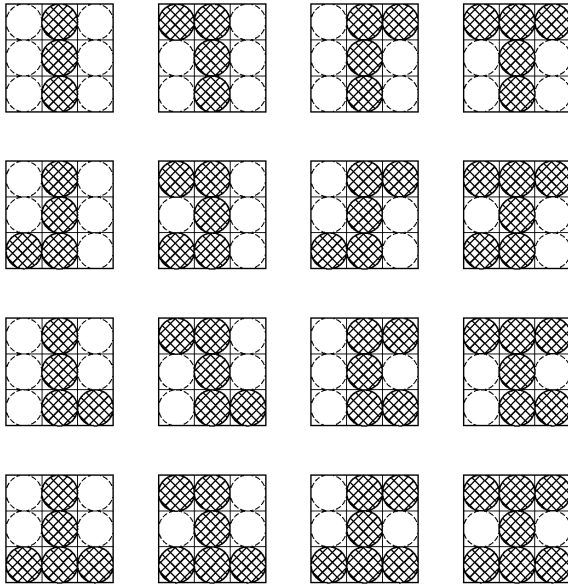
【図 16】

ケース4



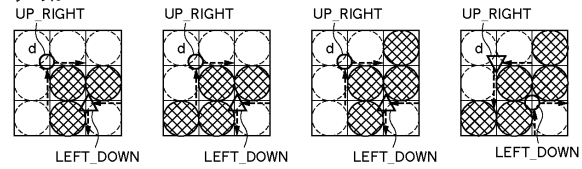
## 【図 17】

ケース5



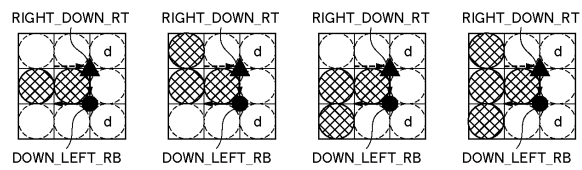
## 【図 18】

ケース6



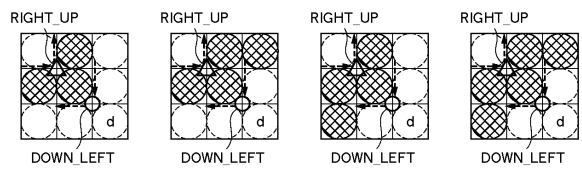
## 【図 20】

ケース8

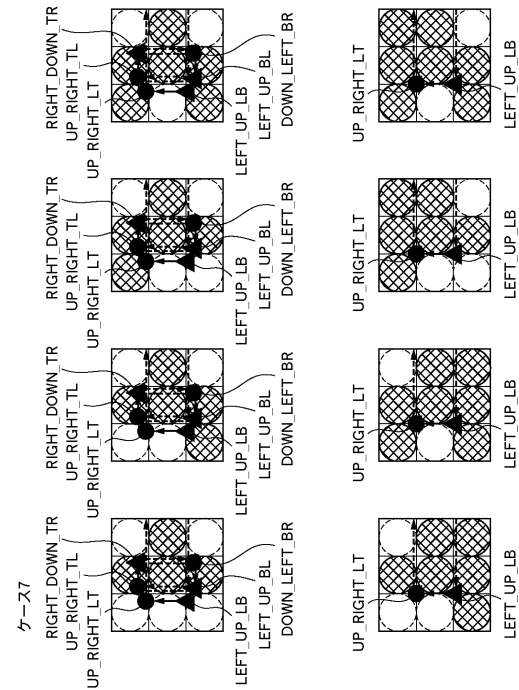


## 【図 21】

ケース9

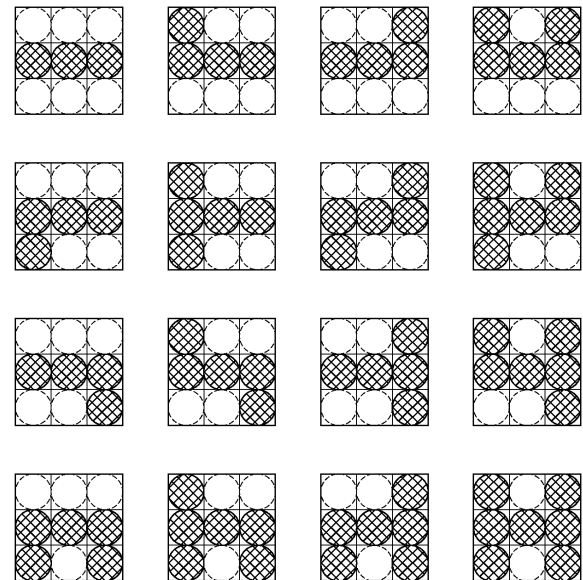


## 【図 19】



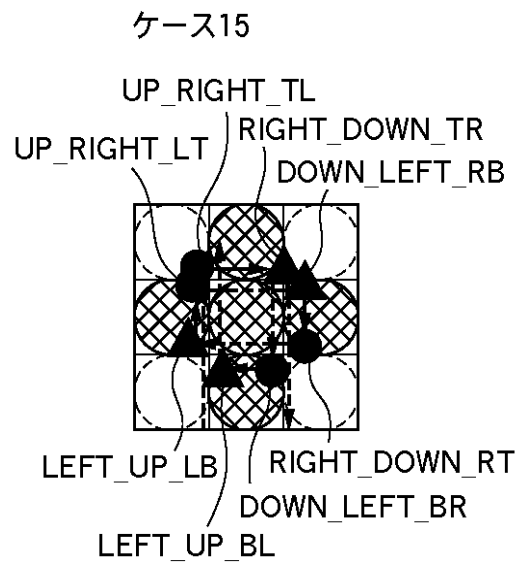
## 【図 22】

ケース10





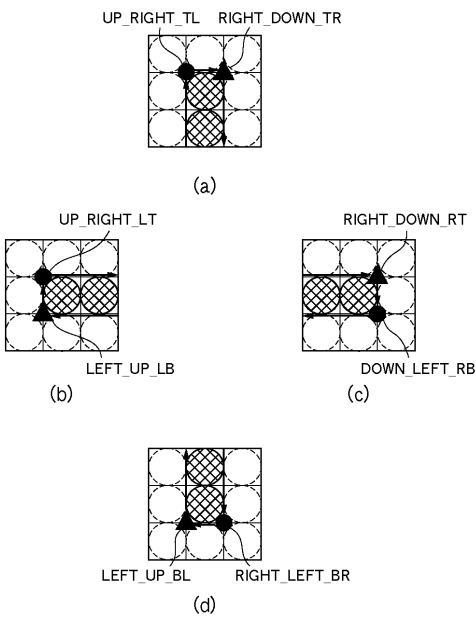
【図 27】



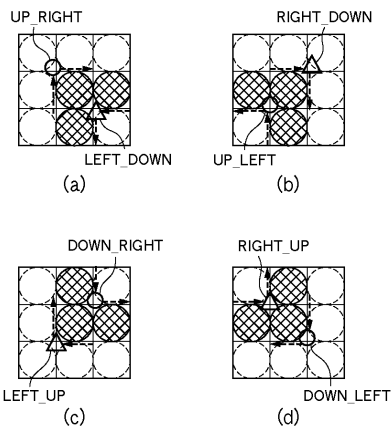
【図 28】

No.	輪郭点情報	端点情報	値	水平(0) 垂直(1)	上向(0) 下向(1)	右向(0) 左向(1)	非選択(0) 選択(1)	端点(1) 非端点(0)
01.	UP_RIGHT_TL	(Top Left)	01H	0	0	0	0	1
02.	UP_RIGHT		02H	0	0	0	1	0
03.	UP_RIGHT_LT	(Left Top)	03H	0	0	0	1	1
04.	UP_LEFT		06H	0	0	1	1	0
05.	DOWN_RIGHT		0aH	0	1	0	1	0
06.	DOWN_LEFT_BR	(Bottom Right)	0dH	0	1	1	0	1
07.	DOWN_LEFT		0eH	0	1	1	1	0
08.	DOWN_LEFT_RB	(Right Bottom)	0fH	0	1	1	1	1
09.	RIGHT_UP		12H	1	0	0	1	0
10.	LEFT_UP_LB	(Left Bottom)	15H	1	0	1	0	1
11.	LEFT_UP		16H	1	0	1	1	0
12.	LEFT_UP_BL	(Bottom Left)	17H	1	0	1	1	1
13.	RIGHT_DOWN_RT	(Right Top)	19H	1	1	0	0	1
14.	RIGHT_DOWN		1aH	1	1	0	1	0
15.	RIGHT_DOWN_TR	(Top Right)	1bH	1	1	0	1	1
16.	RIGHT_DOWN		1eH	1	1	1	1	0

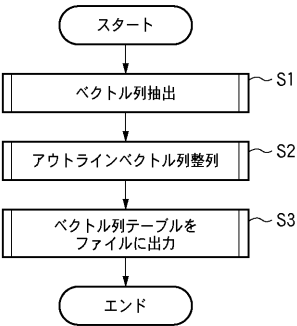
【図 29】



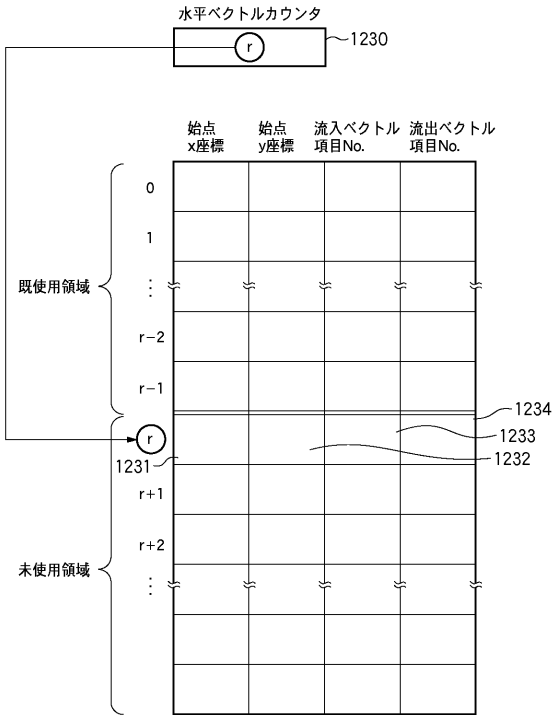
【図 30】



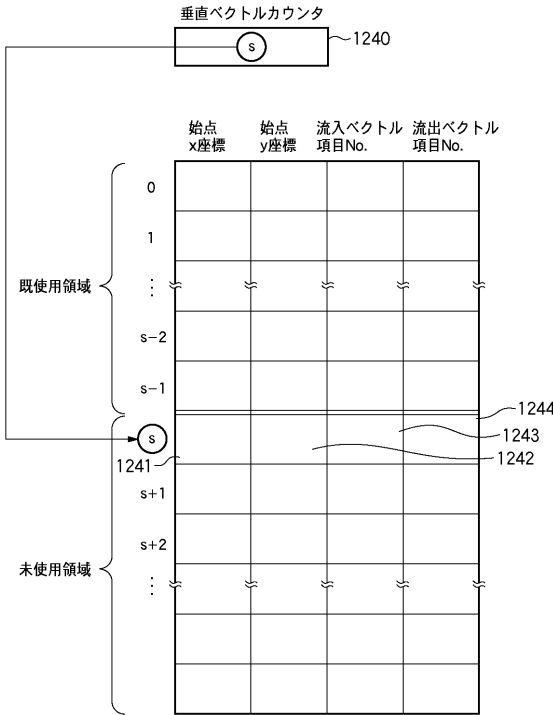
【図 31 A】



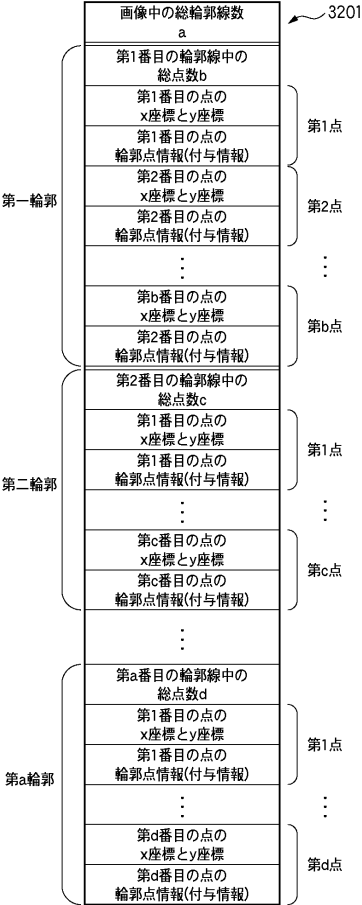
【図 3 1 B】



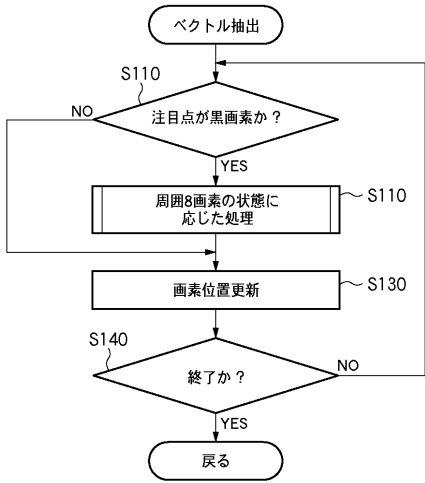
【図 3 1 C】



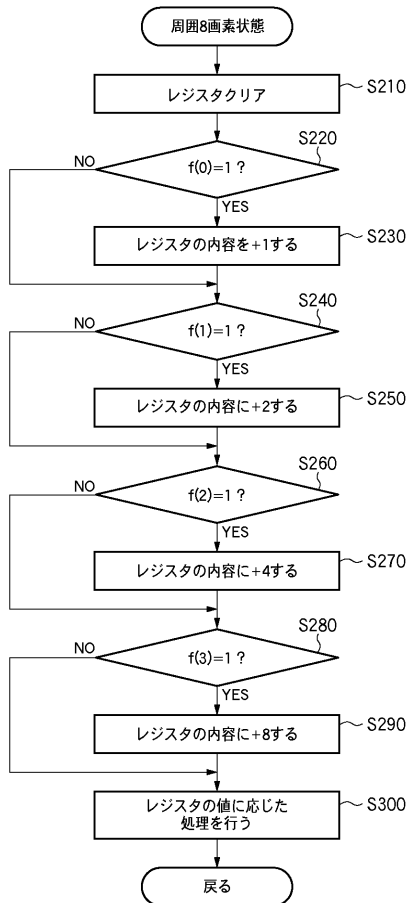
【図 3 2】



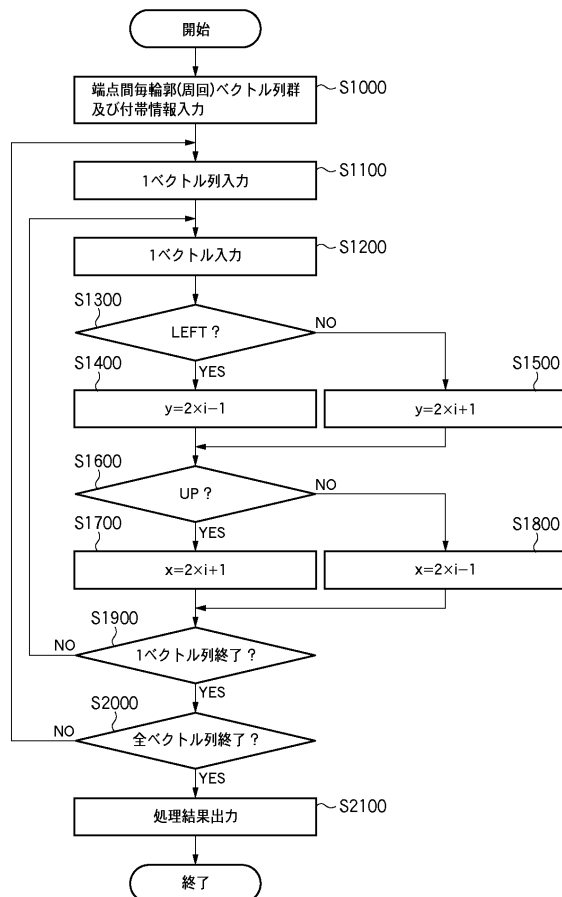
【図 3 3】



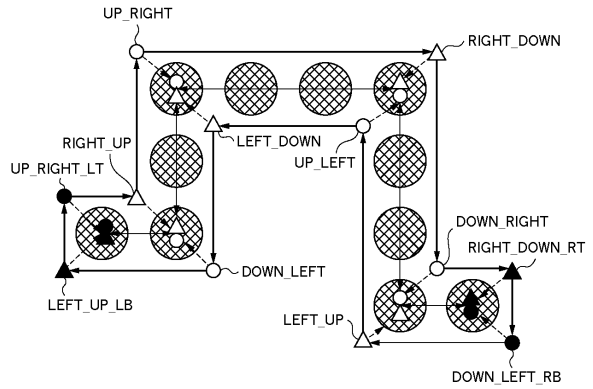
【図 3 4】



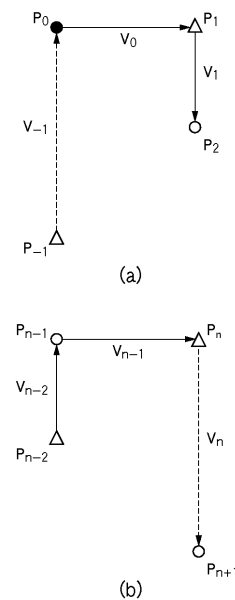
【図 3 6】



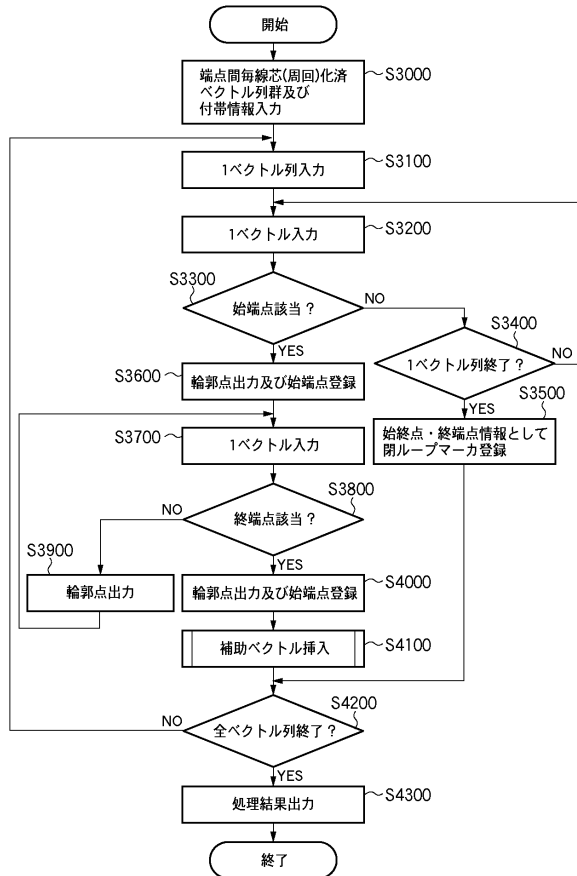
【図 3 5】



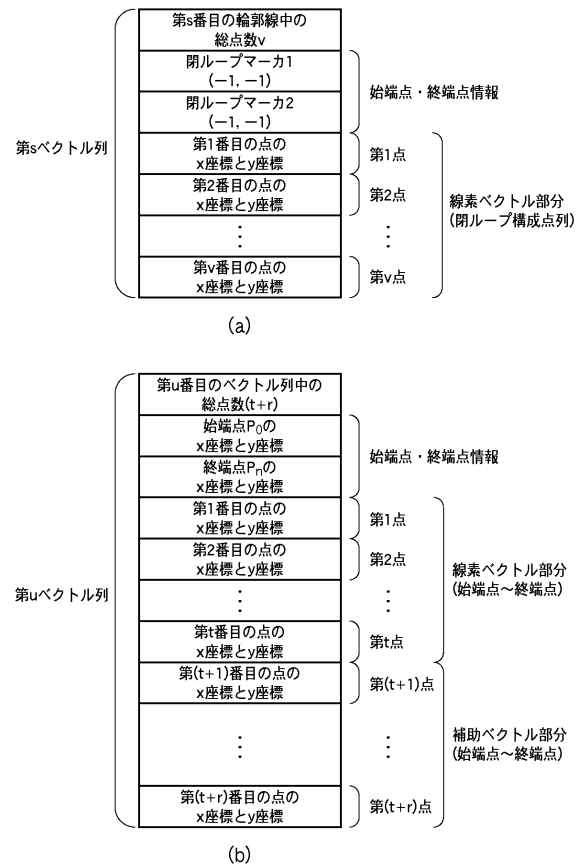
【図 3 7】



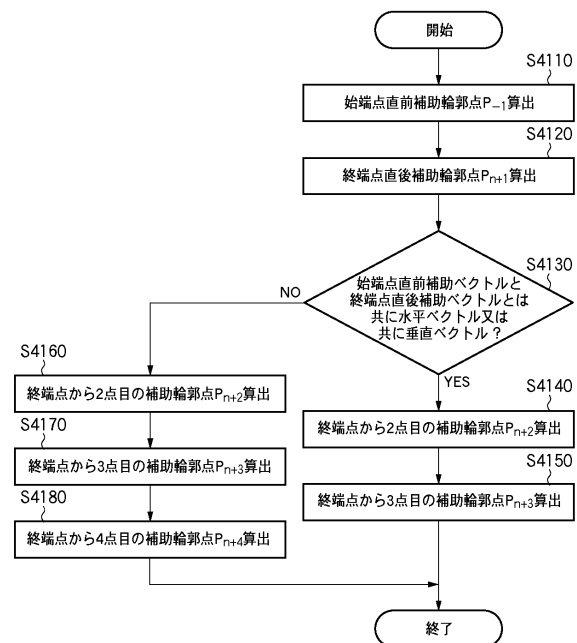
【図 38】



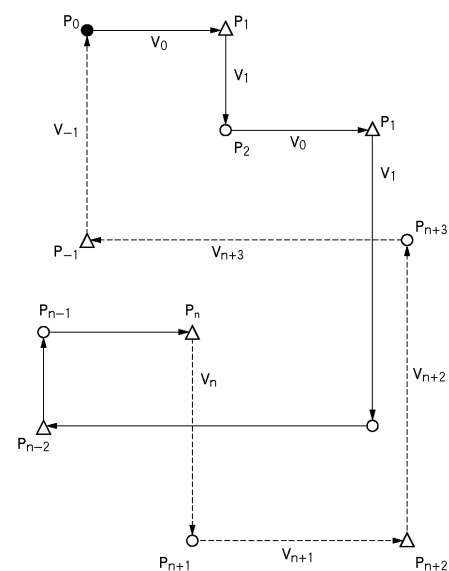
【図 39】



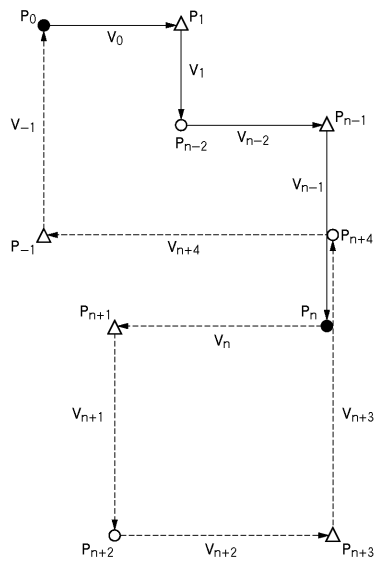
【図 40】



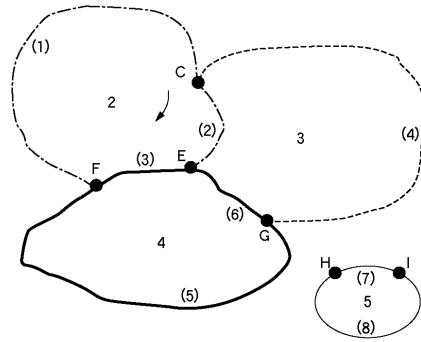
【図 41】



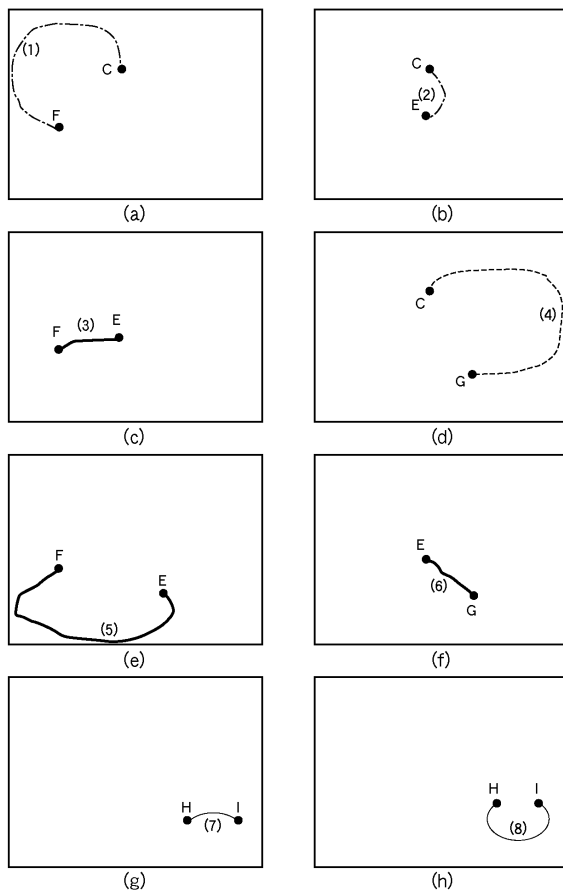
【図 4 2】



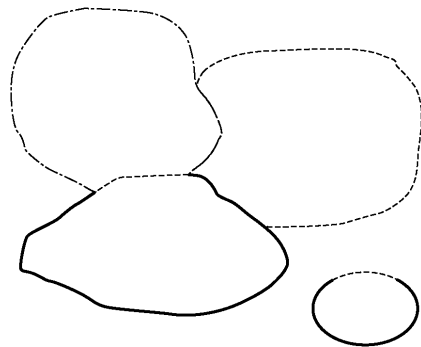
【図 4 3】



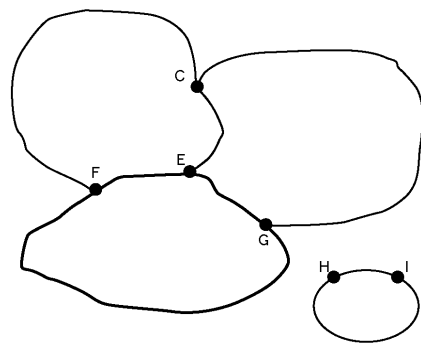
【図 4 4】



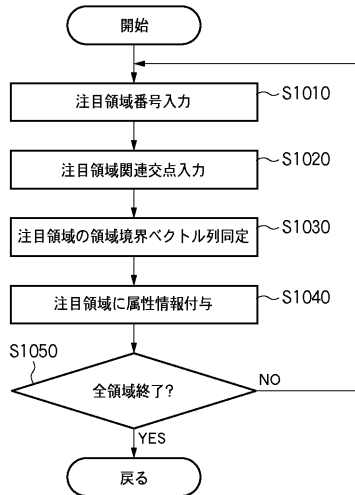
【図 4 5】



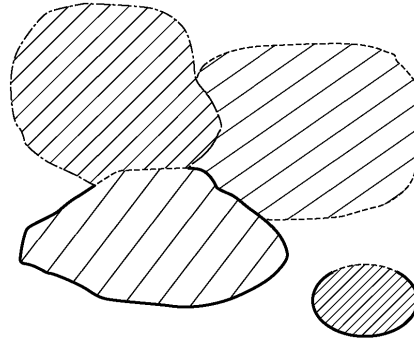
【図 4 6】



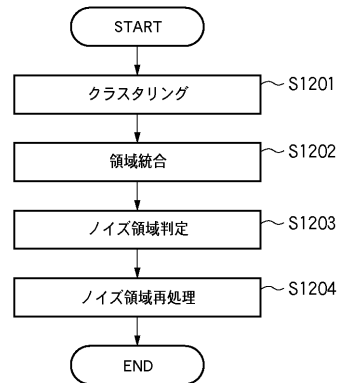
【図 47】



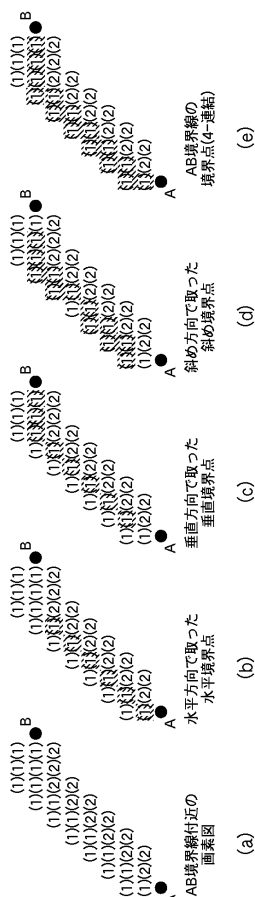
【図 48】



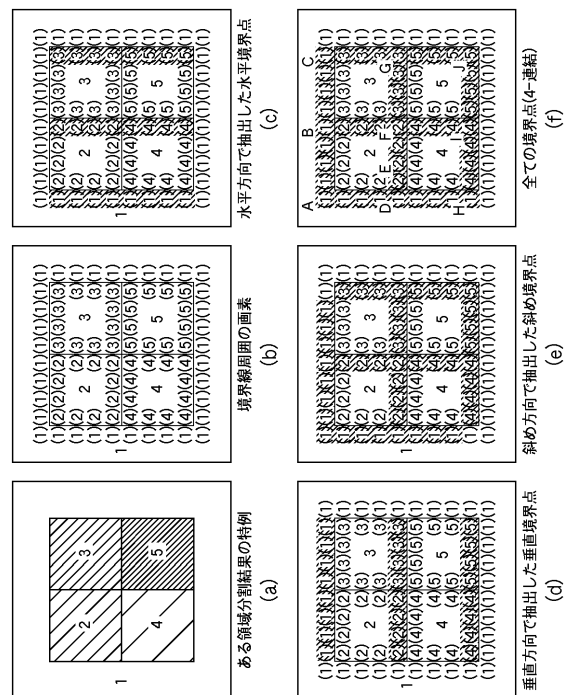
【図 49】



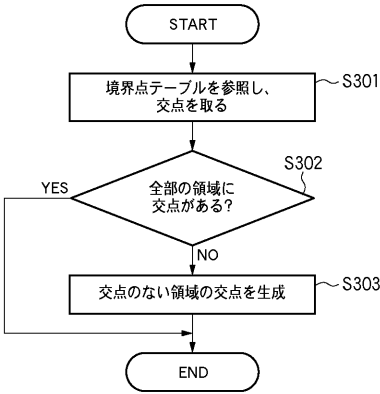
【図 50】



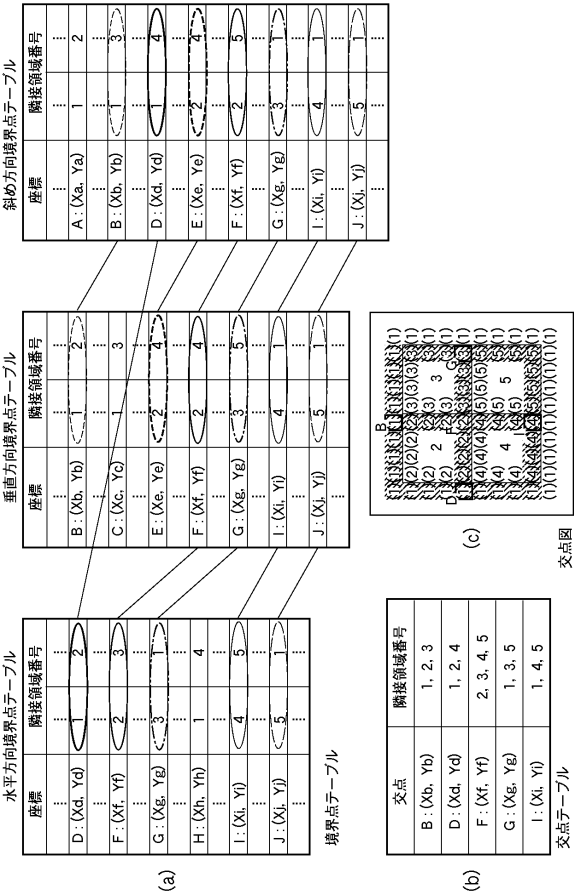
【図 51】



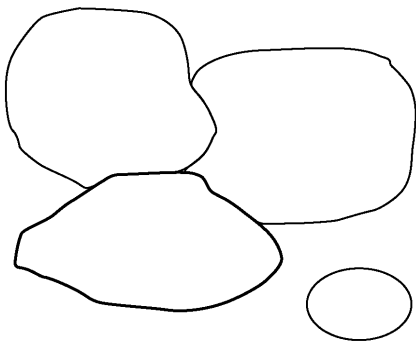
【図 5 2】



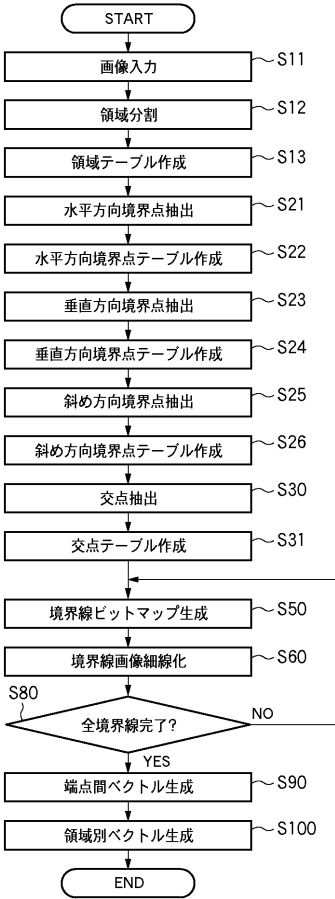
【図 5 3】



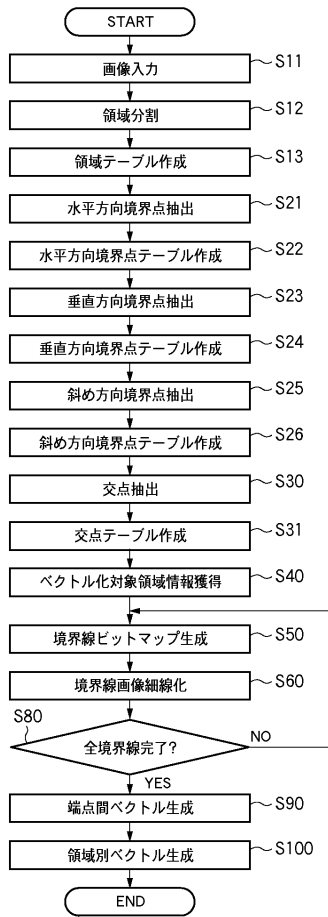
【図 5 4】



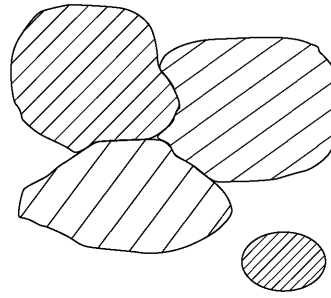
【図 5 5】



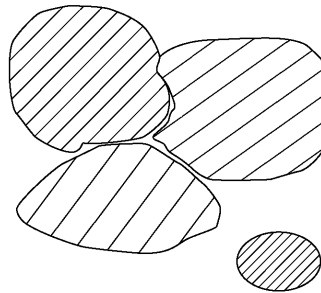
【図 56】



【図 57】



【図 58】



---

フロントページの続き

- (72)発明者 戴 曉艶  
東京都大田区下丸子3丁目30番2号 キヤノン株式会社内
- (72)発明者 大澤 秀史  
東京都大田区下丸子3丁目30番2号 キヤノン株式会社内

審査官 松永 稔

- (56)参考文献 特開平08-214306(JP,A)  
特開2006-107290(JP,A)  
特開2001-351110(JP,A)

- (58)調査した分野(Int.Cl., DB名)  
G06T 9/20