

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4817549号

(P4817549)

(45) 発行日 平成23年11月16日(2011.11.16)

(24) 登録日 平成23年9月9日(2011.9.9)

(51) Int.Cl. F I
G06F 3/12 (2006.01) G O 6 F 3/12 B
G06T 11/40 (2006.01) G O 6 T 11/40 2 O O A

請求項の数 27 (全 31 頁)

(21) 出願番号	特願2001-229426 (P2001-229426)	(73) 特許権者	000001007
(22) 出願日	平成13年7月30日(2001.7.30)		キヤノン株式会社
(65) 公開番号	特開2002-169668 (P2002-169668A)		東京都大田区下丸子3丁目30番2号
(43) 公開日	平成14年6月14日(2002.6.14)	(74) 代理人	100076428
審査請求日	平成20年7月22日(2008.7.22)		弁理士 大塚 康德
(31) 優先権主張番号	PQ9058	(74) 代理人	100112508
(32) 優先日	平成12年7月28日(2000.7.28)		弁理士 高柳 司郎
(33) 優先権主張国	オーストラリア(AU)	(74) 代理人	100115071
			弁理士 大塚 康弘
		(74) 代理人	100116894
			弁理士 木村 秀二

最終頁に続く

(54) 【発明の名称】 ページ記述のレンダリング方法及びシステム

(57) 【特許請求の範囲】

【請求項1】

レンダリングシステム上でページ記述をレンダリングする方法であって、前記ページ記述は1つまたは複数のオブジェクトを含み、

(a) 前記オブジェクトを下から上へのz順に順次受信するステップと、

(b) 1つまたは複数のレンダ命令セットを生成するステップとを含み、前記生成するステップは、受信された各オブジェクトについて、

(ba) 前記受信した現在のオブジェクトに対応するレンダ命令を、既に受信したオブジェクトのレンダリング待ちとなっているレンダ命令を格納している現レンダ命令セットに追加する際に、前記レンダリングシステムのリソースを越えるかどうかを決定するサブステップと、

(bb) 前記リソースを越える場合、

(bba) 前記リソースに空きを発生させるため、前記現レンダ命令セットにあるレンダ命令を、レンダリングエンジンに供給し、背景イメージとして描画させるサブステップと、

(bbb) 前記描画された前記背景イメージへの参照する命令を、新しいレンダ命令セットに追加するサブステップと、

(bbc) 前記受信した前記現在のオブジェクトに対応するレンダ命令を、前記新しいレンダ命令セットに追加するサブステップ

を実行するサブステップと、

10

20

(bc)前記リソースを越えない場合、
(bca)前記受信された現在のオブジェクトに対応するレンダ命令を、前記現レンダ命令セットに追加するサブステップ
を実行するサブステップとを有し、
(c)前記1つまたは複数のレンダ命令セットを、前記レンダリングエンジンを用いて、
画素データとしてレンダリングするステップ
を備えることを特徴とする方法。

【請求項2】

前記リソースを越える場合、前記現レンダ命令セットをキューイングし、前記現レンダ命令セットの前記レンダリングの前記画素データを、メモリにおいて前記背景イメージのために予約された空間に格納するように要求するサブステップをさらに実行することを特徴とする請求項1に記載の方法。

10

【請求項3】

前記リソースを越える場合、
前記現レンダ命令セットをキューイングし、前記現レンダ命令セットの前記レンダリングの前記画素データを、メモリにおいて前記背景イメージのために予約された空間に格納するように要求するサブステップと、
メモリにおける前記空間を、先に割り振られていなかった場合、前記背景イメージのために割り振るサブステップと
をさらに実行することを特徴とする請求項1に記載の方法。

20

【請求項4】

前記リソースを越えない場合、かつ、前記現在の受信されたオブジェクトが前記 z 順における最後のオブジェクトであった場合、
前記現レンダ命令セットをキューイングするサブステップをさらに実行することを特徴とする請求項1に記載の方法。

【請求項5】

前記生成するステップは、
前記現在の受信されたオブジェクトを、対応するレンダ命令に変換するサブステップをさらに含むことを特徴とする請求項1に記載の方法。

【請求項6】

前記レンダリングするステップは、
(ca)前記現レンダ命令セットの実行によって出力される画素データが、メモリにおいて前記背景イメージのために予約された空間に格納されるかどうかを判別するサブステップと、
(cb)前記現レンダ命令セットの前記画素データが、前記メモリにおいて前記背景イメージのために予約された前記空間に格納される場合、
(cba)前記現レンダ命令セットを前記背景イメージとしてレンダリングするサブステップと、
(cbb)前記背景イメージを、前記メモリにおいて予約された前記空間に格納するサブステップとを実行するサブステップと、
(cc)前記現レンダ命令セットの前記画素データが、前記メモリにおいて前記背景イメージのために予約された前記空間に格納されていない場合、
(cd)現レンダ命令セットを出力デバイスへレンダリングするサブステップを実行するサブステップと
を備えることを特徴とする請求項1に記載の方法。

30

40

【請求項7】

前記出力デバイスはプリンタであることを特徴とする請求項6に記載の方法。

【請求項8】

前記出力デバイスは表示画面であることを特徴とする請求項6に記載の方法。

【請求項9】

50

前記現レンダ命令セットの実行によって出力される前記画素データが、前記背景イメージのために予約された前記メモリの前記空間に格納される場合、

前記背景イメージを、前記メモリにおいて予約された前記空間に格納する前に圧縮するサブステップをさらに実行することを特徴とする請求項 6 に記載の方法。

【請求項 10】

前記 1 つまたは複数のオブジェクトのうち 1 つまたは複数の、複数のプリミティブ・オブジェクトのグループを含むことを特徴とする請求項 1 に記載の方法。

【請求項 11】

ページ記述をレンダリングする方法であって、前記ページ記述は 1 つまたは複数のオブジェクトを含み、

(a) レンダリング命令生成部が、前記オブジェクトを下から上への z 順に順次受信するステップを含み、前記受信された各オブジェクトについて、

(aa) 前記レンダリング命令生成部が、前記受信されたオブジェクトを、現在のレンダリストに追加する際に、リソースを越えるかどうかを検査し、越える場合、前記現在のレンダリストをレンダリングエンジンに供給してフレーム・ストア上にレンダリングし、次いで前記現在のレンダリストを空に設定し、前記レンダリングエンジンを用いてレンダリングして得られたイメージの参照をレンダリングリストに追加するサブステップと、

(ab) 前記レンダリング命令生成部が、前記受信されたオブジェクトを、現在のレンダリストに追加する際に、リソースを越えるかどうかを検査し、越えない場合、前記受信されたオブジェクトを前記現在のレンダリストに追加し、前記受信されたオブジェクトが前記 z 順における最後のオブジェクトであった場合、

(aba) 前記フレーム・ストア上のレンダリング済みが存在する場合、前記現在のレンダリストを前記レンダリングエンジンに供給して前記フレーム・ストアと組み合わせるレンダリングして、レンダリングされたページを形成し、前記フレーム・ストア上にレンダリング済みがない場合、前記現在のレンダリストを前記レンダリングエンジンに供給してレンダリングして、前記レンダリングされたページを形成するサブステップと
を実行することを特徴とする方法。

【請求項 12】

前記 1 つまたは複数のオブジェクトのうち 1 つまたは複数の、複数のプリミティブ・オブジェクトのグループを含むことを特徴とする請求項 11 に記載の方法。

【請求項 13】

前記空の現在のレンダリストが基本レンダ命令を含むことを特徴とする請求項 11 に記載の方法。

【請求項 14】

ページ記述をレンダリングするためのシステムであって、前記ページ記述は 1 つまたは複数のオブジェクトを含み、

前記オブジェクトを下から上への z 順に順次受信する手段と、

1 つまたは複数のレンダ命令セットを生成する生成手段とを備え、当該生成手段は、

(i) 前記受信された各オブジェクトについて、前記受信された現在のオブジェクトの対応するレンダ命令を、既に受信したオブジェクトのレンダリング待ちになっているレンダ命令を格納している現レンダ命令セットに追加する際に、前記レンダリングするためのシステムのリソースを越えるかどうかを判別する手段と、

(ii) 前記リソースを越える場合、前記リソースに空きを発生させるため、前記現レンダ命令セットにあるレンダ命令を、レンダリングエンジンに供給し、背景イメージとして描画させる手段と、

(iii) 前記リソースを越える場合、前記描画された前記背景イメージへの参照する命令を、新しいレンダ命令セットに追加する手段と、

(iv) 前記リソースを越える場合、前記受信した前記現在のオブジェクトに対応するレンダ命令を、前記新しいレンダ命令セットに追加する手段と、

(v) 前記リソースを越えない場合、前記受信された現在のオブジェクトに対応するレン

10

20

30

40

50

ダ命令を、前記現レンダ命令セットに追加する手段と

(vi)前記 1 つまたは複数のレンダ命令セットを、前記レンダリングエンジンを用いて、画素データとしてレンダリングする手段と
を備えることを特徴とするシステム。

【請求項 15】

前記生成手段は、

前記リソースを越える場合、前記現レンダ命令セットをキューイングし、前記現レンダ命令セットの前記レンダリングの前記画素データを、メモリにおいて前記背景イメージのために予約された空間に格納するように要求する手段をさらに備えることを特徴とする請求項 14 に記載のシステム。

10

【請求項 16】

前記生成手段は、

前記リソースを越える場合、前記現レンダ命令セットをキューイングし、前記現レンダ命令セットの前記レンダリングの前記画素データを、メモリにおいて前記背景イメージのために予約された空間に格納するように要求する手段と、

メモリにおける前記空間を、前記リソースを越える場合、先に割り振られていなかった場合、前記背景イメージのために割り振る手段と

を更に備えることを特徴とする請求項 14 に記載のシステム。

【請求項 17】

前記生成手段は、

前記リソースを越えない場合、かつ、前記現在の受信されたオブジェクトが前記 z 順における最後のオブジェクトであった場合、前記現レンダ命令セットをキューイングする手段

20

を更に備えることを特徴とする請求項 14 に記載のシステム。

【請求項 18】

前記生成手段は、

前記現在の受信されたオブジェクトを、対応するレンダ命令に変換する手段をさらに備えることを特徴とする請求項 14 に記載のシステム。

【請求項 19】

前記レンダリングする手段は、

前記現レンダ命令セットの実行によって出力される画素データが、メモリにおいて前記背景イメージのために予約された空間に格納されるかどうかを判別する手段と、

前記現レンダ命令セットの実行によって出力される前記画素データが、前記メモリにおいて前記背景イメージのために予約された前記空間に格納される場合、前記現レンダ命令セットを前記背景イメージとしてレンダリングする手段と、

前記レンダリングされた背景イメージを、前記メモリにおいて予約された前記空間に格納する手段と、

30

前記現レンダ命令セットの実行によって出力される前記画素データが、前記メモリにおいて前記背景イメージのために予約された前記空間に格納されない場合、前記現レンダ命令セットを出力デバイスへレンダリングする手段と

40

を備えることを特徴とする請求項 14 に記載のシステム。

【請求項 20】

前記出力デバイスはプリンタであることを特徴とする請求項 19 に記載のシステム。

【請求項 21】

前記出力デバイスは表示画面であることを特徴とする請求項 19 に記載のシステム。

【請求項 22】

前記レンダリングする手段は、

前記現レンダ命令セットの前記画素データが、前記メモリにおいて前記背景イメージのために予約された前記空間に格納される場合、前記背景イメージを、前記メモリにおいて予約された前記空間に格納する前に圧縮する手段

50

をさらに備えることを特徴とする請求項 19 に記載のシステム。

【請求項 23】

ページ記述をレンダリングするためのシステムであって、前記ページ記述は 1 つまたは複数のオブジェクトを含み、

前記オブジェクトを下から上への z 順に順次受信する手段と、

受信された各オブジェクトについて、前記受信されたオブジェクトを、現在のレンダリストに追加する際に、リソースを越えるかどうかを検査し、越える場合、前記現在のレンダリストをフレーム・ストア上にレンダリングし、次いで前記現在のレンダリストを空に設定し、レンダリングして得られたイメージの参照をレンダリストに追加する手段と、

前記受信されたオブジェクトを、現在のレンダリストに追加する際に、リソースを越えるかどうかを検査し、越えない場合、前記受信されたオブジェクトを前記現在のレンダリストに追加する手段と、

前記フレーム・ストア上のレンダリング済みが存在する場合、前記現在のレンダリストを前記フレーム・ストアと組み合わせてレンダリングして、レンダリングされたページを形成し、前記フレーム・ストア上にレンダリング済みがない場合、前記現在のレンダリストをレンダリングして、前記レンダリングされたページを形成する手段と

を備えることを特徴とするシステム。

【請求項 24】

前記 1 つまたは複数のオブジェクトのうち 1 つまたは複数が、複数のプリミティブ・オブジェクトのグループを含むことを特徴とする請求項 23 に記載のシステム。

【請求項 25】

前記空の現在のレンダリストが基本レンダ命令を含むことを特徴とする請求項 23 に記載のシステム。

【請求項 26】

ページ記述をレンダリングするためのコンピュータプログラムであって、前記ページ記述は 1 つまたは複数のオブジェクトを含み、

コンピュータに、

(i) 前記オブジェクトを下から上への z 順に順次受信する手順と、

(ii) 1 つまたは複数のレンダ命令セットを生成する手順とを含み、前記生成する手順は

、
(ii-i) 前記受信した現在のオブジェクトに対応するレンダ命令を、既に受信したオブジェクトのレンダリング待ちとなっているレンダ命令を格納している現レンダ命令セットに追加する際に、前記レンダリングシステムのリソースを越えるかどうかを決定する手順と、

(ii-ii) 前記リソースを越える場合、前記リソースに空気を発生させるため、前記現レンダ命令セットにあるレンダ命令を、レンダリングエンジンに供給し、背景イメージとして描画させる手順と、

(ii-iii) 前記リソースを越える場合、前記描画された前記背景イメージへの参照する命令を、新しいレンダ命令セットに追加する手順と、

(ii-iv) 前記リソースを越える場合、前記受信した現在のオブジェクトに対応するレンダ命令を、前記新しいレンダ命令セットに追加する手順と、

(ii-v) 前記リソースを越えない場合、前記受信された現在のオブジェクトに対応するレンダ命令を、前記現レンダ命令セットに追加する手順とを含み、さらに、

(iii) 前記 1 つまたは複数のレンダリング命令セットを前記レンダリングエンジンを用いて画素データとしてレンダリングする手順

を実行させることを特徴とするコンピュータプログラム。

【請求項 27】

ページ記述をレンダリングするためのコンピュータプログラムであって、前記ページ記述は 1 つまたは複数のオブジェクトを含み、

コンピュータに、

前記オブジェクトを下から上への z 順に順次受信する手順と、

受信された各オブジェクトについて、前記受信されたオブジェクトを、現在のレンダリストに追加する際に、リソースを越えるかどうかを検査し、越える場合、前記現在のレンダリストをフレーム・ストア上にレンダリングし、次いで前記現在のレンダリストを空に設定し、レンダリングして得られたイメージの参照をレンダリングリストに追加する手順と、

前記受信された各オブジェクトについて、前記受信されたオブジェクトを、現在のレンダリストに追加する際に、リソースを越えるかどうかを検査し、越えない場合、受信されたオブジェクトを前記現在のレンダリストに追加する手順と、

前記受信されたオブジェクトが前記 z 順における最後のオブジェクトであり、前記フレーム・ストア上にレンダリング済みが存在する場合、前記現在のレンダリストを前記フレーム・ストアと組み合わせてレンダリングして、レンダリングされたページを形成する手順と、

前記受信されたオブジェクトが前記 z 順における前記最後のオブジェクトであり、前記フレーム・ストア上にレンダリング済みが無かった場合、前記現在のレンダリストをレンダリングして、前記レンダリングされたページを形成する手順

を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】

【0001】

【発明の属する技術分野】

本発明は、ページ記述をレンダリングするための方法およびシステムに関する。本発明はまた、ページ記述をレンダリングするためのコンピュータ・プログラムにも関する。

【0002】

【従来の技術】

比較的近年、印刷ページをプリンタ、表示画面および/または他のロケーション(たとえば、ファイル)にレンダリングするためのグラフィックス・パッケージが存在している。これらのいくつかのグラフィックス・パッケージは、典型的には、高レベルのページ記述言語を、印刷ページのレイアウトにおいて支援するために利用するものである。これらのグラフィックス・パッケージは典型的にはレンダリング・システムを含み、これがページ・レイアウトの高レベルのページ記述を、印刷及び/又は表示されるラスタ・データとして使用することができる画素データに変換する。

【0003】

1つのこのようなレンダリング・システムは、ページまたは画面の画素に基づいたイメージを保持するためのフルページ・フレーム・ストアを含む。フルページ・フレーム・ストアはバッファであり、ページ全体についてプリンタに送信する必要があるあらゆる画素のための空間を有する。これらのシステムでは、高レベルのページ記述において受信されるあらゆるオブジェクトを、直ちにレンダリングすることができる。このシステムの主要な問題は、フレーム・ストアのために必要となるメモリが非常に大きいことであり、特に、各画素を連続色調にする必要があり、これが関連付けられた透過値を有する可能性があるときにそうである。

【0004】

【発明が解決しようとする課題】

オブジェクトを z 順序で順次に適用するというより、むしろレンダリストに適用されたときに、速度においてもメモリ使用においてもより効率的に実行する既存のレンダリング・システムがある。このようなシステムは、レンダリストに必要なとされる記憶装置が使用可能なリソース以内であるときに大変有利であるが、さらにリソースが必要となる場合、中断してしまう。たとえば、オブジェクト・レンダリストのサイズに制限がないのに対して、フレーム・ストアのサイズは大きい制限されており、厳密に印刷ページの解像度およびサイズによって決まる。このようなシステムを使用するとき、これらは、使用可能なリソースを越えるレンダリストと共に提示されたときに障害を起こすか、あるいは、フレーム

10

20

30

40

50

・ストアを使用したオブジェクト順次即時レンダリング方法へと劣化する。

【0005】

本発明の一目的は、既存の構成の1つまたは複数の欠点を実質的に克服、あるいは少なくとも改良することである。

【0006】

【課題を解決するための手段】

本発明の第1の態様によれば、レンダリングシステム上でページ記述をレンダリングする方法を提供する。ここで、ページ記述は1つまたは複数のオブジェクトを含むものであり、この方法は、オブジェクトを下から上への z 順に順次受信するステップと、1つまたは複数のレンダ命令セットを生成するステップとを含み、この生成するステップは、受信された各オブジェクトについて、現在の受信されたオブジェクトが対応するレンダ命令を現レンダ命令セットに追加することにより、レンダリング・システムのリソースを越えるかどうかを決定するサブステップと、リソースを越える場合、新しいレンダ命令セットを作成するサブステップと、レンダ命令を新しいレンダ命令セットに追加して現レンダ命令セットを背景イメージとして描画するサブステップと、新しいレンダ命令セットに、現在の受信されたオブジェクトが対応するレンダ命令を追加するサブステップとを実行するサブステップと、リソースを越えない場合、現在の受信されたオブジェクトが対応するレンダ命令を現レンダ命令セットに追加するサブステップを実行するサブステップとを実行し、さらに、1つまたは複数のレンダ命令セットを画素データとしてレンダリングするステップを含む。

【0007】

本発明の第2の態様によれば、ページ記述をレンダリングする方法を提供する。ここで、ページ記述は1つまたは複数のオブジェクトを含むものであり、この方法は、オブジェクトを下から上への z 順に順次受信するステップを含み、この方法は、受信された各オブジェクトについて、受信されたオブジェクトが現在のレンダリストに追加された場合にリソースを越えるかどうかを検査し、そうであった場合、現在のレンダリストをフレーム・ストア上にレンダリングし、次いで現在のレンダリストを空に設定するサブステップと、受信されたオブジェクトを現在のレンダリストに追加し、受信されたオブジェクトが z 順における最後のオブジェクトであった場合、フレーム・ストア上にレンダリングされた場合、現在のレンダリストをフレーム・ストアと組み合わせてレンダリングして、レンダリングされたページ記述を形成し、そうでない場合、現在のレンダリストをレンダリングして、レンダリングされたページ記述を形成するサブステップとを実行する。

【0008】

本発明の第3の態様によれば、ページ記述をレンダリングするためのシステムを提供する。ここで、ページ記述は1つまたは複数のオブジェクトを含むものであり、このシステムは、オブジェクトを下から上への z 順に順次に受信する手段と、1つまたは複数のレンダ命令セットを生成する手段とを含み、この生成する手段は、受信された各オブジェクトについて、現在の受信されたオブジェクトが対応するレンダ命令を、現レンダ命令セットに追加することにより、レンダリングするためのシステムのリソースを越えるかどうかを決定する手段と、リソースを越える場合、新しいレンダ命令セットを作成する手段と、リソースを越える場合、レンダ命令を新しいレンダ命令セットに追加して現レンダ命令セットを背景イメージとして描画する手段と、リソースを越える場合、新しいレンダ命令セットに、現在の受信されたオブジェクトが対応するレンダ命令を追加する手段と、リソースを越えない場合、現在の受信されたオブジェクトの対応するレンダ命令を現レンダ命令セットに追加する手段と、1つまたは複数のレンダ命令セットを画素データとしてレンダリングする手段とを含む。

【0009】

本発明の第4の態様によれば、ページ記述をレンダリングするためのシステムを提供する。ここで、ページ記述は1つまたは複数のオブジェクトを含むものであり、このシステムは、オブジェクトを下から上への z 順に順次受信する手段と、受信された各オブジェク

トについて、受信されたオブジェクトが現在のレンダリストに追加された場合にリソースを越えるかどうかを検査し、そうであった場合、現在のレンダリストをフレーム・ストア上にレンダリングし、次いで現在のレンダリストを空に設定する手段と、受信されたオブジェクトを現在のレンダリストに追加する手段と、受信されたオブジェクトがz順における最後のオブジェクトであり、フレーム・ストア上にレンダリングされた場合、現在のレンダリストをフレーム・ストアと組み合わせてレンダリングして、レンダリングされたページ記述を形成する手段と、受信されたオブジェクトがz順における最後のオブジェクトであり、フレーム・ストア上にレンダリングされなかった場合、現在のレンダリストをレンダリングして、レンダリングされたページ記述を形成する手段とを含む。

【0010】

本発明の第5の態様によれば、ページ記述をレンダリングするためのコンピュータ・プログラムを提供する。ここでページ記述は1つまたは複数のオブジェクトを含むものである。そして、このコンピュータ・プログラムは、コンピュータに、オブジェクトを下から上へのz順に順次に受信する手順と、1つまたは複数のレンダ命令セットを生成する手順とを含み、この生成するための手順は、受信された各オブジェクトについて、現在の受信されたオブジェクトが対応するレンダ命令を現レンダ命令セットに追加することにより、レンダリング・システムのリソースを越えるかどうかを決定する手順と、リソースを越える場合、新しいレンダ命令セットを作成する手順と、リソースを越える場合、レンダ命令を新しいレンダ命令セットに追加して現レンダ命令セットを背景イメージとして描画する手順と、リソースを越える場合、新しいレンダ命令セットに、現在の受信されたオブジェクトが対応するレンダ命令を追加する手順と、リソースを越えない場合、現在の受信されたオブジェクトの対応するレンダ命令を現レンダ命令セットに追加する手順とを含み、さらに、1つまたは複数のレンダリング命令セットを画素データとしてレンダリングする手順を実行させる。

【0011】

本発明の第6の態様によれば、ページ記述をレンダリングするためのコンピュータ・プログラムを提供する。ここでページ記述は1つまたは複数のオブジェクトを含むものである。そして、このコンピュータ・プログラムは、コンピュータに、オブジェクトを下から上へのz順に順次受信する手順と、受信された各オブジェクトについて、受信されたオブジェクトが現在のレンダリストに追加された場合にリソースを越えるかどうかを検査し、そうであった場合、現在のレンダリストをフレーム・ストア上にレンダリングし、次いで現在のレンダリストを空に設定する手順と、受信されたオブジェクトを現在のレンダリストに追加する手順と、受信されたオブジェクトがz順における最後のオブジェクトであり、フレーム・ストア上にレンダリングされた場合、現在のレンダリストをフレーム・ストアと組み合わせてレンダリングして、レンダリングされたページ記述を形成する手順と、受信されたオブジェクトがz順における最後のオブジェクトであり、フレーム・ストア上にレンダリングされなかった場合、現在のレンダリストをレンダリングして、レンダリングされたページ記述を形成する手順を実行させる。

【0012】

【発明の実施の形態】

本発明のいくつかの好ましい実施形態を、以下、図面を参照して記載する。

【0013】

1つまたは複数の添付の図面のいずれにおいても、同じ参照番号を有するステップおよび/または機能を参照する場合、これらのステップおよび/または機能は、反対の意図が現れない限り、この説明のために同じ機能または動作を有する。

【0014】

図1は、印刷されるラスタ・データとして使用される高レベルのページ記述を、画素にレンダリングかつ提示するために構成されるコンピュータ・システム100を示している。コンピュータ・システム100は、ホスト・プロセッサ102を含み、これはシステム・ランダム・アクセス・メモリ(RAM)103、フォント・メモリ104、レンダ・エン

10

20

30

40

50

ジン 1 0 5、出力デバイス 1 0 6 およびプリンタ・エンジン 1 0 7 に関連付けられる。コンピュータ・システム 1 0 0 は、レンダリング命令生成部ソフトウェア・モジュール 1 0 9、レンダリング命令ローダ・ソフトウェア・モジュール 1 1 0、およびホスト・プロセッサ 1 0 2 上で動作するためのコンプレッサ・ソフトウェア・モジュール 1 1 1 も含む。レンダリング命令生成部ソフトウェア・モジュール 1 0 9、レンダリング命令ローダ・ソフトウェア・モジュール 1 1 0、およびコンプレッサ・ソフトウェア・モジュール 1 1 1 は、別法としてハードウェアにおいて実施することができ、バス・システム 1 0 8 に結合するか、あるいは、ローカル・バス（図示せず）と相互結合することができる。加えて、レンダ・エンジン 1 0 5 は、別法として、ホスト・プロセッサ 1 0 2 上で動作するためのソフトウェア・モジュールとして実施することができる。R A M メモリ 1 0 3 は、レンダリング命令生成部ソフトウェア・モジュール 1 0 9 によって生成された、1 つまたは複数のレンダ命令セットの命令待ち行列 1 1 2 を格納するように適合される。フォント・メモリ 1 0 4 は、その中に 1 つまたは複数のフォントのフォント文字を格納しており、これらがレンダリング・エンジン 1 0 5 によって使用される。レンダ命令生成部 1 0 9 は、1 つまたは複数のレンダリング命令セットを、高レベルのページ記述に応答して生成し、これらを R A M メモリ 1 0 3 における命令待ち行列 1 1 2 上にキューイングする。レンダリング命令ローダ・ソフトウェア・モジュール 1 1 0 は、これらの命令セットを一度に一セット、命令待ち行列 1 1 2 から取り、これをレンダリング・エンジン 1 0 5 にロードし、これが、これらの命令を画素として出力デバイス 1 0 6 およびプリンタ・エンジン 1 0 7 へ出力する。レンダ・エンジン 1 0 5 は時として、命令セットを出力デバイス 1 0 6 へレンダリングする代りに、命令セットをイメージとしてコンプレッサ・ソフトウェア・モジュール 1 1 1 へレンダリングする。コンプレッサ・ソフトウェア・モジュール 1 1 1 はこのイメージを圧縮し、これをメモリ 1 0 3 における命令待ち行列 1 1 2 へ戻すように供給し、これが後にレンダ・エンジン 1 0 5 によって使用される。

【 0 0 1 5 】

上述のコンピュータ・システム 1 0 0 のハードウェア構成要素は、バス・システム 1 0 8 を介して相互接続され、I B M P C / A T タイプのパーソナル・コンピュータおよびそれから発展させられた構成、S u n S p a r c s t a t i o n s など、当技術分野でよく知られているコンピュータ・システムの通常の動作モードにおいて動作可能である。コンピュータ・システム 1 0 0 の構成要素は、分離したコンピューティング環境においても存在できる。これらのソフトウェア・モジュールは、フロッピー・ディスク、光ディスク、メモリ・カードまたは他の適切な媒体など、コンピュータ可読媒体上に格納できることが好ましい。

【 0 0 1 6 】

代替実施形態では、レンダ・エンジン 1 0 5 が生成した画素データを任意選択的に、直接、命令待ち行列 1 1 2 へ送信して圧縮しないようにすることができる。もう 1 つの実施形態では、画素データを、後にレンダ・エンジン 1 0 5 によって使用するために、レンダ・エンジン 1 0 5 自体内で保つことができる。命令待ち行列またはレンダ命令ローダがない、他の実施形態が可能である。これらの場合、レンダ命令生成部 1 0 9 が同期的に命令をレンダ・エンジン 1 0 5 にロードする。さらにもう 1 つの実施形態では、レンダ命令生成部 1 0 9 が、分離したコンピュータに位置し、ネットワークまたはある他の種類のデータ・リンクを介して、命令待ち行列 1 1 2 および / またはレンダ・エンジン 1 0 5 を含むコンピュータに結合されることが可能である。加えて、コンピュータ・システム 1 0 0 は、ページ記述のためのフロー制御を利用することが好ましい。つまり、レンダ・リスト生成部 1 0 9 がページ記述データを受信できない場合、レンダ・リスト生成部 1 0 9 がそれを受信する準備ができるまで、ページ記述データが汚染されないように、データの損失がないようにされる。

【 0 0 1 7 】

高レベルのページ記述は、P O S T S C R I P T 言語など、印刷ページのレイアウトを z 順序で配列されたオブジェクトの形式で記述するための、いかなる知られている高レベル

10

20

30

40

50

のページ記述言語にすることもできる。

【 0 0 1 8 】

このとき、図 1 B を参照して、図 1 A のコンピュータ・システムの機能的データ・フローを示すブロック図を示す。ホスト・プロセッサ 1 0 2 が高レベルのページ記述 1 2 0 を受信する。このような高レベルのページ記述 1 2 0 は、ホスト・プロセッサ 1 0 2 上で動作するグラフィックス・パッケージによって、あるいは、ネットワークを介してこれに接続された、ある他のホスト・プロセッサ上のグラフィックス・アプリケーションから供給することができる。次いで、レンダ命令生成部 1 0 9 が 1 つまたは複数のレンダリング命令セット (1 2 2 、 1 2 4 、 1 2 6) を、高レベルのページ記述 1 2 0 に応答して生成し、これについては以下でより詳細に記載する。これらの 1 つまたは複数のレンダリング命令セット (1 2 2 、 1 2 4 、 1 2 6) が、メモリ 1 0 3 における命令待ち行列 1 1 2 上にキューイングされる。次いで、レンダ命令ローダ 1 1 0 が、これらの 1 つまたは複数のレンダリング命令セット (1 2 2 、 1 2 4 、 1 2 6) を一度に一セット、命令待ち行列 1 1 2 からロードし、これについては以下でより詳細に記載する。次いで、レンダ・エンジン 1 0 5 が 1 つまたは複数のレンダリング命令セットをレンダリングし、これについては以下でより詳細に記載する。レンダリング・エンジン 1 0 5 が出力画素 (イメージ) を出力デバイス 1 0 6 およびプリンタ・エンジン 1 0 7 へ供給する。レンダリング・エンジン 1 0 5 は、ある条件において、出力画素 (イメージ) を出力デバイス 1 0 6 へ供給する代りに、これらをコンプレッサ 1 1 1 へ供給する。コンプレッサ 1 1 1 はこのイメージ 1 2 8 を圧縮し、これが命令待ち行列 1 1 2 へ戻すように供給され、後にレンダ・エンジン 1 0 5 によって使用される。コンピュータ・システム 1 0 0 はパイプライン方式で動作し、レンダ命令生成部 1 1 0 およびレンダリング命令ローダは、同時かつ非同期的な方法で動作することができる。

【 0 0 1 9 】

図 1 C を参照すると、同図は図 1 A のコンピュータ・システムの動作を例示するための、例示的ページ記述のオブジェクトおよび対応する変換されたオブジェクトを示している。レンダ命令生成部 1 0 9 が、ページ記述のオブジェクト (0 、 1 、 2 、 3 、 4 、 5 、 6 および 7) を一度に 1 つ、最低オブジェクト (0) から開始して z 順序の増分において (すなわち、下から上へ) 受信する。レンダ命令生成部 1 0 9 は最初に、ページ記述のための最初のレンダリング命令セットを生成する。この最初の命令セットは、典型的には、ページのレンダリングに必要ないかなる基本命令をも含む。次いで、レンダ命令生成部 1 0 9 が、基本命令および最下位オブジェクト (0) に対応するレンダリング命令を含む命令セットがコンピュータ・システム 1 0 0 のリソース制限を越えさせるかどうかを決定する。この例では、レンダ命令生成部 1 0 9 が、この命令セットではリソース制限を越えないと決定する。次いで、レンダ命令生成部 1 0 9 が、基本レンダリング命令、次の最低オブジェクト (1) に対応するレンダリング命令、および、最低オブジェクト (0) に対応するレンダリング命令を含む命令セットがコンピュータ・システム 1 0 0 のリソース制限を越えさせるかどうかを決定する。この例では、レンダ命令生成部 1 0 9 が、この命令セットではリソース制限を越えないと決定する。レンダ命令生成部 1 0 9 は、リソース制限がこのセットによって越えられると決定するまで、命令をそのセットに追加し続ける。この例では、レンダ命令生成部 1 0 9 が、この組の命令が基本レンダリング命令およびオブジェクト (0 、 1 、 2 、 3 、 4 、 5 および 6) のためのレンダリング命令を含むとき、リソース制限を最初に越えると決定する。次いで、レンダリング命令生成部 1 0 9 が、基本レンダリング命令およびオブジェクト (0 、 1 、 2 、 3 、 4 および 5) のためのレンダリング命令を含む命令セットをレンダ・エンジン 1 0 5 へ、好ましくは命令待ち行列 1 1 2 およびレンダリング命令ローダ 1 1 0 を介して供給する。

【 0 0 2 0 】

次いで、レンダ命令生成部 1 0 9 が新しい初期の基本命令セットを作成し、最初にレンダリング命令を追加して、背景イメージを、オブジェクト (0 、 1 、 2 、 3 、 4 および 5) に対応するページ上に描画し、これを図 1 C でオブジェクト 0 ' として示す。これらのレ

10

20

30

40

50

ンダリング命令は背景イメージ自体を含まないが、背景イメージへの参照を含み、これが続いてレンダ・エンジンによって生成される。次いで、レンダリング命令生成部 109 が進行して、上に記載したものと同一方法で、オブジェクト 6 に対応するレンダリング命令を新規に作成されたセットに追加することによりリソース制限を越えさせるかどうかを決定する。この例では、レンダリング命令生成部 109 が、オブジェクト 6 および 7 のためのレンダリング命令を新規に作成されたセットに追加することによりリソース制限を越えさせないと決定する。

【0021】

レンダ・エンジン 105 が、ある後の時間に、画素データを背景イメージの形式で、最初のレンダリング命令セット（すなわち、オブジェクト 0、1、2、3、4 および 5 に対応する）に回答して生成する。次いで、レンダ・エンジン 105 が画素データを、第 2 のレンダリング命令セット（すなわち、オブジェクト 0'、6 および 7 に対応する）に回答して、かつ、背景イメージすなわちオブジェクト 0' のための第 1 のレンダリング命令セットに回答してレンダ・エンジン 105 によって生成された画素データを使用して、生成する。

【0022】

このように、好ましい実施形態が、ページをレイヤ形式に生成する命令を作成することができる。最初にページ記述に到着したオブジェクトは、よって z 順序において最低であり、これが命令に変換され、次いで、レイヤとして画素にレンダリングされる。これらの画素が、次のレイヤの背景イメージとして使用される。複雑なページ、または大規模オブジェクトを含むページを、幾つかのレイヤのフレーム・ストア上においてレンダリングすることができる。ページのいくつかの低い順序のオブジェクトで、著しいリソースを利用するものを、イメージとしてレンダリングすることができ、これは、ページのさらに高い順序のオブジェクトのための背景イメージとして使用することができる。これには、背景イメージ・オブジェクトが、その対応する数のオブジェクトよりも少ないリソースを利用することができる利点がある。多数のオブジェクトを含むページは、通常、システムのリソース制限を越えるものであり、これをレイヤに変換することができ、したがってリソース制限を越えないようにすることができる。

【0023】

図 2 A は、好ましい実施形態による多数のページのキューイングの一例を例示するブロック図である。好ましい実施形態では、キューイングおよび待ち行列空間リソースの共有の機構により、多数のページを別々に準備かつレンダリングすることができる。好ましい実施形態では、命令待ち行列 112 が、固定あるいは制限された量のメモリにおいて機能する。好ましい実施形態では、ページ全体またはレイヤを、ページ内で z 順序において表すレンダ命令セットが、命令待ち行列 112 においてキューイングされる。別法として、連続的に命令を流すことが可能であり、レンダ命令ローダ 110 はページまたはレイヤの間で区切りを決定し、ページまたはレイヤ全体がロードされたとき、レンダ・エンジンを開始する。

【0024】

1 つのレンダ命令セットが完全に準備され、次いでキューイングされることが好ましい。図 2 A は、命令待ち行列 112 の外観の一例であり、あらゆるページをレンダおよびシステム・リソースの制限内でレンダリングできるときの図である。この例では、各レンダ命令セットがページ全体を表す。これらが、ページ記述から受信される順序でキューイングされ、ページ 1 が最初、ページ 2 が 2 番目、ページ 3 が 3 番目などとなる。

【0025】

図 2 B は、好ましい実施形態による、多数のページのキューイングのもう 1 つの例を例示するブロック図である。図 2 B は、命令待ち行列 112 の外観の一例であり、ページの 1 つが階層化を必要とするときの例を示している。多数のオブジェクトを含むページ 1 が、システムまたはレンダ・エンジンのリソース制限を越える。したがって、ページ 1 がページ 1 a、ページ 1 b およびページ 1 c のレイヤに分割される。ページ 1 a は、ページのも

っとも不明瞭なく（最下位の）レイヤである。これは、その背景が用紙なので、背景イメージを必要としない。ページ 1 a 命令のレンダリングされた出力が、メモリ 1 0 3 においてページ 1 b のための背景イメージとして予約された空間 1 1 3 へ行く。メモリ 1 0 3 における予約された空間 1 1 3 は、ページ 1 b のための背景イメージのためのフレーム・ストアとして動作する。次いで、ページ 1 がもう 1 つのレイヤを必要とする場合、メモリ 1 0 3 において背景イメージのために予約された同じ空間 1 1 3 を、次のレイヤであるページ 1 c のための背景イメージのためのフレーム・ストアとして使用することができる。明瞭にするため、メモリ 1 0 3 においてページ 1 b および 1 c のための背景イメージのために予約された空間 1 1 3 を、別々に示す。ページ 1 b 命令のレンダリングされた出力が、ページ 1 b の背景イメージのために予約された空間 1 1 3 へ行く。これは、ページ 1 b のレンダリングされた出力がこれまで最初から描画されたページ全体の図であるので、適切に作用する。ページ 1 a 命令によって生成された背景イメージは、この図の一部である。そのため、ページ 1 a によって生成された背景イメージを保つ必要はない。ページ 1 c が、ページ 1 b によって生成された背景イメージを使用し、ページの最終的な図を生成し、これがプリンタに向けて送られる。ページ 2 は待ち行列上にあり、階層化の必要はない。

10

【 0 0 2 6 】

図 2 C は、時間におけるさらなる段階の、図 2 B におけるページのキューイングを例示するブロック図である。特に、命令待ち行列 1 1 2 の外観の一例であり、2 つのページで階層化が必要であり、両方のページが同時に待ち行列にあるときの図である。図 2 C は、図 2 B から時間的に連続したものであり、ページ 1 a および 1 b がデキューされている。ページ 1 c は、メモリ 1 0 3 における空間がページ 3 b のための背景イメージのために予約されようとするときに、なお待ち行列上にある。メモリ 1 0 3 における同じ空間 1 1 3 が、ページ 1 c のための背景イメージのために予約されるように、ページ 3 b のための背景イメージのために予約される。明瞭にするため、メモリ 1 0 3 においてページ 1 c および 3 b のための背景イメージのために予約された空間 1 1 3 を、別々に示す。ページ 3 a の出力がこの予約された空間に向けて送られ、ページ 3 b のための背景イメージがこの空間から読み取られる。

20

【 0 0 2 7 】

メモリ 1 0 3 において背景イメージのために予約された空間は、ページが異なるとサイズが変わる。たとえば、あるレイヤが白黒のオブジェクトのみを含む場合、このレイヤを表す圧縮された画素データが、圧縮されたフル・カラーのレイヤよりも少ない空間を取る。そのため、背景イメージのために予約された空間は、白黒またはモノトーンのレイヤではより小さい。これにより、背景イメージのための空間を共有することが少々困難になる。より早く生成されたレイヤがなお待ち行列にあるが、先に背景イメージのために予約された空間が不十分である場合、このレイヤのために必要とされる追加の量の空間が予約され、背景イメージのために保たれた空間上に追加される。予約されたメモリの元の部分もなお使用される。

30

【 0 0 2 8 】

好ましい実施形態における命令待ち行列の実施は、動的メモリ・アロケータを使用し、これが、背景イメージによって使用するために予約されたメモリ上で動作する。この動的メモリ・アロケータはスレッドセーフであり、レンダ命令生成部がメモリをその中に割り振ることができ、非同期的に動作するレンダ命令ローダがメモリを解放することができる。

40

【 0 0 2 9 】

好ましい実施形態では、メモリ 1 0 3 において背景イメージのために予約された空間が、それを必要とするすべてのレイヤの間で共有される。これは、レンダ命令ローダ 1 1 0 が各レイヤを、レンダ・エンジン 1 0 5 にロードするときに待ち行列から除去するので、可能である。レンダ命令ローダは、重複などのために適切でないページをデキューすることができる。後者の場合、十分なメモリを使用可能にしなければならず、適切でないもののデキューイングを、リソース計算において調節しなければならない。

【 0 0 3 0 】

50

しかし、いくつかの他の実施形態では、命令を待ち行列上で保つ必要がある可能性がある。たとえば、ページがエラーなく印刷されたことがわかるまで、命令を保つことができる。印刷または用紙のエラーがあった場合、命令をレンダ・エンジンに再ロードすることができ、ページを再度印刷することができる。完全な命令セットを保って各ページをレンダリングする1つの方法は、最後のレイヤ以外のすべてのレイヤを廃棄し、最後のレイヤに関連付けられた背景イメージを保つことである。このタイプのシステムでは、背景イメージのために予約された空間をページ内で共有することができるが、各ページのための背景イメージが、それ自体の別々の予約された空間を必要とし、これは、有効なデータが多数の背景イメージにおいて同時にある可能性があるためである。もう1つの可能な方法は、最後のレイヤをページのための背景イメージとしてレンダリングし、単にこのページのための背景イメージを保つことである。一般に、これにより空間が節約されるが、コンプレッサがプリンタと同期的に動作してプリンタと同じ速度でデータを受け入れることができない限り、時間が浪費される。次いで、用紙エラー回復のために使用されるイメージを、システム速度の損失なく生成することができる。背景イメージをなお、階層化ページごとに保つ必要がある。代替機構は、ページのすべてのレイヤのためのすべてのレンダ命令データを保つことである。次いで、背景イメージをすべてのページの間で共有することができる。しかし、レイヤを引き起こした制限が、レンダ命令待ち行列のために使用可能なメモリのサイズであった場合、そのレイヤを解放して他のジョブを印刷できるようにしなければならない。すべての上記の手法の混成が可能である。すなわち、用紙エラー回復のために保存されたメモリを最小にするが、スループット速度を最大にする混成機構である。

【0031】

図3Aおよび3Bは、図1Aのレンダ命令生成部によって使用される、1つまたは複数のレンダリング命令セットを生成するための手順のフローチャートを示している。300でレンダ命令生成部109が開始され、301で各単一の印刷ページについて初期化される。最初に、ステップ301で、新しい現レンダ命令セットを作成し、これが、常にあらゆるページで必要とされる基本レンダ命令を含む。レンダ命令生成部109がオブジェクトを1つずつページ記述から受信し、処理する。次に、ステップ302で、現レンダ命令セットによって利用された、すべてのレンダおよびシステム・リソースの現在の使用率を得る。次いで、レンダ命令生成部109がステップ303へ進行し、ページ記述における次のオブジェクトについての十分な情報を検索して、オブジェクトの複雑さおよびサイズを決定する。これが決定される方法については、更に、図4A及び4Bを参照して後述する。

【0032】

次いで、レンダ命令生成部109が判断ブロック304へ進行し、オブジェクトのレンダ命令を現レンダ命令セットに追加することにより、結果として生じる命令セットにリソース制限を越えさせるかどうかを照会する。特に、この結果として生じる命令セットがレンダ・エンジンの複雑さの制限またはメモリ容量のオーバーフローを引き起こすかどうか、あるいは、結果として生じる命令セットにレンダ命令待ち行列において過度なメモリを使用させるかどうかを検査する。

【0033】

判断ブロック304がFALSE(YES)を返し、すなわち制限を越えない場合、レンダ命令生成部がステップ312へ進行し、残りのオブジェクト・データが、必要な場合は受信される。次いで、レンダ命令生成部がステップ313へ進行し、これについては以下でより詳細に記載する。

【0034】

他方では、判断ブロック340がTRUE(YES)であり、すなわち制限を越える場合、レンダ命令生成部109がステップ305へ進行する。ステップ305で、現命令セット(結果として生じる命令セットではない)がレンダ命令ローダ110へ、命令待ち行列112を介して、ページレイヤとしてキューイングされる。加えて、命令が現命令セットに追加され、結果として生じるレンダリングされたページを、メモリにおいて次の命令セ

ット（まだキューイングされていない）における背景画像のために予約される空間に置くことを要求する。ページのこの現レイヤをレンダリングすることによって生成された出力画素データが、ステップ308またはステップ309で後続のレイヤにおいて予約される空間に向けて送られる。出力画素データを圧縮あるいは劣化させて、使用可能な空間に適合させることができる。予約された空間のサイズは、背景イメージ画素データのために必要とされる圧縮率によって決定される。

【0035】

ステップ305の後、レンダ命令生成部109がステップ306へ進行し、新しい現レンダ命令セットが作成される。この新しい現レンダ命令セットは、最初に基本レンダ命令を含み、これが常にあらゆるページで必要とされる。

10

【0036】

ステップ306の後、レンダ命令生成部109が判断ブロック307へ進行し、すでに待ち行列のどこかで背景イメージのために予約された空間があるかどうかの検査が行われる。判断ブロック307がTRUE（YES）を返した場合、レンダ命令生成部109が、予約された空間に関連付けられた参照カウンタを増分する（ステップ308）。判断ブロック307およびステップ308が非可分的に行われて、レンダ命令ローダ110が、予約された空間を、レンダ命令生成部がそれを参照しようとしている間に、デキューすることを防止する。判断ブロック307がFALSE（NO）を返し、つまり待ち行列において背景イメージのための空間がまだ存在していない場合、レンダ命令生成部109がメモリにおいて空間を予約し、予約された空間に関連付けられた参照カウンタを1に設定する（ステップ309）。

20

【0037】

次いで、レンダ命令生成部109がステップ310へ進行する。ステップ310で、レンダ命令生成部109が、イメージをページ全体（背景イメージ）のサイズに描画するための命令を現在のレンダ命令に追加し、これは先にステップ306中に作成されたものである。次いで、結果として生じる現命令セットが、ページをレンダリングするための基本命令、および背景イメージを描画するための命令となる。背景イメージが、この命令セットがレンダ・エンジン107にロードされる準備ができるときまでに、予約された空間に位置付けられる。

【0038】

30

ステップ310の後、レンダ命令生成部109がステップ312へ進行する。判断ブロック304がFALSE（NO）を返した場合も、レンダ命令生成部109がステップ312へ進行する。ステップ312で、レンダ命令生成部109が残りの入力オブジェクトを、必要な場合は受信する。これについては、以下で図5A、5Bを参照してより詳細に説明する。ステップ312の後、次いで、レンダ命令生成部109がオブジェクトをレンダ命令に変換し、これがページ記述の中間データ形式であり（ステップ313）、次いで、これらの変換されたレンダ命令を現レンダ命令セットに追加する（ステップ314）。リソース制限を越えた場合（304）、新しい現命令セットが作成されている。この場合、現セット（ステップ314）が新たな新しい命令セットとなる。他方では、リソース制限を越えていなかった場合（304）、現在のセット（ステップ314）が既存の命令セットである。

40

【0039】

ステップ314の後、レンダ命令生成部109が判断ブロック315へ進行する。判断ブロック315で、現在のオブジェクトがページ上の最後のオブジェクトであるかどうかの検査が行われる。判断ブロック315がTRUE（YES）を返した場合、現レンダ命令セットが、命令待ち行列112上で、レンダ命令ローダ110による後続のローディングのためにキューイングされる（ステップ316）。そうでない場合、判断ブロック315がFALSE（NO）を返した場合、レンダ命令生成部109がステップ302へ戻り、次のオブジェクトを受信し、処理が次のオブジェクトについて再度開始する。

【0040】

50

好ましい実施形態では、システムに渡されたオブジェクトがテキスト文字または形状またはイメージである可能性がある。文字および経路を、一様な色、混合色、イメージ色データ（場合によってはタイル化）、またはパターン色データ（これも場合によってはタイル化）で満たすことができる。オブジェクトを使用してクリップ領域を定義することもできる。レンダ・エンジン 105 が、あらゆるオブジェクトについて、エッジによって境界を画することを必要とする。したがって、レンダ命令生成部 109 がすべての入力オブジェクトのためのエッジを、入力オブジェクトに明示的なアウトラインが定義されていなかった場合であっても、作成する。オブジェクトによって満たされた領域は、常にページ記述によって、明示的あるいは暗示的に定義される。レンダ命令生成部 109 がこの定義を使用してエッジを生成する。そのため、あらゆるオブジェクトがエッジを生成するので、あらゆるオブジェクトをクリップとして使用することができる。レンダ・エンジンはクリップをサポートする。大抵のクリップがただ 1 つのレベルを使用し、色スロットを使用しない。

10

【0041】

このとき、図 4 A および図 4 B を参照して、図 3 A に示すような、オブジェクトのサイズおよび複雑さを決定するためのステップ 303 のサブステップの流れ図を示す。この処理中に、レンダ命令生成部 109 がオブジェクトのためのデータの部分を受信し、この部分は、オブジェクトのサイズおよび複雑さを決定するために十分である。ステップ 302 の後、処理が判断ブロック 420 へ進行し、現在のオブジェクトが文字であるかどうかの検査が行われる。これが文字であった場合、判断ブロック 420 が TRUE (YES) を返し、レンダ命令生成部が判断ブロック 421 へ進行する。判断ブロック 421 で、文字がすでにシステムにおいてフォント・メモリ 104 に格納されているかどうかの検査が行われる。判断ブロック 421 が TRUE (YES) を返した場合、レンダ命令生成部 109 がステップ 422 へ進行し、事前に計算された文字のサイズおよび複雑さが得られる。これが、判断ブロック 304 における後続の使用のために返される。次いで、レンダ命令生成部 109 がステップ 429 へ進行する。

20

【0042】

判断ブロック 421 が FALSE (NO) を返した場合、レンダ命令生成部 109 が判断ブロック 423 へ進行し、文字がビット・マップであるかどうかの検査が行われる。判断ブロック 423 が TRUE (YES) を返した場合、レンダ命令生成部 109 がステップ 424 へ進行し、ビット・マップのサイズ（幅および高さ）が読み取られるが、ビット・マップ・データ自体は読み取られない。ステップ 424 の後、レンダ命令生成部 109 がステップ 425 へ進行し、レンダ命令生成部 109 が、文字によって生成されたエッジ・データのサイズの最悪の推定値を計算し、これは、1 つのチェッカーボード状の画素または類似するものを仮定することによって行われる。これは、現在のオブジェクトのサイズおよび複雑さとして、判断ブロック 304 における後続の使用のために返される。ステップ 425 の後、レンダ命令生成部 109 がステップ 429 へ進行する。

30

【0043】

判断ブロック 423 が FALSE (NO) を返し、すなわち文字がビット・マップでなかった場合、レンダ命令生成部 109 がステップ 427 へ進行する。

40

【0044】

判断ブロック 420 が FALSE (NO) を返し、すなわちオブジェクトが文字でなかった場合、レンダ命令生成部 109 が判断ブロック 426 へ進行し、現在のオブジェクトが経路データによって定義された形状であるかどうかの検査が行われる。判断ブロック 426 が TRUE (YES) を返した場合、レンダ命令生成部がステップ 427 へ進行する。

【0045】

ステップ 427 中に、レンダ命令生成部 109 が形状または文字のための経路データを読み取る。別法として、レンダ命令生成部 109 が経路データの読み取りを延期し、経路データのためのヘッダ内から、形状または文字のサイズおよび複雑さを決定するために十分な情報を検索することができる。これは、好ましい実施形態では実施されていない。これ

50

は、大抵の既存のページ記述言語がこの種類のヘッダ情報を含まないからである。ステップ427の後、レンダ命令生成部109がステップ428へ進行し、経路によって生成されるエッジ・データのサイズの最悪の推定値を、経路における点の数および経路の境界ボックスを使用して計算する。これは、現在のオブジェクトのサイズおよび複雑さの一態様として、判断ブロック304における後続の使用のために返される。ステップ428の後、レンダ命令生成部109がステップ429へ進行する。

【0046】

判断ブロック426がFALSE(NO)を返した場合、レンダ命令生成部がステップ430へ進行する。現在のオブジェクトが文字または形状でなかった場合、これはイメージでなければならない。ステップ430中に、レンダ命令生成部109がイメージのためのヘッダを読み取り、これがイメージ・データのサイズ(幅および高さ)、および各画素のサイズを定義する(ステップ430)。ヘッダは、イメージに適用される変換も含む。ステップ430の後、次いで、レンダ命令生成部109が判断ブロック431へ進行する。

10

【0047】

判断ブロック431で、イメージが圧縮されずに、待ち行列およびレンダ・エンジンにおいてイメージのために可能にされた最大メモリ内に適合するかどうかの検査が行われる。判断ブロック431がTRUE(YES)を返した場合、レンダ命令生成部109が、イメージの4つの隅の点をすべてイメージ変換によって変換することによって生成されるエッジ・データの量を計算する。ステップ432の後、レンダ命令生成部がステップ429へ進行する。

20

【0048】

判断ブロック431がFALSE(NO)を返した場合、レンダ命令生成部109が判断ブロック433へ進行し、イメージが回転されるかどうかの検査が行われる。

【0049】

イメージが回転されないか、あるいは、小さい角度のみ回転される場合、判断ブロック433がFALSE(NO)を返し、レンダ命令生成部がステップ434へ進行する。ステップ434中に、レンダ命令生成部109がリソースを計算し、イメージが水平圧縮されたストリップに分割されることを知る。ステップ434の後、レンダ命令生成部109がステップ436へ進行する。

【0050】

30

他方では、イメージが大幅な角度で回転される場合、判断ブロック433がTRUE(YES)を返し、レンダ命令生成部109がステップ435へ進行する。ステップ435で、レンダ命令生成部109がリソースを計算し、イメージがタイル化圧縮されたフラグメントに分割されることを知る。ステップ435の後、レンダ命令生成部109がステップ436へ進行する。

【0051】

ステップ436中に、レンダ命令生成部109が、イメージの総圧縮データが占有することができる、メモリの最大可能サイズを計算する。これは、現在のオブジェクトのサイズおよび複雑さの1つの特性として、判断ブロック304における後続の使用のために返される。ステップ436の後、レンダ命令生成部がステップ437へ進行し、レンダ・エンジン105において、バッファがイメージ・データを圧縮解除するために必要とされるメモリの量を計算する。これは、現在のオブジェクトのサイズおよび複雑さのもう1つの特性として、判断ブロック304における後続の使用のために返される。ステップ437の後、レンダ命令生成部109がステップ438へ進行し、水平ストリップまたはタイル化フラグメントのエッジ・データに必要とされる、メモリにおける空間を計算する。これは、現在のオブジェクトのサイズおよび複雑さのさらなる特性として、判断ブロック304における後続の使用のために返される。ステップ438の後、レンダ命令生成部109がステップ429へ進行する。

40

【0052】

ステップ429中に、レンダ命令生成部109は、現在のオブジェクトが、好ましい実施

50

形態ではレベルおよび/または色スロットであるいずれかの複雑さのリソースを、現在のレンダ命令におけるいずれかの以前のオブジェクトと共有できるかどうかを、その範囲および充填色に基づいて決定する。オブジェクトがレベルまたは充填色を以前のオブジェクトと共有できない場合、新しいレベルを、それが追加されたときにそれに割り振らなければならない。これらは、現レンダ命令セットに、レンダ・エンジン 105 の制限をオーバーフローさせる可能性がある。また、オブジェクトがある量のメモリをレンダ命令待ち行列およびレンダ・エンジンにおいて使い果たし、これは、レンダおよびシステム・リソースを、そのエッジの生成および追跡において使い果たす。オブジェクトを追加することによっても、これらの制限をオーバーフローする可能性がある。これらのすべての制限に加えて、システムまたはレンダ・エンジンのいずれかに存在する他のいかなる制限も、ステップ 303 で決定されたオブジェクトのサイズおよび複雑さと共に、図 3 A のステップ 304 で検査される。

10

【0053】

このとき、図 5 A および図 5 B を参照して、図 3 B に示すような、現在のオブジェクトのデータの残りを受信するステップ 312 のサブステップの流れ図を示す。この副処理はステップ 310 で開始し、判断ブロック 540 へ進行し、現在のオブジェクトが文字かどうかの検査が行われる。判断ブロック 540 が TRUE (YES) を返した場合、レンダ命令生成部 109 が判断ブロック 541 へ進行し、なお受信するビット・マップ・イメージ・データがあるかどうかの検査が行われる。

【0054】

20

判断ブロック 541 が TRUE (YES) を返した場合、レンダ命令生成部 109 がステップ 542 へ進行し、ビット・マップ・データが受信される。ステップ 542 の後、レンダ命令生成部がステップ 543 へ進行し、ビット・マップ・イメージを追跡してエッジ・データを生成する。次いで、544 で、このエッジ・データがフォント・キャッシュ 104 に、このオブジェクトの文字定義を文字キーとして使用して格納される。このエッジ・データも、現レンダ命令セットの一部を形成する。

【0055】

判断ブロック 541 が FALSE (NO) を返した場合、レンダ命令生成部 109 が判断ブロック 545 へ進行し、なおエッジへ変換する経路データがあるかどうかの検査が行われる。なお経路データがあった場合、判断ブロック 545 が TRUE (YES) を返し、レンダ命令生成部 109 がステップ 546 へ進行し、経路データがエッジ・データに変換される。このエッジ・データも、現レンダ命令セットの一部を形成する。

30

【0056】

判断ブロック 540 が FALSE (NO) を返した場合、レンダ命令生成部 109 が判断ブロック 547 へ進行し、現在のオブジェクトが形状であるかどうかの検査が行われる。現在のオブジェクトが形状であった場合、判断ブロック 547 が TRUE (YES) を返し、レンダ命令生成部 109 がステップ 546 へ進行し、形状の経路データがエッジ・データに変換される。このエッジ・データも、現レンダ命令セットの一部を形成する。

【0057】

判断ブロック 547 が FALSE (NO) を返した場合、現在のオブジェクトがビットマップまたは形状ではなく、そのためイメージでなければならない、レンダ命令生成部 109 が判断ブロック 548 へ進行し、イメージを圧縮する必要があるかどうか検査される。イメージを圧縮する必要がない場合、判断ブロック 548 が FALSE (NO) を返し、レンダ命令生成部 109 がステップ 549 へ進行する。ステップ 549 中に、レンダ命令生成部がイメージ・データを直接現レンダ命令セットに読み込む。次いで、レンダ命令生成部 109 がステップ 550 へ進行し、イメージのためのエッジを生成する。このエッジ・データも、現レンダ命令セットの一部を形成する。

40

【0058】

イメージが圧縮される場合、判断ブロック 548 が TRUE (YES) を返し、レンダ命令生成部がステップ 551 へ進行する。ステップ 551 中に、レンダ命令生成部 109 が

50

イメージ・データの水平ストリップを読み取る。ステップ 551 の後、レンダ命令生成部 109 が判断ブロック 552 へ進行し、イメージが水平ストリップにおいてのみ圧縮されるかどうかの検査が行われる。判断ブロック 552 が TRUE (YES) を返した場合、レンダ命令生成部 109 がステップ 553 へ進行し、ストリップが圧縮される。

【0059】

判断ブロック 552 が FALSE (NO) を返した場合、レンダ命令生成部 109 がステップ 554 へ進行し、ストリップが垂直にタイル・フラグメントに分割される。ステップ 554 の後、命令生成部 109 がステップ 555 へ進行し、次いで、タイル・フラグメントが個別に圧縮される。

【0060】

ステップ 553 またはステップ 555 の後、レンダ命令生成部 109 が判断ブロック 556 へ進行し、圧縮する水平ストリップがそれ以上あるかどうかの検査が行われる。判断ブロック 556 が TRUE (YES) を返した場合、命令生成部がステップ 551 へ戻り、イメージ・データの次の水平ストリップが読み取られる。以前の水平ストリップを読み取るために使用されたものと同じバッファを、再使用することができる。

【0061】

レンダ命令生成部 109 が、イメージ・データの水平ストリップの読み取りおよび処理を、それ以上イメージ・データがなくなるまで継続する。次いで、判断ブロック 556 が FALSE (NO) を返し、レンダ命令生成部がステップ 557 に進む。ステップ 557 中に、レンダ命令生成部 109 がイメージにおけるすべてのストリップまたはフラグメントのためのエッジ・データを作成し、図 3B のステップ 313 の処理を継続する。

【0062】

レンダ・システムの制約により、イメージをデシメートならびに圧縮しなければならない可能性があることが可能である。レンダ・エンジン 105 により、イメージをわずかに 2 倍まで縮小できることが好ましい。したがって、入力イメージがプリンタの解像度の 2 倍以上であった場合、画素を入力イメージから除去してその解像度を低下させなければならない。好ましい実施形態では、これが、x 方向における適切な数の画素を平均化し、y 方向における走査線を省くことによって達成される。解像度の制限は、システム内で処理される、もう 1 つのリソース制限である。

【0063】

いくつかの状況では、圧縮されたイメージ・データが、可能にされた最大空間内に適合しない。これは、大抵のイメージ圧縮方式によって達成される圧縮率がデータ依存であり、所望の圧縮率を達成できないからである。また、イメージ全体を同じ方法において圧縮しなければならず、そうでないとアーチファクトが印刷ページにおいて可視となる。また、イメージ・データがページ記述から読み取られた後、これを再度読み取ることもできない。そのため、イメージの一部が圧縮され、圧縮されたデータが使用可能な空間に適合しないことが判明した場合、既存の圧縮データを圧縮解除し、再度圧縮して、より高い圧縮率を与えなければならない。これは、最初の場合よりも多くのリソースを使用せずに行われる。

【0064】

図 6A および図 6B を参照すると、同図は図 1A のレンダ命令ローダによって使用される、1 つまたは複数のレンダリング命令セットをロードするための手順の流れ図を示している。レンダ命令ローダは継続的にループし、1 つまたは複数のレンダ命令セットが待ち行列に現れるまで待機し、それらを一度に 1 セットずつ処理し、次いで再度待機する。

【0065】

レンダ命令ローダ 110 は、レンダ命令生成部 109 と同時に開始するが、それとは非同期的な方法で動作する。レンダ命令ローダがステップ 600 で開始し、ステップ 660 へ進行し、レンダ命令セットが命令待ち行列 112 上に存在するまで待機する。レンダ命令セットが現れたとき、ステップ 661 で、レンダ命令ローダ 110 が現命令セットを即時にレンダ・エンジンにロードする。レンダ命令ローダがステップ 662 へ進行し、現レン

10

20

30

40

50

ダ命令セットをデキューし、制御情報を現レンダ命令セットのヘッダから抽出し、命令待ち行列 1 1 2 において現レンダ命令セットによって使用された空間を解放する。

【 0 0 6 6 】

ステップ 6 6 2 の後、レンダ命令ローダ 1 1 0 が判断ブロック 6 6 3 へ進行し、現レンダ命令セットが背景イメージを使用するかどうかの検査が行われる。すなわち、現命令セットが、背景イメージを描画するための命令を含むかどうかを検査する。判断ブロック 6 6 3 が TRUE (YES) を返した場合、レンダ命令ローダがステップ 6 6 4 へ進行し、背景イメージの参照カウンタが減分される。この参照カウンタは、上で論じたレンダ命令生成部 1 0 9 によって使用されたものと同じカウンタであり、ローダ 1 1 0 によっても生成部 1 0 9 によってもアクセスすることができる。

10

【 0 0 6 7 】

ステップ 6 6 4 の後、レンダ命令ローダ 1 1 0 が判断ブロック 6 6 5 へ進行し、参照カウンタがゼロ (0) かどうかの検査が行われる。参照カウンタがこのときゼロ (0) だった場合、これは、現在、待ち行列において、メモリにおいて現在の背景イメージと同じ空間を予約する他のレンダ命令セットがないことを意味し、そのため、メモリにおいて予約されたこの空間がもはや必要とされない。現在の背景イメージが、ステップ 6 6 1 中で、先にレンダ・エンジンにロードされたことに留意されたい。したがって、判断ブロック 6 6 5 が TRUE (YES) を返したとき、レンダ命令ローダ 1 1 0 が、予約された空間を解放し、判断ブロック 6 6 7 に進む。判断ブロック 6 6 5 が FALSE (NO) を返した場合、レンダ命令ローダが直接判断ブロック 6 6 7 へ進行する。参照カウンタの減分および検査は、非可分動作において行われなければならない、非同期的にレンダ命令セットを待ち行列に追加するレンダ命令生成部 1 0 9 との正確な動作を保証しなければならない。もう 1 つの実施では、レンダ・エンジンが命令を直接待ち行列から実行し、命令のデキューおよび参照カウンタの減分が、レンダリングが完了した後に行われるようにすることが可能である。

20

【 0 0 6 8 】

判断ブロック 6 6 7 で、現レンダ命令セットがレイヤ、すなわち、後続のレイヤのための背景イメージを生成するかどうかの検査が行われる。これは、現レンダ命令セットのヘッダに含まれた情報から決定することができ、これがレンダ命令生成部 1 0 9 によって追加される。判断ブロック 6 6 7 が FALSE (NO) を返した場合、レンダ命令ローダ 1 1 0 がステップ 6 6 8 へ進行し、レンダ・エンジン 1 0 5 の出力をプリンタ 1 0 6 に向けて送る。レンダ命令ローダ 1 1 0 がステップ 6 6 9 へ進行し、レンダ・エンジンを開始し、次いでステップ 6 7 0 へ進行し、現命令セットのレンダリングが終了するまで待機する。レンダリングが終了した後、レンダリング命令ローダがステップ 6 6 0 へ戻る。もう 1 つの実施形態では、レンダ命令ローダ 1 1 0 が、レンダ・エンジン 1 0 5 におけるメモリが使い尽くされるときのみ待機する。

30

【 0 0 6 9 】

他方では、判断ブロック 6 6 7 が、現レンダ命令セットがレイヤを生成すると決定した場合、レンダ命令ローダ 1 1 0 がステップ 6 7 1 へ進行する。ステップ 6 7 1 中に、レンダ命令ローダ 1 1 0 が、次の命令セットが命令待ち行列 1 1 2 に来るまで待機する。次のレンダ命令セットが待ち行列において受信されたとき、レンダ命令ローダ 1 1 0 がステップ 6 7 2 へ進行する。ステップ 6 7 2 で、レンダ命令ローダ 1 1 0 が、メモリにおいて次のレンダ命令セットのための背景イメージの位置を得る。この段階で、レンダ命令ローダ 1 1 0 は次のレンダ命令セットをデキューしないが、ステップ 6 7 3 へ進行し、レンダ・エンジン 1 0 5 に、現レンダ命令セットのための画素データの出力をコンプレッサ 1 1 1 へ向けて送るように命令する。次いで、レンダ命令ローダ 1 1 0 がステップ 6 7 4 へ進行し、コンプレッサ 1 1 1 に、現レンダ命令セットの圧縮された画素データのためのその出力を、メモリにおいて次のレンダ命令セットのための背景イメージのために予約された空間に向けて送るように命令する。次いで、レンダ命令ローダ 1 1 0 がステップ 6 7 5 へ進行し、レンダ・エンジン 1 0 5 に、現命令セットのレンダリングを開始するように命令する

40

50

。これはまた、コンプレッサ 1 1 1 に、結果として生じる出力画素データを圧縮するように命令し、次いで、これをメモリにおいて予約された空間に格納する。次いで、ステップ 6 7 6 で、レンダリング命令ローダが、レンダリングおよび圧縮が完了するまで待機する。次いで、レンダ命令ローダが判断ブロック 6 7 7 へ進行し、圧縮されたレンダ出力画素が、背景イメージのために使用可能な空間に適合したかどうかの検査が行われる。判断ブロック 6 7 7 が F A L S E (N O) を返した場合、レンダ命令ローダ 1 1 0 がステップ 6 7 8 へ進行し、コンプレッサ 1 1 1 に、より高い圧縮率を達成するように指示する。ステップ 6 7 8 の後、レンダ命令ローダがステップ 6 7 3 へ戻る。他方では、判断ブロック 6 7 7 が T R U E (Y E S) を返し、すなわち、圧縮されたレンダ出力画素が、背景イメージのために使用可能な空間に適合した場合、レンダ命令ローダ 1 1 0 がステップ 6 6 1 に進む。

10

【 0 0 7 0 】

レンダ・エンジン 1 1 0 を、いずれかの知られているレンダ・エンジンにいくらか変更を加えたものに基づかせることができる。レンダ・エンジンは、そのレンダリングされた画素データを、コンプレッサを介して命令待ち行列に戻すように送り直し、また、そのレンダリングされた画素データをプリンタへ向けて送る能力を有するべきである。コンプレッサを、いかなる知られているコンプレッサに基づかせることもできる。別法として、コンプレッサを省くこともでき、背景イメージが圧縮解除される結果となる。後者の状況では、これによりリソース制限がレンダリング・システム上に課せられる可能性がある。

【 0 0 7 1 】

20

好ましいコンピュータ・システムは、次のオブジェクトが適合するかどうかについて、それを中間データに変換する前に検査する利点を有する。時として、次のオブジェクトが適合するかどうかについての検査が、オブジェクトのためのデータがシステム全体によって受信される前であっても行われる。たとえば、大きいイメージが到着しようとしている可能性がある。システムが、このイメージによりシステムにおけるいずれかの制限が越えられるかどうかの検査を、イメージのためのデータが受信される前に行う。より小さいオブジェクト、形状または文字などでは、データがシステムによって受信されるが、オブジェクトが適合するかどうかについての検査が行われるときに、中間データに変換されない。このように、好ましいコンピュータ・システムは、リソースを効率的に使用することも、大きいあるいは複雑なオブジェクトを正確に処理することもできる。さらに、好ましいコンピュータ・システムは、レンダ・エンジン内で著しいリソース制限を有し、これがオブジェクトおよびレンダ命令の、サイズよりもむしろ複雑さに関係付けられる。好ましいコンピュータ・システムは、入力オブジェクトがいずれかのシステムまたはレンダ・リソース制限を越えさせるかどうかを検査する。これは、オブジェクトがいずれかの制限を破らせる可能性がある場合、階層化機構を後押しする。加えて、好ましいコンピュータ・システムは、オブジェクトが受信されたちょうどそのときよりも使用可能になる可能性のある最大可能なメモリの量の知識を有し、最大量のメモリを越えるときのみ階層化を実行する。これは、システムに、より多くのページ記述データを受信できるまで、ページの印刷を待機させる副作用を有する可能性がある。加えて、好ましいコンピュータ・システムは、背景画素データの圧縮を使用して、これをメモリ内に適合させる。圧縮は、有損失または無損失にすることができる。有損失圧縮はなお、わずかにイメージの外観を劣化させる可能性があるが、外観は、画素データを劣化させるよりもよくなる。無損失圧縮は、最終的なイメージの外観の劣化をまったく引き起こさない。

30

40

【 0 0 7 2 】

図 7 を参照すると、同図には図 1 A のレンダ命令生成部によって使用される、1 つまたは複数のレンダリング命令セットを生成するための代替手順のフローチャートが示されている。この代替手順が図 3 A および図 3 B の手順と異なるのは、代替手順が、1 つの単一の z レイヤにおいて合成演算子を使用して共に結合された、多数のプリミティブ・オブジェクトを含む合成オブジェクトを処理することである。つまり、合成されたプリミティブ・オブジェクトのグループが、ペインターズ・アルゴリズムのために、単一の z レイヤと

50

みなされ、合成されたオブジェクトが、合成演算子、および/またはペインターズ・アルゴリズムと互換性のない構造を使用して結合される。したがって、合成されたプリミティブ・オブジェクトのグループが、階層化の視点から非可分的であるように扱われる。これは、合成されたオブジェクトのすべての構成要素が、オブジェクトをレンダ命令セットに追加できるようになる前に、受信されなければならないという意味である。これにより、図3Aおよび図3Bを参照して先に記載したような、1つまたは複数のレンダリング命令セットを生成するための手順がわずかに変更され、これについては以下で明らかになるであろう。しかし、これらの両方の手順には共通のステップが多数あり、理解を容易にするため、これらのステップを同じ参照番号によって参照する。

【0073】

他方では、図3Aおよび図3Bを参照して記載した手順は、zレイヤにつきただ1つの単一のプリミティブ・オブジェクトしか処理できない。この手順がzレイヤにつきただ1つのプリミティブ・オブジェクトを必要とするので、プリミティブ・オブジェクトを受信し、オブジェクトをレンダ命令待ち行列に「ペイント」する動作(ステップ314)を、順次動作にすることができる。これは、オブジェクト・データを、レンダ命令に変換する前に格納する必要がないという意味である。しかし、代替手順では、zレイヤにつき多数のプリミティブ・オブジェクトを処理することができる。そのため、オブジェクト・データおよび関連付けられた合成演算を、それらが受信されるときに格納する必要がある。zレイヤに寄与するすべてのプリミティブ・オブジェクトが受信されたとき、次いで、これらが現在のレンダ命令セットに、一度にまとめて「ペイント」される(ステップ314)。

【0074】

図7に戻り、1つまたは複数のレンダリング命令セットを生成するための代替手順を、以下でより詳細に記載する。レンダ命令生成部109が700で開始され、各単一の印刷ページについて初期化される。次いで702で、代替手順が、ページ記述の次のコマンドを、ページ記述言語(たとえば、PostScript(商標))に従って処理する。次いで704で、代替手順が、コマンドがいくつかのタイプの1つであるかどうかを検査し、次いで代替手順が、コマンド・タイプに応じて、4つの異なる副手順706、708、710および712のいずれか1つへ進行する。スイッチ・タイプのステートメントが、フロー制御においてこの多数の分岐を実行することができる。

【0075】

副手順706は、コマンド・タイプがプリミティブ・オブジェクトを追加するためのものであるときに活動化される。以下で明らかになるように、任意の数のプリミティブ・オブジェクトを受信し、構築中の合成オブジェクトに追加することができる。副手順708は、コマンド・タイプが合成演算子を変更するためのものであるときに活動化される。再度、任意の数のプリミティブ合成演算子を、合成オブジェクトを構築するために設定することができる。副手順710は、合成オブジェクトの構築が完了したときに活動化される。この副手順710は、zレイヤ内のコマンドがペインターズ・アルゴリズムと互換性がないときに、コマンド・タイプ上で活動化させることができる。副手順712は、コマンド・タイプが、他のユーティリティが活動化されるためのものであるときに、活動化される。このようなユーティリティには、クリッピング、診断、レンダ・モード調整などが含まれる可能性がある。

【0076】

これらの4つの副手順が、714を介してステップ702へ戻り、702でページ記述の次のコマンドが処理される。代替手順がステップ716で、ペイント副手順710が最終的に完了した後に終了する。

【0077】

図8Aおよび図8Bを参照すると、同図には図7に示す手順で使用される、プリミティブ・オブジェクトを追加するための副手順706をより詳細に記載されている。プリミティブ・オブジェクトを追加するための副手順706がステップ800で開始し、プリミティブ・オブジェクトに関する基本情報が検索される。副手順706が判断ブロック806へ

進行し、現在のプリミティブ・オブジェクトで既存のオブジェクトを置換するかどうかの検査が行われる。判断ブロック 806 が TRUE (YES) を返した場合、808 で、プリミティブ・オブジェクトが合成オブジェクト内で置換される。たとえば、合成オブジェクトが、ROP4 演算子を使用するようにすることができる。これは、Source、Pattern および Mask のオペランドを取る。Source、Pattern または Mask は、ペイントされる最終的な結合が分かる前に数回置換することができる。プリミティブ・オブジェクトが置換中である場合、これが除去され (808)、これは、もはやこれが合成オブジェクトのための累積リソースに寄与しないからである。次に、ステップ 302 で、副手順 706 が、現レンダ命令セットによって利用された、すべてのレンダおよびシステム・リソースの現在の使用率を、図 3 A および図 3 B を参照して記載したものと類似の様式で得る。次いで、副手順 706 がステップ 303 へ進行し、ページ記述における次のプリミティブ・オブジェクトについての十分な情報を検索し、プリミティブ・オブジェクトの複雑さおよびサイズを決定する。これが決定される方法は、図 4 A および図 4 B の好ましい手順を参照して記載した方法に類似している。

10

【0078】

次に、判断ブロック 810 で、このプリミティブ・オブジェクトを追加することによりホスト・メモリ制限を使い尽くすかどうかの検査が行われる。ホスト・メモリ制限が使い尽くされない場合、副手順 706 がステップ 816 へ進行する。ホスト・メモリ制限を越える場合、判断ブロック 812 で、副手順 706 が、階層化が支援する可能性があるかどうかを検査する。階層化は、すでにオブジェクトが現レンダ命令セットに存在する場合、支援する可能性があり、これは、背景レイヤにレンダリングされたときに、ホスト・メモリを解放する。階層化がメモリを解放しない場合、副手順 706 が、「オブジェクトが複雑すぎる」というエラー 814 信号を返す。他方では、階層化が支援する可能性がある場合、副手順 706 がステップ 305 から 310 (図 8 A 及び図 8 B) へ進行する。これらのステップ 305 から 310 (図 8 A 及び図 8 B) の動作は、図 3 A および図 3 B のステップ 305 から 310 を参照して上で記載したものと実質的に同じであるので、その説明は省略する。

20

【0079】

ステップ 310 (図 8 B) の後、あるいは、判断ブロック 810 (図 8 A) が FALSE (NO) を返したとき、副手順 706 がステップ 816 へ進行し、プリミティブ・オブジェクトのデータの残りが受信される。この代替手順では、このステップが失敗する可能性があることに留意されたい。プリミティブ・オブジェクトを受信するには不十分なホスト・メモリがある可能性がある。不十分なホスト・メモリがある場合、レンダ命令待ち行列が完全に空になるまで待機した後であっても、この実施形態が「オブジェクトが複雑すぎる」というエラーを返す。不十分なホスト・メモリがあった場合、818 で、プリミティブ・オブジェクトが、その複雑さの情報、現在の合成演算子および他のいかなる関連情報と共に保存される。次いで、副手順 706 が 820 で終了し、714 を介して、次のコマンドのためにステップ 702 へ戻る。

30

【0080】

このとき、図 9 を参照して、図 7 に示す手順で使用される、合成演算子を変更するための副手順 708 の流れ図を示す。副手順 708 がステップ 900 で開始し、合成演算子が検索される。検索される合成演算子のタイプは、ステップ 702 で処理されたコマンドのタイプによって決まる。次いで、この合成演算子が現在の合成演算子として保存される。次いで、副手順 708 が 904 で終了し、714 を介して、次のコマンドのためにステップ 702 へ戻る。

40

【0081】

図 10 A および図 10 B を参照すると、これらには図 7 に示す手順で使用される、ペイントするための副手順 710 の流れ図が示されている。この副手順 710 は、ステップ 702 で処理されたコマンドがペインターズ・アルゴリズムと互換性があるときに開始する。これが起こったとき、先に副手順 706 および 708 によって受信かつ格納された、現在

50

構築中のプリミティブ・オブジェクトのグループが完了する。1002で、副手順710が最初に、構築されたプリミティブ・オブジェクトのグループのサイズおよび複雑さを決定する。これは、プリミティブ・オブジェクトの各構成要素の、知られているサイズおよび複雑さの合計に、場合によってはいくらかの追加リソース使用率を加えたものである。たとえば、ある実施では、プリミティブ・オブジェクトを共に合成するために追加のレベルが必要となる可能性がある。次いで、副手順710が、すべてのリソースの現在の使用率を、ステップ302(図3A)に類似した様式で決定する。次いで、副手順が判断ブロック304へ進行し、リソース制限を越えるかどうかの検査が、判断ブロック304(図3A)に類似した様式で行われる。判断ブロック304がFALSE(NO)を返した場合、副手順がステップ313へ進行する。他方では、判断ブロック304がTRUE(YE S)を返した場合、副手順が判断ブロック1004へ進行し、階層化が支援する可能性があるかどうかの検査が行われる。判断ブロック1004が、階層化が支援しないと決定した場合、副手順が「オブジェクトが複雑すぎる」というエラー信号1006を返す。判断ブロック1004が、階層化が支援する可能性があるとして決定した場合、副手順710がステップ305から310(図10B)へ進行する。これらのステップ305から310(図10B)の動作は、図3Aおよび図3Bのステップ305から310を参照して上に記載したものと実質的に同じであり、以下でさらに記載しない。ステップ310の後、副手順710がステップ302へ進行する。

10

【0082】

判断ブロック304(図10A)がFALSEを返した場合、副手順がステップ313へ進行し、オブジェクト・データがレンダ命令に変換され、これはページ記述の中間データ形式であり、次いで314で、これらの変換されたレンダ命令を現レンダ命令セットに追加する。

20

【0083】

ステップ314の後、副手順710が判断ブロック315へ進行する。判断ブロック315で、現在のグループのプリミティブ・オブジェクトが、ページ上のプリミティブ・オブジェクトの最後のグループであるかどうかの検査が行われる。判断ブロック315がTRUE(YE S)を返した場合、現レンダ命令セットが命令待ち行列112上で、レンダ命令ローダ110による後続のローディングのためにキューイングされる(ステップ316)。副手順710が317で終了し、次いで、代替手順が716で終了する。判断ブロック315がFALSE(NO)を返した場合、副手順710がステップ302へ戻り、次のグループのプリミティブ・オブジェクトを受信し、処理が次のグループのプリミティブ・オブジェクトについて再度開始する。

30

【0084】

このように、任意に複雑に構築されたオブジェクトをレンダリングすることができる。しかし、階層化は、z順序付けが適用されるところでしか行うことができない。そのため、合成オブジェクト全体がホスト・メモリに一度に適合しななければならない。これはまた、合成オブジェクトを固定メモリおよび固定リソース環境内でレンダリングできるようにするため、1つのレイヤ内に(メモリおよび複雑さに関して)適合しなければならない。実際は、従来のページ記述言語を使用する際に、オブジェクトが複雑すぎてレンダリングできないことは稀である。

40

【0085】

上述のように、命令待ち行列およびレンダ命令ローダを省くことができる。これらの場合、ただ1つのレンダリング命令セットしかなく、待ち行列の必要性がない。命令生成部が再度、受信されたオブジェクトが現レンダリング命令セットに追加された場合に使用可能なリソースを越えるかどうかを検査する。リソースを越える場合、命令生成部がレンダ・エンジンに、現セット(現在受信されたオブジェクトを含まない)を、メモリ(フレーム・ストア)において予約された背景イメージの空間上にレンダリングするように命令し、次いで、現在のレンダリストを空に設定する。しかし、この空の現レンダリング命令セットは、基本レンダ初期化命令を含む可能性がある。リソースを越えない場合、命令生成部

50

は現レンダリング命令セットを変更しない。次いで、命令生成部が、受信されたオブジェクトを現レンダリング命令セットに追加する。受信されたオブジェクトがz順序における最後のオブジェクトであった場合、かつ、フレーム・ストア上にレンダリングされた場合、命令ローダがレンダ・エンジンに、現レンダリング命令セットをフレーム・ストアと組み合わせてレンダリングして、レンダリングされたページ記述を形成するように要求する。他方では、受信されたオブジェクトがz順序における最後のオブジェクトであった場合、かつ、フレーム・ストア上にレンダリングされなかった場合、命令ローダがレンダ・エンジンに、現セットをレンダリングして、レンダリングされたページ記述を形成するように要求する。

【0086】

10

前述の好ましい方法は、特定の制御フローを含む。好ましい方法の他の多数の変形形態があり、これらは本発明の精神または範囲から逸れることなく、異なる制御フローを使用する。さらに、好ましい方法の1つまたは複数のステップを、順次ではなく並列に実行することができる。前述の好ましい方法を、コンピュータ・プログラムとして実施することができ、これをコンピュータ可読媒体上に格納することができる。方法のステップを、ホスト・プロセッサにこれらのステップの機能を実行させるように命令するためのプログラム・コードとして実施することができる。

【0087】

<産業上の利用可能性>

上記から、本発明の実施形態がコンピュータ・グラフィックスおよび印刷産業に適用可能であることは明らかである。

20

【0088】

前記では、本発明のいくつかの実施形態のみを記載し、本発明の範囲および精神から逸れることなく修正および/または変更をこれに行うことができ、これらの実施形態は例示的であり制限的なものではない。たとえば、好ましい実施形態を、表示画面へのレンダリングにおける使用に適合させることができる。

【図面の簡単な説明】

【図1A】好適な実施形態を組み込む、ページ記述をレンダリングするためのコンピュータシステムの概略ブロック構成図である。

【図1B】好適な実施形態による、図1Aのコンピュータ・システムの機能的データフローを示すブロック図である。

30

【図1C】図1Aのコンピュータ・システムの動作を例示するための、例示的ページ記述のオブジェクトおよび対応する変換されたオブジェクトを示すブロック図である。

【図2A】好適な実施形態による、多数のページのキューイングの一例を例示するブロック図である。

【図2B】好適な実施形態による、多数のページのキューイングのもう1つの例を例示するブロック図である。

【図2C】時間におけるさらなる段階の、図2Bにおけるページのキューイングを例示するブロック図である。

【図3A】、

40

【図3B】図1Aのレンダ命令生成部によって使用される、1つまたは複数のレンダリング命令セットを生成するためのフローチャートである。

【図4A】、

【図4B】図3Aに示すような、オブジェクトのサイズおよび複雑さを決定するステップのサブステップのフローチャートである。

【図5A】、

【図5B】図3Bに示すような、現在のオブジェクトのデータの残りを受信するステップのサブステップのフローチャートである。

【図6A】、

【図6B】図1Aのレンダ命令ローダによって使用される、1つまたは複数のレンダリン

50

グ命令セットをロードするためのフローチャートである。

【図 7】図 1 A のレンダ命令生成部によって使用される、1 つまたは複数のレンダリング命令セットを生成するためのもう 1 つのフローチャートである。

【図 8 A】、

【図 8 B】図 7 に示す手順で使用される、プリミティブ・オブジェクトを追加するための副手順のフローチャートである。

【図 9】図 7 に示す手順で使用される、合成演算子を変更するための副手順のフローチャートである。

【図 10 A】、

【図 10 B】図 7 に示す手順で使用される、ペイントするための副手順のフローチャートである。

10

【図 1 A】

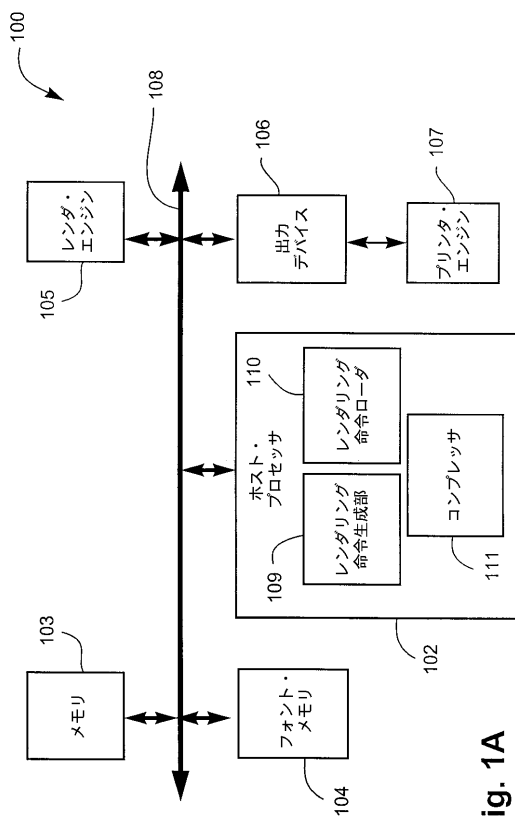


Fig. 1A

【図 1 B】

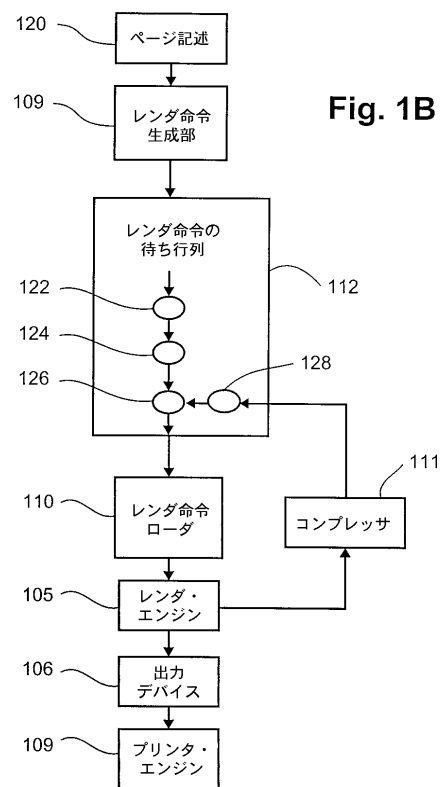


Fig. 1B

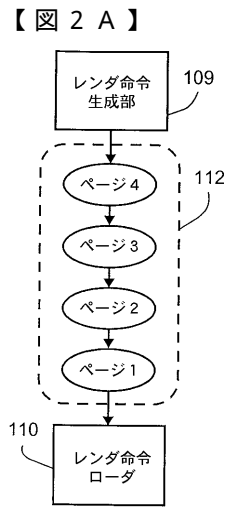
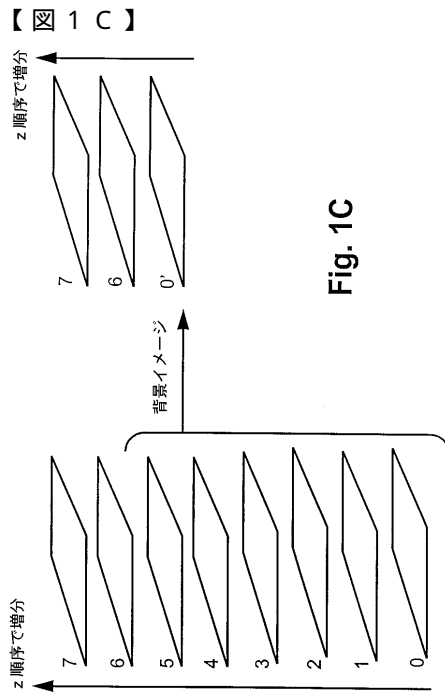


Fig. 2A

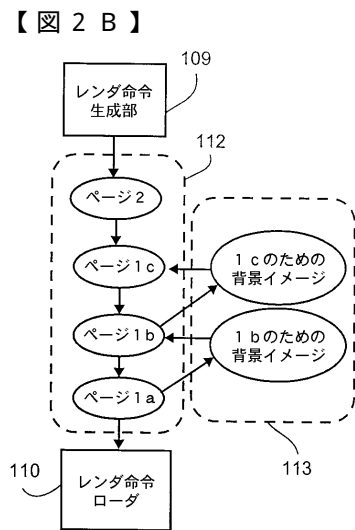


Fig. 2B

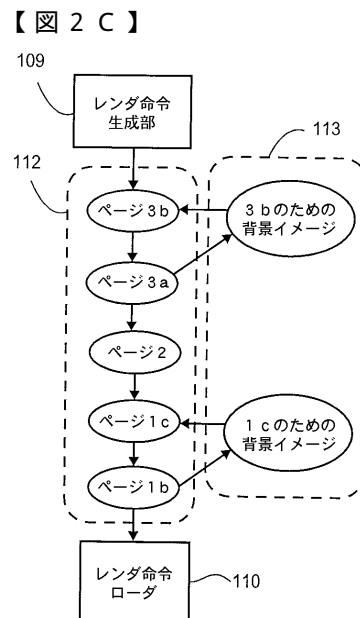
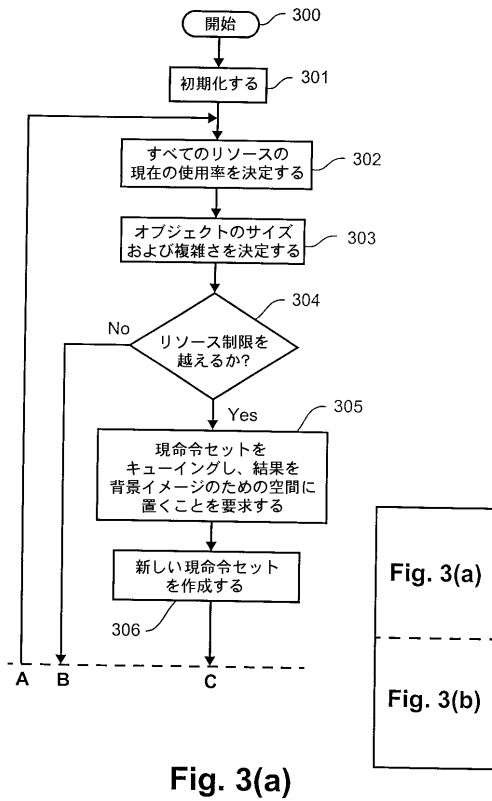
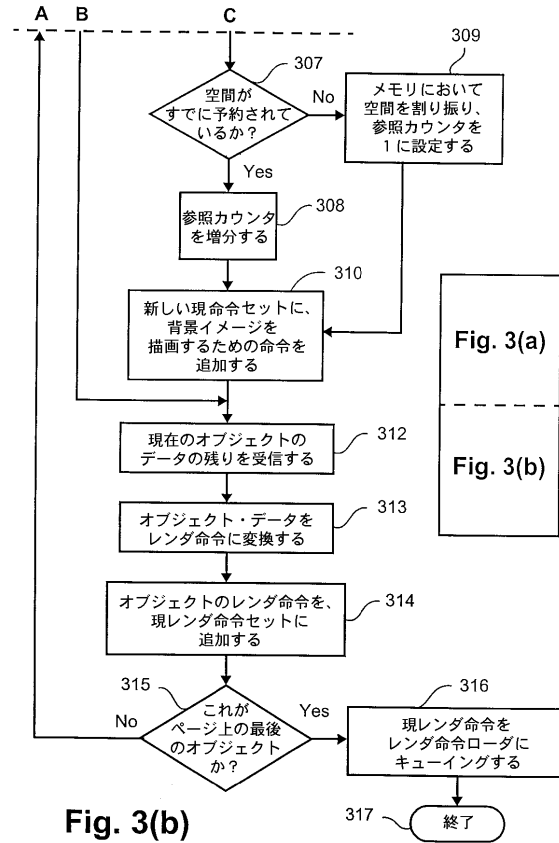


Fig. 2C

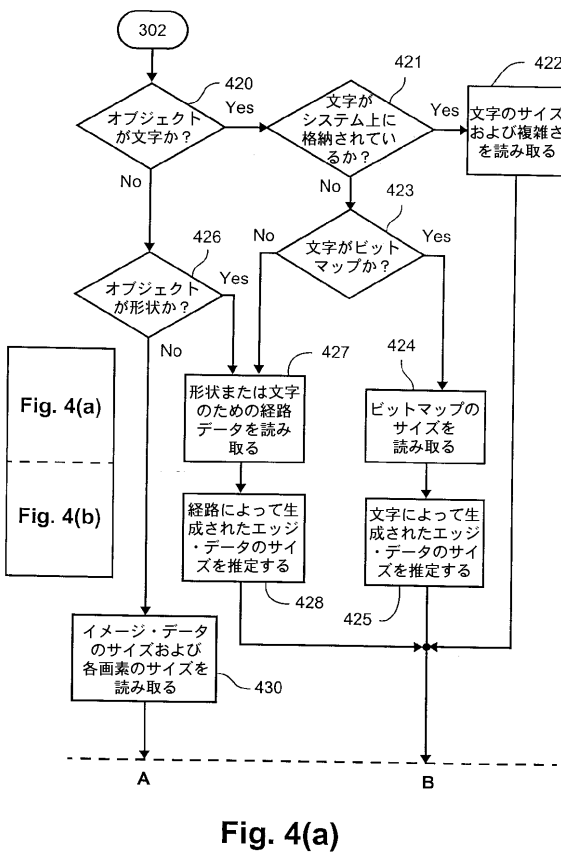
【図 3 A】



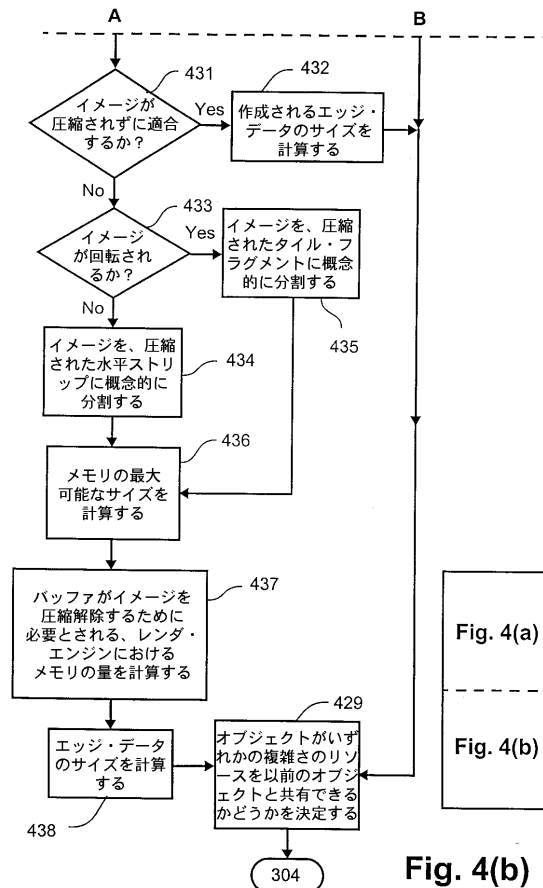
【図 3 B】



【図 4 A】



【図 4 B】



【図 5 A】

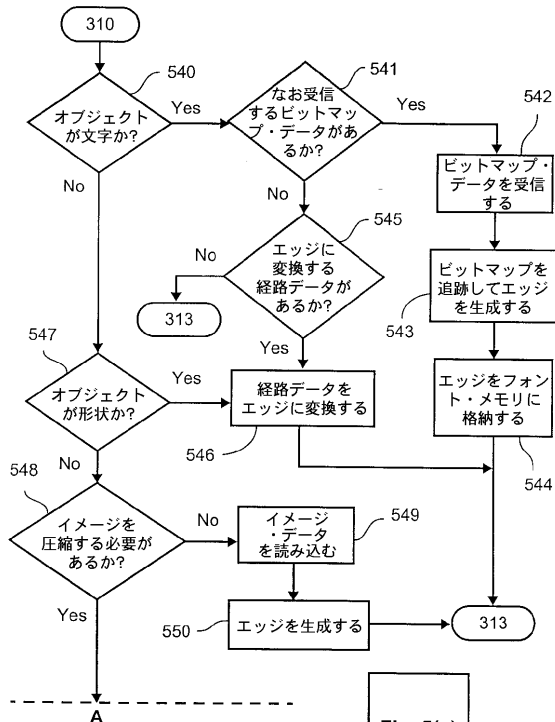


Fig. 5(a)

【図 5 B】

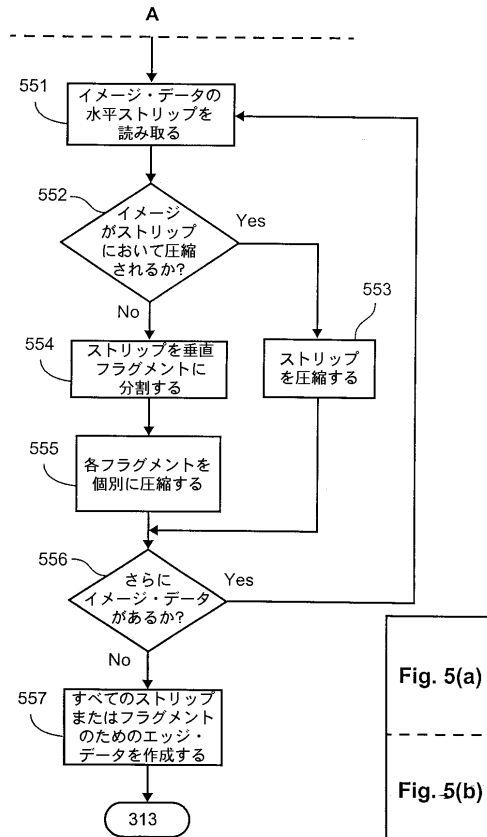
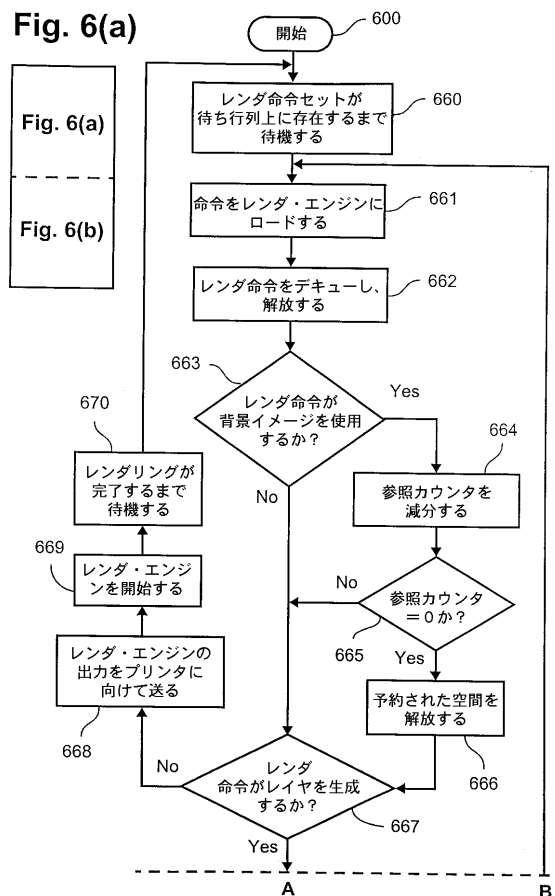


Fig. 5(b)

【図 6 A】

Fig. 6(a)



【図 6 B】

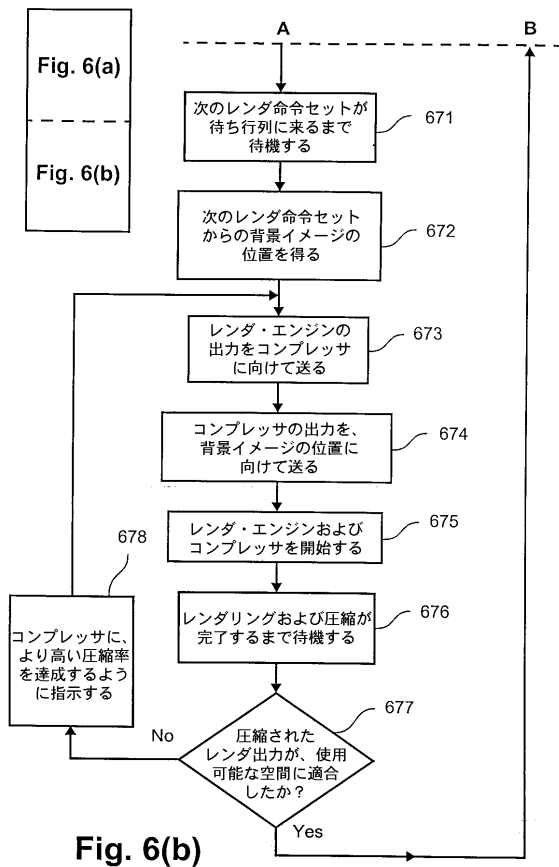


Fig. 6(b)

【図 7】

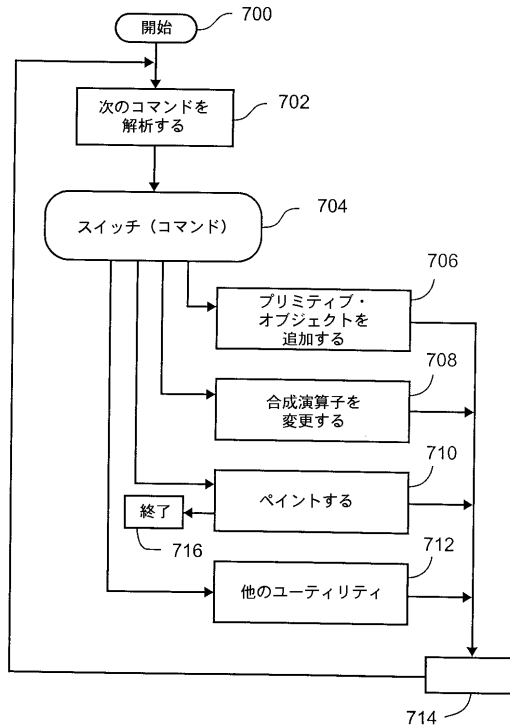
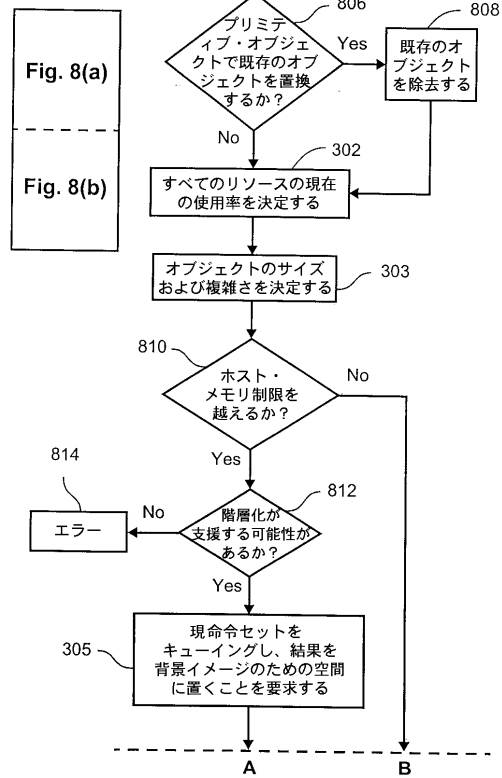


Fig. 7

【図 8 A】

Fig. 8(a)



【図 8 B】

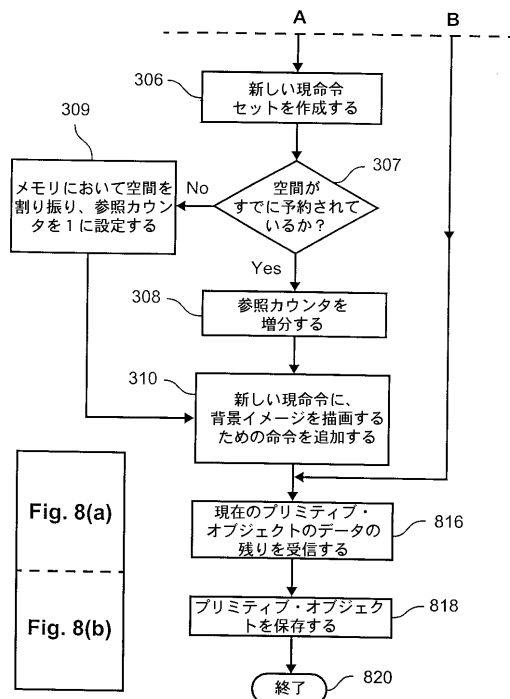


Fig. 8(b)

【図 9】

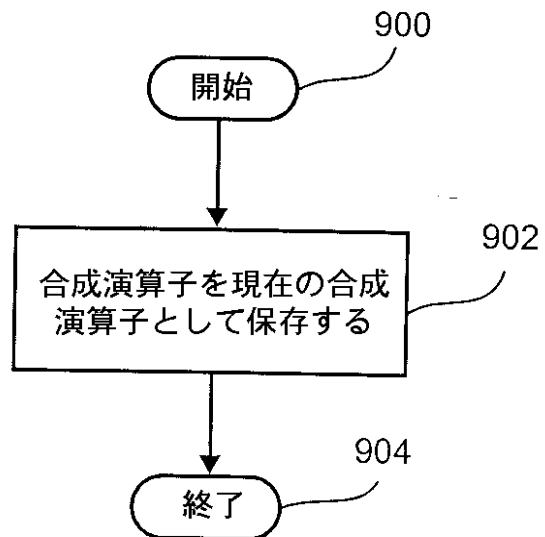
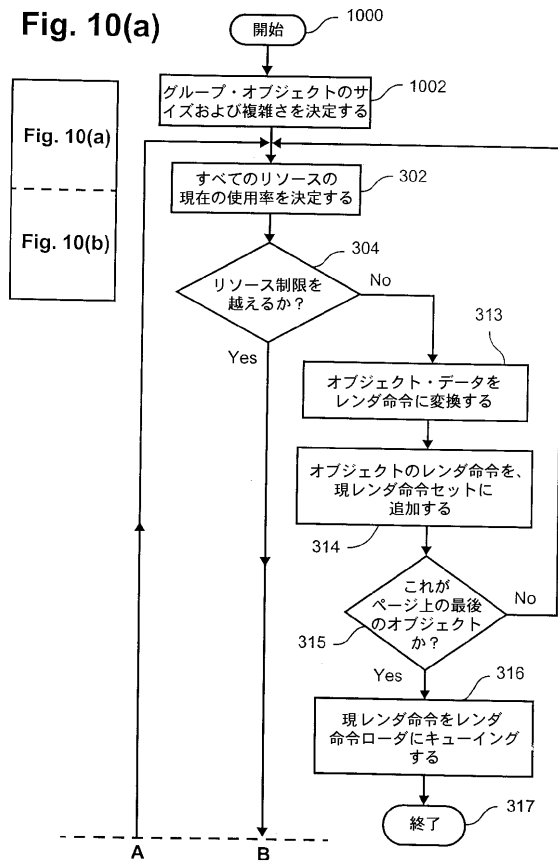


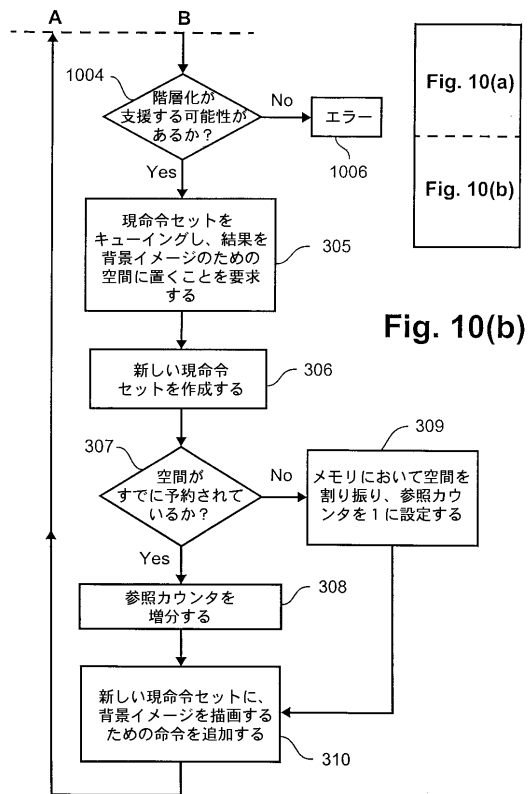
Fig. 9

【図 10 A】

Fig. 10(a)



【図 10 B】



フロントページの続き

(72)発明者 マリヤ ハーセッグ

オーストラリア国 2 1 1 3 ニュー サウス ウェールズ州, ノース ライド, トーマス
ホルト ドライブ 1 キヤノン インフォメーション システムズ リサーチ オーストラリア
プロプライエタリー リミテッド 内

(72)発明者 ティモシー メリック ロング

オーストラリア国 2 1 1 3 ニュー サウス ウェールズ州, ノース ライド, トーマス
ホルト ドライブ 1 キヤノン インフォメーション システムズ リサーチ オーストラリア
プロプライエタリー リミテッド 内

審査官 山口 大志

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

特開平 0 6 - 0 8 7 2 5 1 (J P , A)

特開平 0 5 - 0 2 4 2 9 7 (J P , A)

特開平 1 0 - 1 4 7 0 1 7 (J P , A)

(58)調査した分野(Int.Cl. , D B 名)

G06F 3/12

G06T 11/40

B41J 5/30

B41J 29/38