(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4718763号
(P4718763)

(45)発行日　平成23年7月6日(2011.7.6)　(24)登録日　平成23年4月8日(2011.4.8)

(51)Int.Cl.		F I		
G09G 5/377	(2006.01)	G09G	5/36	520L
G06T 11/00	(2006.01)	G06T	11/00	100A
G09G 5/36	(2006.01)	G09G	5/36	530C
		G09G	5/36	510M

請求項の数 16　外国語出願　(全 37 頁)

(21)出願番号	特願2003-110880 (P2003-110880)
(22)出願日	平成15年4月15日 (2003.4.15)
(65)公開番号	特開2004-4761 (P2004-4761A)
(43)公開日	平成16年1月8日 (2004.1.8)
審査請求日	平成18年3月23日 (2006.3.23)
(31)優先権主張番号	60/372,880
(32)優先日	平成14年4月15日 (2002.4.15)
(33)優先権主張国	米国(US)
(31)優先権主張番号	60/413,060
(32)優先日	平成14年9月24日 (2002.9.24)
(33)優先権主張国	米国(US)
(31)優先権主張番号	10/400,040
(32)優先日	平成15年3月25日 (2003.3.25)
(33)優先権主張国	米国(US)

前置審査

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)復代理人　100115624
弁理士　濱中　淳宏
(74)復代理人　100129171
弁理士　柿沼　健一
(74)代理人　110001243
特許業務法人　谷・阿部特許事務所
(72)発明者　スティーブン　ジェイ．エストロップ
アメリカ合衆国　９８０１４　ワシントン州　カーネーション　ノースイースト　６３　ウェイ　２８２２３

最終頁に続く

(54)【発明の名称】ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進すること

(57)【特許請求の範囲】

【請求項１】

プロセッサ、グラフィックス・デバイス、前記グラフィックス・デバイスに対して画像処理動作に対応する命令を送出するグラフィックス・デバイス・ドライバ、ビデオ・レンダラーおよびビデオ・メモリを含むコンピューティング・システム用のプログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記グラフィックス・デバイス・ドライバ及び前記ビデオ・レンダラーはプログラムモジュールであり、前記プログラムは、前記プロセッサに、

入力されたビデオ・データを前記ビデオ・メモリに割り当てられたウィンドウ内にレンダリングするように制御する前記ビデオ・レンダラーからの、前記グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することができる前記画像処理動作に関する照会であって表示されるべきビデオのタイプについての記述を含む照会を、前記グラフィックス・デバイス・ドライバへ送信させ、

前記グラフィックス・デバイス・ドライバからの、当該グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することができる少なくとも１つの画像処理動作を示す応答を、前記ビデオ・レンダラーで受信させる

命令を与えるように構成されていることを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項２】

前記グラフィックス・デバイス・ドライバは、前記グラフィックス・デバイスを介して

、前記少なくとも１つの画像処理動作を前記ビデオ・レンダラーに提供することができることを特徴とする請求項１に記載のコンピュータ読み取り可能な記録媒体。
【請求項３】
前記ビデオ・レンダラーが別の照会を、前記グラフィックス・デバイス・ドライバに送信し、前記別の照会が、前記少なくとも１つの画像処理動作であるＰｒｏｃＡｍｐ処理動作のために、当該ＰｒｏｃＡｍｐ処理動作の特性についての
最小値、
最大値、
ステップサイズ値、および
デフォルト値
に関連付けられたパラメータを要求する
ことを特徴とする請求項１に記載のコンピュータ読み取り可能な記録媒体。
【請求項４】
前記ビデオ・レンダラーは、前記グラフィックス・デバイス・ドライバに対して、当該グラフィックス・デバイス・ドライバがサポートするグラフィックス・デバイスのリストを問い合わせることを特徴とする請求項１に記載のコンピュータ読み取り可能な記録媒体。
【請求項５】
前記グラフィックス・デバイス・ドライバで、前記表示されるべきビデオのタイプについての前記記述に基づいて前記画像処理能力を調整することを生じさせる命令を記録したことを特徴とする請求項１に記載のコンピュータ読み取り可能な記録媒体。
【請求項６】
前記調整は、前記表示されるべきビデオのタイプについての前記記述に関連した特定のタイプのビデオ・ストリームをサポートするために、前記画像処理能力を調整することを特徴とする請求項５に記載のコンピュータ読み取り可能な記録媒体。
【請求項７】
前記プログラムは、前記プロセッサに、
前記ビデオ・レンダラーからの、前記グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することのできる前記少なくとも１つの画像処理動作についての特性に関する別の照会を、前記グラフィックス・デバイス・ドライバへ送信させ、
前記グラフィックス・デバイス・ドライバからの、当該グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することができる少なくとも１つの画像処理動作についての少なくとも１つの特性を示す別の応答を、前記ビデオ・レンダラーで受信させる
命令を与えるように構成されていることを特徴とする請求項１に記載のコンピュータ読み取り可能な記録媒体。
【請求項８】
前記プログラムは、前記プロセッサに、
前記ビデオ・レンダラーからの、前記グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することのできる前記少なくとも１つの画像処理動作についての、特定の画像処理動作と同時に実行することができるかに関する別の照会を、前記グラフィックス・デバイス・ドライバへ送信させ、
前記グラフィックス・デバイス・ドライバからの、当該グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することができる前記少なくとも１つの画像処理動作についての、前記特定の画像処理動作と同時に実行することができるかを示す別の応答を、前記ビデオ・レンダラーで受信させる
命令を与えるように構成されていることを特徴とする請求項１に記載のコンピュータ読み取り可能な記録媒体。
【請求項９】
プロセッサと、

前記プロセッサに結合されたグラフィックス・デバイスと、
前記プロセッサに結合され、前記プロセッサにより実行されるプログラムモジュールを記録したコンピュータ読み取り可能な記録媒体と
を備え、
前記プログラムモジュールは、ビデオ・レンダラーおよびグラフィックス・デバイス・ドライバを含み、
前記ビデオ・レンダラーは、前記プロセッサに、前記グラフィックス・デバイス・ドライバへの照会であって、前記グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することができる前記画像処理動作に関する照会であり表示されるべきビデオのタイプについての記述を含む照会を送信させるように構成され、
前記グラフィックス・デバイス・ドライバは、前記プロセッサに、前記照会に対する前記ビデオ・レンダラーへの応答であって、当該グラフィックス・デバイス・ドライバが前記ビデオ・レンダラーに提供することができる１つまたは複数のビデオ処理動作および１つまたは複数のＰｒｏｃＡｍｐ処理動作を含む少なくとも１つの画像処理動作を示す応答を送信させるように構成された
ことを特徴とするコンピューティング・デバイス。
【請求項１０】
前記１つまたは複数のビデオ処理動作および前記１つまたは複数のＰｒｏｃＡｍｐ処理動作は、同時に実行されることを特徴とする請求項９に記載のコンピューティング・デバイス。
【請求項１１】
前記グラフィックス・デバイス・ドライバは、前記プロセッサに、
画像処理動作の実行のための前記ビデオ・レンダラーからの命令を受け取り、
前記グラフィックス・デバイスによる前記画像処理を生じさせる
ことを特徴とする請求項９に記載のコンピューティング・デバイス。
【請求項１２】
前記１つまたは複数のＰｒｏｃＡｍｐ処理動作は、
なし、明度、コントラスト、色相、彩度を備えたＰｒｏｃＡｍｐ処理動作の特性のグループから選択され、
前記１つまたは複数のビデオ処理動作は、ＹＵＶ－ＲＧＢ変換動作、Ｘ伸張処理動作、Ｙ伸張処理動作、サブ長方形領域処理動作、アルファ・ブレンド動作を備えたグループから選択されることを特徴とする請求項９に記載のコンピューティング・デバイス。
【請求項１３】
プログラムモジュールであるビデオ・レンダラーとグラフィックス・デバイス・ドライバとの間の相互作用を促進する方法であって、
画像処理動作のうちの利用可能なＰｒｏｃＡｍｐ処理動作についての照会であり、表示されるべきビデオのタイプについての記述を含む照会を、前記ビデオ・レンダラーから前記グラフィックス・デバイス・ドライバへ送信することと、
前記表示されるべきビデオのタイプについての前記記述に基づいて前記グラフィックス・デバイス・ドライバの前記ＰｒｏｃＡｍｐ処理動作を、前記グラフィックス・デバイス・ドライバで生じさせることと、
前記生じたＰｒｏｃＡｍｐ処理動作を有する応答を前記グラフィックス・デバイス・ドライバから前記ビデオ・レンダラーへ送信することと
を含むことを特徴とする方法。
【請求項１４】
前記グラフィックス・デバイス・ドライバによる前記応答は、前記ＰｒｏｃＡｍｐ処理動作と同時に実行されるビデオ処理動作を含むことを特徴とする請求項１３に記載の方法。
【請求項１５】
前記送信することは、

少なくとも１つまたは複数のＰｒｏｃＡｍｐ動作に向けられたレンダリング機能を呼び出すことと、

前記呼び出しに応答して、前記グラフィックス・デバイス・ドライバへのビデオのタイプについての前記記述と関連づけられた１つまたは複数の入力データのパラメータを渡すことにより、利用可能な前記少なくとも１つまたは複数のＰｒｏｃＡｍｐ処理動作を特定することと

をさらに含むことを特徴とする請求項１３に記載の方法。

【請求項１６】

プロセッサと前記プロセッサに結合されたメモリーを含むコンピューティング・システムのためのプログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記プログラムは、前記プロセッサに請求項１３に記載の方法を実行させることを特徴とするコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本開示は、一般に、表示用の画像／図形データを処理することに関し、より詳細には、限定的なものではなく例示として、ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間で、重要な機能だけでなく、情報を伝達するためのプロトコルを使用してこれらの間での相互作用を促進すること（facilitating interaction）に関する。このような情報は、例えばＰｒｏｃＡｍｐ調整操作（adjustment operations）に向けられた照会、応答、命令などを備えることができる。

【０００２】

【従来の技術】

典型的なコンピューティング環境では、グラフィックス・カード（graphics card）またはその類似物は、画像を表示装置に転送すること、および画像の処理の少なくとも一部を取り扱うことを担当している。ビデオ画像に関しては、図形オーバレイ装置および技術がグラフィックス・カードおよびコンピューティング・デバイス全体によってしばしば使用されている。例えば、ＤＶＤまたはインターネットのストリーミング・ソースからのビデオ画像を表示するためには、そのビデオ画像を配置し維持するために図形オーバレイ手順が開始される。

【０００３】

図形オーバレイ手順は、ビデオ画像が表示されることになるスクリーン位置を設定するための長方形（rectangle）とキー・カラーとを選択する。長方形は、所望の高さおよび幅と共に長方形の隅の開始座標で定義することができる。キー・カラーは、通常、明るいピンクなどのめったに見られない色であり、ビデオが表示スクリーン上のデスクトップ最上層に論理的に配置された場合にのみ、そのビデオを定義された長方形内にオーバレイするのを確実にするために使用される。

【０００４】

動作中、グラフィックス・カードは、色ピクセルを表示装置に送っているとき、所与のピクセルの位置が選択された図形オーバレイ長方形内にあるか否かをチェックし、判定する。長方形内にない場合、デフォルト画像データが表示装置に転送される。一方、所与のピクセルの位置が選択された図形オーバレイ長方形内にある場合、グラフィックス・カードは、そのピクセルのデフォルト画像データが選択されたキー・カラーと等しいか否かをチェックして判定する。等しくない場合、所与のピクセルに対して、デフォルト画像データが表示装置に転送される。一方、所与のピクセルの色が選択されたキー・カラーである場合、グラフィックス・カードはその所与のピクセルに対して表示装置にビデオ画像データを転送する。

【０００５】

【非特許文献１】

Microsoft（登録商標）Windows（登録商標）Platform Design Note, "DirectX（登録商標

) VA: Video Acceleration API/DDI", January 23, 2001

【０００６】

【発明が解決しようとする課題】

残念ながら、上述した図形オーバレイ技術にはいくつかの欠点がある。第１に、通常、任意のある時刻に単一の図形オーバレイ手順を有効とするのに十分なハードウェア資源があるにすぎない。何にせよ、前述した図形オーバレイ技術への依存は常に、ハードウェアによって限定されて、可能な同時ビデオ表示数に対して制約を生じる。第２に、上述したピンクまたは他のキー・カラーは、表示されたビデオを含んでいるウィンドウが表示スクリーン上のデスクトップのあちらこちらに激しく移動される場合、可視可能（すなわち、関連付けられた表示装置に表示される）になることがある。

【０００７】

第３に、表示装置上に表示されるビデオ画像はプリント・スクリーン・コマンドによって取り込まれないので、プリント・スクリーン・コマンドは効果的に機能しない。取り込まれないのではなく、キー・カラーがプリント・スクリーン・コマンドによって取り込まれるので、印刷された（またはコピーして貼り付けられた）画像は、ビデオが表示装置上に表示されているその場所にキー・カラーで塗りつぶされた長方形が含まれることになる。

【０００８】

ビデオ画像を表示するための他の技術は、表示装置に転送するためにグラフィックス・プロセッサにビデオ画像を転送する前にビデオ調整を実行するためにホスト・マイクロ・プロセッサを使用することを必要とする。このホスト・プロセッサ技術にもまたいくつかの欠点がある。第１に、典型的なコンピューティング環境のホスト・マイクロ・プロセッサおよび関連付けられたメモリ・サブ・システムは、大きなビデオ画像の処理に関して最適化されていない。したがって、表示可能なビデオ画像のサイズおよび数は極度に制限される。第２に、ホスト・マイクロ・プロセッサが効率よく機能するために、ビデオ画像は、そのホスト・マイクロ・プロセッサによって直接的にアドレス可能なメモリに常駐する必要がある。その結果、圧縮解除および／またはデ・インターレーシングなどの他のタイプのハードウェア加速（hardware acceleration）をそのビデオ画像上で実行することはできない。

【０００９】

すなわち、上述した図形オーバレイ手順および上述したホスト・プロセッサに対する依存などの従来技術は、ビジュアル・アーティファクト（visual artifacts）を生じさせ、あまりにもに低速かつ／またはメモリ資源の使用が非効率的であり、ハードウェアで制限され、ビデオ表示の柔軟性に制約があり、かつ／または全機能のプリント・スクリーン・コマンドを使用可能にしない。したがって、特にビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進することによってこれらおよび他の欠点を補うための方式および／または方法が求められている。

【００１０】

【課題を解決するための手段】

ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進すること（facilitating interaction）は、関連付けられたグラフィックス・ハードウェアの画像処理能力（capabilities）に関する情報をグラフィックス・デバイス・ドライバとビデオ・レンダラーの間で交換されることを許容する通信プロトコルおよび／またはアプリケーション・プログラミング・インターフェース（ＡＰＩ）によって可能とにすることができる。画像処理能力はビデオ処理能力を含み、ビデオ処理能力は、限定的なものではなく例示としてプロセス増幅器（ＰｒｏｃＡｍｐ）制御調整（process amplifier control adjustments）、デ・インターレーシング（de-interlacing）、アスペクト・レシオ補正、色空間変換、フレーム・レート変換、垂直または水平の鏡面効果およびアルファ・ブレンディングを含む。

【００１１】

典型的な方法の実装では、方法は、１つまたは複数のビデオ・レンダラーと少なくとも１

つのグラフィックス・デバイス・ドライバの間の相互作用を促進する。この方法は、１つまたは複数のビデオ・レンダラーの１つのビデオ・レンダラーによって、ビデオ処理能力に関して、少なくとも１つのグラフィックス・デバイス・ドライバに、照会する動作と、少なくとも１つのグラフィックス・デバイス・ドライバによって、少なくとも１つのグラフィックス・デバイス・ドライバがビデオ・レンダラーに提供することのできるビデオ処理能力の少なくとも一部分についてビデオ・レンダラーに通知する動作とを含む。

【００１２】

第１の典型的な媒体の実装では、ビデオ・レンダラーに対するその中の電子的に実行可能な命令は、ビデオ・レンダラーから、ＰｒｏｃＡｍｐ能力に関する情報を要求する照会をグラフィックス・デバイス・ドライバに対して発行する動作と、グラフィックス・デバイス・ドライバから、要求したＰｒｏｃＡｍｐ能力に関する情報を含む応答をビデオ・レンダラーで受信する動作とを含む動作を引き起こす。

【００１３】

第２の典型的な媒体の実装では、グラフィックス・デバイス・ドライバに対するその中の電子的に実行可能な命令は、ビデオ・レンダラーから、ＰｒｏｃＡｍｐ能力に関する情報を要求する照会をグラフィックス・デバイス・ドライバで受信する動作と、グラフィックス・デバイス・ドライバから、ＰｒｏｃＡｍｐ能力に関して要求された情報を含む応答をビデオ・レンダラーに送信する動作とを含む動作を引き起こす。

【００１４】

典型的なシステムの実装では、システムは、ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進する。このシステムは、表示されるべきビデオに適用することのできるＰｒｏｃＡｍｐ能力に関する情報を要求する照会を準備するように構成されたビデオ・レンダリング論理と、表示されるべきビデオにどのようなＰｒｏｃＡｍｐ能力を適用することができるかを示す応答を準備するように構成されたグラフィックス・デバイス・ドライビング論理とを含む。

【００１５】

他の方法、システム、機器、プロトコル、媒体、構成などの実装態様を本明細書に記載する。

【００１６】

【発明の実施の形態】

類似の、かつ／または対応する態様、機能および構成要素を参照するために、すべての図面を通して同じ番号が使用される。

【００１７】

典型的なビデオ処理パイプラインおよびＰｒｏｃＡｍｐ調整

ＰｒｏｃＡｍｐ調整を有する典型的なビデオ処理パイプライン

図１は、ＰｒｏｃＡｍｐ調整動作１０４（ProcAmp adjustment operation）を含む第１のビデオ処理パイプライン１００である。第１のビデオ処理パイプライン１００は、グラフィックス・カードなどのグラフィックス・ハードウェアを使用して実装することができる。これは、（ｉ）３つの画像メモリ・ブロック１０２、１０６および１０８と、（ｉｉ）少なくとも１つの画像処理動作１０４とを含む。画像メモリ・ブロック１０２は、ＹＵＶビデオ・イメージ・オフスクリーン・プレーン・サーフェスを含む。図示するＰｒｏｃＡｍｐ調整動作１０４を含んでいる画像処理動作１０４は、画像メモリ・ブロック１０６を提供するために画像メモリ・ブロック１０２に適用される。画像メモリ・ブロック１０６は、画像調整動作を実行するグラフィックス・ハードウェアのパラメータおよび能力に応じてＹＵＶオフスクリーン・プレーン・サーフェスまたはＹＵＶテクスチャを含む。

【００１８】

１つまたは複数の追加の画像処理動作の後（図１には明示せず）、グラフィックス・ハードウェアは、ＲＧＢレンダー・ターゲットを含む画像メモリ・ブロック１０８を生成する。画像メモリ・ブロック１０８のＲＧＢレンダー・ターゲットは、追加の画像処理動作なしにグラフィックス・ハードウェアによって表示装置上に表示することができる。同様に

、画像データを画像メモリ・ブロック１０８から表示装置に転送している期間に画像データを他のメモリから取り出す必要がないように、画像メモリ・ブロック１０８は表示装置のスクリーンの各ピクセルに対する画像データを含んでいる。

【００１９】

ＰｒｏｃＡｍｐ調整動作１０４は、１つまたは複数のプロセス増幅器（ＰｒｏｃＡｍｐ）調整に関するものである。ＰｒｏｃＡｍｐ調整の概念は、ビデオが記憶され、操作され、アナログ技術を使用して表示されたときに始まった。しかし、ＰｒｏｃＡｍｐ調整動作１０４は、現在、デジタル技術を使用して実行することができる。このようなＰｒｏｃＡｍｐ調整動作１０４は、少なくとも明度、コントラスト、彩度、および色相のビデオ特性の１つまたは複数に向けられた１つまたは複数の動作を含むことができる。

【００２０】

典型的なＰｒｏｃＡｍｐ関連のビデオ特性

明度、コントラスト、彩度、および色相に関連した以下の記述は、それらの値を操作するために可能な、かつ／または提案された設定に関連して、典型的な記載の実装に関するものである。他のＰｒｏｃＡｍｐ調整のガイドラインを別法として使用することもできる。

【００２１】

明度：　明度は「黒設定」としても知られている。すなわち、明度は、ゲイン（コントラスト）と混同されるべきではない。明度は、各特定の表示シナリオで「ビューイング・ブラック」レベル（「ｖｉｅｗｉｎｇ　ｂｌａｃｋ」ｌｅｖｅｌ）を設定するために使用される。機能的には、ピクチャ内のすべてのルミナンス・ワード（ｌｕｍｉｎａｎｃｅ　ｗｏｒｄｓ）から、同一数の量子化ステップ（ビット）を加算または減算する。明度は、オフセットといくつかのルミナンス・ワードの和が０よりも小さいかまたはフル・レンジよりも大きい場合は、クリッピング状態を作ることができ、また一般にそうなる。通常、明度はコントラスト制御と相互作用する。

【００２２】

コントラスト：　コントラストはピクチャの明るさの「ゲイン」である。コントラストは、ピクチャ内の相対的明るさを暗い値に変更するために使用される。機能的には、入力値の範囲をより小さい範囲にまたはより大きい範囲にマッピングする線形の正または負のゲインである。設定値（例えば、ゲインが変更した際に変更されない値）は、通常はコード０と等しいが、名目上のビューイング・ブラック設定値に関連付けられたコード・ワード（code word）の方がより適切である。コントラストのゲイン構造は、一般に、この設定値を通過する線形転送ランプ（linear transfer ramp）である。コントラスト機能は、一般に、ゲインが１対１以外にセットされている場合は計算された値を丸めることが必然的に含まれ、この丸めは、一般に、視覚的アーティファクトの生成「輪郭形成（contouring）」を防止するためのプログラマチック・ディザリングを含む。

【００２３】

彩度：　彩度はコントラストの論理的同等物である。彩度は、「０クロマ」（例えば記載の実装では、ＹＵＶのコード１２８またはＲＧＢのコード０）周辺の設定値を有するゲイン機能である。

【００２４】

色相：　色相は、クロミナンス成分の位相関係である。色相は、典型的には有効範囲を－１８０から＋１８０までとし、デフォルト値を０として、度で指定される。コンポーネント・システム（例えばＹＵＶまたはＲＧＢ）の色相は、有効なクロミナンス／明るさの関係を維持するために３つの成分が共に変化する３部分からなる変数である。

【００２５】

ＹＵＶ色空間での典型的なＰｒｏｃＡｍｐ関連調整

ＹＵＶ色空間の明度、コントラスト、彩度、および色相の処理に関する以下の記述は、それらの値を操作するために可能な、かつ／または提案された設定に関連して一例として記載した実装に関するものである。他の調整のガイドラインを別法として使用することもできる。一般に、ＹＵＶ色空間で作業することによって、ビデオ・ストリームのＰｒｏｃＡ

ｍｐ調整制御に必要とされる計算が簡略化される。
【００２６】
Ｙ処理：　黒レベルを０に位置付けるためにＹの値から１６を減算する。これは、コントラスト調整が黒のレベルを変更しないようにするため、ＤＣオフセットを除去する。Ｙの値が１６よりも小さい場合があるので、この処理ではこの時点で、負のＹの値がサポートされる必要がある。コントラストは、ＹＵＶピクセル値に定数を掛けることによって調整される。（ＵとＶが調整される場合、コントラストが変更された場合はいつでも色シフトが生じることになる。）明度特性値（brightness property value）がコントラスト調整されたＹ値に加算（またはＹ値から減算）される。これによってコントラスト調整によるＤＣオフセットの発生は防止される。最後に、ブラック・レベルを１６に再位置付けするために、値１６が再び加算される。Ｙ値を処理するための典型的な方程式は以下の通りである。
Ｙ’＝（（Ｙ－１６）×Ｃ）＋Ｂ＋１６
上式で、Ｃはコントラスト値であり、Ｂは明度値である。
【００２７】
ＵＶ処理：　範囲を０周辺に位置付けるためにＵ値とＶ値の両方からまず１２８が減算される。以下のようにＵ値とＶ値とを混合することによって色相特性だけが実装される。
Ｕ’＝（Ｕ－１２８）×Ｃｏｓ（Ｈ）＋（Ｖ－１２８）×Ｓｉｎ（Ｈ）、および
Ｖ’＝（Ｖ－１２８）×Ｃｏｓ（Ｈ）－（Ｕ－１２８）×Ｓｉｎ（Ｈ）
上式で、Ｈは所望の色相角度を示している。
【００２８】
彩度は、彩度値と共に定数をＵとＶに掛けることによって調整される。最後に、値１２８がＵとＶの両方に再度加算される。ＵＶデータでの色相と彩度の結合された処理は以下の通りである。
Ｕ’＝（（（Ｕ－１２８）×Ｃｏｓ（Ｈ）＋（Ｖ－１２８）×Ｓｉｎ（Ｈ））×Ｃ×Ｓ）＋１２８、および
Ｖ’＝（（（Ｖ－１２８）×Ｃｏｓ（Ｈ）－（Ｕ－１２８）×Ｓｉｎ（Ｈ））×Ｃ×Ｓ）＋１２８
上式で、Ｃは上記Ｙ’の方程式同様コントラスト値、Ｈは色相角度、Ｓは彩度である。
【００２９】
２つの処理動作による典型的なビデオ処理パイプライン
図２は、ＲＧＢレンダー・ターゲット１０８に達するために２つのビデオ処理動作２０２と２０６を含む第２のビデオ処理パイプライン２００である。第２のビデオ処理パイプライン２００は、（ｉ）３つの画像メモリ・ブロック１０２、２０４、および１０８と、（ｉｉ）２つの画像処理動作２０２および２０６を含む。
【００３０】
第２のビデオ処理パイプライン２００に関しては、一般に、画像メモリ・ブロック２０４はＲＧＢテクスチャを含んでいる。画像メモリ・ブロック２０４は、画像処理動作２０２の適用後に画像メモリ・ブロック１０２から生じる。画像メモリ・ブロック１０８は、画像処理動作２０６後に画像メモリ・ブロック２０４から生成される。
【００３１】
ＰｒｏｃＡｍｐ制御調整に加えて他の画像処理動作を実行することができる。例えば、以下の典型的なビデオ処理動作の１つまたは複数のどの動作でも、それを表示装置のスクリーン面上に表示する前にビデオ画像データに適用することができる。
１．ＰｒｏｃＡｍｐ制御調整
２．デ・インターレーシング
３．アスペクト・レシオ補正
４．色空間変換、および
５．垂直または水平の鏡面効果処理（mirroring）およびアルファ・ブレンディング
可能ならば、ビデオ画像の処理中に消費される全体的なメモリ帯域幅を低減するために、

所望のビデオ（および／または他の画像）処理動作を可能な限り少ない動作に結合する。処理動作を結合することができる程度は、一般に、グラフィックス・ハードウェアの能力によって決定される。通常、色空間変換処理とアスペクト・レシオ補正処理は、ビデオ・ストリームの大部分ではないにしてもその多くに適用される。しかし、垂直／水平の鏡面効果およびアルファ・ブレンディングはそこまでに頻繁には適用されない。

【００３２】

第２のビデオ処理パイプライン２００に関しては、ＰｒｏｃＡｍｐ調整処理と色空間変換処理が画像処理動作２０２に組み合わされる。アスペクト・レシオ補正処理は、画像処理動作２０６によって実行される。任意選択で、垂直／水平の鏡面効果処理および／またはアルファ・ブレンディングを画像処理動作２０６に組み合わせることができる。図示するように、第２のビデオ処理パイプライン２００を実装するグラフィックス・ハードウェアは、ＲＧＢレンダー・ターゲットのような画像メモリ・ブロック１０８を作成するために２つの画像処理動作と３つの画像メモリ・ブロックを使用する。しかし、より効率よいグラフィックス・ハードウェアがあるかも知れない。

【００３３】

１つの処理動作による典型的なビデオ処理パイプライン

図３は、ＲＧＢレンダー・ターゲット１０８に達するために１つのビデオ処理動作３０２を含む第３のビデオ処理パイプライン３００である。一般に、第３のビデオ処理パイプライン３００は、１つの画像処理動作３０２と２つの画像メモリ・ブロック１０２および１０８を使用してグラフィックス・ハードウェアによって実装される。具体的には、画像メモリ・ブロック１０８は、画像処理動作３０２によって画像メモリ・ブロック１０２から生成される。画像処理動作３０２は、図示するように、後述するような複数のビデオ処理動作を含む。

【００３４】

画像処理動作３０２が、ＰｒｏｃＡｍｐ調整処理、色空間変換処理、およびアスペクト・レシオ補正処理を組み合わせるので、第３のビデオ処理パイプライン３００は（図２の）第２のビデオ処理パイプライン２００よりも短い。したがって所与のビデオ処理パイプラインのステージ数は、関連付けられたグラフィックス・ハードウェアの能力だけでなく、ビデオ画像を表示するソフトウェア（例えば、アプリケーション、オペレーティング・システム・コンポーネントなど）によって要求される画像処理動作の数およびタイプに依存する。典型的なソフトウェア、グラフィックス・ハードウェアなどは図４を参照して以下でさらに記述する。

【００３５】

典型的なビデオ関連ソフトウェアおよびグラフィックス・ハードウェア

図４は、ビデオ・レンダラー４１０とグラフィックス・デバイス・ドライバ４２２の間の相互作用を促進するように構成されたコンピューティングまたは他の電子デバイスのある種の機能要素を示すブロック図４００である。これらの様々な典型的な要素および／または機能は、ハードウェア、ソフトウェア、ファームウェア、これらのいくつかの組合せなどによって実装可能である。このようなハードウェア、ソフトウェア、ファームウェア、これらのいくつかの組合せなどを、本明細書では一括して、また個別に論理（logic）と称する。

【００３６】

ブロック図４００の構成はビデオ・データ処理機器またはシステムの一例に過ぎない。図示および説明された要素および／または機能の１つまたは複数は、ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進するための機能を損なうことなく結合されたり、再構成されたり、増加させたり、あるいは省略したりすることができることを理解されたい。

【００３７】

機器またはシステム４００は、例えば中央処理装置（ＣＰＵ）によって実行される命令を含むことができる変換論理４０８、グラフィックス・プロセッシング・ユニットおよび／

またはこれらの組合せを含む。変換論理４０８は、少なくとも１つのソース４０６から符号化されたビデオ・データを受信するように構成されている。ソース４０６からの符号化されたビデオ・データは、何らかの方法で符号化されており（例えばＭＰＥＧ－２など）、変換論理４０８はこの符号化されたビデオ・データを復号するように構成されている。

【００３８】

一例として、ソース４０６は、磁気ディスクおよび関連するディスク・ドライブ、光ディスクおよび関連するディスク・ドライブ、磁気テープおよび関連するテープ・ドライブ、固体メモリ、送信された信号、伝送媒体、または符号化されたビデオ・データを変換論理４０８に送信または他の手段で提供するように構成された類似のソースを含むことができる。ソース４０６のさらなる例は、図７を参照して後述する。ある種の実装では、ソース４０６は、ネットワーク・ソースおよび遠隔ソースなどの複数のソース構成要素を含むことができる。図示するように、ソース４０６はインターネット４０４および遠隔ディスク・ベースの記憶装置４０２を含む。

【００３９】

変換論理４０８によって出力された復号済みビデオ・データは少なくとも１つのビデオ・レンダラー４１０に提供される。限定的なものではなく例示として、ビデオ・レンダラー４１０は、Ｍｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システム（ＯＳ）のＶｉｄｅｏ　Ｍｉｘｅｒ　ａｎｄ　Ｒｅｎｄｅｒｅｒ（ＶＭＲ）を使用して実現することができる。記載の実装では、ビデオ・レンダラー４１０は、変換論理４０８がビデオ・ストリームを復号するのを補助し、ビデオ処理動作を実行させ、クローズド・キャプション（ＣＣ）またはＤＶＤサブ・ピクチャ画像などの任意の他の補助画像をビデオ画像とブレンドすることなどを行うように構成されている。また、ビデオ・レンダラー４１０は、適宜、表示装置４３６上での最終的な表示のために、グラフィック・インターフェース論理４１２にビデオ画像データを適提出するか、または提出をさせる。

【００４０】

このようにして、結果として得られるレンダーリングされたビデオ・データは、グラフィック・インターフェース論理４１２に提供される。限定的なものではなく例示として、グラフィック・インターフェース論理４１２は、例えばＤｉｒｅｃｔＤｒａｗ（登録商標）、Ｄｉｒｅｃｔ３Ｄ（登録商標）、および／または他の類似の論理を含むことができる。グラフィック・インターフェース論理４１２は、ビデオ・レンダラー４１０とグラフィックス・デバイス４２４の間にインターフェースを提供するように構成されている。図示するように、グラフィックス・デバイス４２４は、グラフィックス・プロセッサ・ユニット（ＧＰＵ）４２６、ビデオ・メモリ４３２、およびデジタル－アナログ・コンバータ（ＤＡＣ）４３４を含む。限定的なものではなく例示として、グラフィックス・デバイス４２４は、コンピューティングまたは他の電子デバイス内に構成されたビデオ・グラフィックス・カードとして実現することができる。

【００４１】

グラフィック・インターフェース論理４１２によって出力された画像データは、デバイス・ドライバ・インターフェース（ＤＤＩ）４１４を使用してグラフィックス・デバイス・ドライバ４２２に提供される。図４では、デバイス・ドライバ・インターフェース４１４は、それに関連付けられた少なくとも１つのアプリケーション・プログラミング・インターフェース（ＡＰＩ）４１６を有するように示されている。デバイス・ドライバ・インターフェース４１４は、ビデオ・レンダラー４１０とグラフィックス・デバイス・ドライバ４２２の間のインターフェースをサポートし、かつ／または設定するように構成されている。

【００４２】

機器／システム４００で、また記載の実装に関して図示されているように、デバイス・ドライバ・インターフェース４１４とグラフィックス・デバイス・ドライバ４２２は、関連付けられたオペレーティング・システム環境とグラフィックス・デバイス４２４に関して

ユーザ・モード４１８またはカーネル・モード４２０の部分としてさらに分類することができる。したがって、ビデオ・レンダラー４１０とデバイス・ドライバ・インターフェース４１４はユーザ・モード４１８の部分であり、グラフィックス・デバイス・ドライバ４２２はカーネル・モード４２０の部分である。少なくともデバイス・ドライバ・インターフェース４１４とグラフィックス・デバイス・ドライバ４２２の間で発生するこれらの通信は、ユーザ・モード４１８とカーネル・モード４２０の間を横断する。

【００４３】

この記載された実装では、したがってビデオ・レンダラー４１０によって出力されたビデオ画像データは、グラフィックス・プロセッサ・ユニット４２６に提供される。グラフィックス・プロセッサ・ユニット４２６は、１つまたは複数の画像処理動作を実行するように構成可能である。これらの画像処理動作は、それぞれにＰｒｏｃＡｍｐ調整論理４２８および／または他のビデオ処理動作論理４３０によって示されるようなＰｒｏｃＡｍｐ調整および／または他のビデオ処理動作を含む。ＰｒｏｃＡｍｐ調整動作と、デ・インターレーシングおよびフレーム・レート変換などの他の典型的なビデオ処理動作は上記に加えて下記でもさらに説明する。

【００４４】

グラフィックス・プロセッサ・ユニット４２６からの出力はビデオ・メモリ４３２に提供される。ビデオ・メモリ４３２が読み取られると、その結果得られる画像データは、表示装置４３６上に、表示装置４３６によって表示するのに適した対応するアナログ・ビデオ信号を出力するデジタル－アナログ・コンバータ４３４に転送することができる。他の構成では、表示装置４３６は、デジタル－アナログ・コンバータ４３４によるアナログ変換なしに、ビデオ・メモリ４３２からのデジタル画像データを表示することを可能にすることができる。

【００４５】

ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の典型的なプロトコル

図５は、ビデオ・レンダラー４１０とグラフィックス・デバイス・ドライバ４２２の間の典型的なプロトコルを示す通信／信号送受のダイヤグラム５００である。この典型的なプロトコルは、ＰｒｏｃＡｍｐ調整などのビデオ（または他の画像）処理動作の実行を促進する。このようなビデオ処理動作は、ユーザによって起動され、制御されたビデオ表示アプリケーション（例えば、教唆アプリケーション（instigating application））によって要求／指定される動作を含むことができる。

【００４６】

通信／信号送受の線図５００は、ビデオ・レンダラー４１０とグラフィックス・デバイス・ドライバ４２２の間の複数の通信交換と通信伝送を含む。任意選択で、通信は、任意の適切なＡＰＩ　４１６と共に（図４の）グラフィック・インターフェース４１２および／またはデバイス・ドライバ・インターフェース４１４によってイネーブルされ、かつ／または補助されることができる。

【００４７】

通信交換（communication exchange）５０２は、ビデオ処理（video processing；ＶＰ）能力の設定に向けられている。具体的には、ビデオ・レンダラー４１０は、グラフィックス・デバイス・ドライバ４２２に備えられている、またグラフィックス・デバイス・ドライバ４２２によって提供されるべきビデオ処理能力に関して、伝送５０２Ａでグラフィックス・デバイス・ドライバ４２２に対して要求または照会する。応答５０２Ｂで、グラフィックス・デバイス・ドライバ４２２は、割り当てられたビデオ処理能力についてビデオ・レンダラー４１０に通知する。

【００４８】

割り当てられるビデオ処理能力は、グラフィックス・デバイス・ドライバ４２２が実行することのできるビデオ処理動作を含む。これらは、ＰｒｏｃＡｍｐ制御調整動作、デ・インターレーシング動作、アスペクト・レシオ補正動作、色空間変換動作、垂直／水平の鏡面効果処理およびアルファ・ブレンディング、フレーム・レート変換動作などの１つまた

は複数を含むことができる。グラフィックス・デバイス・ドライバ４２２は、残されているビデオ処理動作の帯域幅についてすべてを提供したり、または一部を提供したりすることを選択することができる。残されているビデオ処理動作の帯域幅のすべてよりも少なく割り当てることによって、グラフィックス・デバイス・ドライバ４２２は後続の要求のための予備の追加ビデオ処理動作帯域幅を保持することができる。

【００４９】

通信交換５０４は、指定されたビデオ処理動作（video processing operation；ＶＰ　Ｏｐ）に関する制御特性能力の設定に方向づけられている。ビデオ・レンダラー４１０からグラフィックス・デバイス・ドライバ４２２に送られる要求５０４Ａでは、ビデオ・レンダラー４１０は、応答５０２Ｂで割り当てられた特定のビデオ処理動作を指定する。要求５０４Ａは、特定のビデオ処理動作に関してグラフィックス・デバイス・ドライバ４２２はどのような、またはどの特性能力を実行することができるかに関する問い合わせを含むこともできる。応答５０４Ｂで、グラフィックス・デバイス・ドライバ４２２は、指定された特定のビデオ処理動作に関して使用可能な特性能力に関してビデオ・レンダラー４１０に通知する。通信交換５０４は、例えば特定のビデオ処理動作に対する複数の制御特性能力がない場合は、省略することができる。

【００５０】

通信交換５０６は、他の割り当てられたビデオ処理動作のどれを、指定された特定のビデオ処理動作と同時に実行することができるかの設定に向けられている。要求５０６Ａで、ビデオ・レンダラー４１０は、ビデオ処理動作がある場合にはどのビデオ処理動作が特定のビデオ処理動作と同時に実行することができるかを決定するためにグラフィックス・デバイス・ドライバ４２２に対して照会を発行する。グラフィックス・デバイス・ドライバ４２２は、応答５０６Ｂで、ビデオ・レンダラー４１０に、グラフィックス・デバイス・ドライバ４２２が特定のビデオ処理動作と同時に実行することができるビデオ処理動作を通知する。一例として、（ｉ）伝送５０４Ａおよび５０６Ａおよび／または（ｉｉ）伝送５０４Ｂおよび５０６Ｂはそれぞれに単一の照会および応答伝送に組み合わせることができることに留意されたい。

【００５１】

通信交換５０８は、特定のビデオ処理動作の指定された制御特性に対する値の設定に向けられている。要求５０８Ａで、ビデオ・レンダラー４１０は、特定のビデオ処理動作に対する制御特性を問い合わせで指定する。指定される制御特性は、応答５０４Ｂで提供された使用可能な制御特性から選択することができる。グラフィックス・デバイス・ドライバ４２２は、特定のビデオ処理動作に対する指定された制御特性に対する値をビデオ・レンダラー４１０に提供する。これらの値は、ビデオ・レンダラー４１０がグラフィックス・デバイス・ドライバ４２２に対して特定のビデオ処理動作を実行するよう命令する際にフレームワークとして使用することができる数値による設定値、範囲などであってよい。通信交換５０８は、応答５０４Ｂで示されている各使用可能な制御特性に関して反復することができる。別法として、１つのそのような通信交換５０８は、使用可能な制御特性の（すべての制御特性を含めて）複数の制御特性に向けられることができる。

【００５２】

通信交換５１０は、ビデオ処理ストリーム・オブジェクト（video processing stream object）を開始することに向けられている。命令５１０Ａで、ビデオ・レンダラー４１０は、ビデオ処理ストリーム・オブジェクトを開くためにグラフィックス・デバイス・ドライバ４２２にコマンドを送る。このコマンドは、表示装置４３６へのビデオ画像の提示を試みるアプリケーションまたは他のソフトウェアのために送信することができる。応答５１０Ｂで、グラフィックス・デバイス・ドライバ４２２は、要求しているビデオ・レンダラー４１０にビデオ処理ストリーム・オブジェクトに対するハンドルを戻す。

【００５３】

伝送５１２Ａで、ビデオ・レンダラー４１０はグラフィックス・デバイス・ドライバ４２２に対して特定の、または別の割り当てられたビデオ処理動作を実行するよう命令する。

このビデオ処理動作実行コマンドは、特定のビデオ処理動作に対する１つまたは複数の制御特性に対する値を設定かつ／または変更するために選択された数を含むことができる。応答で、グラフィックス・デバイス・ドライバ４２２は、伝送５１２Ａで要求されたようなビデオ処理動作５１２Ｂを実行する。通常、少なくとも１つのビデオ・レンダラー４１０が、ビデオを表示すべき各アプリケーションに割り振られる。そのような教唆アプリケーションが例えばビデオ処理動作を要求するときはいつでも、任意選択の再フォーマッティング、翻訳などの後で、ビデオ・レンダラー４１０はそのような要求をビデオ処理動作命令としてグラフィックス・デバイス・ドライバ４２２に転送する。

【００５４】

ビデオ処理動作実行コマンド５１２Ａおよびその結果得られるビデオ処理動作５１２Ｂは、ビデオ処理ストリーム・オブジェクトが存在する間は、希望に応じて反復することができる。ビデオが完了するか、または関連ソフトウェアが終了したとき、ビデオ処理ストリーム・オブジェクトをクローズさせる命令５１４がビデオ・レンダラー４１０からグラフィックス・デバイス・ドライバ４２２に送信される。

【００５５】

例えば図４、５、および６の方法は、複数のブロックおよび／または複数の伝送に分割された図面で示される。しかし、この方法が記載され、かつ／または示されている順番および／またはレイアウトは限定と解釈されることを意図したものではなく、ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進する１つまたは複数のシステム、方法、媒体、プロトコル、構成などを実装するために、ブロック／伝送の任意の数を結合し、かつ／または任意の順序で再構成することができる。さらに、本明細書の記述には図４の実装（ならびに図７の典型的なシステム環境）などの特定の実装、および典型的なＡＰＩへの参照が含まれるが、本方法は、任意の適切なハードウェア、ソフトウェア、ファームウェア、またはこれらの組合せで、また任意の適切な１つ以上のプログラミング言語、１つ以上の符合化機構、１つ以上のプロトコル・パラダイム、１つ以上のグラフィックス・セットアップ、などを使用して実装することができる。

【００５６】

典型的な汎用ＡＰＩ実装態様

図６は、ビデオ・レンダラー４１０とグラフィックス・デバイス・ドライバ４２２の間の相互作用を促進する典型的な方法を示す流れ図６００である。図６に反映されている説明した実装はＰｒｏｃＡｍｐ調整動作に向けられているが、これはこのように限定されるものではない。限定されるのではなく、この典型的な汎用ＡＰＩ実装態様の少なくともある種の態様は、１つまたは複数の他のビデオ（または一般的な画像）処理動作によって使用することができる。

【００５７】

流れ図６００では、ビデオ・レンダラー４１０は６０２～６１８の９ブロックに関連付けられ、グラフィックス・デバイス・ドライバ４２２は６２０～６３０の６ブロックに関連付けられている。６０２～６１８と６２０～６３０の各ブロックは、それぞれにビデオ・レンダラー４１０とグラフィックス・デバイス・ドライバ４２２によって、またはこれらのために実行される少なくとも１つの動作に対応する。

【００５８】

流れ図６００について、典型的な汎用ＡＰＩに照らして後述する。本明細書に記載のこれらの汎用ＡＰＩは、方法、機器論理などの２つの機能グループに分割することができる。第１のグループは、グラフィックス・デバイスのビデオ処理能力を決定するために使用することができる。第２のグループは、ビデオ処理動作ストリーム・オブジェクト（video processing operation stream objects）を作成し、使用するために使用することができる。

【００５９】

これらの典型的な汎用ＡＰＩは、デバイス・ドライバ・インターフェース４１４の一部として図示されている（図４の）ＡＰＩ　４１６に対応させることができ、グラフィック・

インターフェース４１２をサポートし、グラフィックス・デバイス・ドライバ４２２とのインターフェースをとる。したがってＡＰＩ　４１６は、ユーザ・モード部分４１８内のデバイス・ドライバ・インターフェース４１４の一部として図示されている。しかし、このようなＡＰＩ　４１６は、別法としてデバイス・ドライバ・インターフェース４１４以外の他の論理に配置し、かつ／またはこれらの論理によって機能させることができる。このような他の論理は、単なる一例としてではあるが、ビデオ・レンダラー４１０、グラフィック・インターフェース４１２、カーネル・モード部分４２０の一部などを含む。

【００６０】

本節で後述する汎用ＡＰＩは、グラフィックス・デバイス・ドライバに連動して表示されているビデオ・コンテンツに対するいくつかのビデオ処理動作（例えばＰｒｏｃＡｍｐ調整、フレーム・レート変換など）の任意数をサポートするために、例えば、Ｍｉｃｒｏｓｏｆｔ（登録商標）ＤｉｒｅｃｔＸ（登録商標）Ｖｉｄｅｏ　Ａｃｃｅｌｅｒａｔｉｏｎ（ＶＡ）を拡張／強化などするために使用することができる。追加の関連情報は、参照によりその全体を本明細書に組み込んだ文献（例えば、非特許文献１参照）に記載されている。

【００６１】

本明細書では流れ図６００の動作をパーソナル・コンピュータ用Ｍｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システムの現在の展開に特に適用可能なＡＰＩに特有の表現で記載したが、そのブロックならびに本明細書に記載の他の実装も他のオペレーティング・システムおよび／または他の電子デバイスに適用可能であることを理解されたい。

【００６２】

後述する例では、１つ以上のビデオ処理動作の出力は、ターゲットＤｉｒｅｃｔＤｒａｗ（登録商標）サーフェスなどのＲＧＢレンダー・ターゲット・フォーマットで提供される。このようなことを行うことによって、従来のハードウェア・オーバレイ技術の必要性が排除される。また、表示装置上に表示可能なスクリーン全体は、存在する任意数のビデオ画像も含めて、１つのメモリ位置に存在しており、したがってプリント・スクリーン・コマンドによってビデオ画像を取り込むことができる。これにより、このプリント・スクリーン・キャプチャは文書に貼り付けられ、ファイルに追加され、直接的に印刷されることなどが可能である。

【００６３】

流れ図６００では、関連付けられたグラフィックス・ハードウェアがＰｒｏｃＡｍｐ調整ビデオ処理動作を実行することができることが既にグラフィックス・デバイス・ドライバ４２２によってビデオ・レンダラー４１０に通知済みである可能性があるか、またはビデオ・レンダラー４１０は以下のようにＰｒｏｃＡｍｐ能力の有無またはそれらの欠損を判定することができる。ブロック６０２で、ビデオ・レンダラー４１０は、表示されるべきビデオの記述を提供し、ＰｒｏｃＡｍｐ制御特性に関する図形処理能力を要求する。

【００６４】

ビデオ・レンダラー４１０は、ブロック６０２とブロック６２０の間の伝送の矢で示されるように、１つまたは複数の伝送によってビデオ記述の提供および／または制御特性要求をグラフィックス・デバイス・ドライバ４２２に対して行う。ビデオについての記述は、グラフィックス・デバイス・ドライバ４２２がビデオのタイプに基づいて、使用可能なビデオ処理能力／可能なビデオ処理能力などを調整することを可能にする。例えば所定の能力のセットをビデオのいくつかの異なるタイプのそれぞれに対してセットアップすることができる。

【００６５】

ブロック６２０で、グラフィックス・デバイス・ドライバ４２２は、ビデオ・レンダラー４１０に使用可能なＰｒｏｃＡｍｐ制御特性のリストを提供する。このリストは、明度、コントラスト、色相および彩度の１つまたは複数を含んでいても含んでいなくてもよい。ブロック６０４では、ビデオ・レンダラー４１０はグラフィックス・デバイス・ドライバ

４２２から使用可能なＰｒｏｃＡｍｐ制御特性を受信する。ブロック６２０および６２２の動作は、ブロック６０２の１つ以上の通信に応答して実行することができる。別法として、ビデオ・レンダラー４１０は、ブロック６２２の動作を誘発するために別個の問い合わせを行うことができる。

【００６６】

ブロック６２２で、グラフィックス・デバイス・ドライバ４２２は、ビデオ・レンダラー４１０にＰｒｏｃＡｍｐ調整動作と同時／並行して実行することができる可能性のあるビデオ処理動作を提供する。このようなビデオ処理動作は、ＹＵＶ－ＲＧＢ、Ｘ伸張（StretchX）、Ｙ伸張（StretchY）、サブ長方形（SubRects）およびアルファ・ブレンド（AlphaBlend）の１つまたは複数を含む場合があり、または１つも含まない場合もある。他のこのようなビデオ処理動作は、デ・インターレーシング、フレーム・レート変換などを含むことができる。ブロック６０６で、ビデオ・レンダラー４１０は、可能な同時ビデオ処理動作をグラフィックス・デバイス・ドライバ４２２から受信する。

【００６７】

ブロック６０２、６０４、６０６、６２０および６２２の動作の少なくとも一部を実装する典型的な汎用ＡＰＩは、以下のように提供される。

【００６８】

ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＣａｐｓ

このＡＰＩは、ビデオ・レンダラー４１０が、ＰｒｏｃＡｍｐ制御デバイスの入力要件と、ＰｒｏｃＡｍｐ調整動作が実行されているのと同時にサポートすることのできる任意の追加ビデオ処理動作とに関する情報を決定するためにグラフィックス・デバイス・ドライバ４２２に照会することを可能にする。

```
HRESULT
ProcAmpControlQueryCaps(
      [in]DXVA_VideoDesc* lpVideoDescription,
      [out]DXVA_ProcAmpControlCaps* lpProcAmpCaps
      );
```

【００６９】

グラフィックス・デバイス・ドライバ４２２は、ｌｐＰｒｏｃＡｍｐＣａｐｓに対して出力されたＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＣａｐｓ構造で当該モードに対するその能力を報告する。

```
typedef struct_DXVA_ProcAmpControlCaps {
      DWORD Size;
      DWORD InputPool;
      D3DFORMAT OutputFrameFormat;
      DWORD ProcAmpControlProps;
      DWORD VideoProcessingCaps;
} DXVA_ProcAmpControlCaps;
```

Ｓｉｚｅフィールドはデータ構造のサイズを示しており、異なるバージョンが異なるデータ構造サイズを有している場合は特にバージョン・インジケータとして使用することができる。

【００７０】

ＩｎｐｕｔＰｏｏｌフィールドは、ビデオ・ソース・サーフェスがそこから割り付けられ

るべきメモリ・プールを示す。例えば、メモリ・プールはグラフィックス・カード上のローカル・ビデオ・メモリ、特別タグ付きシステム・メモリ（例えば加速グラフィックス・ポート（ＡＧＰ）メモリ）、一般的なシステム・メモリなどに配置することができる。Ｄ３ＤおよびＤｉｒｅｃｔＤｒａｗドキュメンテーションも有効メモリ・プール位置についての記述を提供する。

【００７１】

ＯｕｔｐｕｔＦｒａｍｅＦｏｒｍａｔフィールドは、出力されたフレームのＤｉｒｅｃｔ３Ｄサーフェス・フォーマットを示す。ＰｒｏｃＡｍｐデバイスは入力されたサーフェス・フォーマットと一致するサーフェス・フォーマットでフレームを出力することができる。このフィールドは、ビデオ・レンダラー４１０がＰｒｏｃＡｍｐ制御ハードウェアに対して出力フレーム・サーフェスの正確なフォーマットを供給できることを保証する。ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＹＵＶ２ＲＧＢフラッグ（下記参照のこと）がＶｉｄｅｏＰｒｏｃｅｓｓｉｎｇＣａｐｓフィールドに戻された場合、ビデオ・レンダラー４１０は、有効な出力フォーマットがＲＧＢ３２などのＲＧＢフォーマットに加えてこのフィールドによって指定されるとすることができることに留意されたい。ＲＧＢ３２は、赤、緑、青チャネルのそれぞれに対する８ビットの精度と、８ビットの未使用エリアとを有するＲＧＢフォーマットである。

【００７２】

ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＰｒｏｐフィールドは、ハードウェアが実行することのできるＰｒｏｃＡｍｐ動作を特定する。グラフィックス・デバイス・ドライバ４２２は、サポートするＰｒｏｃＡｍｐ動作の組合せの以下の論理を戻す。

・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｎｏｎｅ。ハードウェアはどのようなＰｒｏｃＡｍｐ制御動作をもサポートしない。

・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｂｒｉｇｈｔｎｅｓｓ。ＰｒｏｃＡｍｐ制御ハードウェアは、ビデオ画像に対して明度調整を実行することができる。

・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｃｏｎｔｒａｓｔ。ＰｒｏｃＡｍｐ制御ハードウェアは、ビデオ画像に対してコントラスト調整を実行することができる。

・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｈｕｅ。ＰｒｏｃＡｍｐ制御ハードウェアは、ビデオ画像に対して色相調整を実行することができる。

・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｓａｔｕｒａｔｉｏｎ。ＰｒｏｃＡｍｐ制御ハードウェアは、ビデオ画像に対して彩度調整を実行することができる。

【００７３】

ＶｉｄｅｏＰｒｏｃｅｓｓｉｎｇＣａｐｓフィールドは、要求されたＰｒｏｃＡｍｐ調整と並行して実行することのできる他の動作を特定する。以下のフラグは可能な動作を特定する。

・ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＹＵＶ２ＲＧＢ。ＰｒｏｃＡｍｐ制御ハードウェアは、ビデオをＹＵＶ色空間からＲＧＢ色空間に変換することができる。使用されるＲＧＢフォーマットは、各色成分に対して８ビットまたはそれ以上の精度を有することができる。これが可能ならば、ビデオ・レンダラー４１０内のバッファ・コピーを防止することができる。ＲＧＢ色空間からＹＵＶ色空間への変換に対しては、このフラグは必要条件ではないということに留意されたい。

・ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＳｔｒｅｔｃｈＸ。ＰｒｏｃＡｍｐ制御ハードウェアが水平に伸張または縮小することができる場合、ビデオがＰｒｏｃＡｍｐ調整を受けながら、同時に、アスペクト・レシオ補正を実行することができる。

・ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＳｔｒｅｔｃｈＹ。アスペクト・レシオ調整は、アプリケーションで定義済みの合成空間内でビデオ画像をスケール変更するために一般的なピクチャ・サイズ変更動作と組み合わされる場合がある。これはどちらかというとめったにない場合である。ビデオをアプリケーション・ウィンドウに適合させるようサイズ変更するスケール変更を実行することは、ＰｒｏｃＡｍｐ調整のためのスケール変更と同時に実行することができる。これらのスケール変更を一緒に実行することによって累積的

なアーティファクトが防止される。
・ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＳｕｂＲｅｃｔｓ。このフラグは、ハードウェアが、全体画像に加えて、画像の長方形の（サブ）領域で動作することができることを特定する。ＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＢｌｔデータ構造のソース長方形によって、この長方形の領域を特定することができる。
・ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＡｌｐｈａＢｌｅｎｄ。アルファ・ブレンディングは、透明度および／または不透明度のレベルの設定などによって他の図形情報がどのように表示されるかを制御することができる。したがって、アルファ値は、色の透明度、すなわちその色が背景色とブレンドされる程度を示すことができる。このようなアルファ値は、完全な透明色から完全な不透明色までの範囲で変化させることができる。

【００７４】
動作の際、アルファ・ブレンディングは、ソースと背景色データのピクセルごとのブレンディングを使用して達成することができる。アルファ・ブレンディング動作を実行するために、所与のソース色の３つの色成分（赤、緑、青）のそれぞれを背景色の対応する成分とブレンドすることができる。典型的な実装では、色は一般に、アルファ、赤、緑、および青のそれぞれに対して８ビットずつの３２ビット値で表示することができる。

【００７５】
ここでもまた、この機能を使用することによってビデオ・レンダラー４１０によるバッファ・コピーが防止される。しかし、アプリケーションはビデオ・ストリームに関連付けられた一定のアルファ値をめったに変更しないので、これもめったに使用されない機能である。

【００７６】
流れ図６００のブロック６０８で、ビデオ・レンダラー４１０は、ブロック６０４で受信した中からＰｒｏｃＡｍｐ制御特性を選択する。ブロック６１０で、ビデオ・レンダラー４１０は、選択したＰｒｏｃＡｍｐ制御特性に対する１つまたは複数の値をグラフィックス・デバイス・ドライバ４２２に要求する。ブロック６２４で、グラフィックス・デバイス・ドライバ４２２は、要求されたＰｒｏｃＡｍｐ制御特性に対する値をビデオ・レンダラー４１０に送る。このような値は、デフォルト値、増分値、最小値、最大値などの１つまたは複数に関係付けることができる。

【００７７】
ブロック６１２で、ビデオ・レンダラー４１０は、グラフィックス・デバイス・ドライバ４２２から受信し、これにより、選択したＰｒｏｃＡｍｐ制御特性に関する１つまたは複数の値について、通知を受ける。ブロック６１２からブロック６０８に対する矢で示されるように、ブロック６０８、６１０、６１２および６２４の動作は、その１つまたは複数に対して反復して、使用可能なＰｒｏｃＡｍｐ制御特性のすべてを含めることができる。別法として、ビデオ・レンダラー４１０は、グラフィックス・デバイス・ドライバ４２２に対して、複数の伝送を有する単一の通信交換によって使用可能なＰｒｏｃＡｍｐ制御特性のすべてを含めた複数を照会することができる。

【００７８】
ブロック６０８、６１０、６１２、および６２４の動作の少なくとも一部を実装する典型的な汎用ＡＰＩは以下のように提供される。
ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＲａｎｇｅ
各ＰｒｏｃＡｍｐ特性（明度、コントラスト、彩度、色相、など）に関しては、ビデオ・レンダラー４１０は、最小、最大、ステップサイズ（増分）、デフォルト値などを決定するためにグラフィックス・デバイス・ドライバ４２２に照会する。ハードウェアが特定のＰｒｏｃＡｍｐ制御特性をサポートしない場合、グラフィックス・デバイス・ドライバ４２２は、ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＲａｎｇｅ機能に応答して「Ｅ＿ＮＯＴＩＭＰＬ」を戻すことができる。

【００７９】
グラフィックス・デバイス・ドライバ４２２は、異なるＰｒｏｃＡｍｐ制御特性に対して

希望する任意の値を戻すことができるが、以下の設定値を一例として提供する（一覧の数字はすべて浮動小数点である）。

特性	最小	最大	デフォルト値	増分
明度	-100.0F	100.0F	0.0F	0.1F
コントラスト	0.0F	10.0F	1.0F	0.01F
彩度	0.0F	10.0F	1.0F	0.01F
色相	-180.0F	180.0F	0.0F	0.1F

【００８０】

デフォルト値がビデオ・ストリームのヌル変換をもたらす場合、教唆するアプリケーション（instigating application）がＰｒｏｃＡｍｐ制御特性を１つも変更しないならば、ビデオ・レンダラー４１０はそのビデオ・パイプラインでＰｒｏｃＡｍｐ調整の段階をバイパスすることができる。

```
HRESULT
ProcAmpControlQueryRange(
      [in]DWORD VideoProperty,
      [in]DXVA_VideoDesc* lpVideoDescription,
      [out]DXVA_VideoPropertyRange* lpPropRange
      );
```

【００８１】

ＶｉｄｅｏＰｒｏｐｅｒｔｙは、グラフィックス・デバイス・ドライバ４２２が情報を戻すことを既に要求している（１つまたは複数の）ＰｒｏｃＡｍｐ制御特性を特定する。記載の実装では、このフィールドに対する可能なパラメータ値は以下の通りである。

・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｂｒｉｇｈｔｎｅｓｓ；
・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｃｏｎｔｒａｓｔ；
・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｈｕｅ；
・ＤＸＶＡ＿ＰｒｏｃＡｍｐ＿Ｓａｔｕｒａｔｉｏｎ．

【００８２】

ｌｐＶｉｄｅｏＤｅｓｃｒｉｐｔｉｏｎは、ＰｒｏｃＡｍｐ調整が適用されようとしているビデオについての記述をグラフィックス・デバイス・ドライバ４２２に提供する。グラフィックス・デバイス・ドライバ４２２は、そのＰｒｏｃＡｍｐ機能サポートを特定のビデオ・ストリーム記述タイプに合わせて調整することができる。

【００８３】

ｌｐＰｒｏｐＲａｎｇｅは、ＶｉｄｅｏＰｒｏｐｅｒｔｙパラメータ／フィールドによって指定されたＰｒｏｃＡｍｐ制御特性に関する範囲（最小および最大）、ステップサイズ、およびデフォルト値を特定する。

```
typedef struct_DXVA_VideoPropertyRange {
      FLOAT MinValue;
      FLOAT MaxValue;
      FLOAT DefaultValue;
      FLOAT StepSize;
} DXVA_VideoPropertyRange, *LPDXVA_VideoPropertyRange;
```

【００８４】

流れ図６００のブロック６１４で、ビデオ・レンダラー４１０は、オープンＰｒｏｃＡｍｐストリーム・オブジェクト・コマンドをグラフィックス・デバイス・ドライバ４２２に

送る。これに応答して、グラフィックス・デバイス・ドライバ４２２は、ブロック６２６でＰｒｏｃＡｍｐストリーム・オブジェクトを開く。ブロック６１６で、ビデオ・レンダラー４１０は、ＰｒｏｃＡｍｐ調整動作を実行するようグラフィックス・デバイス・ドライバ４２２に命令する。これに応答して、グラフィックス・デバイス・ドライバ４２２は、要求されたＰｒｏｃＡｍｐ調整動作をブロック６２８で実行する。

【００８５】

ブロック６１６で曲線の矢が示すように、ビデオ・レンダラー４１０は、所望の限りグラフィックス・デバイス・ドライバ４２２に対して実行ＰｒｏｃＡｍｐ調整動作命令を送り続けることができる（例えばビデオ・ストリームを表示する教唆アプリケーションによって要求されるときはいつでも）。ブロック６１８で、ビデオ・レンダラー４１０は、ＰｒｏｃＡｍｐストリーム・オブジェクトを閉じるようグラフィックス・デバイス・ドライバ４２２に命令する。グラフィックス・デバイス・ドライバ４２２は次いで、ブロック６３０でＰｒｏｃＡｍｐストリーム・オブジェクトを閉じる。

【００８６】

ブロック６１４、６１６、６１８、６２６、６２８および６３０の少なくとも一部を実装する典型的な汎用ＡＰＩは以下のように提供される。

【００８７】

ＰｒｏｃＡｍｐＳｔｒｅａｍオブジェクト

ビデオ・レンダラー４１０がＰｒｏｃＡｍｐ制御ハードウェアの能力を決定した後で、ＰｒｏｃＡｍｐＳｔｒｅａオブジェクトを作成することができる。ＰｒｏｃＡｍｐＳｔｒｅａｍオブジェクトを作成することによって、グラフィックス・デバイス・ドライバ４２２は、要求された１つ以上のＰｒｏｃＡｍｐ調整動作を実行するために必要なハードウェア資源を確保することができる。

【００８８】

ＰｒｏｃＡｍｐＯｐｅｎＳｔｒｅａｍ

ＰｒｏｃＡｍｐＯｐｅｎＳｔｒｅａｍメソッドはＰｒｏｃＡｍｐＳｔｒｅａｍオブジェクトを作成する。

```
IIRESULT
ProcAmpOpenStream(
      [in]LPDXVA_VideoDesc lpVideoDescription,
      [out] HDXVA_ProcAmpStream* lphCcStrm
);
```

【００８９】

ＨＤＸＶＡ＿ＰｒｏｃＡｍｐＳｔｒｅａｍ出力パラメータはＰｒｏｃＡｍｐＳｔｒｅａｍオブジェクトに対するハンドルであり、そこに向けられた将来のコール（call）でこのストリームを識別するために使用される。

【００９０】

ＰｒｏｃＡｍｐＢｌｔ

ＰｒｏｃＡｍｐＢｌｔメソッドは、ビット・ブロック転送動作中に宛先サーフェスにその出力を書き込むことによってＰｒｏｃＡｍｐ調整動作を実行する。

```
HRESULT
ProcAmpBlt(
      [in]HDXVA_ProcAmpStream hCcStrm
      [in]LPDDSURFACE lpDDSDstSurface,
      [in]LPDDSURFACE lpDDSSrcSurface,
      [in]DXVA_ProcAmpBlt* ccBlt
      );
```

【００９１】
ソース長方形と宛先長方形とは、サブ長方形ＰｒｏｃＡｍｐ調整または伸張のために使用される。伸張に対するサポートは、任意選択である（および、ＣａｐｓフラグによＵて報告される）。同様に、サブ長方形に対するサポートも強制的ではない。
【００９２】
宛先サーフェスは、オフスクリーン・プレーン・サーフェス、Ｄ３Ｄレンダー・ターゲット、Ｄ３Ｄテクスチャ、レンダー・ターゲットでもあるＤ３Ｄテクスチャなどであってよい。宛先サーフェスは、例えばローカル・ビデオ・メモリに割り付けることができる。ＹＵＶ－ＲＧＢ色空間変換がＰｒｏｃＡｍｐ調整動作と共に実行されていない限り、宛先サーフェスのピクセル・フォーマットはＤＸＶＡ＿ＰｒｏｃＡｍｐＣａｐｓ構造で示されるものである。これらの場合、宛先サーフェス・フォーマットは、各色成分に対して８ビット以上の精度のＲＧＢフォーマットである。
【００９３】
ＰｒｏｃＡｍｐＣｌｏｓｅＳｔｒｅａｍ
ＰｒｏｃＡｍｐＣｌｏｓｅＳｔｒｅａｍメソッドはＰｒｏｃＡｍｐＳｔｒｅａｍオブジェクトを閉じ、識別されたストリームに関連付けられたハードウェア資源を解放するように、グラフィックス・デバイス・ドライバ４２２に命令する。

```
HRESULT
ProcAmpCloseStream(
      HDXVA_ProcAmpStream hCcStrm
      );
```

【００９４】
典型的な専用ＡＰＩの実装
本節で後述する特定の状態および典型的ＡＰＩは、パーソナル・コンピュータ用の既存のＭｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システムのサブセットに特に適用可能である。しかし、後述する、擬似コードのある種の態様と同様に、この原理は、他のオペレーティング・システムおよび／または他の環境で（そのままで、またはルーチンを修正して）使用できることを理解されたい。
【００９５】
ＰｒｏｃＡｍｐインターフェースに対するＤＤＩマッピング
既存のＭｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システムのサブセットに対するＤＤＩインフラストラクチャとの互換性のために、前節で記載のＡＰＩはＤｉｒｅｃｔＤｒａｗおよびＤｉｒｅｃｔＸＶＡ用の既存のＤＤＩに「マッピング」することができる。本節では、既存のＤｉｒｅｃｔＤｒａｗおよびＤＸ－ＶＡ　ＤＤＩにマッピングするＰｒｏｃＡｍｐインターフェースについて説明する。
【００９６】
ＤＸ－ＶＡ　ＤＤＩは、それ自体が「ＤＸ－ＶＡコンテナ」と「ＤＸ－ＶＡデバイス」の

２つの機能グループに分割される。ＤＸ－ＶＡコンテナＤＤＩグループの目的は、表示ハードウェアに含まれる様々なＤＤＩ－ＶＡデバイスの数および能力を決定することである。したがって、ＤＸ－ＶＡドライバは単一のコンテナしか有することはできないが、複数のＤＸ－ＶＡデバイスをサポートすることができる。

【００９７】

ＰｒｏｃＡｍｐＱｕｅｒｙＣａｐｓ　ｃａｌｌ　ｏｎをＤＸ－ＶＡコンテナ・グループのＤＤＩエントリーポイントのどれにマッピングすることも実現可能でない。その理由は、他のＤＸ－ＶＡとは異なり、このコンテナ・メソッドが型付きパラメータを使用するからである。しかし、ＤＸ－ＶＡデバイスＤＤＩグループは型付きパラメータを使用しないので、このデバイス・グループ内のメソッドにＰｒｏｃＡｍｐ制御インターフェースをマッピングすることは実現可能である。本節では、ＰｒｏｃＡｍｐインターフェースをどのようにしてＤＸ－ＶＡデバイスＤＤＩにマッピングすることができるかの具体例を記載する。

【００９８】

デ・インターレース・コンテナ・デバイス

ＤＸ－ＶＡデバイス・メソッドは型付きパラメータを使用しないので、多くの異なる目的に再利用することができる。しかし、ＤＸ－ＶＡデバイス・メソッドはＤＸ－ＶＡデバイスとの関連でしか使用することができないので、第１のタスクは特別の「コンテナ・デバイス」を定義し、作成することである。

【００９９】

参照により本明細書に組み込んだ米国特許出願第１０/２７３,５０５号明細書「Methods And Apparatuses For Facilitating Processing Of Interlaced Video Images For Progressive Video Displays」には、デ・インターレース・コンテナ・デバイスについての記述が含まれている。本明細書では、この米国特許出願第１０/２７３,５０５号明細書に記載されているデ・インターレース・コンテナ・デバイスをＰｒｏｃＡｍｐＱｕｅｒｙＣａｐｓ機能のために再利用する。

【０１００】

ＤＸ－ＶＡデ・インターレース・コンテナ・デバイスはソフトウェア構成のみなので、物理的デバイスに含まれるどのような機能的ハードウェアをも示さない。下記のＰｒｏｃＡｍｐ制御サンプル（デバイス）ドライバ擬似コードは、ドライバがどのようにしてコンテナ・デバイスを実装することができるかを示している。

【０１０１】

ユーザ・モードの構成要素からＤＤＩを呼び出す

（ビデオ）レンダラーなどのユーザ・モードの構成要素からのＤＤＩを使用するための８タスクの典型的なシーケンスを以下に示す。

【０１０２】

１．ドライバがサポートするＤＸ－ＶＡデバイスのリストを入手するためにＧｅｔＭｏＣｏｍｐＧｕｉｄｓを呼び出す。

【０１０３】

２．「デ・インターレース・コンテナ・デバイス」ＧＵＩＤがある場合、このＤＸ－ＶＡデバイスのインスタンスを作成するためにＣｒｅａｔｅＭｏＣｏｍｐを呼び出す。このコンテナ・デバイスＧＵＩＤは以下のように定義される。

```
DEFINE_GUID(DXVA_DeinterlaceContainerDevice,
0x0e85cb93,0x3046,0x4ff0,0xae,0xcc,0xd5,0x8c,0xb5,0xf0,0x35,0xfc);
```

【０１０４】

３．ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＭｏｄｅＣａｐｓ動作を特定するｄｗＦｕｎｃｔｉｏｎパラメータを有するＲｅｎｄｅｒＭｏｃｏｍｐを呼び出す。ここでもまた、ドライバに入力パラメータを渡すためにｌｐＩｎｐｕｔＤａｔａパラメータが使用され、当該ドライバはｌｐＯｕｔｐｕｔＤａｔａパラメータによってその出力を戻す。

【０１０５】
４．ハードウェアがサポートするＰｒｏｃＡｍｐ調整特性ごとに、レンダラーはＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＲａｎｇｅ動作を特定するｄｗＦｕｎｃｔｉｏｎパラメータを有するＲｅｎｄｅｒＭｏｃｏｍｐを呼び出す。入力パラメータをドライバに渡すためにｌｐＩｎｐｕｔＤａｔａパラメータが使用され、当該ドライバはｌｐＯｕｔｐｕｔＤａｔａパラメータによってその出力を戻す。
【０１０６】
５．レンダラーは、ハードウェアのＰｒｏｃＡｍｐ調整能力の決定後、ＰｒｏｃＡｍｐ制御デバイスのインスタンスを作成するためにＣｒｅａｔｅＭｏｃｏｍｐを呼び出す。ＰｒｏｃＡｍｐ制御デバイスＧＵＩＤは以下のように定義される。

```
DEFINE_GUID(DXVA_ProcAmpControlDevice,
0x9f200913,0x2ffd,0x4056,0x9f,0x1e,0xe1,0xb5,0x08,0xf2,0x2d,0xcf);
```

【０１０７】
６．レンダラーは次いで、ＰｒｏｃＡｍｐ調整動作ごとにＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＢｌｔＦｎＣｏｄｅの機能パラメータを有する、ＰｒｏｃＡｍｐ制御デバイスの、ＲｅｎｄｅｒＭｏｃｏｍｐを呼び出す。
【０１０８】
７．レンダラーは、ＰｒｏｃＡｍｐ動作を実行する必要がなくなると、ＤｅｓｔｒｏｙＭｏｃｏｍｐを呼び出す。
【０１０９】
８．ドライバはＰｒｏｃＡｍｐ制御デバイスが使用したどの資源も解放する。
【０１１０】
ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＣａｐｓ
このメソッドは、デ・インターレース・コンテナ・デバイスのｃａｌｌ　ｔｏｔｈｅ　ＲｅｎｄｅｒＭｏＣｏｍｐメソッドに直接的にマッピングする。ＤＤ＿ＲＥＮＤＥＲＭＯＣＯＭＰＤＡＴＡ構造は以下のように完成される。
・ｄｗＮｕｍＢｕｆｆｅｒｓは０である。
・ｌｐＢｕｆｆｅｒＩｎｆｏはＮＵＬＬである。
・ｄｗＦｕｎｃｔｉｏｎは、ＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＣａｐｓＦｎＣｏｄｅと定義される。
・ｌｐＩｎｐｕｔＤａｔａはＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃ構造を指す。
・ｌｐＯｕｔｐｕｔＤａｔａはＤＸＶＡ＿ＰｒｏｃＡｍｐＣａｐｓ構造を指す。
【０１１１】
最初にＢｅｇｉｎＭｏＣｏｍｐＦｒａｍｅまたはＥｎｄＭｏＣｏｍｐＦｒａｍｅを呼び出さずに、ＤＸ－ＶＡコンテナ・デバイスのＲｅｎｄｅｒＭｏＣｏｍｐメソッドを、呼び出すことができることに留意されたい。
【０１１２】
ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＲａｎｇｅ
このメソッドは、デ・インターレース・コンテナ・デバイスのｃａｌｌ　ｔｏｔｈｅ　ＲｅｎｄｅｒＭｏＣｏｍｐメソッドに直接的にマッピングする。ＤＤ＿ＲＥＮＤＥＲＭＯＣＯＭＰＤＡＴＡ構造は以下のように完成される。
・ｄｗＮｕｍＢｕｆｆｅｒｓは０である。
・ｌｐＢｕｆｆｅｒＩｎｆｏはＮＵＬＬである。
・ｄｗＦｕｎｃｔｉｏｎは、ＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＲａｎｇｅＦｎＣｏｄｅと定義される。
・ｌｐＩｎｐｕｔＤａｔａは、ＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＱｕｅｒｙＲａｎｇｅ構造を指す。

```
typedef struct _DXVA_ProcAmpQueryRange {
    DWORD              Size;
    DWORD              VideoProperty;
    DXVA_VideoDesc      VideoDesc;
} DXVA_ProcAmpControlQueryRange,
*LpDXVA_ProcAmpControlQueryRange;
```

・ｌｐＯｕｔｐｕｔＤａｔａはＤＸＶＡ＿ＶｉｄｅｏＰｒｏｐｅｒｔｙＲａｎｇｅ構造を指すことになる。

【０１１３】

最初にＢｅｇｉｎＭｏＣｏｍｐＦｒａｍｅまたはＥｎｄＭｏＣｏｍｐＦｒａｍｅを呼び出さずに、ＤＸ－ＶＡコンテナ・デバイスのＲｅｎｄｅｒＭｏＣｏｍｐメソッドを、呼び出すことができることに留意されたい。

【０１１４】

ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＯｐｅｎＳｔｒｅａｍ

ＧＵＩＤがＰｒｏｃＡｍｐデバイスＧＵＩＤであり、ｐＵｎｃｏｍｐＤａｔａがデータを含まない構造（オール０）を指し、ｐＤａｔａがＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃ構造を指す場合、このメソッドは、ＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＣｒｅａｔｅＭｏＣｏｍｐメソッドに直接マッピングする。

【０１１５】

ドライバが圧縮されたビデオのアクセレートされた復号をサポートする場合、レンダラーは、２つのＤＸ－ＶＡデバイス、ＤｉｒｅｃｔＸ　ＶＡビデオの復号仕様で定義されているように実際のビデオ復号作業を実行するために一方を、ＰｒｏｃＡｍｐ調整専用に使用されるために他方を、を生成するためにそのドライバを呼び出すことができる。

【０１１６】

＊＊例：ＣｒｅａｔｅＭｏＣｏｍｐのＰｒｏｃＡｍｐＣｏｎｔｒｏｌＯｐｅｎＳｔｒｅａｍへのマッピング＊＊

下記の典型的な擬似コードは、ドライバがどのようにしてＣｒｅａｔｅＭｏＣｏｍｐ　ＤＤＩ　ｃａｌｌをｃａｌｌｓ　ｔｏ　ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＯｐｅｎＳｔｒｅａｍにマッピングすることができるかを示している。この擬似コードは、ＣｒｅａｔｅＭｏＣｏｍｐ機能がＰｒｏｃＡｍｐに対してどのように使用されるかを示している。ドライバが、ＭＰＥＧ－２ビデオ・ストリームの復号などの他のＤＸ－ＶＡ機能をサポートする場合、追加ＤＸ－ＶＡ　ＧＵＩＤの処理を含めるために下記のサンプル・コードを拡張することができる。

【０１１７】

```
  DWORD APIENTRY
  CreateMoComp(
    LPDDHAL_CREATEMOCOMPDATA lpData
    )
  {
    // Make sure its a guid we like.
    if (!ValidDXVAGuid(lpData->lpGuid)) {

      DbgLog((LOG_ERROR, 1,
           TEXT("No formats supported for this GUID")));

      lpData->ddRVal = E_INVALIDARG;
  return DDHAL_DRIVER_HANDLED;
}

// Look for the deinterlace container device GUID
if(*lpData->lpGuid == DXVA_DeinterlaceContainerDevice) {

  DXVA_DeinterlaceContainerDeviceClass* lpDev =
    new DXVA_DeinterlaceContainerDeviceClass(
        *lpData.->lpGuid,
        DXVA_DeviceContainer);

  if(lpDev) {
    lpData->ddRVal = DD_OK;
  }
  else {
    lpData->ddRVal = E_OUTOFMEMORY;
  }
```

```
  lpData->lpMoComp->lpDriverReserved1 =
    (LPVOID)(DXVA_DeviceBaseClass*)lpDev;
  return DDHAL_DRIVER_HANDLED;
}

// Look for the ProcAmp Control device GUID
if(*lpData->lpGuid == DXVA_ProcAmpControlDevice) {

  DXVA_ProcAmpControlDeviceClass* lpDev =
    new DXVA_ProcAmpControlDeviceClass(
        *lpData->lpGuid,
         DXVA_DeviceProcAmpControl);

    if(lpDev) {
      LPDXVA_VideoDesc lpVideoDescription =
         (LPDXVA_VideoDesc)lpData->lpData;
      lpData->ddRVal =
          lpDev->ProcAmpControlOpenStream(
            lpVideoDescription);

      if (lpData->ddRVal != DD_OK) {
        delete lpDev;
        lpDev = NULL;
      }
    }
    else {
      lpData->ddRVal = E_OUTOFMEMORY;
    }
```

```
    lpData->lpMoComp->lpDriverReserved1 =
      (LPVOID)(DXVA_DeviceBaseClass*)lpDev;
    return DDHAL_DRIVER_HANDLED;
  }
  lpData->ddRVal = DDERR_CURRENTLYNOTAVAIL;
  return DDHAL_DRIVER_HANDLED;
}
```

【０１１８】

＊＊例：ＧｅｔＭｏＣｏｍｐＧｕｉｄｓの実装＊＊

ＣｒｅａｔｅＭｏＣｏｍｐ　ＤＤＩ機能の他に、ドライバはＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＧｅｔＭｏＣｏｍｐＧｕｉｄｓメソッドを実装することもできる。下記の典型的な擬似コードは、この機能をドライバで実装する１つの方法を示している。

```
// This is a list of DV-VA device GUIDs supported by
// the driver - this list includes decoder, ProcAmp and
// the de-interlacing container device. There is no significance to
// the order of the GUIDs on the list.
DWORD g_dwDXVANumSupportedGUIDs = 2;
const (GUID* g_DXVASupportedGUIDs[2] = {
       &DXVA_DeinterlaceContainerDevice,
       &DXVA_ProcAmpControlDevice
};

DWORD APIENTRY
GetMoCompGuids(
      PDD_GETMOCOMPGUIDSDATA lpData
      )
{
      DWORD dwNumToCopy;

      // Check to see if this is a GUID request or a count request
     if (lpData->lpGuids) {
            dwNumToCopy =
min(g_dwDXVANumSupportedGUIDs,
              lpData->dwNumGuids);
        for (DWORD i =0; i < dwNumToCopy; i++) {
                lpData->lpGuids[i] =
 *g_DXVAsupportedGUIDs[i];
            }
      }
      else {
            dwNumToCopy = g_dwDXVANumSupportedGUIDs;
```

```
        }

        lpData->dwNumGuids = dwNumToCopy;
        lpData->ddRVal = DD_OK;
        return DDHAL_DRIVER_HANDLED;
    }
```

【０１１９】
ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＢｌｔ
このメソッドは、ＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＲｅｎｄｅｒＭｏＣｏｍｐメソッドに直接マッピングする。ここで、
・ｄｗＮｕｍＢｕｆｆｅｒｓは２である。
・ｌｐＢｕｆｆｅｒＩｎｆｏは２つのサーフェスの１つのアレイを指す。このアレイの第１の要素は宛先サーフェスであり、このアレイの第２の要素はソース・サーフェスである。
・ｄｗＦｕｎｃｔｉｏｎは、ＤＸＶＡ＿ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＢｌｔＦｎＣｏｄｅと定義される。
・ｌｐＩｎｐｕｔＤａｔａは下記の構造を指す。

```
    typedef struct_DXVA_ProcAmpControlBlt {
        DWORD       Size;
        RECT        DstRect;
        RECT        SrctRect;
        FLOAT       Alpha;
        FLOAT       Brightness;
        FLOAT       Contrast;
        FLOAT       Hue;
        FLOAT       Saturation;
    } DXVA_ProcAmpControlBlt;
```

・ｌｐＯｕｔｐｕｔＤａｔａはＮＵＬＬである。
【０１２０】
ＰｒｏｃＡｍｐに使用されるＤＸ－ＶＡデバイスに関して、ＢｅｇｉｎＭｏＣｏｍｐＦｒａｍｅまたはＥｎｄＭｏＣｏｍｐＦｒａｍｅを呼び出さずに、ＲｅｎｄｅｒＭｏＣｏｍｐを呼び出すことができることに留意されたい。
【０１２１】
＊＊例：ＲｅｎｄｅｒＭｏＣｏｍｐのＰｒｏｃＡｍｐＣｏｎｔｒｏｌＢｌｔへのマッピング＊＊
下記の典型的な擬似コードは、ドライバがどのようにしてＲｅｎｄｅｒＭｏＣｏｍｐ　ＤＤＩ　ｃａｌｌをｃａｌｌｓ　ｔｏ　ＰｒｏｃＡｍｐＢｌｔにマッピングすることができるかを示している。サンプル・コードは、ＰｒｏｃＡｍｐ調整のためにＲｅｎｄｅｒＭｏＣｏｍｐ機能をどのように使用することができるかを示している。ドライバが、ＭＰＥＧ－２ビデオ・ストリームの復号などの他のＤＸ－ＶＡ機能をサポートしている場合、下記のサンプル・コードを拡張して、追加ＤＸ－ＶＡ　ＧＵＩＤの処理を含めることができる。

【０１２２】

```
DWORD APIENTRY
RenderMoComp(
  LPDDHAL_RENDERMOCOMPDATA lpData
  )
{
  if (lpData->dwFunction = = DXVA_ProcAmpControlBltFnCode)
  {
    DXVA_ProcAmpControlDeviceClass* pDXVADev =
          (DXVA_ProcAmpControlDeviceClass*)pDXVABase;

    DXVA_ProcAmpControlBlt* lpBlt =
          (DXVA_ProcAmpControlBlt*)lpData->lpInputData;

    LPDDMCBUFFERINFO lpBuffInfo = lpData->lpBufferInfo;

    lpData->ddRVal = pDXVADev->ProcAmpControlBlt(
          lpBuffInfo[0].lpCompSurface,
          lpBuffInfo[1].lpCompSurface,
          lpBlt);
    return DDHAL_DRIVER_HANDLED;
  }
  lpData->ddRVal = E_INVALIDARG;
  return DDHAL_DRIVER_HANDLED;
}
```

【０１２３】

ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＣｌｏｓｅＳｔｒｅａｍ

このメソッドは、ＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＤｅｓｔｒｏｙＭｏＣｏｍｐメソッドに直接マッピングする。

【０１２４】

＊＊例：ＤｅｓｔｒｏｙＭｏＣｏｍｐのＰｒｏｃＡｍｐＣｏｎｔｒｏｌＣｌｏｓｅＳｔｒｅａｍへのマッピング＊＊

下記の典型的な擬似コードは、ドライバがどのようにしてＤｅｓｔｒｏｙＭｏＣｏｍｐＤＤＩ　ｃａｌｌをｃａｌｌｓ　ｔｏ　ＰｒｏｃＡｍｐＣｏｎｔｒｏｌＣｌｏｓｅＳｔｒｅａｍにマッピングすることができるかを示している。サンプル・コードは、ＰｒｏｃＡｍｐ調整のためにＤｅｓｔｒｏｙＭｏＣｏｍｐ機能をどのように使用することができるかを示している。ドライバが、ＭＰＥＧ－２ビデオ・ストリームの復号などの他のＤＸ－ＶＡ機能をサポートしている場合、下記のサンプル・コードを拡張して、追加ＤＸ－ＶＡＧＵＩＤの処理を含めることができる。

【０１２５】

```
DWORD APIENTRY
DestroyMoComp(
  LPDDHAL_DESTROYMOCOMPDATA lpData
  )
{
  DXVA_DeviceBaseClass* pDXVABase =
(DXVA_DeviceBaseClass*)
    lpData->lpMoComp->lpDriverReserved 1;

  if(pDXVABase == NULL) {
    lpData->ddRVal = E_POINTER;
    return DDHAL_DRIVER_HANDLED;
  }

  switch (pDXVABase->m_DeviceType) {
  case DXVA_DeviceContainer:
    lpData->ddRVal = S_OK;
    delete pDXVABase;
    break;

  case DXVA_DeviceProcAmpControl:
    {
       DXVA_ProcAmpControlDeviceClass* pDXVADev =
           (DXVA_ProcAmpControlDeviceClass*)pDXVABase;
       lpData->ddRVal = pDXVADev->ProcAmpControlCloseStream();
       delete pDXVADev;
    }
            break;
          }

          return DDHAL_DRIVER_HANDLED;
        }
```

【０１２６】

コンピュータまたは他の電子デバイスの典型的な動作環境

図７は、本明細書に記載のようにビデオ・レンダラーとグラフィックス・デバイス・ドラ

イバの間の相互作用を促進するための少なくとも１つのシステム、デバイス、構成要素、構成、プロトコル、方法、メソッド、プロセス、これらのいくつかの組合せなどで（全体的にまたは部分的に）実装することができる典型的なコンピューティング（または一般的な電子デバイス）の動作環境７００を示している。コンピューティング環境７００は、後述するコンピュータおよびネットワーク・アーキテクチャでまたはスタンドアロン状況で使用することができる。

【０１２７】

典型的な電子デバイスの動作環境７００は、環境の一例に過ぎず、適用可能な電子（コンピュータ、ゲーム・コンソール、テレビジョンなどを含めて）アーキテクチャの用途または機能の範囲に関してどのような限定をも意図しない。また、電子デバイス環境７００は図７に示すどの構成要素またはこれら構成要素のどのような組合せに関するいかなる依存または要件を有するものと理解されるべきではない。

【０１２８】

さらに、ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進することは、多数の他の汎用または専用電子デバイス（コンピューティング・システムを含めて）環境または構成によって実装することができる。使用に適したものとなることのできる周知の電子（デバイス）システム、環境および／または構成の実装例は、限定的ではないが、パーソナル・コンピュータ、サーバ・コンピュータ、シン・クライアント、シック（thick）・クライアント、パーソナル・デジタル・アシスタント（ＰＤＡ）、または携帯電話、ハンドヘルドまたはラップトップ・デバイス、マルチ・プロセッサ・システム、マイクロ・プロセッサ・ベースのシステム、セットトップボックス、プログラム可能な家庭用電化製品、ビデオ・ゲームマシン、ゲーム・コンソール、携帯用またはハンドヘルドのゲーム装置、ネットワークＰＣ、ミニ・コンピュータ、メーンフレーム・コンピュータ、上記システムまたはデバイスのどれかを含む分散型コンピューティング環境、これらのいくつかの組合せなどを含む。

【０１２９】

ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進するための実装は、電子的に実行可能な命令の一般的状況において説明することができる。一般に、電子的に実行可能な命令には、特定のタスクを実行するかまたは特定の抽象データ型を実装するルーチン、プログラム、オブジェクト、構成要素、データ構造などが含まれる。本明細書のある種の実装に記載されているようにビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進することは、通信リンクおよび／またはネットワークによって接続されている遠隔にリンクされた処理装置によってタスクが実行される分散型コンピューティング環境で実行することもできる。特に分散型コンピューティング環境では、電子的に実行可能な命令は、別個の記憶媒体に配置し、異なるプロセッサによって実行し、かつ／または伝送媒体を介して伝播することができる。

【０１３０】

電子デバイス環境７００は、計算能力および／または処理能力機能を有するどのような電子デバイスをも含むことができるコンピュータ７０２形式の汎用コンピューティング・デバイスを含む。コンピュータ７０２の構成要素は、限定的なものではないが、１つまたは複数のプロセッサまたは処理装置７０４、システム・メモリ７０６、およびプロセッサ７０４を含めて様々なシステム・コンポーネントをシステム・メモリ７０６に結合するシステム・バス７０８を含むことができる。

【０１３１】

システム・バス７０８は、メモリ・バスまたはメモリ・コントローラ、周辺バス、加速グラフィックス・ポート、各種バス・アーキテクチャのどれかを使用するプロセッサまたはローカル・バスを含めて有線または無線バス構造のいくつかのタイプのどれか１つまたは複数を示している。一例として、このようなアーキテクチャは、業界標準アーキテクチャ（ＩＳＡ）バス、マイクロ・チャネル・アーキテクチャ（ＭＣＡ）バス、拡張ＩＳＡ（ＥＩＳＡ）バス、ビデオ電子標準協会（ＶＥＳＡ）ローカル・バス、メザニン・バスとして

も知られる周辺装置相互接続（ＰＣＩ）バス、これらのいくつかの組合せなどを含むことができる。

【０１３２】

コンピュータ７０２は、通常、各種の電子的にアクセス可能な媒体を含む。このような媒体は、コンピュータ７０２または他の電子デバイスによってアクセス可能などのような使用可能な媒体でもあってよく、揮発性媒体と不揮発性媒体、取り外し可能媒体と取り外し不可能媒体、および記憶媒体と伝送媒体を含む。

【０１３３】

システム・メモリ７０６は、ランダム・アクセス・メモリ（ＲＡＭ）７１０のような揮発性メモリおよび／または読み取り専用メモリ（ＲＯＭ）７１２のような不揮発性メモリの形式の、電子的にアクセス可能な記憶媒体を含む。起動時などにコンピュータ７０２内の素子間での情報転送を支援する基本ルーチンを含んでいる基本入出力システム（ＢＩＯＳ）７１４は、通常はＲＯＭ７１２に記憶される。ＲＡＭ７１０は、通常、処理装置７０４によって直接的にアクセス可能、かつ／または現行で操作されているデータおよび／またはプログラム・モジュール／命令を収容している。

【０１３４】

コンピュータ７０２はまた、他の取り外し可能／取り外し不可能な、かつ／または揮発性／不揮発性の記憶媒体を含む。一例として、図７は、（典型的な）取り外し不可能で不揮発性の磁気媒体（単独では図示せず）から読み取り、そこに書き込むためのハードディスク・ドライブまたはディスク・アレイ７１６と、（典型的な）取り外し可能で不揮発性の磁気ディスク７２０（例えば「フロッピー（登録商標）・ディスク」）から読み取り、そこに書き込むための磁気ディスク・ドライブ７１８と、ＣＤ－ＲＯＭ、ＤＶＤまたは他の光学媒体などの（典型的な）取り外し可能で不揮発性の光ディスク７２４から読み取り、かつ／またはそこに書き込むための光ディスク・ドライブ７２２を示している。ハードディスク・ドライブ７１６、磁気ディスク・ドライブ７１８および光ディスク・ドライブ７２２は、１つまたは複数の記憶媒体インターフェース７２６によってそれぞれにシステム・バス７０８に接続されている。別法として、ハードディスク・ドライブ７１６と、磁気ディスク・ドライブ７１８と、光ディスク・ドライブ７２２は、１つまたは複数の他の単独の、または結合されたインターフェース（図示せず）によってシステム・バス７０８に接続することができる。

【０１３５】

ディスク・ドライブおよびその関連付けられた電子的にアクセス可能な媒体は、データ構造、プログラム・モジュール、およびコンピュータ７０２に対する他のデータなどの電子的に実行可能な命令の不揮発性記憶を提供する。典型的なコンピュータ７０２は、ハードディスク７１６、取り外し可能な磁気ディスク７２０、および取り外し可能な光ディスク７２４を示しているが、他のタイプの電子的にアクセス可能な媒体が、磁気カセットまたは他の磁気記憶媒体、フラッシュ・メモリ、ＣＤ－ＲＯＭ、デジタル多用途ディスク（ＤＶＤ）または他の光記憶装置、ＲＡＭ、ＲＯＭ、電子的消去書込み可能読み取り専用メモリ（ＥＥＰＲＯＭ）などの電子デバイスによってアクセス可能な命令を格納することができることが理解されよう。そのような媒体は、いわゆる専用の、または配線された集積回路（ＩＣ）チップを含むこともできる。すなわち、典型的な電子システムおよび環境７００の記憶媒体を実現するために、どのような電子的にアクセス可能な媒体でも使用することができる。

【０１３６】

一般例としてオペレーティング・システム７２８、１つまたは複数のアプリケーション・プログラム７３０、他のプログラム・モジュール７３２、およびプログラム・データ７３４を含めてプログラム・モジュール（または命令の他の単位またはセット）はいくつでも、ハードディスク７１６、磁気ディスク７２０、光ディスク７２４、ＲＯＭ７１２および／またはＲＡＭ７１０に記憶することができる。限定ではないが一般例として、ビデオ・レンダラー４１０、グラフィック・インターフェース４１２およびデバイス・ドライバ・

インターフェース４１４（すべて図４に示す）はオペレーティング・システム７２８の一部であってよい。グラフィックス・デバイス・ドライバ４２２は、オペレーティング・システム７２８と密接に連結し、かつ／または一体型の関係を任意選択で有するプログラム・モジュール７３２の一部であってよい。同様に、Ｗｉｎｄｏｗｓ（登録商標）Ｍｅｄｉａ（登録商標）９などの教唆プログラムはアプリケーション・プログラム７３０の一例である。現在システム・メモリにある画像制御および／またはグラフィックス・データはプログラム・データ７３４の一部であってよい。

【０１３７】

例えばＰｒｏｃＡｍｐまたは他のビデオ設定を変更中のユーザは、キーボード７３６およびポインティング・デバイス７３８（例えば「マウス」）などの入力デバイスによってコンピュータ７０２にコマンドおよび／または情報を入力することができる。他の入力デバイス７４０（具体的には図示せず）は、マイクロフォン、ジョイスティック、ゲーム・パッド、衛星放送受信アンテナ、シリアル・ポート、スキャナなどを含むことができる。これらおよび他の入力デバイスは、システム・バス７０８に結合されている入出力インターフェース７４２を介して処理装置７０４に接続されている。しかし、これらおよび／または出力デバイスは、代わりにパラレル・ポート、ゲーム・ポート、ユニバーサル・シリアル・バス（ＵＳＢ）ポート、ＩＥＥＥ１３９４（「Ｆｉｒｅｗｉｒｅ」）インターフェース、ＩＥＥＥ８０２．１１無線インターフェース、Ｂｌｕｅｔｏｏｔｈ（登録商標）無線インターフェースなどの他のインターフェースおよびバス構造によって接続することができる。

【０１３８】

モニタ／ビュー・スクリーン７４４（図４の表示装置４３６の一例）または他のタイプの表示装置を、ビデオ・アダプタ７４６などのインターフェースを介してシステム・バス７０８に接続することもできる。ビデオ・アダプタ７４６（または他の構成要素）は、グラフィックス中心の計算を処理するため、また要求の厳しい表示要件を取り扱うためにグラフィックス・カード（グラフィックス・デバイス４２４の一例である）であっても、またはこれを含んでいてもよい。典型的には、グラフィックス・カードは、図形操作の迅速な性能を促進するために、（ＧＰＵ４２６などの）ＧＰＵ、ビデオＲＡＭ（ＶＲＡＭ）（ビデオ・メモリ４３２の一例である）などを含む。モニタ７４４の他に、他の出力周辺装置は、入出力インターフェース７４２を介してコンピュータ７０２に接続することができるスピーカ（図示せず）およびプリンタ７４８などの構成要素を含むことができる。

【０１３９】

コンピュータ７０２は、遠隔コンピューティング・デバイス７５０などの１つまたは複数の遠隔コンピュータへの論理接続を使用してネットワーク接続された環境で動作することができる。一例として、遠隔コンピューティング・デバイス７５０は、パーソナル・コンピュータ、ポータブル・コンピュータ（例えばラップトップ・コンピュータ、タブレット・コンピュータ、ＰＤＡ、移動局など）、パーム・サイズまたはポケット・サイズのコンピュータ、ゲーム装置、サーバ、ルータ、ネットワーク・コンピュータ、ピア・デバイス、この他の共通ネットワーク・ノードまたは上記に列挙した以外のタイプのコンピュータなどであってよい。しかし、遠隔コンピューティング・デバイス７５０は、コンピュータ７０２に関して本明細書に記載の要素および機能の多くまたはすべてを含むことができるポータブル・コンピュータとして図示されている。

【０１４０】

コンピュータ７０２と遠隔コンピュータ７５０の間の論理接続はローカル・エリア・ネットワーク（ＬＡＮ）７５２および一般的なワイド・エリア・ネットワーク（ＷＡＮ）７５４として図示されている。このようなネットワーク接続環境は、オフィス、全社的なコンピュータ・ネットワーク、イントラネット、インターネット、固定電話網および携帯電話網、この他の無線ネットワーク、ゲーム用ネットワーク、これらのいくつかの組合せなどで普及している。

【０１４１】

ＬＡＮネットワーク接続環境で実装される際、コンピュータ７０２は一般にネットワーク・インターフェースまたはアダプタ７５６を介してＬＡＮ７５２に接続される。ＷＡＮネットワーク接続環境で実装される際、コンピュータ７０２は、通常、ＷＡＮ７５４を介して通信を確立するためのモデム７５８または他の手段を含む。コンピュータ７０２内蔵型であっても外付けであってもよいモデム７５８は、入出力インターフェース７４２または他の１つ以上の適切な機構を介してシステム・バス７０８に接続することができる。図示したネットワーク接続は一例であって、コンピュータ７０２と７５０の間で１つ以上の通信リンクを確立する他の手段を使用することができることが理解されよう。

【０１４２】

電子デバイス環境７００などで示されているネットワーク接続された環境では、コンピュータ７０２に関連して図示されているプログラム・モジュールまたは他の命令、またはそれらの一部は、その全体または一部を遠隔記憶装置に記憶することができる。一例として、遠隔アプリケーション・プログラム７６０は遠隔コンピュータ７５０のメモリ・コンポーネントに常駐しているが、コンピュータ７０２を介して使用可能あるいはアクセス可能であってもよい。同様に、説明の目的から、アプリケーション・プログラム７３０およびオペレーティング・システム７２８などの他の電子的に実行可能な命令を本明細書では別個のブロックとして図示したが、このようなプログラム、構成要素、および他の命令は様々な時点でコンピューティング・デバイス７０２（および／または遠隔コンピューティング・デバイス７５０）の異なる記憶装置に常駐し、コンピュータ７０２（および／または遠隔コンピューティング・デバイス７５０）の１つ以上のデータ・プロセッサ７０４によって実行されることが理解されよう。

【０１４３】

システム、媒体、メソッド、プロトコル、方法、プロセス、構成、および他の実装を、構造、論理、アルゴリズム、および機能の特徴に特有の言語および／または図面で説明したが、首記の特許請求の範囲で規定される本発明は、ここで説明した特定の特徴または図面に必ずしも限定されるものではないということを理解されたい。限定されるものではなく、この特定の特徴および図面は、請求する本発明を実装する典型的な形態として開示されているものである。

【図面の簡単な説明】

【図１】ＰｒｏｃＡｍｐ調整動作を含む第１のビデオ処理パイプラインを示す図である。

【図２】ＲＧＢレンダー・ターゲットに達するための２つのビデオ処理動作を含む第２のビデオ処理パイプラインを示す図である。

【図３】ＲＧＢレンダー・ターゲットに達するための１つのビデオ処理動作を含む第３のビデオ処理パイプラインを示す図である。

【図４】ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進するように構成されたコンピューティング・デバイスまたは他の電子デバイスのある種の機能要素を示すブロック図である。

【図５】ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の典型的なプロトコルを示す通信／信号送受の図である。

【図６】ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進する典型的なメソッドを示す流れ図である。

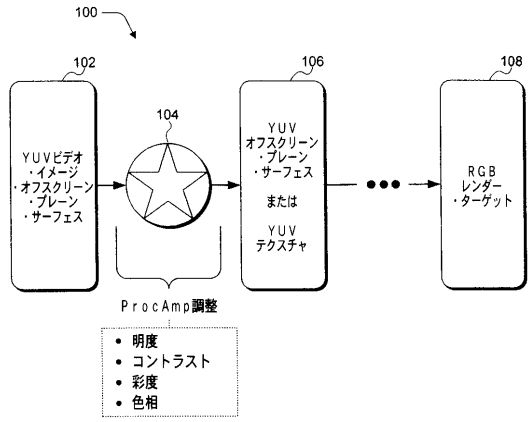
【図７】本明細書に記載のように、ビデオ・レンダラーとグラフィックス・デバイス・ドライバの間の相互作用を促進する少なくとも１つの態様で（全体的にまたは部分的に）実装することのできる典型的なコンピューティング（または一般的な電子デバイス）動作環境を示す図である。

【符号の説明】

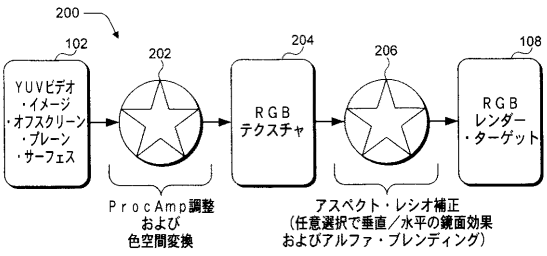
４１０　ビデオ・レンダラー
４１２　グラフィックス・インターフェース
４１４　デバイス・ドライバ・インターフェース
４２２　グラフィックス・デバイス・ドライバ

４２６　グラフィックス・プロセッサ・ユニット（ＧＰＵ）
４２４　グラフィックス・デバイス
４３２　ビデオ・メモリ

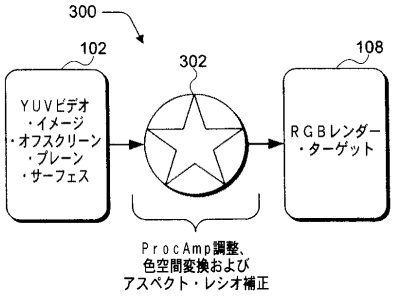
【図１】



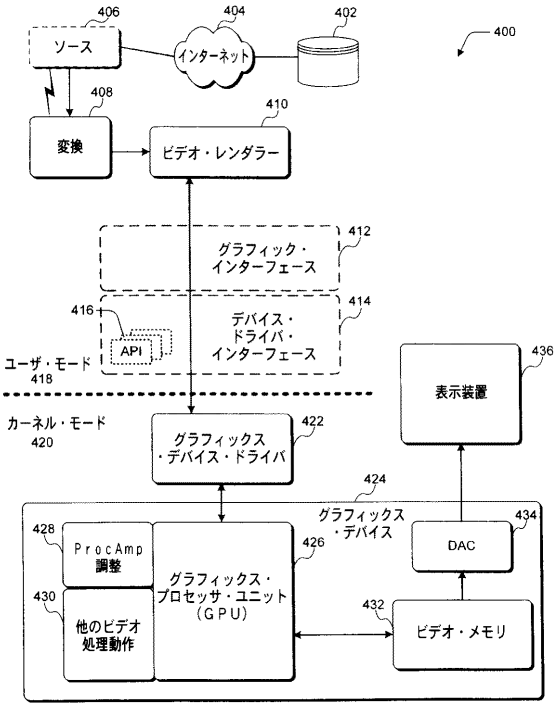
【図２】



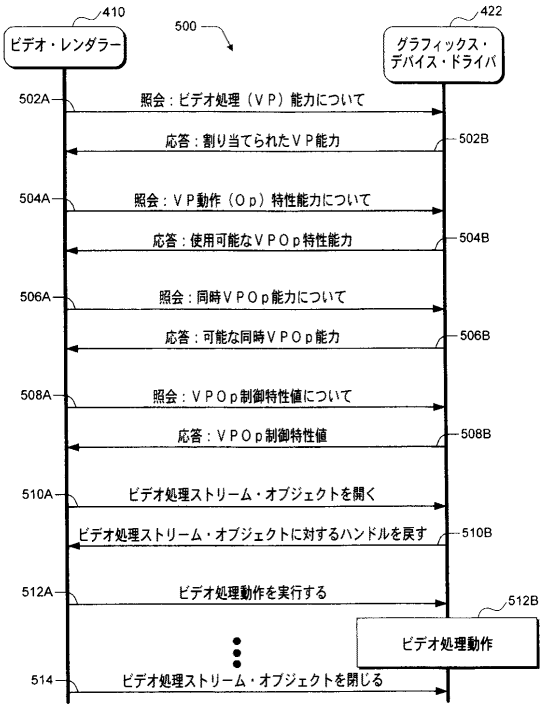
【図３】



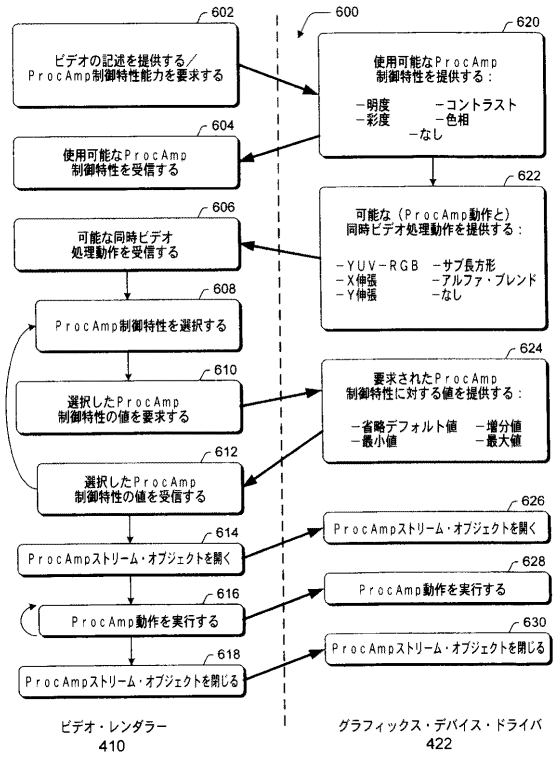
【図４】


400
406
ソース
404
インターネット
402
408
変換
410
ビデオ・レンダラー
412
グラフィック・
インターフェース
414
デバイス・
ドライバ・
インターフェース
416
API
ユーザ・モード
418
カーネル・モード
420
436
表示装置
422
グラフィックス
・デバイス・ドライバ
424
グラフィックス
・デバイス
428
ProcAmp
調整
426
グラフィックス・
プロセッサ・ユニット
（GPU）
430
他のビデオ
処理動作
DAC
434
432
ビデオ・メモリ

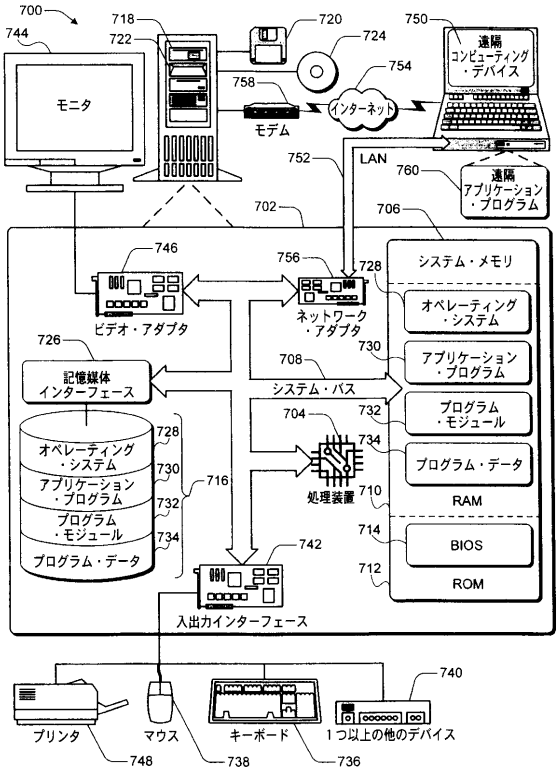
【図５】


410
ビデオ・レンダラー
500
422
グラフィックス・
デバイス・ドライバ
502A
照会：ビデオ処理（ＶＰ）能力について
応答：割り当てられたＶＰ能力
502B
504A
照会：ＶＰ動作（Ｏｐ）特性能力について
応答：使用可能なＶＰＯｐ特性能力
504B
506A
照会：同時ＶＰＯｐ能力について
応答：可能な同時ＶＰＯｐ能力
506B
508A
照会：ＶＰＯｐ制御特性値について
応答：ＶＰＯｐ制御特性値
508B
510A
ビデオ処理ストリーム・オブジェクトを開く
ビデオ処理ストリーム・オブジェクトに対するハンドルを戻す
510B
512A
ビデオ処理動作を実行する
512B
ビデオ処理動作
514
ビデオ処理ストリーム・オブジェクトを閉じる

【図６】


602
ビデオの記述を提供する／
ProcAmp制御特性能力を要求する
600
620
使用可能なProcAmp
制御特性を提供する：
－明度 －コントラスト
－彩度 －色相
－なし
604
使用可能なProcAmp
制御特性を受信する
622
可能な（ProcAmp動作と）
同時ビデオ処理動作を提供する：
－YUV－RGB －サブ長方形
－X伸張 －アルファ・ブレンド
－Y伸張 －なし
606
可能な同時ビデオ
処理動作を受信する
608
ProcAmp制御特性を選択する
624
要求されたProcAmp
制御特性に対する値を提供する：
－省略デフォルト値 －増分値
－最小値 －最大値
610
選択したProcAmp
制御特性の値を要求する
612
選択したProcAmp
制御特性の値を受信する
626
ProcAmpストリーム・オブジェクトを開く
614
ProcAmpストリーム・オブジェクトを開く
628
ProcAmp動作を実行する
616
ProcAmp動作を実行する
630
ProcAmpストリーム・オブジェクトを閉じる
618
ProcAmpストリーム・オブジェクトを閉じる
ビデオ・レンダラー
410
グラフィックス・デバイス・ドライバ
422

【図７】


700
744
モニタ
718
722
720
724
754
758
モデム
インターネット
750
遠隔
コンピューティング
・デバイス
752
LAN
760
遠隔
アプリケーション
・プログラム
702
706
システム・メモリ
746
ビデオ・アダプタ
756
728
ネットワーク
・アダプタ
オペレーティング
・システム
726
記憶媒体
インターフェース
730
アプリケーション
・プログラム
708
システム・バス
704
732
プログラム
・モジュール
734
プログラム・データ
処理装置
710
RAM
714
BIOS
712
ROM
オペレーティング
・システム
728
アプリケーション
・プログラム
730
プログラム
・モジュール
732
プログラム・データ
734
716
742
入出力インターフェース
740
プリンタ
748
マウス
738
キーボード
736
１つ以上の他のデバイス

フロントページの続き

審査官　居島　一仁

(56)参考文献　特開平０９－００６５７２（ＪＰ，Ａ）
特開平０６－０２８１１７（ＪＰ，Ａ）
米国特許第０５７１５４５９（ＵＳ，Ａ）
特開２００４－０２９７４４（ＪＰ，Ａ）
特表２００２－５１７８５５（ＪＰ，Ａ）
国際公開第９９／０６４９５２（WO，Ａ１）
特開平１１－１７５２９４（ＪＰ，Ａ）
米国特許第０６３２３８７５（ＵＳ，Ｂ１）
特開平１０－２７５０７２（ＪＰ，Ａ）
特開平１１－１８４６４９（ＪＰ，Ａ）
特開２００１－０８４２４６（ＪＰ，Ａ）
特開２０００－２９３６０８（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G09G3/00-5/42