(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2005-165674
(P2005-165674A)
(43)公開日　平成17年6月23日(2005.6.23)

(51)Int.Cl.[7]	F I			テーマコード（参考）
G06T 11/60	G06T	11/60	100A	5B021
G06F 3/12	G06F	3/12	L	5B050
G06T 3/00	G06T	3/00	400Z	5B057
G06T 7/00	G06T	7/00	250	5C076
G06T 7/60	G06T	7/60	150H	5L096

審査請求 未請求　請求項の数 18　ＯＬ　（全 32 頁）　最終頁に続く

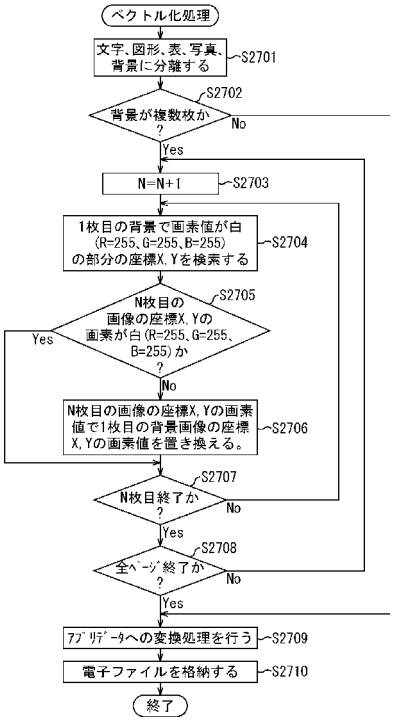
(21)出願番号　特願2003-403658 (P2003-403658)
(22)出願日　平成15年12月2日 (2003.12.2)

(71)出願人　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100090273
弁理士　國分　孝悦
(72)発明者　矢口　博之
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　松久保　勇志
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　西川　英一
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、画像処理方法、及びコンピュータプログラム

(57)【要約】
【課題】　イメージ画像を複数のオブジェクトに分離した際に、背景オブジェクトが白く抜けた状態になることを防止する。
【解決手段】　１枚目の原稿画像の背景オブジェクトの文字が抜けたところ２１０１と、円グラフが抜けたところ２１０２とを、他の原稿画像の画素を用いて補間することにより、１枚目の原稿画像の背景オブジェクトで白く抜けてしまっているところを消すことができるようにして、背景オブジェクトを再利用することができるようにする。
【選択図】　図２７



【特許請求の範囲】
【請求項１】
複数ページのイメージ画像のそれぞれを、背景画像を含む複数の画像に分離する分離手段と、
前記分離手段により分離された、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間する補間手段とを有することを特徴とする画像処理装置。
【請求項２】
前記分離手段により分離された背景画像の非背景領域を検索する検索手段を有し、
前記補間手段は、前記検索手段により検索された、所定ページの非背景領域の画素を、他のページの背景画像の画素に置き換え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３】
前記補間手段により補間された背景画像を記憶する記憶手段と、
前記記憶手段により記憶された背景画像を用いて、複数ページのイメージ画像を形成する画像形成手段とを有す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】
前記分離手段により分離された、所定ページの背景画像の位置と、他のページの背景画像の位置との差分情報を求める差分演算手段を有し、
前記補間手段は、前記差分演算手段により求められた差分情報に基づいて、前記所定ページの背景画像と、前記他のページの背景画像との位置合わせを行い、位置合わせを行った、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間することを特徴とする請求項１～３の何れか１項に記載の画像処理装置。
【請求項５】
前記分離手段により分離された背景画像以外のオブジェクトの外周を求める外周演算手段と、
前記外周演算手段により求められた外周の最も大きい領域を求める領域演算手段と、
前記領域演算手段により求められた領域を、前記分離手段により分離された背景画像からマスクするマスク手段とを有し、
前記差分演算手段は、前記マスク手段により一部の領域がマスクされた、所定ページの背景画像と、他のページの背景画像とを用いて、前記差分情報を求める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】
前記分離手段により分離された背景画像以外のオブジェクトの外周を求める外周演算手段と、
前記外周演算手段により求められた外周を、所定の許容値だけ拡大する外周拡大手段と、
前記外周拡大手段により拡大された外周によって囲まれる領域を、前記分離手段により分離された背景画像からマスクするマスク手段とを有し、
前記差分演算手段は、前記マスク手段により一部の領域がマスクされた、所定ページの背景画像と、他のページの背景画像とを用いて、前記差分情報を求める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７】
前記補間手段により所定の画素値に補間されなかった領域を、その周囲の領域の画素を用いて補間する第２の補間手段を有することを特徴とする請求項１～６の何れか１項に記載の画像処理装置。
【請求項８】
イメージ画像を読み取る読み取り手段を有し、
前記分離手段は、前記読み取り手段により読み取られた複数ページのイメージ画像のそれぞれを、背景画像を含む複数の画像に分離することを特徴とする請求項１～７の何れか１項に記載の画像処理装置。

【請求項９】
前記補間手段により補間された背景画像が、複数ページに亘り共通であることを指示するためのユーザインタフェースを有することを特徴とする請求項１～８の何れか１項に記載の画像処理装置。
【請求項１０】
複数ページのイメージ画像のそれぞれを、背景画像を含む複数の画像に分離する分離ステップと、
前記分離ステップにより分離された、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間する補間ステップとを有することを特徴とする画像処理方法。
【請求項１１】
前記分離ステップにより分離された背景画像の非背景領域を検索する検索ステップを有し、
前記補間ステップは、前記検索ステップにより検索された、所定ページの非背景領域の画素を、他のページの背景画像の画素に置き換えることを特徴とする請求項１０に記載の画像処理方法。
【請求項１２】
前記補間ステップにより補間された背景画像を記憶する記憶ステップと、
前記記憶ステップにより記憶された背景画像を用いて、複数ページのイメージ画像を形成する画像形成ステップとを有することを特徴とする請求項１０又は１１に記載の画像処理方法。
【請求項１３】
前記分離ステップにより分離された、所定ページの背景画像の位置と、他のページの背景画像の位置との差分情報を求める差分演算ステップを有し、
前記補間ステップは、前記差分演算ステップにより求められた差分情報に基づいて、前記所定ページの背景画像と、前記他のページの背景画像との位置合わせを行い、位置合わせを行った、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間することを特徴とする請求項１０～１２の何れか１項に記載の画像処理方法。
【請求項１４】
前記分離ステップにより分離された背景画像以外のオブジェクトの外周を求める外周演算ステップと、
前記外周演算ステップにより求められた外周の最も大きい領域を求める領域演算ステップと、
前記領域演算ステップにより求められた領域を、前記分離ステップにより分離された背景画像からマスクするマスクステップとを有し、
前記差分演算ステップは、前記マスクステップにより一部の領域がマスクされた、所定ページの背景画像と、他のページの背景画像とを用いて、前記差分情報を求めることを特徴とする請求項１３に記載の画像処理方法。
【請求項１５】
前記分離ステップにより分離された背景画像以外のオブジェクトの外周を求める外周演算ステップと、
前記外周演算ステップにより求められた外周を、所定の許容値だけ拡大する外周拡大ステップと、
前記外周拡大ステップにより拡大された外周によって囲まれる領域を、前記分離ステップにより分離された背景画像からマスクするマスクステップとを有し、
前記差分演算ステップは、前記マスクステップにより一部の領域がマスクされた、所定ページの背景画像と、他のページの背景画像とを用いて、前記差分情報を求めることを特徴とする請求項１３に記載の画像処理方法。
【請求項１６】
前記補間ステップにより所定の画素値に補間されなかった領域を、その周囲の領域の画素を用いて補間する第２の補間ステップを有することを特徴とする請求項１０～１５の何

れか１項に記載の画像処理方法。
【請求項１７】
イメージ画像を読み取る読み取りステップを有し、
前記分離ステップは、前記読み取りステップにより読み取られた複数ページのイメージ画像のそれぞれを、背景画像を含む複数の画像に分離することを特徴とする請求項１０～１６の何れか１項に記載の画像処理方法。
【請求項１８】
前記請求項１０～１７の何れか１項に記載の画像処理方法におけるステップをコンピュータに実行させることを特徴とするコンピュータプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、画像処理装置、画像処理方法、及びコンピュータプログラムに関し、特に、イメージ画像を複数のオブジェクトに分離して再利用できるようにするために用いて好適なものである。
【背景技術】
【０００２】
近年、環境問題が叫ばれる中、オフィスでのペーパーレス化が急速に進んでいる。このような背景の下、機能が拡張されたデジタル複合機（以降、ＭＦＰ（Multi Function Printer）と記す）では、画像ファイルを記録する際に、該画像ファイルが存在する画像記憶装置内のポインター情報を、文書の表紙或いは記載情報中に付加情報として記録しておき、文書を複写等して再利用する際に、このポインター情報からオリジナルの画像ファイル（電子ファイル）の格納場所を検出し、検出した格納場所に格納されている画像ファイルの元情報を直接用いることができるようにして、紙文書全体の保存を削減することができるようにする技術があった。
また、ポインター情報から検索した画像ファイルが見当たらない場合、あるいはポインター情報が検出できない場合でも、イメージ画像を文字、写真、図形、表、及び背景画像のようなオブジェクトに分離して、イメージ画像をコンピュータ（ＰＣ）上で再利用できるような形式に変換するイメージ画像のベクトル化の技術が提案されている（例えば、特許文献１を参照。）。
【０００３】
【特許文献１】特開２００２－２４７９９号公報
【発明の開示】
【発明が解決しようとする課題】
【０００４】
しかしながら、前述した従来の技術で行われているイメージ画像のベクトル化では、文字、写真、図形、及び表を抽出した後に残る背景画像は、元のオブジェクトの形状が白く抜けた状態で保存されており、前記背景画像を再利用できないという問題がある。
また、もともとは共通の背景画像であるにも関わらず、ページ毎にオブジェクトの抜け方が異なるため、背景画像をベージ毎に別の画像として管理しなくてはならない。このため、ハードディスク（ＨＤＤ）などのデータ保存領域を無駄に使ってしまうという問題もあった。
【０００５】
本発明は、前述の問題点に鑑みてなされたものであり、イメージ画像を複数のオブジェクトに分離した際に、背景画像が白く抜けた状態になることを可及的に防止することを第１の目的とする。
また、イメージ画像を再利用する際に使用するデータ量を可及的に低減させることを第２の目的とする。
【課題を解決するための手段】
【０００６】

本発明の画像処理装置は、複数ページのイメージ画像のそれぞれを、背景画像を含む複数の画像に分離する分離手段と、前記分離手段により分離された、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間する補間手段とを有することを特徴とする。

【０００７】

本発明の画像処理方法は、複数ページのイメージ画像のそれぞれを、背景画像を含む複数の画像に分離する分離ステップと、前記分離ステップにより分離された、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間する補間ステップとを有することを特徴とする。

【０００８】

本発明のコンピュータプログラムは、前記記載の画像処理方法におけるステップをコンピュータに実行させることを特徴とする。

【発明の効果】

【０００９】

本発明によれば、イメージ画像から分離された、所定ページの背景画像の画素を、他のページの背景画像の画素を用いて補間するようにしたので、前記所定ページの背景画像で白く抜けてしまっているところを、前記他のページの背景画像を用いて消すことができる。これにより、背景画像が白く抜けた状態になることを可及的に防止することができ、背景画像を再利用することが可能になる。

また、本発明の他の特徴によれば、補間された所定ページの背景画像を記憶し、記憶した背景画像を使用して、複数ページの背景画像を形成するようにしたので、複数ページのイメージ画像に対して、同一の背景画像を１枚だけ持つようにすることができる。これにより、背景画像をイメージ画像毎に記憶する必要がなくなり、記憶容量を小さくすることができ、イメージ画像を効率良く管理することが可能になる。

【発明を実施するための最良の形態】

【００１０】

（第１の実施の形態）

次に、図面を参照しながら、本発明の第１の実施の形態について説明する。

図１は、本実施の形態における画像処理システムの構成の一例を示すブロック図である。

この画像処理システムは、オフィス１０とオフィス２０とがインターネット１０４を介して相互に接続された環境で実現される。オフィス１０内に構築されたＬＡＮ１０７には、ＭＦＰ１００と、ＭＦＰ１００を制御するマネージメントＰＣ１０１と、クライアントＰＣ（外部記憶手段）１０２と、文書管理サーバ１０６ａと、そのデータベース１０５ａと、プロキシサーバ１０３ａとが接続されている。

【００１１】

オフィス内のＬＡＮ１０７及びオフィス２０内のＬＡＮ１０８は、プロキシサーバ１０３ａ、１０３ｂを介してインターネット１０４に接続される。本実施の形態のＭＦＰ１００は、紙文書を読み取る画像読み取り部と、前記画像読み取り部で読み取った画像信号に対して画像処理を行う画像処理部の一部とを担当する。そして、ＭＦＰ１００で生成された画像信号は、ＬＡＮ１０９を用いてマネージメントＰＣ１０１に入力される。マネージメントＰＣ１０１は、通常のパーソナルコンピュータ（ＰＣ）であり、内部に画像記憶手段、画像処理手段、表示手段、及び入力手段を有するが、その一部がＭＦＰ１００に一体化して構成されている。

【００１２】

図２は、ＭＦＰ１００の構成の一例を示すブロック図である。

図２において、オートドキュメントフィーダー（以降ＡＤＦと記す）を含む画像読み取り部１１０は、束状の原稿画像、或いは１枚の原稿画像を図示しない光源で照射し、原稿からの反射像を、レンズを用いて固体撮像素子上に結像する。そして、固体撮像素子からラスター状の画像読み取り信号を、６００ＤＰＩの密度のイメージ情報として得る。通常

の複写機能は、この画像信号をデータ処理部１１５で画像処理して記録信号を生成する。そして、生成した記録信号を記録装置１１２に出力し、紙上に画像を形成する。なお、複数ページ毎に複写する場合には、記録装置１１１に１ページ分の記録データを一旦記憶保持した後、複数ページの記録データを記録装置１１２に順次出力して紙上に画像を形成する。

【００１３】

一方、クライアントＰＣ１０２から出力されるプリントデータは、ＬＡＮ１０７からネットワークＩＦ１１４を経てデータ処理装置１１５に入力される。入力されたプリントデータは、データ処理装置１１５で記録可能なラスターデータに変換された後、記録装置１１２で紙上に記録画像として形成される。

ＭＦＰ１００に対する操作者の指示は、ＭＦＰ１００に装備されたキー操作部と、マネージメントＰＣ１０１に装備されたキーボード及びマウス等からなる入力装置１１３とから行われる。これら一連の動作は、データ処理装置１１５内の図示しない制御部で制御される。

一方、操作入力の状態表示及び処理中の画像データの表示は、表示装置１１６で行われる。なお、記憶装置１１１は、マネージメントＰＣ１０１からも制御され、これらＭＦＰ１００とマネージメントＰＣ１０１とのデータの授受及び制御は、ネットワークＩＦ１１７および直結したＬＡＮ１０９を用いて行われる。

【００１４】

［処理概要］

次に、図３のフローチャートを参照しながら、本実施の形態よる画像処理全体の概要を説明する。

図３において、まず、ＭＦＰ１００の画像読み取り部１１０を動作させて１枚の原稿をラスター状に走査し、イメージ画像の入力処理を行って６００ＤＰＩ－８ビットの画像信号を得る（ステップＳ１２０のイメージ情報入力処理）。そして、前記画像信号をデータ処理部１１５で前処理し、前処理した画像信号を、記憶装置１１１に１ページ分の画像データとして保存する。

【００１５】

マネージメントＰＣ１０１に配設されているＣＰＵは、前記保存された画像信号に基づいて、文字部分及び線画部分と、ハーフトーンの画像部分とに領域を分離する。そして、文字部分については、更に段落で塊として纏まっているブロック毎に、或いは、線で構成された表及び図形に分離し、各々セグメント化する。

一方、ハーフトーンで表現される画像部分は、矩形に分離されたブロックの画像部分や、背景部分等、所謂ブロック毎に独立したオブジェクトに分割する（ステップＳ１２１のブロックセレクション処理）。

このとき、原稿画像中に付加情報として記録された２次元バーコード、或いはＵＲＬに該当するオブジェクトを検出する。そして、ＵＲＬについてはＯＣＲ（Optical Character Reader）で文字認識し、２次元バーコードについてはそのマークを解読して（ステップＳ１２２のOCR/OMR処理）、原稿のオリジナル電子ファイルが格納されている記憶装置内のポインター情報を検出する（ステップＳ１２３のポインター情報検出処理）。

なお、ポインター情報を付加する手段としては、文字と文字の間隔に情報を埋め込む方法や、ハーフトーンの画像に埋め込む方法等、直接可視化されない所謂電子透かしによる方法も有る。また、ポインター情報とは、オリジナル電子ファイルや、その格納場所を指し示すものであり、本発明の目的を達成可能なものであればいかなる構成を採用してもよい。

【００１６】

そして、ポインター情報が検出された場合（ステップＳ１２４でＹＥＳと判定された場合）には、ステップＳ１２５に進み、ポインター情報で示されたアドレスからオリジナル電子ファイルを検索する。

オリジナル電子ファイルは、図１においてクライアントＰＣ１０２に配設されているハ

ードディスク内、或いはオフィス１０、２０のＬＡＮ１０７、１０８に接続された文書管理サーバ１０５ａ、１０５ｂに配設されたデータベース内、或いはＭＦＰ１００自体に配設されている記憶装置１１１のいずれかに格納されている。したがって、ステップＳ１２３で得られたポインター情報（アドレス情報）に従って、これらの記憶装置内を検索する。

【００１７】

そして、ステップＳ１２５でオリジナル電子ファイルが見つからなかった場合、又は見つかったがＰＤＦあるいはtiffに代表される所謂イメージファイルであった場合、又はポインター情報自体が存在しなかった場合（ステップＳ１２４、Ｓ１２５でＮＯと判定された場合）には、ステップＳ１２６に進む。一方、ステップＳ１２５でオリジナル電子ファイルが見つかった場合（ステップＳ１２５でＹＥＳと判定された場合）には、ステップＳ１３３に進み、オリジナル電子ファイルが格納されているアドレスをユーザに通知する。

【００１８】

ステップＳ１２６は、ファイル検索処理（所謂文書検索処理ルーチン）である。

まず、ステップＳ１２２で各文字ブロックに対して行ったＯＣＲ処理の結果から、単語を抽出して全文検索を行ったり、各オブジェクトの配列と各オブジェクトの属性とから所謂レイアウト検索を行ったりする。

以上のような検索の結果、類似度の高い電子ファイルが見つかった場合、サムネイル画像等を表示し、複数のサムネイル画像の中から、操作者の選択が必要なら操作者の入力操作よってファイルの特定を行う（ステップＳ１２７の候補表示／選択処理）。なお、候補が１つのファイルの場合には、自動的にステップＳ１２８からステップＳ１３３に分岐し、オリジナル電子ファイルが格納されているアドレスをユーザに通知する。

ステップＳ１２６のファイル検索処理で電子ファイルが見つからなかった場合、或いは、見つかったがＰＤＦあるいはｔｉｆｆに代表される所謂イメージファイルであった場合には、ステップＳ１２８からステップＳ１２９に進む。

【００１９】

ステップＳ１２９において、イメージデータをベクトルデータへ変換する処理を行い、オリジナル電子ファイルに近い電子ファイルにイメージデータを変換する（ステップＳ１２９のベクトル化処理）。

具体的に説明すると、先ず、ステップＳ１２２でＯＣＲ処理された文字ブロックに対しては、更に文字のサイズ、スタイル、及び字体を認識し、原稿を走査して得られた文字に可視的に忠実なフォントデータに変換する。一方、線で構成される表や、図形ブロックに対してはアウトライン化する。また、画像ブロックに対しては、イメージデータとして個別のＪＰＥＧファイルとして処理する。

これらのベクトル化処理は、オブジェクト毎に行い、更に各オブジェクトのレイアウト情報を保存して、例えば、アプリデータ（rtf：Rich Text Format）に変換して（ステップＳ１３０のアプリデータへの変換処理）、電子ファイルとして記憶装置１１１に格納する（ステップＳ１３１の電子ファイル格納処理）。

【００２０】

ベクトル化した原稿画像は、以降同様の処理を行う際に電子ファイルとして直接検索出来るように、先ず、ステップＳ１３２において、検索の為のインデックス情報を生成して検索用インデックスファイルに追加する。更に、ステップＳ１３４において、操作者が行いたい処理が、記録であると判断すれば、ステップＳ１３５に進み、ポインター情報をイメージデータとしてファイルに付加する。

ファイル検索処理で電子ファイルが特定できた場合も同様に、以降からは電子ファイルを直接特定出来るようにする為に、ステップＳ１２８からステップＳ１３３に進み、オリジナル電子ファイルが格納されているアドレスをユーザに通知し、紙に記録する場合は、ステップＳ１３６において同様にポインター情報を電子ファイルに付加する。

なお、ステップＳ１２５でポインター情報から電子ファイルが特定できた場合、又はステップＳ１２６のファイル検索処理で電子ファイルが特定できた場合、及びステップＳ１

２９のベクトル化処理により電子ファイルに変換した場合には、ステップＳ１３３において、オリジナル電子ファイルが格納されているアドレスをユーザに通知する。

【００２１】

以上の手順によって得られた電子ファイルは、オリジナルの電子情報もしくはそれに非常に近いベクトル情報として編集可能であるので、電子ファイル自体を用いて、例えば文書の加工、蓄積、伝送、及び記録をステップＳ１３６で行うことが可能になる。また、これらの処理を行うことにより、イメージデータを直接用いる場合に比べて情報量を削減することができる。したがって、記憶装置１１１などにおける蓄積効率を高め、伝送時間を短縮し、さらに、高品位なデータとして記録表示することができる。

【００２２】

以下、各処理ブロックに対して詳細に説明する。

［ブロックセレクション処理］

先ずステップＳ１２１で示すブロックセレクション処理について説明する。

ブロックセレクション処理とは、図４の左に示すステップＳ１２０で読み取った一頁のイメージデータを、同じく図４の右に示すように、オブジェクト毎の塊（ブロック）として認識し、これら認識したブロックの各々が文字、図画、写真、線、及び表等のうち、どの属性に属するかを判定し、判定した結果に基づいて、前記イメージデータを異なる属性を持つ領域に分割する処理である。

【００２３】

このようなブロックセレクション処理の具体例を以下に説明する。

先ず、入力されたイメージデータ（画像）を白黒に二値化し、輪郭線追跡を行って、黒画素輪郭で囲まれる画素の塊を抽出する。面積の大きい黒画素の塊については、内部にある白画素に対しても輪郭線追跡を行い白画素の塊を抽出する。さらに、一定面積以上の白画素の塊の内部からは再帰的に黒画素の塊を抽出する。

【００２４】

このようにして得られた黒画素の塊を、大きさ及び形状で分類し、異なる属性を持つ領域へ分類していく。たとえば、縦横比が１に近く、大きさが一定の範囲のものを文字相当の画素塊とし、さらに近接する文字が整列良くグループ化可能な部分を文字領域とする。また、扁平な画素塊を線領域とする。また、一定の大きさ以上でかつ四角系の白画素塊を整列よく内包する黒画素塊の占める範囲を表領域とする。また、不定形の画素塊が散在している領域を写真領域とする。さらに、それ以外の任意形状の画素塊を図画領域とする。なお、分類方法は、前述したものに限定されないということは勿論である。

以上のようなブロックセレクション処理で得られた各ブロックに対するブロック情報の一例を図５に示す。

これらのブロック毎の情報は、以降に説明するベクトル化、或いは検索の為の情報として用いる。

【００２５】

［ポインター情報の検出］

次に、ステップＳ１２２で示す、ファイルの格納位置をイメージ画像から抽出する為のOCR/OMR処理について説明する。

図６は、原稿画像中に付加された２次元バーコード（ＱＲコードシンボル）を復号して、データ文字列を出力する手順を示すフローチャートである。また、図７は、２次元バーコードが付加された原稿３１０の一例を示す図である。

まず、データ処理装置１１５内のページメモリに格納された原稿３１０を表すイメージ画像をＣＰＵ（不図示）で走査して、先に説明したブロックセレクション処理の結果から所定の２次元バーコードシンボル３１１の位置を検出する。ＱＲコードの位置検出パターンは、２次元バーコードシンボル３１１の４隅のうちの３隅に配置される同一の位置検出要素パターンから構成される（ステップＳ３００）。

【００２６】

次に、前記位置検出パターンに隣接する形式情報を復元し、２次元バーコードシンボル

３１１に適用されている誤り訂正レベル及びマスクパターンを得る（ステップＳ３０１）。

そして、２次元バードコードシンボル３１１の型番を決定した後（ステップＳ３０２）、前記形式情報で得られたマスクパターンを使って、符号化領域ビットパターンをＸＯＲ（排他的論理和）演算することによってマスク処理を解除する（ステップＳ３０３）。

次に、モデルに対応する２次元バードーードの配置規則に従い、シンボルキャラクタを読み取り、メッセージのデータ及び誤り訂正コード語を復元する（ステップＳ３０４）。ここで、モデルとは、２次元バーコードや１次元バーコードの標準フォーマットのことを指す。本実施の形態では、ＱＲコードシンボルの標準フォーマットモデルのことを指し、この標準フォーマットのデータ配置規則にのっとって、２次元バーコードを解析する。

次に、復元されたコード上に、誤りがあるかどうかの検出を行い（ステップＳ３０５）、誤りが検出された場合、ステップＳ３０６に進み、これを訂正する。

次に、誤り訂正されたデータコード語を復元する。そして、モード指示子および文字数指示子に基づいて、前記復元したデータコード語をセグメントに分割する（ステップ３０７）。

最後に、仕様モードに基づいてデータ文字を復号し、結果を出力する（ステップＳ３０８）。

なお、２次元バーコード内に組み込まれたデータは、対応するファイルのアドレス情報を表しており、このアドレス情報は、例えばファイルサーバ名及びファイル名からなるパス情報で構成される。或いは、前記アドレス情報は、対応するファイルへのＵＲＬで構成される。

【００２７】

また、本実施の形態では、ポインター情報が２次元バーコードを用いて付与された原稿３１０について説明したが、ポインター情報が文字列で直接記録される場合には、所定のルールに従った文字列のブロックを、先のブロックセレクション処理で検出し、ポインター情報を示す文字列の各文字を文字認識することで、オリジナル電子ファイルのアドレス情報を直接得ることが可能である。

【００２８】

また、図７に示した原稿（文書）３１０の第１の文字ブロック３１２、或いは第２の文字ブロック３１３の文字列に対して、隣接する文字と文字との間等に、視認し難い程度の変調を加え、その文字と文字との間隔に情報を埋め込むことでもポインター情報を付与できる。このようなポインター情報は、所謂透かし情報であり、後述する文字認識処理を行う際に各文字の間隔を検出することにより得られる。また、自然画３１４の中に電子透かしとしてポインター情報を付加することも可能である。

【００２９】

［ポインター情報によるファイル検索］

次に、図３で先に説明したステップＳ１２５及びステップＳ１２８で示す、ポインター情報から電子ファイルを検索する手順の一例について図８のフローチャートを使用して説明する。

まず、ポインター情報に含まれるアドレスに基づいて、ファイルサーバを特定する。（ステップＳ４００）

ここで、ファイルサーバとは、クライアントＰＣ１０２や、データベース１０５を内蔵する文書管理サーバ１０６や、記憶装置１１１を内蔵するＭＦＰ１００自身を指す。

また、アドレスとは、ＵＲＬや、サーバ名とファイル名とからなるパス情報である。

【００３０】

ファイルサーバが特定できたら、ファイルサーバに対してアドレスを転送する（ステップＳ４０１）。

ファイルサーバは，アドレスを受信すると、該当するファイルを検索する（ステップＳ４０２）。

検索の結果、ファイルが存在しない場合（ステップＳ４０３でＮｏの場合）には、本フ

ローチャートによる処理を終了する。
一方、該当するファイルが存在した場合（ステップＳ４０３でＹｅｓの場合）には、図３で説明したように、ファイルのアドレスを通知すると共に（ステップＳ１３３）、ユーザの希望する処理が画像ファイルデータの取得であれば、ＭＦＰ１００に対してファイルを転送する（ステップＳ４０８）。
【００３１】
［ファイル検索処理］
次に、図３のステップＳ１２６で示すファイル検索処理の詳細について図５及び図１０を使用して説明を行う。
ステップＳ１２６の処理は、前述したように、ステップＳ１２４で入力原稿（入力ファイル）にポインター情報が存在しなかった場合、または、ポインター情報は在るが電子ファイルが見つからなかった場合、或いは電子ファイルがイメージファイルであった場合に行われる。
【００３２】
ここでは、ステップＳ１２２の結果、抽出された各ブロック及び入力ファイルが、図５に示す情報（ブロック情報５１、入力ファイル情報５２）を備えるものとする。情報内容として、属性、座標位置、幅と高さのサイズ、及びＯＣＲ情報の有無を例としてあげる。属性は、文字（１）、線（４）、写真（５）、絵（２）、表（３）、及びその他に分類する。また簡単に説明を行うため、ブロックは座標Ｘの小さい順、（即ち、例えば、Ｘ１＜Ｘ２＜Ｘ３＜Ｘ４＜Ｘ５＜Ｘ６）にブロック１、ブロック２、ブロック３、ブロック４、ブロック５、ブロック６と名前をつけている。ブロックの総数は、入力ファイル中の全ブロック数であり、図５の場合、ブロック総数Ｎは６である。
【００３３】
以下、これらの情報を使用して、データベース１０５内から、入力ファイルに類似したファイルのレイアウト検索を行うフローチャートを図１０に示す。ここで、データベースファイルは、図５と同様の情報を備えることを前提とする。
図１０のフローチャートの流れは、入力ファイルとデータベース１０５中のファイルとを順次比較するものである。
まず、ステップＳ５１０にて、後述する類似率などの初期化を行う。
次に、ステップＳ５１１にて、入力ファイルにおけるブロックの総数と、データベース１０５中のファイルにおけるブロックの総数との比較を行う。ここで、この比較の結果、ブロックの総数の差が所定の範囲内の場合、さらにファイル内のブロックの情報を順次比較する（ステップＳ５１２、Ｓ５１４、Ｓ５１６）。
【００３４】
ブロックの情報の比較は、ステップＳ５１３、Ｓ５１５、Ｓ５１８にて、属性類似率、サイズ類似率、ＯＣＲ類似率をそれぞれ算出して行われ、入力ファイルの全ブロックに対する処理が終了したら（ステップＳ５１９のＹｅｓ）、ステップＳ５２２にてそれらをもとに総合類似率を算出する。各類似率の算出方法については、公知の技術が用いられるので説明を省略する。
【００３５】
なお、ステップＳ５１２、Ｓ５１４、Ｓ５１６にて、入力ファイルにおけるブロックの情報と、データベース１０５中のファイル（以下、データベースファイルと記す）におけるブロックの情報とが一致または所定の範囲内にない場合には、ステップＳ５２１に進む。そして、ステップＳ５２１にて、次のブロックへ処理を移行する。具体的に説明すると、入力ファイルにおけるブロックの総数ｎが、データベースファイルにおけるブロックの総数Ｎ以上である場合には、入力ファイルの次のブロックへ処理を移行する。一方、入力ファイルのブロックの総数ｎが、データベースファイルにおけるブロックの総数Ｎよりも少ない場合には、データベースファイルにおける次のブロックへ処理を移行する。
そして、入力ファイルにおけるブロックの情報と、データベースファイルにおけるブロックの情報とが一致または所定の範囲内になるまで、ステップＳ５１２～Ｓ５１６の処理

を繰り返し行う。
　また、ステップＳ５１９にて、入力ファイルの全ブロックに対する処理が終了していないと判定した場合には、前記ステップＳ５２１に進む。
【００３６】
　ステップＳ５２３にて、総合類似率が、予め設定された閾値Ｔｈより高いと判定すれば、ステップＳ５２４にてそのファイルを類似候補としてあげる。但し、図中のＮ、Ｗ、Ｈは、入力ファイルのブロック総数、各ブロック幅、各ブロック高さとし、ΔＮ、ΔＷ、ΔＨは、入力ファイルのブロック情報を基準として誤差を考慮したものである。ｎ、ｗ、ｈは、データベースファイルのブロック総数、各ブロック幅、各ブロック高さとする。また、不図示ではあるが、ステップＳ５１４にて、サイズの比較とともに、位置情報ＸＹの比較などを行ってもよい。
【００３７】
　以上のような検索の結果、類似度が閾値Ｔｈより高い場合（ステップＳ５２３でＹｅｓの場合）には、ステップＳ５２４において候補として保存されたデータベースファイルをサムネイル等で表示する（図３のステップＳ１２７）。複数の中から操作者の選択が必要なら操作者の入力操作よってファイルの特定を行う。最後に、ステップＳ５２５にて、データベース１０５内の全てのファイルに対する処理が終了したか否かを判定し、終了したら、図１０に示すフローチャートの処理を終了する。
　なお、ステップＳ５２５にて、データベース１０５内の全てのファイルに対する処理が終了していないと判定された場合と、ステップＳ５２３にて、総合類似度が閾値Ｔｈより高くないと判定された場合と、ステップＳ５１１にて、入力ファイルにおけるブロックの総数Ｎと、データベースファイルにおけるブロックの総数ｎとの差が所定の範囲内にない場合には、ステップＳ５２６に進む。そして、ステップＳ５２６にて、データベース１０５の次のファイルへ移行し、ステップＳ５１０に進む。
【００３８】
［ベクトル化処理］
　次に、図３のステップＳ１２９で示されるベクトル化処理について詳説する。
　ファイルサーバにオリジナル電子ファイルが存在しない場合は、図４に示すようなイメージデータを、ブロック毎にベクトル化する。
【００３９】
　まず、文字ブロックに対しては各文字に対して文字認識処理を行う。
『文字認識』
　文字認識処理では、文字単位で切り出された画像に対し、パターンマッチの一手法を用いて文字認識を行い、対応する文字コードを得る。この文字認識処理は、文字画像から得られる特徴を数十次元の数値列に変換した観測特徴ベクトルと、あらかじめ字種毎に求められている辞書特徴ベクトルとを比較し、最も距離の近い字種を認識結果とする処理である。特徴ベクトルの抽出には、種々の公知手法があり、たとえば、文字をメッシュ状に分割し、各メッシュ内の文字線を方向別に線素としてカウントしたメッシュ数次元ベクトルを用いる方法がある。
【００４０】
　ブロックセレクション（ステップＳ１２１）で抽出された文字領域に対して文字認識を行う場合には、まず該当領域に対して、横書き及び縦書きの判定を行い、各々対応する方向に行を切り出し、その後、文字を切り出して文字画像を得る。横書き及び縦書きの判定は、該当する領域内で、画素値に対する水平及び垂直の射影を取り、水平射影の分散が大きい場合は横書き領域、垂直射影の分散が大きい場合は縦書き領域と判断すればよい。文字列および文字への分解は、横書きならば水平方向の射影を利用して行を切り出し、さらに切り出された行に対する垂直方向の射影から、文字を切り出すことで行う。縦書きの文字領域に対しては、水平と垂直を逆にすればよい。なお、このとき、文字のサイズを検出することが出来る。
【００４１】

『フォント認識』
文字認識の際に用いる、字種数ぶんの辞書特徴ベクトルを、文字形状種すなわちフォント種に対して複数用意し、マッチングの際に文字コードとともにフォント種を出力することで、文字のフォントを認識することが出来る。
【００４２】
『文字のベクトル化』
前記文字認識及びフォント認識よって得られた、文字コード及びフォント情報と、各々あらかじめ用意されたアウトラインデータとを用いて、文字部分の情報をベクトルデータに変換する。なお、元原稿がカラーの場合は、カラー画像から各文字の色を抽出してベクトルデータとともに記録する。
以上の処理により、文字ブロックに属するイメージ情報を、形状、大きさ、及び色がほぼ忠実なベクトルデータに変換することが出来る。
【００４３】
『文字以外の部分のベクトル化』
ブロックセレクション処理（ステップＳ１２１）で、図画、線、及び表に属するとされた領域を対象に、抽出された画素塊の輪郭をベクトルデータに変換する。具体的には、輪郭をなす画素の点列を角と看做される点で区切って、各区間を部分的な直線あるいは曲線で近似する。角とは曲率が極大となる点である。この曲率が極大となる点は、図１１に図示するように、任意点Ｐｉに対し左右にｋ個の離れた２つの点Pi-k、Pi+kの間に弦を引いたときに、この弦と点Ｐｉとの距離が極大となる点として求められる。さらに、点Pi-k、Pi+kの間の弦の長さを弧の長さで除した値（弦の長さ／弧の長さ）をＲとし、この値Ｒが閾値以下である点を角とみなすことができる。角によって分割された後の各区間において、直線については、点列に対する最小二乗法などを用いてベクトル化することができる。また、曲線については、３次スプライン関数などを用いてベクトル化することができる。
また、対象が内輪郭を持つ場合、ブロックセレクション処理（ステップＳ１２１）で抽出した白画素輪郭の点列を用いて、同様に部分的直線あるいは曲線で前記内輪郭を近似する。
【００４４】
以上のように、輪郭の区分線近似を用いれば、任意の形状の図形のアウトラインをベクトル化することができる。元原稿がカラーの場合は、カラー画像から図形の色を抽出してベクトルデータとともに記録する。
さらに、図１２に示すように、ある区間で外輪郭と、内輪郭あるいは別の外輪郭とが近接している場合、２つの輪郭線をひとまとめにし、太さを持った線として表現することができる。具体的には、ある輪郭の点Ｐｉから最短距離となる別の輪郭上の点Ｑｉまで線を引く。そして、これら２点Ｐｉ、Ｑｉ間の距離ＰＱｉが平均的に一定長以下の場合、２点Ｐｉ、Ｑｉ間の距離ＰＱｉの中点を点列として直線あるいは曲線で近似し、その太さを２点Ｐｉ、Ｑｉ間の距離ＰＱｉの平均値とする。線や、線の集合体である表罫線は、前記のような太さを持つ線の集合として効率よくベクトル表現することができる。
【００４５】
なお、先に説明した文字ブロックに対する文字認識処理を用いたベクトル化では、文字認識処理の結果、辞書からの距離が最も近い文字を認識結果として用いるようにした。しかしながら、この距離が所定値以上の場合は、文字認識処理により得られた文字が必ずしも本来の文字に一致せず、形状が類似する文字に誤認識している場合が多い。従って、本実施の形態では、この様な文字に対しては、前記のように、一般的な線画と同じに扱い、その文字をアウトライン化する。このようにすれば、従来の文字認識処理では誤認識を起こす文字に対しても、誤った文字にベクトル化されず、可視的にイメージデータに忠実なアウトライン化によるベクトル化を行える。
また、本実施の形態では、写真と判定されたブロックに対しては、ベクトル化することが出来ない為、イメージデータのままとする。
【００４６】

［図形認識］
前述したように、任意の形状の図形のアウトラインをベクトル化した後、これらベクトル化された区分線（以降、ベクトルデータと記す）を図形オブジェクト毎にグループ化する処理について説明する。
図１３は、ベクトルデータを図形オブジェクト毎にグループ化するまでの手順の一例を説明するフローチャートを示している。
まず、各ベクトルデータの始点と終点とを算出する（ステップＳ７００）。
次に、各ベクトルの始点の情報と、終点の情報とを用いて、図形要素を検出する（ステップＳ７０１）。ここで、図形要素の検出とは、区分線が構成している閉図形を検出することである。検出に際しては、閉形状を構成する各ベクトルが、その両端にそれぞれ連結するベクトルを有しているという原理を応用する。
【００４７】
次に、図形要素内に存在する他の図形要素、もしくは区分線をグループ化し、一つの図形オブジェクトとする（ステップＳ７０２）。また、図形要素内に他の図形要素、及び区分線が存在しない場合には、その図形要素を図形オブジェクトとする。
【００４８】
図１４は、図形要素を検出する手順の一例を説明するフローチャートを示している。
先ず、ベクトルデータの両端に連結していない不要なベクトルを除去し、閉図形構成ベクトルを抽出する（ステップＳ７１０）。
次に、閉図形構成ベクトルの中から、その閉図形構成ベクトルの始点を開始点とし、時計回りに順にベクトルを追っていく。そして、開始点に戻るまでベクトルを追っていき、通過したベクトルを、全て一つの図形要素を構成する閉図形としてグループ化する（ステップＳ７１１）。また、閉図形の内部にある閉図形構成ベクトルも全てグループ化する。さらに、まだグループ化されていないベクトルの始点を開始点とし、同様の処理を繰り返す。
最後に、ステップＳ７１０で除去された不要なベクトルのうち、ステップＳ７１１で閉図形としてグループ化されたベクトルに接合しているもの（閉図形連結ベクトル）を検出し、これらを一つの図形要素としてグループ化する（ステップＳ７１２）。
以上によって図形ブロックを個別に再利用可能な図形オブジェクトとして扱うことが可能になる。
【００４９】
［アプリデータへの変換処理］
ところで、一頁分のイメージデータをブロックセレクション処理（ステップＳ１２１）し、ベクトル化処理（ステップＳ１２９）した結果は、図１５に示す様な中間データ形式のファイルとなるが、このようなデータ形式はドキュメント・アナリシス・アウトプット・フォーマット（DAOF）と呼ばれる。
【００５０】
図１５は前記DAOFのデータ構造の一例を示す図である。
図１５において、７９１はヘッダ（Header）であり、処理対象の文書画像データに関する情報が保持される。レイアウト記述データ部７９２では、文書画像データ中のTEXT（文字）、TITLE（タイトル）、CAPTION（キャプション）、LINEART（線画）、PICTURE（自然画）、FRAME（枠）、及びTABLE（表）等の属性毎に認識された各ブロックの属性情報と、その矩形アドレス情報とを保持する。
【００５１】
文字認識記述データ部７９３では、TEXT（文字）、TITLE（タイトル）、及びCAPTION（キャプション）等のTEXTブロックを文字認識して得られる文字認識結果を保持する。
表記述データ部７９４では、TABLE（表）ブロックの構造の詳細を格納する。画像記述データ部７９５は、PICTURE（自然画）やLINEART（線画）等のブロックのイメージデータを文書画像データから切り出して保持する。
【００５２】

このようなDAOFは、中間データとしてのみならず、それ自体がファイル化されて保存される場合もあるが、このファイルの状態では、所謂一般の文書作成アプリケーションで個々のオブジェクトを再利用することは出来ない。そこで、次に、このDAOFからアプリデータに変換する処理（図３のステップＳ１３０）について詳説する。

【００５３】

図１６は、この処理における全体の手順の概略を説明するフローチャートである。

まず、ステップＳ８０００では、DAOFデータの入力を行う。

次に、ステップＳ８００２では、アプリデータの元となる文書構造ツリーの生成を行う。

次に、ステップＳ８００４では、ステップＳ８００２で生成された文書構造ツリーを元に、DAOF内の実データを流し込み、実際のアプリデータを生成する。

【００５４】

図１７は、ステップＳ８００２において文書構造ツリーを生成する手順の一例を説明するフローチャートである。図１８は、文書構造ツリーの具体的な内容の一例を説明する図である。

全体制御の基本ルールとして、処理の流れはミクロブロック（単一ブロック）からマクロブロック（ブロックの集合体）へ移行する。以後の説明において、ブロックとは、ミクロブロック、及びマクロブロックの全体を指すこととする。

まず、ステップＳ８１００では、縦方向の関連性を元に、ブロック単位で再グループ化する。スタート直後はミクロブロック単位での判定となる。

ここで、関連性とは、距離が近いことや、ブロック幅（横方向の場合は高さ）がほぼ同一であることなどで定義することができる。

また、距離、幅、及び高さなどの情報は、前記DAOFを参照し、抽出する。

図１８（ａ）は、実際のページ構成、図１８（ｂ）は、その文書構造ツリーを示した図である。

ステップＳ８１００の結果、ブロックＴ３、Ｔ４、Ｔ５が同じ階層の１つのグループＶ１として再グループ化される。また、ブロックＴ６、Ｔ７が、同じ階層の１つのグループＶ２として再グループ化される。

【００５５】

次に、ステップＳ８１０２では、縦方向のセパレータの有無をチェックする。セパレータは、例えば物理的にはDAOF中でライン属性を持つオブジェクトである。また論理的な意味としては、アプリ中で明示的にブロックを分割する要素である。ここでセパレータを検出した場合は、同じ階層で再分割する。

【００５６】

次に、ステップＳ８１０４では、分割がこれ以上存在し得ないか否かをグループ長を利用して判定する。

ここで、縦方向のグループ長がページ高さとなっている場合には、分割がこれ以上存在し得ないと判定し、文書構造ツリー生成処理を終了する。

図１８の場合は、セパレータもなく、グループの高さはページ高さではないので、ステップＳ８１０６に進む。

【００５７】

次に、ステップＳ８１０６では、横方向の関連性を元に、ブロック単位で再グループ化する。ここもスタート直後の第１回目はミクロブロック単位で判定を行うことになる。

前記関連性、及びその判定情報の定義は、縦方向の場合と同じである。

図１８の場合は、ブロックＴ１、Ｔ２が１つのグループＨ１として再グループ化される。また、ブロックＶ１、Ｖ２が１つのグループＨ２として再グループ化される。なお、これらのグループＨ１、Ｈ２は、前述したグループＶ１、Ｖ２の１つ上の同じ階層のグループとして生成される。

【００５８】

次に、ステップＳ８１０８では、横方向セパレータの有無をチェックする。

図１８では、セパレータＳ１があるので、これをツリーに登録する。このようにして、グループＨ１、Ｈ２及びセパレータＳ１という階層が生成される。

次に、ステップＳ８１１０では、分割がこれ以上存在し得ないか否かをグループ長を利用して判定する。

ここで、横方向のグループ長がページ幅となっている場合には、分割がこれ以上存在し得ないと判定し、文書構造ツリー生成処理を終了する。

一方、そうでない場合は、ステップＳ８１００に戻り、もう一段上の階層で、縦方向の関連性を元にブロックを再グループ化し、ステップＳ８１００～Ｓ８１１０までの処理を繰り返す。

【００５９】

図１８の場合は、分割幅がページ幅になっているので、ここで処理を終了し、最後にページ全体を表す最上位階層のブロックＶ０が文書構造ツリーに付加される。

以上のようにして文書構造ツリーが完成した後、その文書構造ツリー内の情報を元にステップＳ８００４においてアプリデータの生成を行う。

図１８の場合は、具体的には、以下のようになる。

すなわち、グループＨ１には、横方向に２つのブロックＴ１、Ｔ２があるので、２カラムとし、ブロックＴ１の内部情報（DAOFを参照、文字認識結果の文章、画像など）を出力した後、カラムを変え、ブロックＴ２の内部情報を出力し、その後セパレータＳ１の内部情報を出力する。

グループＨ２は、横方向に２つのブロックＶ１、Ｖ２があるので、２カラムとし、ブロックＶ１では、ブロックＴ３、Ｔ４、Ｔ５の順にその内部情報を出力し、その後カラムを変え、ブロックＶ２のブロックＴ６、Ｔ７の内部情報を出力する。

以上によりアプリデータへの変換処理が行える。

【００６０】

［ポインター情報付加処理］

次に、図３のステップＳ１３５で示すポインター情報付加処理について説明する。

今、処理すべき文書がファイル検索処理で特定された場合、あるいはベクトル化によってオリジナル電子ファイルが再生できた場合において、その文書を記録処理する場合には、紙への記録の際にポインター情報を付与するようにする。このようにすることで、その文書を用いて再度各種処理を行う場合に、オリジナル電子ファイルのデータを簡単に取得できる。

【００６１】

図１９はポインター情報としてのデータ文字列を２次元バーコード（QRコードシンボル：JIS X0510 311）にて符号化して画像中に付加する手順の一例を示すフローチャートである。

２次元バーコード内に組み込むデータは、対応するファイルのアドレス情報を表しており、例えばファイルサーバ名及びファイル名からなるパス情報で構成される。この他、対応するファイルへアクセスするためのＵＲＬや、対応するファイルが格納されているデータベース１０５内で管理されるファイルＩＤや、ＭＦＰ１００自体に配設されている記憶装置内で管理されるファイルＩＤ等で、２次元バーコード内に組み込むデータが構成されるようにしてもよい。

【００６２】

図１９において、まず、符号化する種種の異なる文字を識別するために、入力データ列を分析する。また、誤り検出及び誤り訂正レベルを選択し、入力データを収容できる最小型番を選択する（ステップＳ９００）。

次に、入力データ列を所定のビット列に変換し、必要に応じてデータのモード（数字、英数字、８ビットバイト、漢字等）を表す指示子や、終端パターンを付加する。さらに、所定のビット列に変換され、指示子や終端パターンが付加された入力データ列を所定のビットコード語に変換する（ステップＳ９０１）。

【００６３】

このとき、誤り訂正を行うため、データコード語列を型番及び誤り訂正レベルに応じて所定のブロック数に分割し、ブロック毎に誤り訂正コード語を生成し、データコード語列の後に付加する（ステップＳ９０２）。

このステップＳ９０２で得られた各ブロックのデータコード語を接続し、各ブロックの誤り訂正コード語、並びに必要に応じて剰余コード語を後続する（ステップＳ９０３）。

【００６４】

次に、位置検出パターン、分離パターン、タイミングパターン及び位置合わせパターン等とともに、コード語モジュールをマトリクスに配置する（ステップＳ９０４）。

更に、シンボルの符号化領域に対して最適なマスクパターンを選択して、マスク処理パターンをステップＳ９０４で得られたモジュールに変換する(ステップＳ９０５)。この変換は、ＸＯＲ（排他的論理和）演算などにより行う。

最後に、形式情報及び型番情報を生成して、ステップＳ９０５で得られたモジュールに付加し、２次元コードシンボルを完成する（ステップＳ９０６）。

【００６５】

前記のようにしてアドレス情報が組み込まれた２次元バーコードは、例えば、クライアントＰＣ１０２からの指示に基づいて、電子ファイルをプリントデータとして記録装置１１２を用いて、紙上に記録画像として形成する場合に、データ処理装置１１５内で記録可能なラスターデータに変換された後に、ラスターデータ上の所定の箇所に付加されて画像形成される。ここで、画像形成された紙上のポインター情報が、画像読み取り部１１０で読み取られることにより、前述したステップＳ１２３の処理にてオリジナル電子ファイルの格納場所を検出することができる。したがって、ユーザは、オリジナル電子ファイルの格納場所を容易に知ることができる。

なお、同様の目的で付加情報を付与する手段は、本実施の形態で説明した２次元バーコードの他に、例えば、ポインター情報を直接文字列で文書に付加する方法や、文書内の文字列、特に文字と文字との間隔を変調して情報を埋め込む方法や、文書中の中間調画像中に埋め込む方法等、一般に電子透かしと呼ばれる方法を適用することが出来る。

【００６６】

［ファイルアクセス権に関する変形例］

我々が扱う文書ファイルの中には、第３者による再利用を制限すべき物がある。しかしながら、前述した図８のフローチャートに示す手法ではファイルサーバに蓄積されたファイルは全て自由にアクセスでき、ファイル全体、或いはその一部のオブジェクトを全て再利用することが可能なことを前提に説明した。

【００６７】

これに対し、ここでは、アクセス権の制限が有る電子ファイルをポインター情報から検索する手順の一例を図９のフローチャートを使用して説明する。

ステップＳ４００からステップＳ４０３までは、図８と同様の為、説明を省略する。ファイルが特定された場合（ステップＳ４０３でＹｅｓの場合）、ファイルサーバはそのファイルのアクセス権情報を調べ、アクセス制限がある場合（ステップＳ４０４でＹｅｓ）には、ＭＦＰ１００に対してパスワードの送信を要求する（ステップＳ４０５）。

次に、ＭＦＰ１００は、操作者に対してパスワードの入力を促し、入力されたパスワードをファイルサーバに送信する（ステップＳ４０６）。

次に、ファイルサーバは、送信されたパスワードを照合し、一致した場合には（ステップＳ４０７のＹｅｓ）、図３で説明した様に、ファイルのアドレスを通知する（ステップＳ１３４）。このとき、ユーザの希望する処理が画像ファイルデータの取得であれば、ＭＦＰ１００に対してファイルを転送する（ステップＳ４０８）。

なお、アクセス権の制御を行う為の認証の方法は、ステップＳ４０５、Ｓ４０６に示したパスワードによる方法に限定されず、例えば、指紋認証等の一般に広く用いられている生体認証や、カードによる認証等、全ての認証手段を用いることが出来る。

【００６８】

また、本変形例では、紙文書に付加的に付与されたポインター情報によりファイルを特

定した場合についての例を示したが、図３のステップＳ１２６～Ｓ１２８で示す、所謂ファイル検索処理でファイルを特定した場合においても同様の制御が可能である。

一方、ファイルサーバ内からファイルを特定出来なかった場合（ステップＳ４０３でＮｏの場合）には、図３のステップＳ１２９～Ｓ１３２で説明したベクトル化処理に対しても、制限を加えることが出来る。即ち紙文書を走査して得られたイメージ情報（画像）から文書に対するアクセス権の制限の存在を検出した場合には、認証確認が取れた場合にのみベクトル化処理を行うことで、機密性の高い文書の使用に制限をかけることが出来る。

【００６９】

［ファイル特定に関する変形例］

前述した図３のフローチャートに示す手法で、原稿を走査して得られるイメージ情報（画像）からオリジナル電子データを特定するには、文書中に付与されたポインター情報に従うか、或いは文書中に記載された各オブジェクト情報に従うかのいずれかに依る。しかしながら、元のファイルをより正確に特定するには、前記ポインター情報と前記各オブジェクト情報との両方に従うようにすれば良い。

【００７０】

即ち、原稿中から得られるポインター情報から元のファイルの存在を検出することが出来たとしても、文書中のオブジェクト情報を更に使って、例えば、レイアウト情報に従うレイアウト検索や、文字認識されたキーワードに従う全文検索を、検出されたファイルに対して行う。そして、高い一致が得られた場合に、検出したファイルを、正式にオリジナル電子ファイルであると特定する。これは、例えば、ポインター情報の下位の部分が曖昧であったり、誤り訂正でも訂正できなかったりした場合に、検索の範囲を絞り込んでファイルを特定することが出来る為、確度の高いファイルの特定をより高速に行える。

【００７１】

［ベクトル化に関する変形例］

前述した図３のフローチャートに示す手法では、ファイル検索処理で、オリジナル電子ファイルの特定が出来ない場合に、イメージ画像全体に対してベクトル化処理を行うようにした。しかしながら、例えば、一般の文書の場合、文書中のオブジェクトの全てが新規に作成された物ではなく、一部のオブジェクトは他のファイルから流用して作成される場合がある。

例えば、背景オブジェクト（壁紙）については、文書作成アプリケーションで予め容易されている幾つかのパターンの中から選択して用いるのが通常である。

従って、このようなオブジェクトは、文書ファイルデータベースの中の他の文書ファイル中に存在している可能性が高く、又、再利用可能なベクトルデータとして存在する可能性が高い。

【００７２】

このような背景から、図３のステップＳ１２９におけるベクトル化処理の変形例として、以下のような手法が挙げられる。

まず、ブロックセレクション処理で個別のオブジェクトに分割された各オブジェクトに対して、オブジェクト単位でデータベースの中から一致するオブジェクトを一部に含むファイルを検索する。そして、一致したオブジェクトに対して、ファイルからオブジェクト単位で個別にベクトルデータを取得する。

これにより、文書全体をベクトル化する必要が無くなり、より高速にベクトル化することが出来、且つベクトル化による画質劣化を防止することが出来る。

【００７３】

一方、図３のステップＳ１２６～Ｓ１２８におけるファイル検索処理で、オリジナル電子ファイルがＰＤＦとして特定できた場合、ＰＤＦファイルが、その文書の文字オブジェクトに対して既に文字認識された文字コードを、付加ファイルとして有している場合がある。

このようなＰＤＦファイルをベクトル化する際には、文字コードファイルを用いることにより、ステップＳ１２９以降のベクトル化処理の中の文字認識処理を省くことが出来る

。即ち、ベクトル化処理をより高速に処理することが可能になる。

【００７４】

以下、本実施の形態の中心となる背景部分の補間、および共用について説明する。

まず、図２０及び図２２に示した原稿を読み込んだときの背景を補間する方法について説明である。図２０は、１枚目の原稿画像を示す図である。

前述したベクトル化処理により、領域２００１は文字と認識される。また、領域２００２は円グラフなので図と認識される。

図２１は、図２０に示した原稿画像から背景オブジェクト以外のオブジェクトを抽出した後に残る背景画像を示す図である。図２１において、領域２１０１は文字が抜けたところ、領域２１０２は円グラフが抜けたところを示している。オブジェクトが抜けたところは、画素のレベルは白、輝度であればＲ＝２５５,Ｇ＝２５５,Ｂ＝２５５になる。

【００７５】

図２２は、２枚目の原稿画像を示す図である。図２０に示した原稿画像と同様に、ベクトル化処理により、領域２２０１は写真として認識される。また、領域２２０２は、文字として認識される。

図２３は、図２２に示した原稿画像から背景オブジェクト以外のオブジェクトを抽出した後に残る背景画像（背景オブジェクト）を示す図である。図２３において、領域２３０１は写真が抜けたところ、領域２３０２は文字が抜けたところを示している。

【００７６】

図２４に、図２１に示した背景画像と、図２２に示した背景画像とを重ねた画像を示す。図２４では、分かり易くするために、図２３の背景で白く抜けた部分（ＡＧＥ）を黒で示す。

図２４において、領域２４０１は、図２１と図２３に示した背景画像の白く抜けた部分が重なるところを示す。

そして、本実施の形態では、２枚の画像を用いることにより背景を補間し、図２５のように、図２１に示した背景画像の白く抜けた部分を、図２３に示した背景画像で補間した画像を生成する。背景画像の枚数を多くすることで、図２６のように、白い部分がない完全な背景画像を生成することも可能である。

【００７７】

ここで、図２７のフローチャートを参照しながら、背景画像を補間する際の手順の一例を、図３に示したベクトル化処理、アプリケーションデータへの変換処理、及び電子ファイル格納処理に適用した場合の動作について説明する。

まず、ステップＳ２７０１において、原稿画像を文字、図形、表、写真、及び背景に分離する。

次に、ステップＳ２７０２において、原稿画像が複数枚あるかどうかを判断する。この判断の結果、原稿画像が複数枚ないときは、背景画像（背景オブジェクト）を補間できないため、後述するステップＳ２７０３～Ｓ２７０８を省略してステップＳ２７０９に進む。

【００７８】

一方、原稿が複数枚のときは、１枚目の背景画像を他の背景画像で補間する。すなわち、ステップＳ２７０３において、Ｎの値をインクリメントする。使用枚数Ｎの初期値は１である。したがって、最初にこの処理を通ると、Ｎは２になる。なお、Ｎの最大値は、補間に使用する背景画像の枚数である。

次に、ステップＳ２７０４において、１枚目の背景画像で画素値が白の部分の座標（Ｘ，Ｙ）を検出する。この座標の検出は、オブジェクトを抜いた部分の座標を管理しておいて、その管理している座標（Ｘ，Ｙ）を使ってもよいし、主走査方向（Ｘ方向）の画素と、副走査方向（Ｙ方向）の画素をカウントするカウンタを使って、全画素をサーチして、画素値が白のところを検索してもよい。

【００７９】

次に、ステップＳ２７０５において、補間に使おうとしているＮ枚目の画像の座標（Ｘ

，Ｙ）の画素が白でないことを確認し、白なら後述するステップＳ２７０６を省略してステップＳ２７０７に進む。一方、白でなければ背景領域と判断し、ステップＳ２７０６に進み、１枚目の背景画像の座標（Ｘ，Ｙ）の画素値を、Ｎ枚目の背景画像の座標（Ｘ，Ｙ）の画素値で置き換える。

【００８０】

次に、ステップＳ２７０７において、Ｎ枚目の背景画像を用いて画素値の置き換えが終了かどうかを判断し、終了でなければ、ステップＳ２７０４に戻り、Ｎ枚目の背景画像を用いた画素値の置き換えが終了するまで、ステップＳ２７０４～Ｓ２７０７を繰り返す。一方、Ｎ枚目の画像を用いて画素値の置き換えが終了するとステップＳ２７０８に進む。

【００８１】

次に、ステップＳ２７０８において、全てのページを用いて画素値の置き換えが終了したかどうかを判断し、終了でなければ、ステップＳ２７０３に戻り、全てのページを用いて画素値の置き換えが終了するまで、ステップＳ２７０３～Ｓ２７０８を繰り返す。一方、全てのページを用いて画素値の置き換えが終了すると、ステップＳ２７０９に進み、画素値が置き換えられた背景画像をアプリデータに変換する。

最後に、ステップＳ２７１０において、アプリデータに変換された背景画像を電子ファイルとして格納する。

【００８２】

図２８に従来のページ管理の概念と本実施の形態のページ管理の概念との差を示す。従来のページ管理では、文字、写真、図形、表、及び背景というようなデータをページ毎に持っていた。これに対し、本実施の形態のページ管理では、背景部分が共通なので、背景オブジェクト（背景画像）を共通のオブジェクト（背景画像）として管理し、各ページのオブジェクト（画像）を読み出すときに、前記共通のオブジェクト（背景画像）として管理している背景データも併せて読み出すようにした。これにより、管理するデータ量を可及的に低減させながら、各ページのオブジェクト（画像）を確実に読み出すことができる。

【００８３】

（第２の実施の形態）

次に、本発明の第２の実施の形態について説明する。本実施の形態と、前述した第１の実施の形態では、背景画像（背景オブジェクト）を補間する際の処理が異なるだけであり、他の部分は、同一である。したがって、以下の説明において、第１の実施の形態と同一の部分についての詳細な説明を省略する。

【００８４】

複写機やスキャナにより原稿のスキャンを行う場合には、オートドキュメントフィーダー（以下、ＡＤＦと記す）が使用されることが多い。ＡＤＦでは、原稿を１枚ずつスキャナのプラテンガラス上に給送し、スキャンを行って画像を取り込む。この際、原稿の給送位置が僅かでもずれると、同じ背景画像を持つ原稿を使ったとしても、図２９に示す原稿２９０４、２９０５のように入力された画素の位置がずれる可能性が大きくなる。

そこで、本実施の形態では、図３０に示すように全ページに存在する背景以外のオブジェクト２９０２ａ～２９０２ｃの外周２９０１を求める。具体的に説明すると、文字オブジェクト２９０２ａの上側の輪郭線と、写真オブジェクト２９０２ｂの左側及び下側の輪郭線と、図形オブジェクト２９０２ｃの右側の輪郭線とにより形成される長方形の外周２９０１を求める。外周の求め方は本発明の本質に関係するところではないため、ここでは具体的な説明を行わない。図３０に示した例においては、背景以外のオブジェクト２９０２ａ～２９０２ｃの外周２９０１とは、背景以外のオブジェクト２９０２ａ～２９０２ｃの全てを囲む最小の四角形の外周である。

そして、ＡＤＦにおける原稿給送のための機械の精度により保証されているズレの最大値分だけ外周２９０１を広げる。そして、その広げた外周２０９３の分だけ、各ページの原稿画像をソフトウェア処理で消去する。

【００８５】

その後、１枚目の画像と、それ以外の画像とのズレ量（移動量）２９０６を検出する。ズレの方向と量は、動画の動きベクトルを検出する技術、すなわち公知の全探索法や勾配法を使用する。

【００８６】

ここで、図３１のフローチャートを参照しながら、ＡＤＦでの原稿のズレを考慮して背景画像を補間する際の手順の一例を、図３に示したベクトル化処理、アプリケーションデータへの変換処理、及び電子ファイル格納処理に適用した場合の動作について説明する。

先ず、ステップＳ３００１において、図２７のステップＳ２７０１と同様に、原稿画像を複数のオブジェクトに分離する。

次に、ステップＳ３００２において、共通の背景を使用することが指定されているかどうかを、入力装置１１３を用いたユーザの操作内容に基づいて判断する。この判断の結果、共通の背景を使用することが指定されていない場合には、後述するステップＳ３００３～Ｓ３０１０を省略してステップＳ３０１１に進む。一方、共通の背景を使用することが指定されている場合には、ステップＳ３００３に進み、図３０に示すようなオブジェクトの外周２９０１を検出する。

【００８７】

次に、ステップＳ３００４において、図２７のステップＳ２７０３と同様に、Ｎの値をインクリメントする。

次に、ステップＳ３００５において、ステップＳ３００３で検出されたオブジェクトの外周２９０１を画像から引いて、１枚目とＮ枚目の背景のズレ（移動量）２９０６を求める。

次に、ステップＳ３００６において、図２７のステップＳ２７０４と同様に、１枚目の背景画像で画素値が白の部分の座標（Ｘ，Ｙ）を検出する。

【００８８】

次に、ステップＳ３００７において、ステップＳ３００６で検出した座標（Ｘ，Ｙ）に、ステップＳ３００５で求められたズレ（移動量）２９０６を加え、Ｎ枚目の画像における座標（Ｘ，Ｙ）の画素が白でないことを確認し、白なら後述するステップＳ３００８を省略してステップＳ３００９に進む。一方、白でなければその座標（Ｘ，Ｙ）が背景領域と判断し、ステップＳ３００８に進む。そして、１枚目の背景画像におけるズレ（移動量）２９０６が加えられた座標の画素値を、Ｎ枚目の背景画像における座標（Ｘ，Ｙ）の画素値に置き換える。

【００８９】

そして、ステップＳ３００９において、図２７のステップＳ２７０７と同様に、Ｎ枚目の画像を用いた画素値の置き換えが終了したと判断され、さらに、ステップＳ３０１１において、図２７のステップＳ２７０８と同様に、全てのページを用いて画素値の置き換えが終了したと判断されると、ステップＳ３０１１に進む。このステップＳ３０１１では、複数枚の原稿（全てのページ）を使っても補間仕切れなかった１枚目の背景画像の補間処理を行う。

【００９０】

例えば、図３２のように、１枚目の背景画像で補間仕切れなかった部分を注目画素３２０１として、その周囲の画素を参照する。そして、周囲の画素の平均を注目画素３２０１の画素値とする。この他、背景画素の平均を注目画素３２０１の画素値としてもよい。このような処理によって、背景画像は全て埋まり、白く抜き出されたままになることを防止することができる。

【００９１】

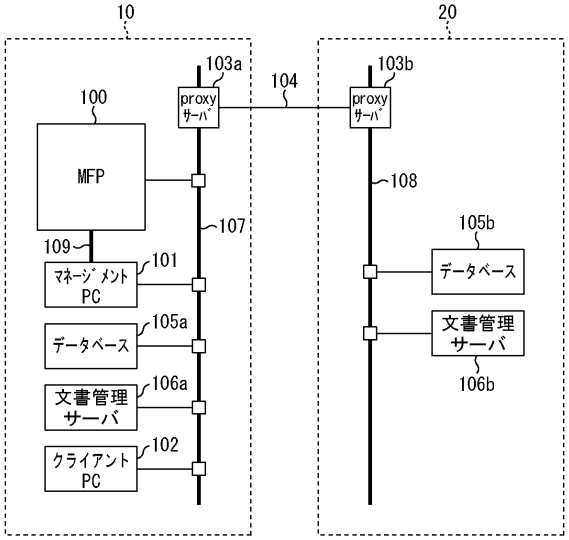
なお、本実施の形態では、求めた外周２９０１を、原稿給送のための機械の精度により保証されているズレの最大値分だけ広げ、その広げた外周２０９３の分だけ各ページの原稿画像を消去してズレ（移動量）２９０６を求めるようにしたが、原稿の複数ページ（好ましくは全ページ）について外周２９０１を求め、求めた外周２９０１の中から最も長い外周により囲まれる領域を求め、求めた領域を各ページの原稿画像から消去してズレ（移

動量）２９０６を求めるようにしてもよい。
【００９２】
　また、本実施の形態では、１枚目の背景画像の位置をずらして（座標を変更して）、１枚目の背景画像と、Ｎ枚目の背景画像との位置を合わせるようにしたが、Ｎ枚目の背景画像の位置を調整して、位置合わせを行うようにしてもよい。また、１枚目の背景画像と、Ｎ枚目の背景画像との双方の位置をずらして、位置合わせを行うようにしてもよい。
【００９３】
（第３の実施の形態）
　次に、本発明の第３の実施の形態について説明する。本実施の形態と、前述した第１及び第２の実施の形態では、背景画像（背景オブジェクト）を補間する際の処理が異なるだけであり、他の部分は、同一である。したがって、以下の説明において、第１及び第２の実施の形態と同一の部分についての詳細な説明を省略する。
　前述した第１及び第２の実施の形態では、背景画像からオブジェクトが抜けたところを、白画像として検索した。これに対し、本実施の形態では、オブジェクトが抜けた位置をビットプレーンで管理するようにする。例えば、画像と同じサイズの１ビットのビットプレーンの属性データを持ち、その属性データを０で初期化しておいてオブジェクトが抜けたところに１を立てることで、オブジェクトが抜けた位置を検出できるようにする。このような状態で、ビットプレーンを参照し、１が立てられている位置において、画素値の置き換えを行う。
【００９４】
　例えば、第１の実施の形態で説明した図２７のステップＳ２７０４～Ｓ２７０６又は図３１のステップＳ３００６～Ｓ３００８を、以上のような処理とすることにより、前述した第１及び第２の実施の形態と同様に、背景画像（オブジェクト）を補間することができる。
【００９５】
（本発明の他の実施の形態）
　上述した実施の形態の機能を実現するべく各種のデバイスを動作させるように、該各種デバイスと接続された装置あるいはシステム内のコンピュータに対し、前記実施の形態の機能を実現するためのソフトウェアのプログラムコードを供給し、そのシステムあるいは装置のコンピュータ（ＣＰＵあるいはＭＰＵ）に格納されたプログラムに従って前記各種デバイスを動作させることによって実施したものも、本発明の範疇に含まれる。
【００９６】
　また、この場合、前記ソフトウェアのプログラムコード自体が上述した実施の形態の機能を実現することになり、そのプログラムコード自体、及びそのプログラムコードをコンピュータに供給するための手段、例えば、かかるプログラムコードを格納した記録媒体は本発明を構成する。かかるプログラムコードを記憶する記録媒体としては、例えばフレキシブルディスク、ハードディスク、光ディスク、光磁気ディスク、ＣＤ－ＲＯＭ、磁気テープ、不揮発性のメモリカード、ＲＯＭ等を用いることができる。
【００９７】
　また、コンピュータが供給されたプログラムコードを実行することにより、上述の実施の形態の機能が実現されるだけでなく、そのプログラムコードがコンピュータにおいて稼働しているＯＳ（オペレーティングシステム）あるいは他のアプリケーションソフト等と共同して上述の実施の形態の機能が実現される場合にもかかるプログラムコードは本発明の実施の形態に含まれることは言うまでもない。
【００９８】
　さらに、供給されたプログラムコードがコンピュータの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格納された後、そのプログラムコードの指示に基づいてその機能拡張ボードや機能拡張ユニットに備わるＣＰＵ等が実際の処理の一部または全部を行い、その処理によって上述した実施の形態の機能が実現される場合にも本発明に含まれることは言うまでもない。

【図面の簡単な説明】
【００９９】
【図１】本発明の第１の実施の形態を示し、画像処理システムの構成の一例を示すブロック図である。
【図２】本発明の第１の実施の形態を示し、ＭＦＰの構成の一例を示すブロック図である。
【図３】本発明の第１の実施の形態を示し、画像処理全体の概要を説明するフローチャートである。
【図４】本発明の第１の実施の形態を示し、イメージ情報と、前記イメージ情報をブロックセレクション処理して得られるブロックの一例を示す図である。
【図５】本発明の第１の実施の形態を示し、ブロックセレクション処理で得られた各ブロックに対するブロック情報の一例を表形式で示す図である。
【図６】本発明の第１の実施の形態を示し、原稿画像中に付加された２次元バーコードを復号して、データ文字列を出力する手順の一例を説明するフローチャートである。
【図７】本発明の第１の実施の形態を示し、２次元バーコードが付加された原稿の一例を示す図である。
【図８】本発明の第１の実施の形態を示し、ポインター情報から電子ファイルを検索する手順の一例を説明するフローチャートである。
【図９】本発明の第１の実施の形態を示し、アクセス権の制限が有る電子ファイルをポインター情報から検索する手順の一例を説明するフローチャートである。
【図１０】本発明の第１の実施の形態を示し、ポインター情報を用いずに電子ファイルを検索する手順の一例を説明するフローチャートである。
【図１１】本発明の第１の実施の形態を示し、曲率が極大となる点を示す図である。
【図１２】本発明の第１の実施の形態を示し、２つの輪郭線をひとまとめにし、太さを持った線として表現する様子を示す図である。
【図１３】本発明の第１の実施の形態を示し、ベクトルデータを図形オブジェクト毎にグループ化するまでの手順の一例を説明するフローチャートである。
【図１４】本発明の第１の実施の形態を示し、図形要素を検出する手順の一例を説明するフローチャートである。
【図１５】本発明の第１の実施の形態を示し、ドキュメント・アナリシス・アウトプット・フォーマット（DAOF）データ構造の一例を示す図である。
【図１６】本発明の第１の実施の形態を示し、DAOFからアプリデータに変換する手順の概略の一例を説明するフローチャートである。
【図１７】本発明の第１の実施の形態を示し、文書構造ツリーを生成する手順の一例を説明するフローチャートである。
【図１８】本発明の第１の実施の形態を示し、文書構造ツリーの具体的な内容の一例を説明する図である。
【図１９】本発明の第１の実施の形態を示し、ポインター情報としてのデータ文字列を２次元バーコードにて符号化して画像中に付加する手順の一例を説明するフローチャートである。
【図２０】本発明の第１の実施の形態を示し、１枚目の原稿画像を示す図である。
【図２１】本発明の第１の実施の形態を示し、図２０に示した原稿画像から背景オブジェクト以外のオブジェクトを抽出した後に残る背景画像を示す図である。
【図２２】本発明の第１の実施の形態を示し、２枚目の原稿画像を示す図である。
【図２３】本発明の第１の実施の形態を示し、図２２に示した原稿画像から背景オブジェクト以外のオブジェクトを抽出した後に残る背景画像を示す図である。
【図２４】本発明の第１の実施の形態を示し、図２１に示した背景画像と、図２２に示した背景画像とを重ねた画像を示す図である。
【図２５】本発明の第１の実施の形態を示し、図２１に示した背景画像の白く抜けた部分を、図２３に示した背景画像で補間した画像を示す図である。

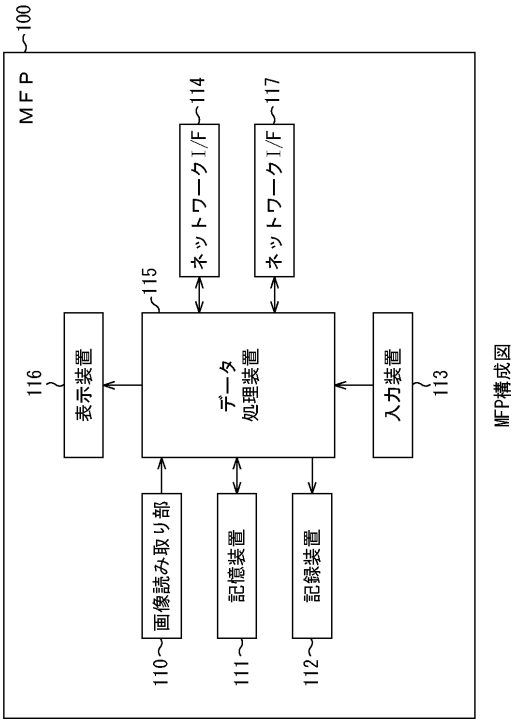
【図２６】本発明の第１の実施の形態を示し、図２１に示した背景画像の白く抜けた部分を、複数枚の背景画像で完全に補間した画像を示す図である。
【図２７】本発明の第１の実施の形態を示し、背景画像を補間する際の手順の一例を説明するフローチャートである。
【図２８】本発明の第１の実施の形態を示し、従来のページ管理の概念と、本実施の形態のページ管理とを示す図である。
【図２９】本発明の第２の実施の形態を示し、位置がずれて入力された２枚の原稿画像を示す図である。
【図３０】本発明の第２の実施の形態を示し、背景以外のオブジェクトの外周が求められた原稿を示す図である。
【図３１】本発明の第２の実施の形態を示し、原稿のズレを考慮して背景画像を補間する際の手順の一例を説明するフローチャートである。
【図３２】本発明の第２の実施の形態を示し、背景画像で補間仕切れなかった注目画素と、その周囲の画素を示す図である。
【符号の説明】
【０１００】
１０、２０　オフィス
１００　ＭＦＰ
１０１　マネージメントＰＣ
１０２　クライアントＰＣ
１０３　プロキシサーバ
１０４　インターネット
１０５　データベース
１０６　文書管理サーバ
１０７～１０９　ＬＡＮ
１１０　画像読み取り部
１１１　記憶装置
１１２　記録装置
１１３　入力装置
１１４、１１７　ネットワークＩ／Ｆ
１１５　データ処理装置
１１６　表示装置

【図１】



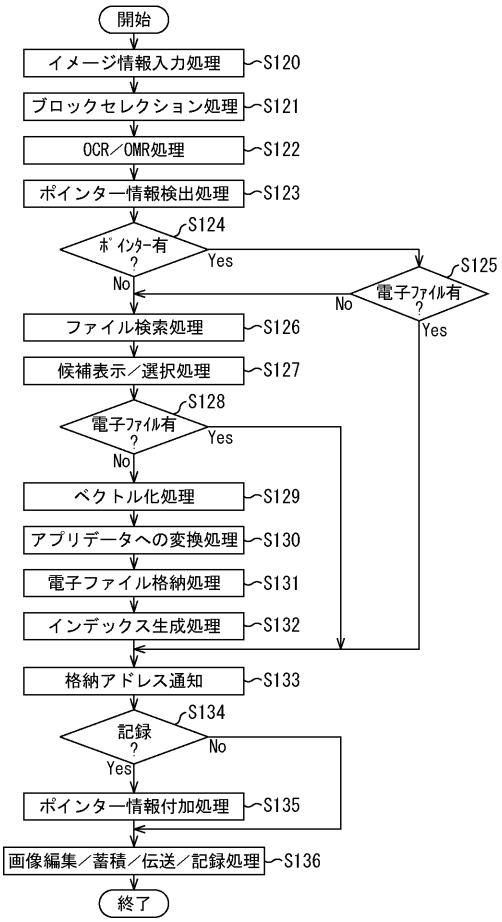
システム構成図

【図２】

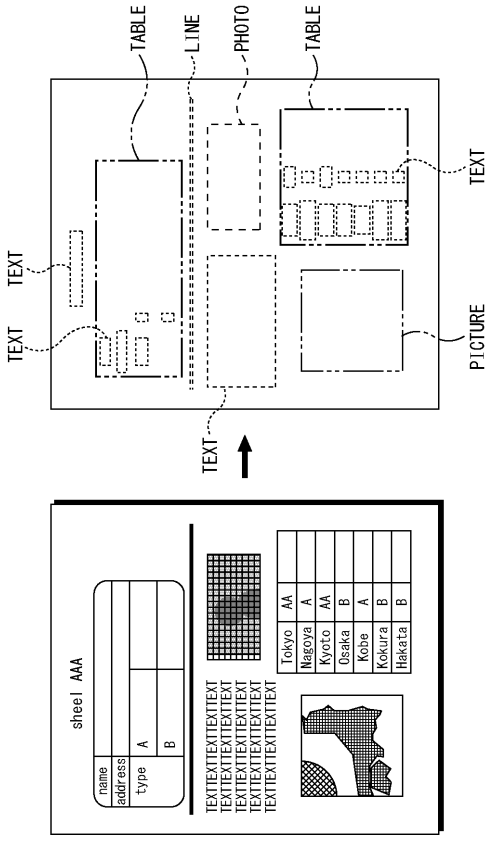


MFP構成図

【図３】



【図４】



【図５】

51 ブロック情報

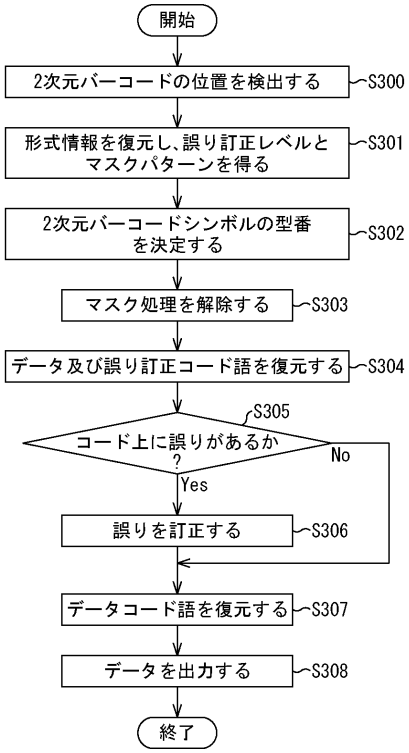
	属性	座標 X	座標 Y	幅 W	高さ H	OCR 情報
ブロック1	1	X1	Y1	W1	H1	有
ブロック2	3	X2	Y2	W2	H2	有
ブロック3	2	X3	Y3	W3	H3	無
ブロック4	1	X4	Y4	W4	H4	有
ブロック5	3	X5	Y5	W5	H5	有
ブロック6	5	X6	Y6	W6	H6	無

＊ 属性1：text 2：picture 3：table 4：line 5：photo

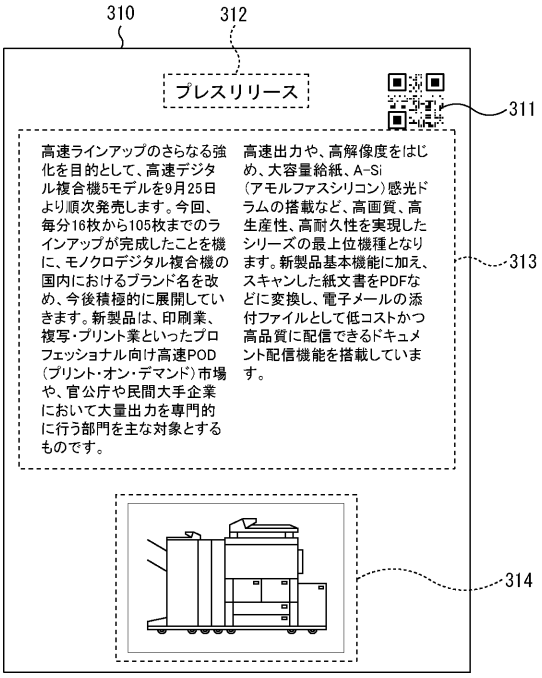
52 入力ファイル情報

ブロック総数	N（=6）

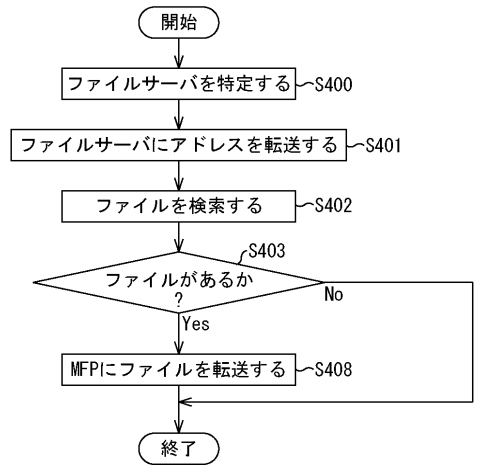
【図６】



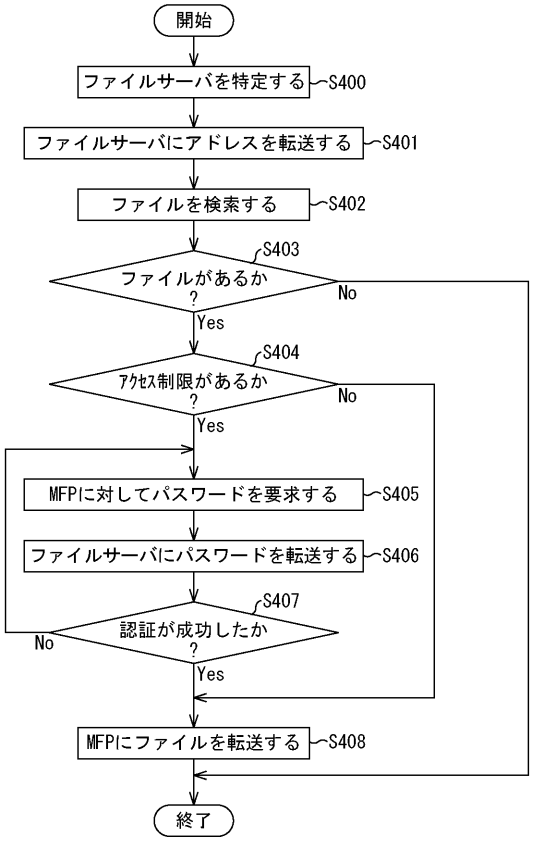
【図７】



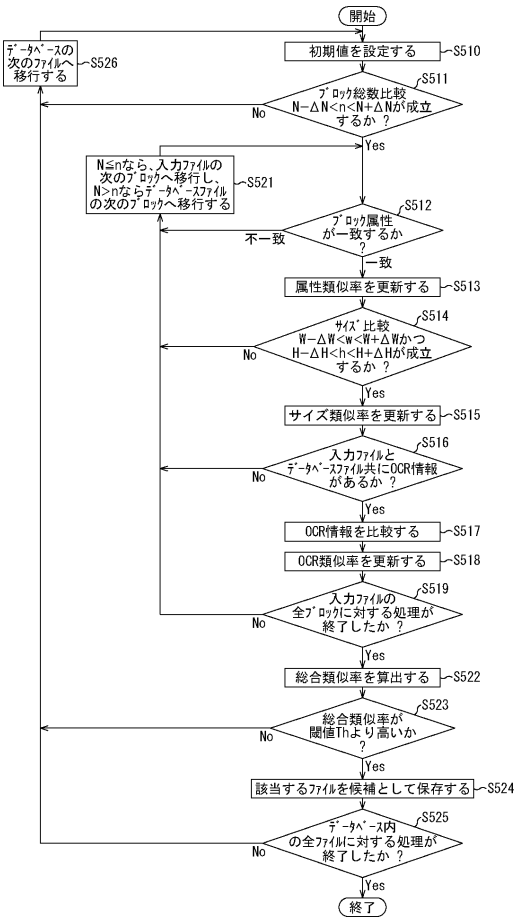
【図８】



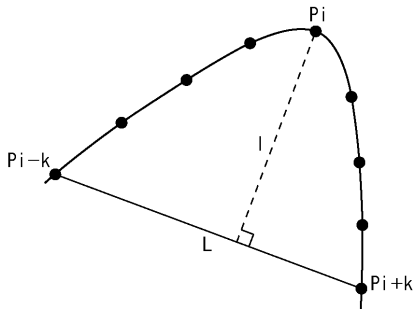
【図９】



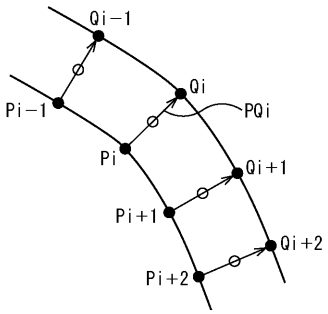
【図１０】



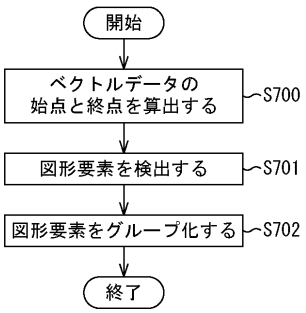
【図１１】



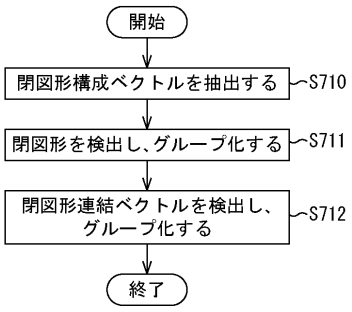
【図１２】



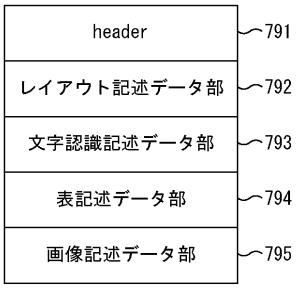
【図１３】



【図１４】

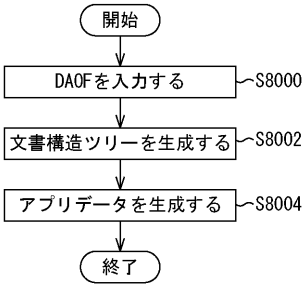


【図１５】



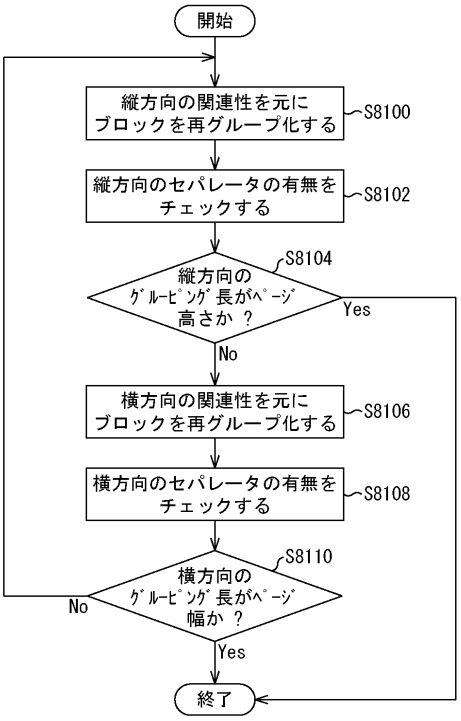
DAOF

【図１６】



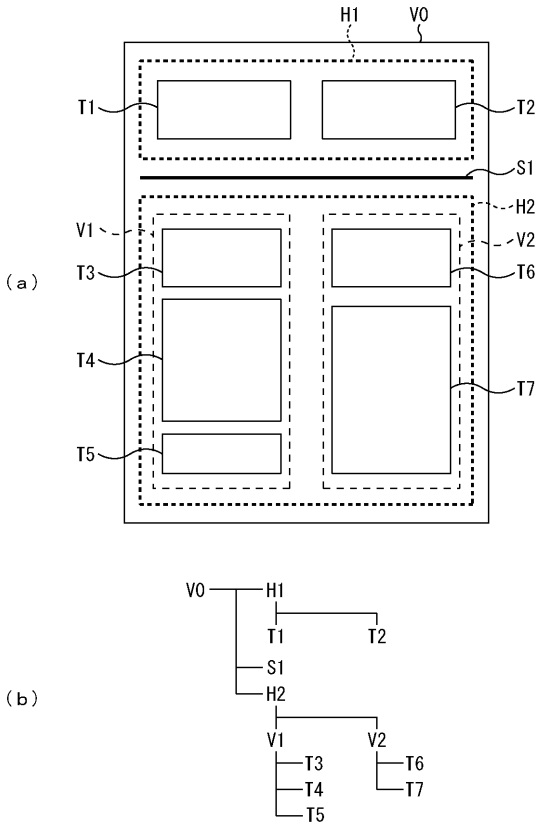
アプリデータ変換

【図１７】



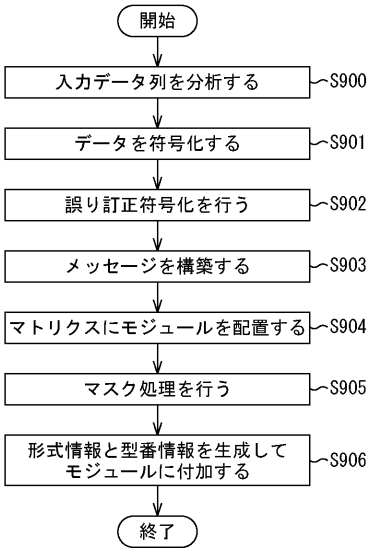
文書構造ツリー生成

【図１８】

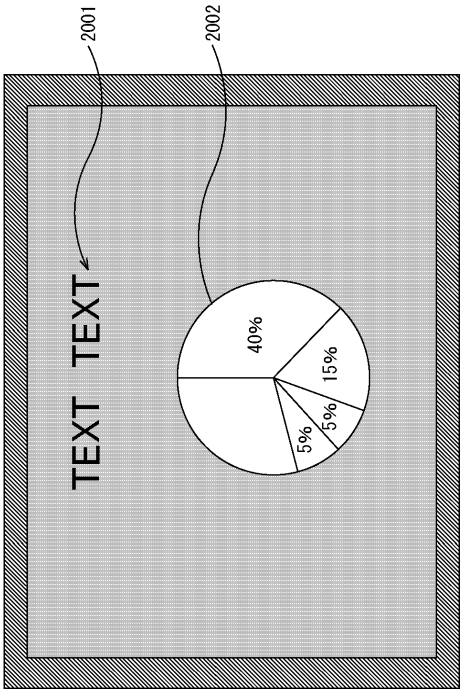


文書構造ツリー説明図

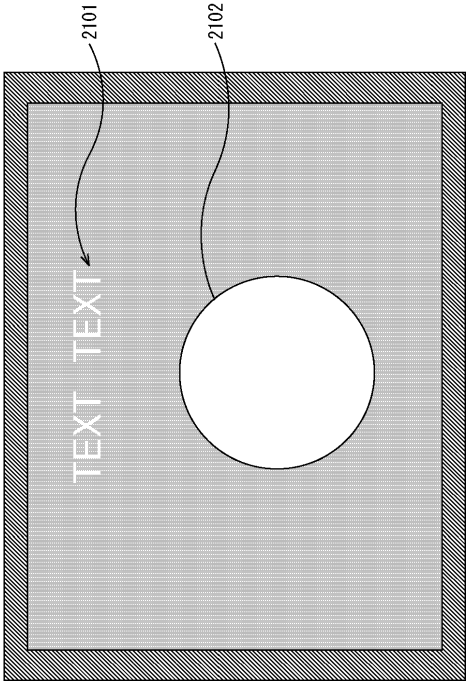
【図１９】



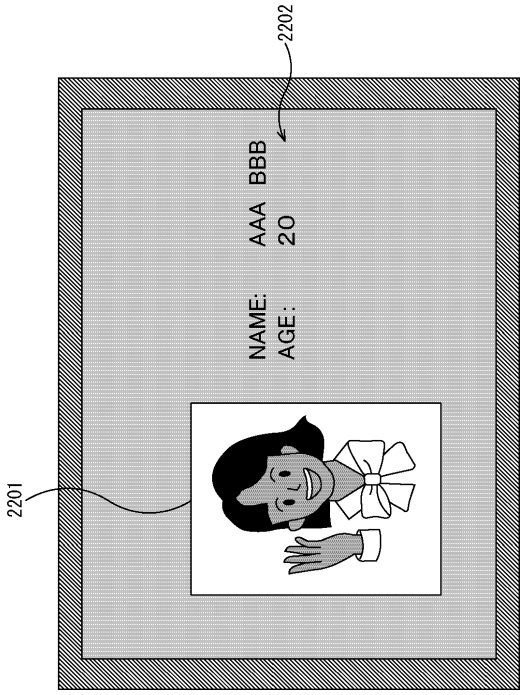
【図２０】



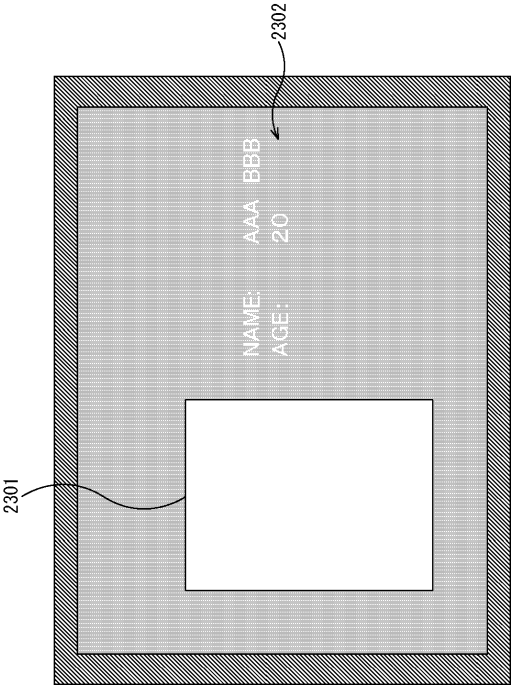
【図２１】



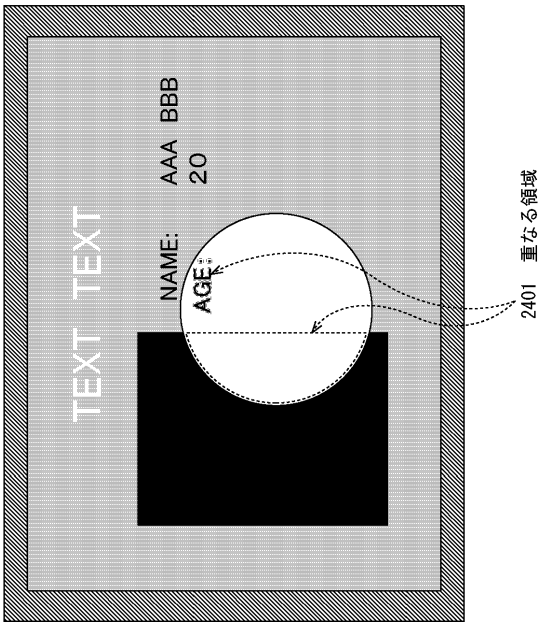
【図２２】



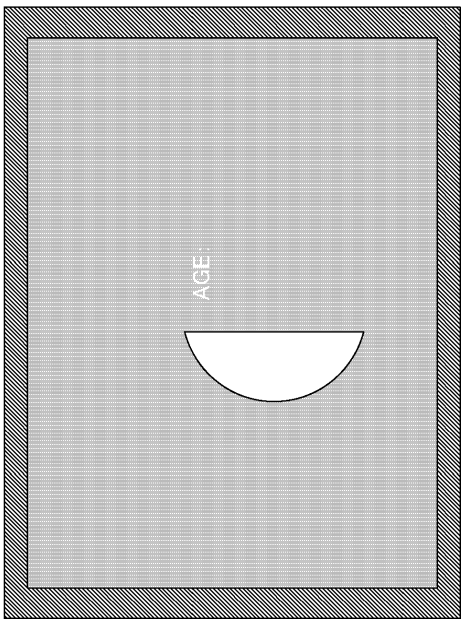
【図２３】



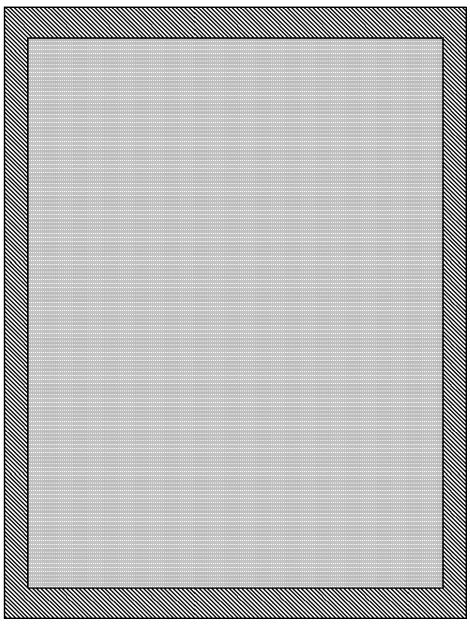
【図２４】



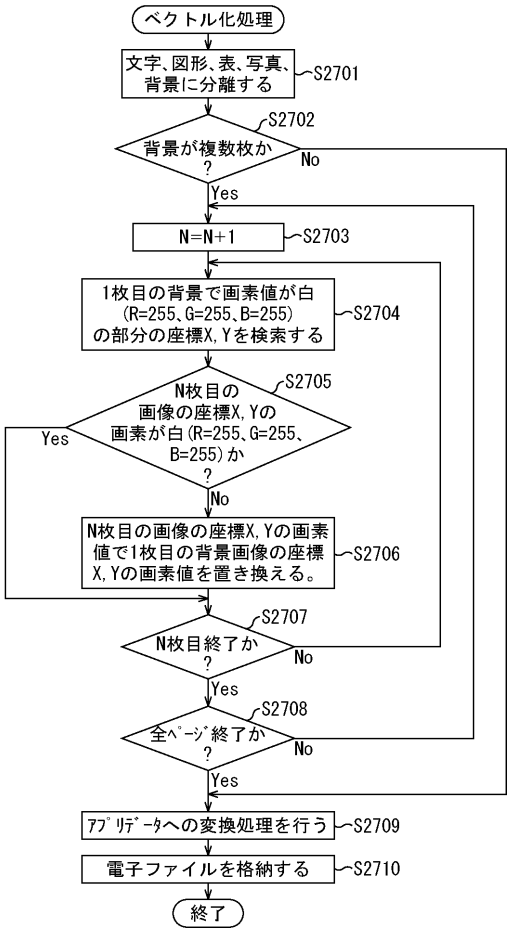
【図２５】



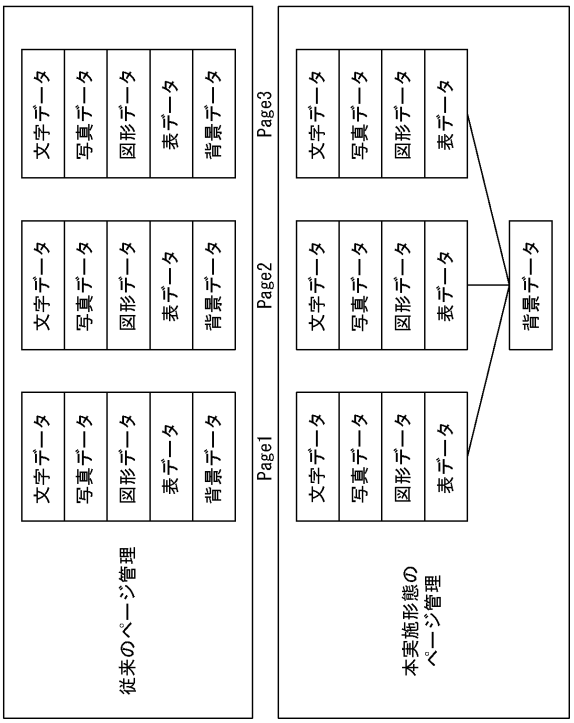
【図２６】



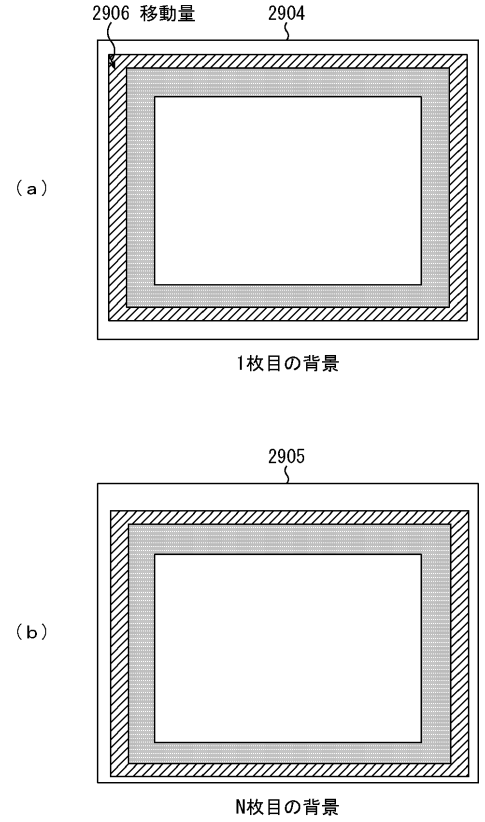
【図２７】



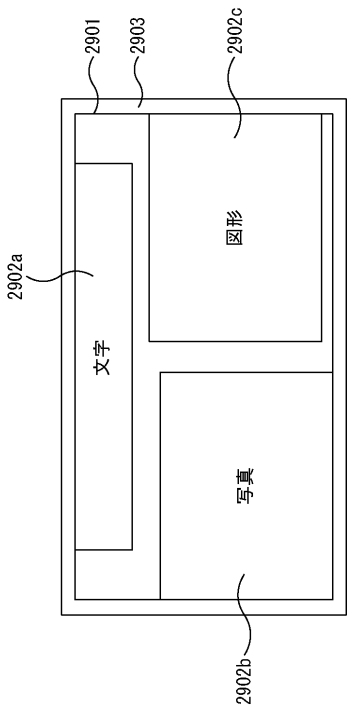
【図２８】



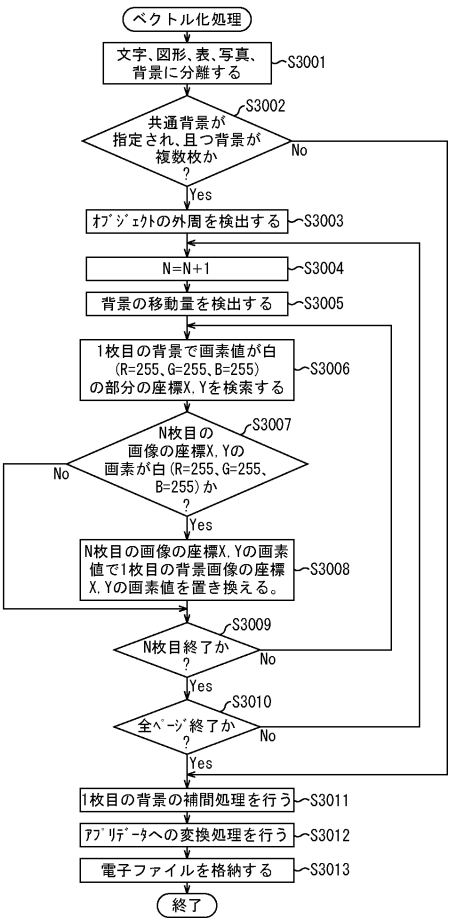
【図２９】



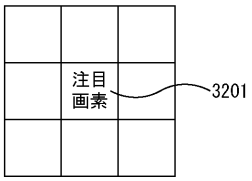
【図３０】



【図３１】



【図３２】



フロントページの続き

(51)Int.Cl.[7]	FI		テーマコード（参考）
Ｇ０６Ｔ 11/80	Ｇ０６Ｔ 11/80	Ａ	
Ｈ０４Ｎ 1/387	Ｈ０４Ｎ 1/387		

(72)発明者 辻 博之
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 加藤 進一
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 木虎 正和
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 関口 賢三
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 吉田 廣義
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

Ｆターム(参考) 5B021 AA19 LG07 LG08
5B050 AA10 BA06 BA10 BA16 BA18 BA20 CA08 EA03 EA12 EA19
FA02 GA08
5B057 AA20 BA11 CA12 CB12 CF10 CH01 DB02 DC03 DC33
5C076 AA02 AA13 AA36 BA02 BA03 BA04 BA06
5L096 AA06 BA07 DA01 EA33 GA08 GA10 MA03