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

特許第5123282号
(P5123282)

(45)発行日　平成25年1月23日(2013.1.23)　(24)登録日　平成24年11月2日(2012.11.2)

(51)Int.Cl.			F I		
H04N	7/01	(2006.01)	H04N	7/01	G
G09G	5/00	(2006.01)	H04N	7/01	Z
G09G	5/391	(2006.01)	G09G	5/00	520V
G09G	3/20	(2006.01)	G09G	3/20	650E
			G09G	3/20	650J

請求項の数 36　(全 37 頁)

(21)出願番号　特願2009-285191 (P2009-285191)
(22)出願日　平成21年12月16日 (2009.12.16)
(62)分割の表示　特願2003-110878 (P2003-110878)の分割
原出願日　平成15年4月15日 (2003.4.15)
(65)公開番号　特開2010-68544 (P2010-68544A)
(43)公開日　平成22年3月25日 (2010.3.25)
審査請求日　平成21年12月16日 (2009.12.16)
(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　10/273,505
(32)優先日　平成14年10月18日 (2002.10.18)
(33)優先権主張国　米国(US)

(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)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(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

審査官　益戸　宏

最終頁に続く

(54)【発明の名称】プログレッシブ・ビデオ表示のためのインターレース・ビデオ画像の処理を容易にする方法および装置

(57)【特許請求の範囲】
【請求項１】
グラフィックス・デバイス・ドライバに関連付けられたグラフィックス・デバイスによって処理されるべきインターレースされたビデオ・データの記述を、前記グラフィックス・デバイス・ドライバに識別させること、をレンダラーにさせること、
前記グラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を、前記レンダラーに識別させることを、前記グラフィックス・デバイス・ドライバにさせること、
前記レンダラーに、少なくとも１つの識別されたグラフィックス処理能力を選択すること、かつ前記少なくとも１つの識別されたグラフィックス処理能力に対する入力要件を前記グラフィックス・デバイス・ドライバに要求することをさせること、および、
前記少なくとも１つの識別されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を、前記レンダラーに対して識別することを、前記グラフィックス・デバイス・ドライバにさせること
を備えることを特徴とする方法。
【請求項２】
レンダリング・ロジックと、
グラフィックス・デバイス・ロジックと、
前記レンダリング・ロジックと前記グラフィックス・デバイス・ロジックとに動作可能に結合されるインターフェース・ロジックと

を備える装置であって、
前記インターフェース・ロジックを使用することによって、
前記レンダリング・ロジックは、前記グラフィックス・デバイス・ロジックに関連付けられたグラフィックス・デバイスによって処理されるべきインターレースされたビデオ・データの記述を前記グラフィックス・デバイスへ提供し、
前記グラフィックス・デバイス・ロジックは、前記グラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を識別し、
前記レンダリング・ロジックは、少なくとも１つの選択されたグラフィックス処理能力に対する入力要件を前記グラフィックス・デバイス・ロジックに要求し、および、
前記グラフィックス・デバイス・ロジックは、前記少なくとも１つの選択されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を識別する
ことを特徴とする装置。
【請求項３】
前記インターフェース・ロジックは、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）を含むことを特徴とする請求項２に記載の装置。
【請求項４】
グラフィックス・デバイス・ドライバに関連付けられたグラフィックス・デバイスによって処理されるべきインターレースされたビデオ・データの記述を前記グラフィックス・デバイス・ドライバに対してレンダラーが識別すること、
前記グラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を前記レンダラーに対して前記グラフィックス・デバイス・ドライバが識別すること、
前記レンダラーが、少なくとも１つの識別されたグラフィックス処理能力を選択し、かつ前記少なくとも１つの識別されたグラフィックス処理能力に対する入力要件を前記グラフィックス・デバイス・ドライバに要求すること、および、
前記少なくとも１つの識別されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を前記レンダラーに対して前記グラフィックス・デバイス・ドライバが識別すること
を可能にするインターフェース能力を提供することを備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行可能命令を有することを特徴とするコンピュータ可読記録媒体。
【請求項５】
前記インターフェース能力は、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）を含むことを特徴とする請求項４に記載のコンピュータ可読記録媒体。
【請求項６】
レンダラーとグラフィックス・デバイス・ドライバをインターフェースする方法であって、
ビデオ・データをインターレース解除する際に、グラフィックス・デバイス・ドライバに、関連付けられたグラフィックス・デバイスによって実行することができる少なくとも１つのグラフィックス処理能力に関して、レンダラーが照会することを可能にし、
前記少なくとも１つのグラフィックス処理能力を前記レンダラーに対して識別することによって、前記グラフィックス・デバイス・ドライバが前記照会に応答することを可能にし、
前記識別された少なくとも１つのグラフィックス処理能力に関連付けられた少なくとも１つの入力要件に関して、前記レンダラーが前記グラフィックス・デバイス・ドライバに追加照会することを可能にし、
前記グラフィックス処理能力に関連付けられた前記少なくとも１つの入力要件を前記レンダラーに対して識別することによって、前記グラフィックス・デバイス・ドライバが前記追加照会に応答することを可能にする

ように構成可能な少なくとも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）を提供すること
を備えることを特徴とする方法。
【請求項７】
レンダリング・ロジックと、グラフィックス・デバイス・ドライバ・ロジックと、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）とを提供するように動作可能に構成された少なくとも１つの処理ユニットを備える装置であって、
前記レンダリング・ロジックは、少なくとも１つの処理ユニットに動作可能に結合されている場合、ビデオ・データをインターレース解除する際に、少なくとも１つのＡＰＩを介して前記グラフィックス・デバイス・ドライバ・ロジックに照会し、関連付けられたグラフィックス・デバイスによって実行することができる少なくとも１つのグラフィックス処理能力を決定し、
グラフィックス・デバイス・ドライバ・ロジックは、前記少なくとも１つのＡＰＩを介して、前記少なくとも１つのグラフィックス処理能力を前記レンダリング・ロジックに対して識別して応答し、
レンダリング・ロジックは、前記識別された少なくとも１つのグラフィックス処理能力に関連付けられた少なくとも１つの入力要件に関して、前記少なくとも１つのＡＰＩを介して前記グラフィックス・デバイス・ドライバ・ロジックに追加照会し、
前記グラフィックス・デバイス・ドライバ・ロジックは、前記少なくとも１つのＡＰＩを介して、前記グラフィックス処理能力に関連付けられた前記少なくとも１つの入力要件を前記レンダリング・ロジックに対して識別して応答する
ように構成されていることを特徴とする装置。
【請求項８】
インターフェース環境を確立することを備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行可能命令を有するコンピュータ可読媒体であって、前記インターフェース環境を確立する際に、
ビデオ・データをインターレース解除する際に、グラフィックス・デバイス・ドライバ・ロジックに、関連付けられたグラフィックス・デバイスによって実行することができる少なくとも１つのグラフィックス処理能力に関して、レンダリング・ロジックが照会することを可能とし、
前記少なくとも１つのグラフィックス処理能力を前記レンダリング・ロジックに対して識別することによって前記グラフィックス・デバイス・ドライバ・ロジックが前記照会に応答することを可能とし、
前記レンダリング・ロジックは、前記識別された少なくとも１つのグラフィックス処理能力に関連付けられた少なくとも１つの入力要件に関して、前記グラフィックス・デバイス・ドライバ・ロジックに追加照会することを可能とし、
前記グラフィックス処理能力に関連付けられた前記少なくとも１つの入力要件を前記レンダリング・ロジックに対して識別することによって、前記グラフィックス・デバイス・ドライバ・ロジックが前記追加照会に応答することを可能とする
ことを特徴とするコンピュータ可読記録媒体。
【請求項９】
前記インターフェース環境は少なくとも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）を使用して動作可能に提供されることを特徴とする請求項８に記載のコンピュータ可読記録媒体。
【請求項１０】
ビデオ・データのインターレース解除をサポートするように構成されたレンダリング・ロジックと、
グラフィックス・デバイス・ドライバ・ロジックと、および、
インターフェース・ロジックと
を備えた装置であって、前記インターフェース・ロジックは、

前記グラフィックス・デバイス・ドライバ・ロジックに関連付けられ、前記ビデオ・データのインターレース解除をサポートするように構成されている、グラフィックス・デバイスによって実行することができるグラフィックス処理能力を要求する照会を前記レンダリング・ロジックから受け取り、
前記照会を前記グラフィックス・デバイス・ドライバ・ロジックに伝達し、
前記少なくとも１つのグラフィックス処理能力を識別する、前記照会に対する、応答を前記グラフィックス・デバイス・ドライバ・ロジックから受け取り、
前記応答を前記レンダリング・ロジックに伝達する
ように動作可能に構成されていることを特徴とする装置。
【請求項１１】
前記インターフェース・ロジックは、さらに、
前記識別された少なくとも１つのグラフィックス処理能力に関連付けられた少なくとも１つの入力要件に関しての別の照会を、前記レンダリング・ロジックから受け取り、
前記別の照会を前記グラフィックス・デバイス・ドライバに伝達し、
前記グラフィックス処理能力に関連付けられた前記少なくとも１つの入力要件を識別する別の応答を、前記グラフィックス・デバイス・ドライバ・ロジックから受け取り、
前記別の応答を前記レンダリング・ロジックに伝達する
ように動作可能に構成されることを特徴とする請求項１０に記載の装置。
【請求項１２】
前記インターフェース・ロジックは、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）を含むことを特徴とする請求項１１に記載の装置。
【請求項１３】
レンダリング・プロセスと、グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスとの間で通信する方法であって、
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を備える少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行することを備え、
前記レンダリング・プロセスとグラフィックス・デバイスとによって処理されるべきビデオ・データの前記記述、および前記グラフィックス・デバイスが提供することのできる前記少なくとも１つの対応する処理能力は、前記ビデオ・データをインターレース解除することに関連付けられている
ことを特徴とする方法。
【請求項１４】
前記少なくとも１つの対応する処理能力は、ＢＯＢライン倍加能力、切替適応能力、３次元適応能力、および運動ベクトル・ステアード能力を備える一群の処理能力、から選択された少なくとも１つの処理能力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】
レンダリング・プロセスと、グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスとの間で通信する方法であって、
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって前記照会コールを受け取

り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を備える少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行することを備え、
前記レンダリング・プロセスおよびグラフィックス・デバイスによって処理されるべきビデオ・データの前記記述、および前記グラフィックス・デバイスが提供することのできる前記少なくとも１つの対応する処理能力は、前記ビデオ・データをフレーム・レート変換することに関連付けられていることを特徴とする方法。
【請求項１６】
前記少なくとも１つの対応する処理能力は、フレーム反復／ドロップ能力、線形時相補間能力、および運動ベクトル・ステアード能力を備える一群の処理能力、から選択された少なくとも１つの処理能力を含むことを特徴とする請求項１５に記載の方法。
【請求項１７】
レンダリング・プロセスと、グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスとの間で通信する方法であって、
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を備える少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行することを備え、
前記少なくとも１つの肯定応答パラメータは、前記少なくとも１つの対応する処理能力に関連付けられた少なくとも１つのグローバル一意識別子（ＧＵＩＤ）を含み、
前記照会肯定応答コールは、複数の対応する処理能力に関連付けられた複数のグローバル一意識別子（ＧＵＩＤ）を含み、
前記複数のＧＵＩＤは、前記複数の対応する処理能力に関連付けられた少なくとも１つの比較因子に対応する順番で配列されることを特徴とする方法。
【請求項１８】
前記少なくとも１つの比較因子は、前記複数の対応する処理能力に関連付けられた品質因子を含むことを特徴とする請求項１７に記載の方法。
【請求項１９】
レンダリング・プロセスと、グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスとの間で通信する方法であって、
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を備える少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行することを備え、
前記照会コールは、ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓ

コールを含むことを特徴とする方法。
【請求項２０】
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって、前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を含んでいる少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行すること
を備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行命令を有し、
前記レンダリング・プロセスおよび前記グラフィックス・デバイスによって処理されるべきビデオ・データの前記記述、および前記グラフィックス・デバイスが提供することのできる前記少なくとも１つの対応する処理能力は、前記ビデオ・データをインターレース解除することに関連付けられていること
を特徴とするコンピュータ可読記録媒体。
【請求項２１】
前記少なくとも１つの対応する処理能力は、ＢＯＢライン倍加能力、切替適応能力、３次元適応能力、および運動ベクトル・ステアード能力を備える一群の処理能力、から選択された少なくとも１つの処理能力を含むことを特徴とする請求項２０に記載のコンピュータ可読記録媒体。
【請求項２２】
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって、前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を含んでいる少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行すること
を備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行命令を有し、
前記レンダリング・プロセスおよびグラフィックス・デバイスによって処理されるべきビデオ・データの前記記述、および前記グラフィックス・デバイスが提供することのできる前記少なくとも１つの対応する処理能力は、前記ビデオ・データをフレーム・レート変換することに関連付けられていることを特徴とするコンピュータ可読記録媒体。
【請求項２３】
前記少なくとも１つの対応する処理能力は、フレーム反復／ドロップ能力、線形時相補間能力、および運動ベクトル・ステアード能力、を備える一群の処理能力から選択された少なくとも１つの処理能力を含むことを特徴とする請求項２０に記載のコンピュータ可読記録媒体。
【請求項２４】
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、

前記グラフィックス・デバイス・ドライバ・プロセスによって、前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を含んでいる少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行すること
を備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行命令を有し、
前記少なくとも１つの肯定応答パラメータは、前記少なくとも１つの対応する処理能力に関連付けられた少なくとも１つのグローバル一意識別子（ＧＵＩＤ）を含み、
前記照会肯定応答コールは、複数の対応する処理能力に関連付けられた複数のグローバル一意識別子（ＧＵＩＤ）を含み、
前記複数のＧＵＩＤは、前記複数の対応する処理能力に関連付けられた少なくとも１つの比較因子に対応する順番で配列されることを特徴とするコンピュータ可読記録媒体。
【請求項２５】
前記少なくとも１つの比較因子は、前記複数の対応する処理能力に関連付けられた品質因子を含むことを特徴とする請求項２４に記載のコンピュータ可読記録媒体。
【請求項２６】
レンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータを有する照会コールを、前記レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイス・ドライバ・プロセスによって、前記照会コールを受け取り、前記照会コールを構文解析して、前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに含まれるビデオ・データの前記記述に基づいた、前記グラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の、識別子を含んでいる少なくとも１つの肯定応答パラメータを有する照会肯定応答コールを、前記グラフィックス・デバイス・ドライバ・プロセスによって発行すること
を備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行命令を有し、
前記照会コールは、ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓコールを含むことを特徴とするコンピュータ可読記録媒体。
【請求項２７】
レンダリング・プロセスと、グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスとの間で通信する方法であって、
選択されたビデオ・データを処理するためにグラフィックス・デバイスが提供することのできる選択された対応する処理能力の識別子を備える少なくとも１つのコール・パラメータを有するビデオ・インターレース解除に関する照会コールを、レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスによって前記照会コールを受け取り、前記照会コールを構文解析して前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに基づき、前記選択された対応する処理能力に関連付けられた少なくとも１つの入力要件、を備える少なくとも１つの肯定応答パラメータを有する照会肯定応答を、前記グラフィックス・デバイス・ドライバ・プロセスによって発行すること
を備えることを特徴とする方法。
【請求項２８】
前記照会コールは、ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅｓＣａｐｓコールを含むことを特徴とする請求項２７に記載の方法。

【請求項２９】
前記少なくとも１つの応答パラメータは、以前の出力フレーム・パラメータ、入力プール・パラメータ、前方参照サンプル・パラメータ、後方参照サンプル・パラメータ、出力フレーム形式パラメータ、ビデオ処理能力パラメータ、インターレース解除副長方形パラメータ、インターレース解除ＹＵＶ－ＲＧＢパラメータ、インターレース解除ストレッチＸパラメータ、インターレース解除ストレッチＹパラメータ、インターレース解除アルファ・ブレンド・パラメータ、色補正パラメータ、ガンマ補正パラメータ、回転パラメータ、およびシアリング・パラメータを備える一群の肯定応答パラメータから選択された少なくとも１つの肯定応答パラメータを含むことを特徴とする請求項２７に記載の方法。
【請求項３０】
選択されたビデオ・データを処理するためにグラフィックス・デバイスが提供することのできる選択された対応する処理能力の識別子を備える少なくとも１つのコール・パラメータを有するビデオ・インターレース解除に関する照会コールを、レンダリング・プロセスによって発行すること、
前記グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスによって前記照会コールを受け取り、前記照会コールを構文解析して前記少なくとも１つのコール・パラメータを取り出すこと、および、
前記コール・パラメータに基づき、前記選択された対応する処理能力に関連付けられた少なくとも１つの入力要件を含んでいる少なくとも１つの応答パラメータを有する照会応答を、前記グラフィックス・デバイス・ドライバ・プロセスによって発行すること
を備える動作を少なくとも１つの処理装置に実行させるためのコンピュータ実行可能命令を有することを特徴とするコンピュータ可読記録媒体。
【請求項３１】
前記照会コールは、ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅｓＣａｐｓコールを含むことを特徴とする請求項３０に記載のコンピュータ可読記録媒体。
【請求項３２】
前記少なくとも１つの応答パラメータは、以前の出力フレーム・パラメータ、入力プール・パラメータ、前方参照サンプル・パラメータ、後方参照サンプル・パラメータ、出力フレーム形式パラメータ、ビデオ処理能力パラメータ、インターレース解除副長方形パラメータ、インターレース解除ＹＵＶ－ＲＧＢパラメータ、インターレース解除ストレッチＸパラメータ、インターレース解除ストレッチＹパラメータ、インターレース解除アルファ・ブレンド・パラメータ、色補正パラメータ、ガンマ補正パラメータ、回転パラメータ、およびシアリング・パラメータを備える一群の応答パラメータから選択された少なくとも１つの応答パラメータを含むことを特徴とする請求項３０に記載のコンピュータ可読記録媒体。
【請求項３３】
レンダリングする手段と、
グラフィックス・デバイスをドライブする手段と、を備え、
前記レンダリングする手段は、
前記レンダリングする手段と前記グラフィックス・デバイスをドライブする手段により処理されるべきビデオ・データの記述を含んだ少なくとも１つのコール・パラメータを有する照会コールを発行する手段を有し、
前記グラフィックス・デバイスをドライブする手段は、
前記少なくとも１つのコール・パラメータを検索するために前記照会コールを受け取りそして前記照会コールを構文分析するための手段と、
前記コール・パラメータに含まれる前記ビデオ・データの記述に基づいて前記グラフィックス・デバイスをドライブする手段が提供できる少なくとも１つのグラフィックス処理能力のための識別子を含む少なくとも１つの肯定応答パラメータを含む照会肯定応答コールを発行するための手段を有し、
（ｉ）前記レンダリングする手段と前記グラフィックス・デバイスをドライブする手

段により処理されるべき前記ビデオ・データの記述、及び（ｉｉ）前記グラフィックス・デバイスをドライブする手段が提供できる少なくとも１つのグラフィックス処理能力の両方が、前記ビデオ・データのインターレース解除に関連していること
を特徴とする装置。
【請求項３４】
レンダリングする手段と、
グラフィックス・デバイスをドライブする手段と、を備え、
前記レンダリングする手段は、
前記レンダリングする手段と前記グラフィックス・デバイスをドライブする手段により処理されるべきビデオ・データの記述を含んだ少なくとも１つのコール・パラメータを有する照会コールを発行する手段を有し、
前記グラフィックス・デバイスをドライブする手段は、
前記少なくとも１つのコール・パラメータを検索するために前記照会コールを受け取りそして前記照会コールを構文分析するための手段と、
前記コール・パラメータに含まれる前記ビデオ・データの記述に基づいて前記グラフィックス・デバイスをドライブする手段が提供できる少なくとも１つのグラフィックス処理能力のための識別子を含む少なくとも１つの肯定応答パラメータを含む照会肯定応答コールを発行するための手段を有し、
（ｉ）前記レンダリングする手段と前記グラフィックス・デバイスをドライブする手段により処理されるべき前記ビデオ・データの記述、及び（ｉｉ）前記グラフィックス・デバイスをドライブする手段が提供できる少なくとも１つのグラフィックス処理能力の両方が、前記ビデオ・データのフレーム・レート変換に関連していること
を特徴とする装置。
【請求項３５】
前記少なくとも１つのグラフィックス処理能力が、ＢＯＢライン倍加能力と、切替適応化能力と、三次元適応化能力と、運動ベクトル・ステアード能力とを含む処理能力のグループから選択された少なくとも１つの処理能力を含むことを特徴とする請求項３４に記載の装置。
【請求項３６】
前記少なくとも１つのグラフィックス処理能力が、フレーム反復／ドロップ能力と、線型時相補間能力と、三次元適応化能力と、運動ベクトル・ステアード能力とを含む処理能力のグループから選択された少なくとも１つの処理能力を含むことを特徴とする請求項３４に記載の装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、一般に、ビデオ処理およびビデオ表示に関し、より詳細には、プログレッシブ・ビデオ表示のためのインターレースされたビデオ画像の処理を容易にする方法および装置に関する。
【背景技術】
【０００２】
特許証のためのこの米国本特許出願は、２００２年４月１５日出願の、「Methods and Apparatuses for Facilitating De-Interlacing of Video Images」という名称の特許証番号６０／３７６，８８０の同時継続米国仮特許出願の優先権の特典を主張し、参照によりその開示全体を本明細書に組み込む。
【０００３】
ビデオ画像シーケンスは、一般に、プログレッシブ・シーケンスとインターレースされたシーケンスの２つのタイプに分類することができる。プログレッシブ・ビデオでは、ビデオ・データのフレームを構成するすべての画素が時間的に同時にサンプリングされる。インターレースされたビデオでは、ビデオ画像の１つ置きのラインが時間的に交互にサン

プリングされる。したがってその結果得られるインターレースされたビデオ・フレームは２つの「フィールド」から構成されており、各フィールドの高さはそのビデオ・フレームの半分の高さになる。

【０００４】

インターレースされたビデオ・シーケンスの例としては、ＴＶ信号およびＤＶカメラ一体型ビデオの出力がある。プログレッシブ・ビデオ・シーケンスの例としては、ある種のＤＶＤおよびＷｉｎｄｏｗｓ（登録商標）Ｍｅｄｉａ　Ｖｉｄｅｏエンコーダの出力などのコンピュータ生成ビデオがある。さらに、一部のプロ仕様のビデオ・カメラもまた、プログレッシブ・ビデオを生成する。

【０００５】

コンピュータのモニタおよび他の類似の表示装置は「プログレッシブ・ディスプレイ・システム」の例であり、したがって追加的な処理をまったく必要とせずにプログレッシブ・ビデオを表示することができる。インターレースされたビデオは、最初に「インターレース解除」されない限りコンピュータのモニタ上に表示することはできない。

【０００６】

従来型パーソナル・コンピュータ（ＰＣ）でビデオをインターレース解除するための現在の技術では、グラフィックス・プロセッサ（graphics processor）によって「グラフィックス・オーバーレイ・デバイス（graphics overlay device）」が使用されることが必要となる。グラフィックス・オーバーレイ・デバイスの使用にはいくつかの制約および制限があり、したがってこれらのデバイスを使用することは将来的には評価が低くなる可能性がある。例えば、大部分のＰＣにはグラフィックス・オーバーレイ・デバイスが１つしかない場合がある。さらに、大部分のグラフィックス・オーバーレイ・デバイスは、ＢＯＢアルゴリズムなどの粗雑なインターレース解除アルゴリズムしか提供しない。

【発明の開示】

【発明が解決しようとする課題】

【０００７】

この結果、これらおよび他の理由から、ビデオのインターレース解除を容易にするための改善された方法および装置が求められている。

【課題を解決するための手段】

【０００８】

本発明のある種の典型的な側面によれば、ビデオのインターレース解除を容易にするための方法および装置が提供される。例として、ある種の実装には、コンピュータのモニタおよび／または他の類似の表示装置上に表示することができる単一ビデオ・フレームを生成するためにビデオ・フィールドをインターレース解除するようグラフィックス・プロセッサ・ロジック（すなわち、ハードウェア、ファームウェアおよび／またはソフトウェア）に指示するための、インターフェースをとる方法および装置が提供されている。これは、グラフィックス・オーバーレイ・デバイスを必要とせずに達成することができる。

【０００９】

本発明のある種の実装によれば、グラフィックス・デバイス・ドライバに、対応するグラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を、レンダラーに対して識別することを、させることと、その識別されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を、そのレンダラーに対して識別することを、そのグラフィックス・デバイス・ドライバにさせることとを含む方法が提供される。

【００１０】

本発明の他の実装では、グラフィックス・デバイス・ドライバは、対応するグラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を、レンダラーに対して識別し、その識別されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を、レンダラーに対して識別するように、動作可能に構成されている。

【００１１】

本発明のさらに別の実装によれば、グラフィックス・デバイス・ドライバに、対応するグラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を、レンダラーに対して識別すること、を実行させ、またその識別されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を、そのレンダラーに対して識別することをそのグラフィックス・デバイス・ドライバにさせることを、少なくとも１つの処理装置に実行させるためのコンピュータ実行可能命令を含むコンピュータ可読媒体が提供される。

【００１２】

上記および他の要件は、グラフィックス・デバイス・ドライバに関連付けられたグラフィックス・デバイスによって処理されるべきインターレースされたビデオ・データの記述を、そのグラフィックス・デバイス・ドライバに対して識別することを、レンダラーにさせること、を含む方法によっても達成される。この方法は、インターレース解除および／または他の適用可能な処理を提供するために使用することができるグラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力をそのレンダラーに対して識別することを、そのグラフィックス・デバイス・ドライバにさせること、およびそのレンダラーにその識別されたグラフィックス処理能力の少なくとも１つを選択すること、かつその１つ以上の選択されたグラフィックス処理能力に対する入力要件をそのグラフィックス・デバイス・ドライバに要求することをさせることをさらに含む。この方法はまた、その１つ以上の識別されたグラフィックス処理能力に関連付けられた入力要件を、そのレンダラーに対して識別することを、そのグラフィックス・デバイス・ドライバにさせることも含む。

【００１３】

ある種の実装では、レンダリング・ロジック、グラフィックス・デバイス・ロジック、およびインターフェース・ロジックを含む装置が提供される。インターフェース・ロジックはレンダリング・ロジックとグラフィックス・デバイス・ロジックを動作可能に結合する。その結果、レンダリング・ロジックは、グラフィックス・デバイス・ロジックに関連付けられたグラフィックス・デバイスによって処理されるべきインターレースされたビデオ・データの記述を提供することができ、グラフィックス・デバイス・ロジックは、グラフィックス・デバイスに関連付けられた少なくとも１つのグラフィックス処理能力を識別することができ、レンダリング・ロジックは、少なくとも１つの選択されたグラフィックス処理能力に対する入力要件をグラフィックス・デバイス・ロジックに要求することができ、グラフィックス・デバイス・ロジックは、その少なくとも１つの選択されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件を識別することができる。

【００１４】

さらに別の典型的な実装では、ビデオ・データをインターレース解除する際に、関連付けられたグラフィックス・デバイスが実行することのできる少なくとも１つのグラフィックス処理能力に関して、グラフィックス・デバイス・ドライバに対する照会（query）をレンダラーから受け取ること、その照会をそのグラフィックス・デバイス・ドライバに伝達すること、そのレンダラーに対して１つ以上のグラフィックス処理能力を識別するその照会への応答をそのグラフィックス・デバイス・ドライバから受け取ること、およびその応答をそのレンダラーに伝達することを含む方法が提供される。別の実装では、この方法は、その識別されたグラフィックス処理能力に関連付けられた少なくとも１つの入力要件に関して、グラフィックス・デバイス・ドライバに対する追加照会をそのレンダラーから受け取ること、その追加照会をそのグラフィックス・デバイス・ドライバに伝達すること、そのグラフィックス処理能力に関連付けられたその１つ以上の入力要件を識別するその追加照会に対する追加応答をそのグラフィックス・デバイス・ドライバから受け取ること、およびその追加応答をそのレンダラーに伝達することも含む。

【００１５】

本発明のさらに別の実装では、レンダリング・プロセスと、グラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスとの間で通信する方法

が提供される。ここで、例えば、この方法は、そのレンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述を備える少なくとも１つのコール・パラメータ（call parameter）を有する照会コール（query call）、をレンダリング・プロセスによって発行すること、グラフィックス・デバイス・ドライバ・プロセスによってその照会コールを受け取り、その照会コールを構文解析して、１つ以上のコール・パラメータを取り出すこと、およびビデオ・データの記述に基づいてグラフィックス・デバイスが提供することのできる少なくとも１つの対応する処理能力の識別子を含んでいる少なくとも１つの肯定応答パラメータ（acknowledgment parameter）を有する照会肯定応答コール（query acknowledgment call）を、グラフィックス・デバイス・ドライバ・プロセスによって発行することを含む。

【００１６】

別の典型的な方法は、選択されたビデオ・データを処理するためにグラフィックス・デバイスが提供することのできる選択された対応する処理能力の識別子を含んでいる少なくとも１つのコール・パラメータを有する照会コールをレンダリング・プロセスによって発行すること、そのグラフィックス・デバイスに関連付けられたグラフィックス・デバイス・ドライバ・プロセスによってその照会コールを受け取り、１つ以上のコール・パラメータを取り出すためにその照会コールを構文解析すること、およびその選択された対応する処理能力に関連付けられた少なくとも１つの入力要件を含んでいる少なくとも１つの肯定応答パラメータ（acknowledgment parameter）を有する照会肯定応答（query acknowledgment）を、グラフィックス・デバイス・ドライバ・プロセスによって発行することを含む。

【００１７】

本発明のさらに別の態様によれば、レンダリング・プロセスと、関連付けられたグラフィックス・デバイス・ドライバを有するグラフィックス・デバイスとの間にビデオ・ストリーム・オブジェクトを確立する方法が提供される。ここで、例えば、この方法は、ストリーミングされたビデオ・データを処理する期間にグラフィックス・デバイスが実行することが可能な選択された対応する処理能力の識別子と、そのストリーミングされたビデオ・データを処理する期間にレンダリング・プロセスとグラフィックス・デバイスによって処理されるべきビデオ・データの記述とを含む少なくとも２つのコール・パラメータを有するオープン・ストリーム・コールをレンダリング・プロセスによって発行することを含む。この方法は、対応するプロセスによってオープン・ストリーム・コールを受け取り、コール・パラメータを取り出すためにそのオープン・ストリーム・コールを構文解析すること、その対応するプロセスが提供するビデオ・ストリーム・オブジェクトに対応するストリーム識別子を、その対応するプロセスによって発行することをさらに含む。

【００１８】

さらに別の実装では、ビデオ・ストリーム・オブジェクトを閉じる方法が提供される。ここで、例えば、この方法は、既に確立されたビデオ・ストリーム・オブジェクトに対応するストリーム識別子を含んでいる少なくとも１つのコール・パラメータを有するクローズ・ストリーム・コールを、レンダリング・プロセスによって発行すること、対応するプロセスによって、このクローズ・ストリーム・ストリームコールを受信し、コール・パラメータを取り出すためにこのクローズ・ストリーム・コールを構文解析すること、およびその事前に確立されたビデオ・ストリーム・オブジェクトを、その対応するプロセスに閉じさせることを含む。

【００１９】

上記および他の要件の一部は、インターレースされたビデオ・データを操作する際に使用する方法によっても扱われる。この方法は、幅、高さおよびストライドを有するビデオ・サーフェスに関連付けられたトップ・フィールド・データとボトム・フィールド・データとを含むインターレースされたビデオ・データにアクセスすることを含む。この方法は、トップ・フィールド・データとボトム・フィールド・データを分離することと、その高さよりも低い再解釈された高さとそのストライドよりも大きな再解釈されたストライドと

を再解釈されたビデオ・サーフェスが有するように、その再解釈されたビデオ・サーフェスの分離されたトップとボトムのフィールド・データを構成することによって再解釈されたビデオ・サーフェスを生成することをさらに含む。

【図面の簡単な説明】

【００２０】

【図１】本発明のある種の実装態様での使用に適した典型的なコンピュータ・システムを全般的に示すブロック図である。

【図２】本発明のある種の典型的な実装に合せて、２つのインターレースされたフィールドを含んでいるビデオ・サーフェスを示す説明図である。

【図３】本発明のある種の典型的な実装に合せて、ビデオ信号をレンダリングするように構成されたコンピュータまたは類似の装置のある種の機能素子を示すブロック図である。

【図４】本発明のある種の典型的な実装に合せて、ビデオ信号を処理する方法を示す流れ図である。

【発明を実施するための形態】

【００２１】

添付の図面を参照して以下の詳細な説明を読むことによって、本発明の様々な方法および装置のより徹底した理解が得られよう。

【００２２】

図面に目を向けると、そこには、類似の要素が類似の参照番号で示され、本発明は、適切なコンピューティング環境で実装されているように描かれている。必須ではないが、本発明は、パーソナル・コンピュータによって実行される、プログラム・モジュールなどの、コンピュータ実行可能命令という一般的な状況で説明することにする。一般に、プログラム・モジュールには、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、コンポーネント、データ構造等が含まれる。さらに、本発明は、ハンドヘルド・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、またはプログラム可能な民生電子機器、ネットワークＰＣ、ミニ・コンピュータ、メーンフレーム・コンピュータ等を含む他のコンピュータ・システム構成によっても実行することができることが当業者には理解されよう。

【００２３】

典型的なビデオ・ディスプレイ環境

図１は、以下で説明する方法および装置を実装することができる適切なコンピューティング環境１２０の一例を示している。典型的なコンピューティング環境１２０は適切なコンピューティング環境の一例にすぎず、本明細書に記載の改良された方法およびシステムの用途または機能の範囲に関していかなる限定をも示唆することを意図するものではない。また、コンピューティング環境１２０は、コンピューティング環境１２０に示される構成要素のどれか１つまたはそれらの組合せに関するいかなる依存状態または必要条件を有するものと解釈されるべきでもない。

【００２４】

本明細書に記載の改良された方法および装置は、多数の他の汎用または専用コンピューティング・システム環境または構成によって使用可能である。適切と思われる周知のコンピューティング・システム、環境および／または構成の例としては、限定はしないが、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・クライアント、ハンドヘルドまたはラップトップ・デバイス、マルチプロセス・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、プログラム可能な民生電子機器、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等がある。

【００２５】

図１に示すように、コンピューティング環境１２０には、コンピュータ１３０の形態の汎用コンピューティング・デバイスが含まれる。コンピュータ１３０の構成要素としては、１つまたは複数のプロセッサまたは処理装置１３２、システム・メモリ１３４、および

システム・メモリ１３４を含めて様々なシステム・コンポーネントをプロセッサ１３２に結合するバス１３６を含むことができる。

【００２６】

バス１３６は、メモリ・バスまたはメモリ・コントローラ、周辺バス、アクセラレィティッド・グラフィックスポート、および様々なバス・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、数種類のバス構造のどれか１つまたは複数を表している。限定はしないが一例として、このようなアーキテクチャには、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、およびＭｅｚｚａｎｉｎｅバスとしても知られる周辺コンポーネント相互接続（ＰＣＩ）バスが含まれる。

【００２７】

コンピュータ１３０は、通常、様々なコンピュータ可読媒体を含む。このような媒体は、コンピュータ１３０によってアクセス可能な任意の使用可能な媒体であってよく、揮発性および不揮発性媒体、取り外し可能および非取り外し可能な媒体を含む。

【００２８】

図１では、システム・メモリ１３４は、ランダム・アクセス・メモリ（ＲＡＭ）１４０などの揮発性メモリ形式、および／または読み出し専用メモリ（ＲＯＭ）１３８などの不揮発性メモリ形式のコンピュータ可読媒体を含む。起動時などにコンピュータ１３０内の素子間で情報を転送するために役立つ基本ルーチンを含んでいる基本入出力システム（ＢＩＯＳ）１４２がＲＯＭ１３８に記憶されている。ＲＡＭ１４０は、通常、プロセッサ１３２によって即座にアクセス可能であり、かつ／または現在動作中のデータおよび／またはプログラム・モジュールを含んでいる。

【００２９】

コンピュータ１３０は、他の取り外し可能／非取り外し可能、揮発性／不揮発性のコンピュータ記憶媒体をさらに含むことができる。例えば、図１は、非取り外し可能な不揮発性磁気媒体（図示せず、一般には「ハード・ドライブ」と称する）に対して読み出しと書き込みを行うハードディスク・ドライブ１４４と、取り外し可能な不揮発性磁気ディスク１４８（例えば「フロッピー（登録商標）ディスク」）に対して読み出しと書き込みを行う磁気ディスク・ドライブ１４６と、ＣＤ－ＲＯＭ／Ｒ／ＲＷ、ＤＶＤ－ＲＯＭ／Ｒ／ＲＷ／＋Ｒ／ＲＡＭ、または他の光媒体などの取り外し可能な不揮発性光ディスク１５２に対して読み出しと書き込みを行う光ディスク・ドライブ１５０、を示している。ハードディスク・ドライブ１４４、磁気ディスク・ドライブ１４６、および光ディスク・ドライブ１５０は、それぞれに１つまたは複数のインターフェース１５４によってバス１３６に接続される。

【００３０】

ドライブおよび関連付けられたコンピュータ可読媒体は、コンピュータ可読命令、データ構造、プログラム・モジュール、および他のデータの不揮発性ストレージをコンピュータ１３０に提供する。本明細書に記載の典型的な環境はハードディスク、取り外し可能な磁気ディスク１４８、および取り外し可能な光ディスク１５２を使用しているが、この典型的な操作環境では、コンピュータによってアクセス可能なデータを記憶することができる他のタイプのコンピュータ可読媒体、例えば磁気カセット、フラッシュメモリ・カード、デジタル・ビデオ・ディスク、ランダム・アクセス・メモリ（ＲＡＭ）、読み取り専用メモリ（ＲＯＭ）等も使用することができるということを当業者には理解されたい。

【００３１】

オペレーティング・システム１５８、１つまたは複数のアプリケーション・プログラム１６０、他のプログラム・モジュール１６２およびプログラムデータ１６４を含めて多数のプログラム・モジュールを、ハードディスク、磁気ディスク１４８、光ディスク１５２、ＲＯＭ１３８、またはＲＡＭ１４０上に記憶することができる。

【００３２】

本明細書に記載の改良された方法およびシステムは、オペレーティング・システム１５８、１つまたは複数のアプリケーション・プログラム１６０、他のプログラム・モジュール１６２、および／またはプログラムデータ１６４、の範囲内で実装することができる。

【００３３】

ユーザは、キーボード１６６およびポインティング・デバイス１６８（「マウス」等）等の入力装置を介してコンピュータ１３０にコマンドおよび情報を提供することができる。この他の入力装置としては（図示せず）、マイクロフォン、ジョイスティック、ゲームパッド、衛星放送用アンテナ、シリアルポート、スキャナー、カメラ等を含むことができる。これらおよび他の入力装置は、バス１３６に結合されているユーザ入力インターフェース１７０を介して処理装置１３２に接続されているが、パラレルポート、ゲームポート、またはユニバーサル・シリアル・バス（ＵＳＢ）などの他のインターフェースおよびバス構造を介して接続することもできる。

【００３４】

モニタ１７２または他のタイプの表示装置も、ビデオ・アダプタ１７４などのインターフェースを介してバス１３６に接続される。モニタ１７２の他に、パーソナル・コンピュータは、通常、出力周辺インターフェース１７５を介して接続することができるスピーカーおよびプリンタなどの他の周辺出力装置（図示せず）を含む。ビデオ・アダプタ１７４は、通常、ビデオ・グラフィックス・デバイス（video graphics device）を含む。

【００３５】

コンピュータ１３０は、遠隔コンピュータ１８２等の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。遠隔コンピュータ１８２は、コンピュータ１３０に関して本明細書に記載した要素および特徴の多くまたはすべてを含むことができる。

【００３６】

図１に示した論理接続は、ローカル・エリア・ネットワーク（ＬＡＮ）１７７と一般的な広域ネットワーク（ＷＡＮ）１７９である。このようなネットワーク接続環境は、オフィス、企業規模のコンピュータ・ネットワーク、イントラネット、およびインターネットでは普及している。

【００３７】

ＬＡＮネットワーク接続環境で使用されるとき、コンピュータ１３０はネットワーク・インターフェースまたはアダプタ１８６を介してＬＡＮ１７７に接続される。ＷＡＮネットワーク接続環境で使用されるとき、コンピュータは、通常、モデム１７８またはＷＡＮ１７９を介して通信を確立する他の手段を含む。内蔵であっても外付けでもよいモデム１７８は、ユーザ入力インターフェース１７０または他の適切な機構を介してシステム・バス１３６に接続することができる。

【００３８】

図１に示されているのは、インターネットを介したＷＡＮの具体的な実装である。ここでは、コンピュータ１３０は、インターネット１８０を介して少なくとも１つの遠隔コンピュータ１８２との通信を確立するためにモデム１７８を使用している。

【００３９】

ネットワーク接続された環境では、コンピュータ１３０に関して示したプログラム・モジュールまたはその一部は、遠隔メモリ・ストレージ・デバイスに記憶することができる。したがって、図１に示すように、遠隔アプリケーション・プログラム１８９は遠隔コンピュータ１８２のメモリ・デバイス上に常駐することができる。図示し説明したネットワーク接続は一例であって、コンピュータ間に通信リンクを確立する他の手段を使用することもできることが理解されよう。

【００４０】

ビデオのインターレース解除

上記の通り、ビデオ画像データをコンピュータのモニタ上または他の類似の表示装置上に正確に表示することができるようにある種のビデオ画像データをインターレース解除す

る必要がある。既に発売されているＭｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システムは、例えば、ビデオをコンピュータのモニタ上に表示する際にそのビデオをインターレース解除するためには「グラフィックス・オーバーレイ・デバイス」に依存してきた。この技術または他の類似の技術にはいくつかの欠点があるが、それらの欠点には例えば以下のものが含まれる。（１）コンピュータには「グラフィックス・オーバーレイ・デバイス」が１つしかなく、したがって単一のビデオ・ストリームしか正確に表示することはできない。（２）グラフィックス・オーバーレイ・デバイスが、コンピュータのデスクトップ・ディスプレイに「ｋｅｙｅｄ（嵌めこまれ）」されてしまう。すなわちエンド・ユーザは印刷画面キーを押しても、その時点で表示されているビデオ画像ではなく基調色（key color）を取り込むだけだということである。（３）ユーザがウィンドウを新しい位置にドラッグすると、ビデオが表示されるべき場所に基調色が時折点滅しながら表示される。（４）ユーザがビデオ再生ウィンドウの上に半透明のウィンドウをドラッグすると、基調色はそのビデオ画像ではなくユーザのウィンドウとブレンドされる。

【００４１】

さらに、現代のグラフィックス・オーバーレイ・デバイスが使用するインターレース解除するための一般的な方法は、非常に品質が低い。これは、テレビジョンなどの民生用デバイスの品質よりも遥かに低い水準である。

【００４２】

当業者には、現行の技術にはこれら以外の他の欠点もあることに気付かれよう。

【００４３】

本明細書で提示する新しい方法および装置は、グラフィックス・オーバーレイ・デバイスを必要とせずにグラフィックス・プロセッサがビデオ画像をインターレース解除することを可能にする。提示する技術は、より高度なインターレース解除アルゴリズム、特に従来の装置と比較した場合に、表示されるビデオ画像の視覚的品質が大幅に向上するようなアルゴリズム、をグラフィックス・プロセッサが使用することを可能にする。

【００４４】

インターレースされたフィールドの典型的な再解釈

図２の上部には、２つのインターリーブされたフィールド、すなわちトップ・フィールド２０２とボトム・フィールド２０４とを有するように示されるインターリーブされたビデオ・サーフェス２００が示されている。ビデオ・サーフェス２００は、幅２０６、高さ２０８、およびストライド２１０を有する。図２の下部には、再解釈されたビデオ・サーフェス２００′が、分離したトップ・フィールド２０２′とボトム・フィールド２０４′とを有するように示されている。ここで、ビデオ・サーフェス２００′は、対応するインターレースされたビデオ・サーフェス２００の高さの１／２の高さ２０８′と、対応するインターレースされたビデオ・サーフェス２００のストライドの２倍のストライド２１０′とを有する。

【００４５】

レンダラーとグラフィックス・デバイスの間のインターフェース

図３は、図１に示されているようなコンピュータ内でビデオ信号をレンダリングする際に使用されるある種の要素／機能を示すブロック図である。これらの様々な要素および／または機能は、ハードウェアおよび／またはソフトウェアで適切なものとして実装可能である。これらの要素および／または機能の多くは周知の技術を代表するものである。本発明のある側面では、グラフィックス・インターフェースとグラフィックス・デバイス・ドライバとの間のインターフェース（例えばデバイス・ドライバ・インターフェース、ＤＤＩ）は、グラフィックス・オーバレイを必要とせず、かつ／またはレンダラーの機能を改善して、グラフィックス・デバイス・ドライバ／ロジックとより良好な通信を行うように変更され、インターレース、フレーム・レート変換、および／またはグラフィックス・デバイスの他の機能の改善点を活用する。したがって、図３の典型的な装置３００の記述に続いて、適用可能な方法が、図４に、さらに、本発明のある追加実装例に従ってアプリケ

ーション・プログラミング・インターフェース（ＡＰＩ）の形式で、提示されている。
【００４６】
装置３００は、変換ロジック３０２を含む。この変換ロジックは、例えば中央処理装置、グラフィックス処理装置および／またはこれらの組合せによって実行される命令を含むことができる。変換ロジック３０２は、少なくとも１つのソース３０４から符号化されたビデオ・データを受け取るように構成されている。ソース３０４からのこの符号化されたビデオ・データはある種の方式（例えばＭＰＥＧ－２等）で符号化されており、変換ロジック３０２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。一例として、ソース３０４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、メモリ、送信される信号、または変換ロジック３０２に符号化されたビデオ・データを分配または他の方法で提供するように構成された他の類似のソースを含むことができる。ある種の実装態様では、ソースは、インターネット３０６および遠隔ソース３０８に代表されるようなネットワークおよび遠隔ソースを含むことができる。
【００４７】
変換ロジック３０２によって出力された復号されたビデオ・データ入力は、少なくとも１つのレンダラー３１０に提供される。レンダラー３１０は、ビデオ・ストリームを復号する変換ロジック３０２を支援し、ビデオ画像が表示装置１７２のアスペクト比と一致するようにビデオ画像をアスペクト比を補正し、クローズド・キャプションまたはＤＶＤサブ・ピクチャ画像などの他の補助的画像データをこのビデオ画像とブレンドし、次いで表示装置１７２上に表示するために適宜そのビデオ画像データをグラフィックス・インターフェース・ロジック（graphics interface logic）３１２に送出するように構成されている。この結果得られたレンダリングされたビデオ・データは、次いでグラフィック・インターフェース・ロジック３１２に提供される。グラフィック・インターフェース・ロジック３１２は、例えばＤｉｒｅｃｔＤｒａｗ（登録商標）、Ｄｉｒｅｃｔ３Ｄ（登録商標）、および／または類似のロジックを含むことができる。グラフィック・インターフェース・ロジック３１２は、レンダラー３１０とグラフィックス・デバイス３３２と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。
【００４８】
グラフィック・インターフェース・ロジック３１２によるデータ出力は、デバイス・ドライバ・インターフェース３１４を使用してグラフィックス・デバイス・ドライバ３２０に提供される。ここで、デバイス・ドライバ・インターフェース３１４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）３１６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３１４は、レンダラー３１０とグラフィックス・デバイス・ドライバ３２０の間のインターフェースをサポートするように構成されている。
【００４９】
図３に示されるように、ある種の実装では、デバイス・ドライバ・インターフェース３１４とグラフィックス・デバイス・ドライバ３２０は、グラフィックス・デバイス３３２の操作環境に関してユーザ・モード・ロジック３１８またはカーネル・モード・ロジック３１９のどちらかの部分にさらに分類することができる。
【００５０】
この例では、レンダラー３１０によるビデオ・データ出力は最終的にグラフィックス・プロセッサ・ユニット（ＧＰＵ）３２２に提供される。これはインターレース解除ロジック３２４、フレームレート・コンバータ・ロジック３２６（任意選択）、および／またはビデオ・データの他の処理を実行するように構成可能である。この例では、これらの名称が示す通り、インターレース解除ロジック３２４はビデオ・データをインターレース解除するように構成されており、フレームレート・コンバータ・ロジック３２６はフレーム・レートを必要／所望に応じて変更するように構成されている。
【００５１】

ＧＰＵ３２２からの出力はビデオ・メモリ３２８に提供される。ビデオ・メモリ３２８が読み取られると、その結果生じるデータは次いでデジタル－アナログコンバータ（ＤＡＣ）３３０に提供され、このデジタル－アナログ・コンバータ（ＤＡＣ）３３０は表示装置１７２が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。別の構成では、表示装置１７２は、ＤＡＣ３３０を必要とせずにビデオ・メモリ３２８からのデジタル・データを処理するように構成することができる。図示したように、グラフィックス・デバイス３３２は、ＧＰＵ３２２、ビデオ・メモリ３２８およびＤＡＣ３３０を含むことができる。ある種の実装では、グラフィックス・デバイス３３２は、ＰＣまたは類似のデバイス内で構成することができるビデオ・グラフィック・カードの形態を取る。

【００５２】

図３に示す構成はビデオ・データ処理装置の一例にすぎない。当業者には、本発明の方法および装置がこの他の実現可能な構成も利することができることが理解されよう。

【００５３】

典型的なインターフェース操作

次に図４の流れ図を参照すると、レンダラー３１０とグラフィックス・デバイス３３２をインターフェースする方法４００が示されている。１つまたは複数のＡＰＩ３１６を、方法４００を実行するかまたは他の方法でサポートするように、構成することができる。

【００５４】

動作４０２で、レンダラー３１０は、ビデオ・データの記述をグラフィックス・デバイス・ドライバ３２０に提供する。ここで、このビデオ・データは、処理されて最終的に表示装置１７２を介して表示されるように、予め選択される。これに呼応して、グラフィックス・デバイス・ドライバ３２０は動作４０４で、グラフィックス・デバイス３３２についての適用可能な、または他の方法で使用可能なグラフィックス処理能力を識別する。この識別されたグラフィックス能力はＧＰＵ３２２または他の類似のロジックに関連付けられる。ある種の実装では、例えば１つまたは複数のインターレース解除機能、レート・コンバータ機能および／または他の適用可能な機能を、ある種の方法で識別することができる。

【００５５】

動作４０６では、レンダラー３１０または他の関連付けられたロジックは、識別されたグラフィックス処理能力の少なくとも１つを選択する。動作４０８では、レンダラー３１０は、その選択された１つ以上のグラフィックス処理能力に関する任意の適用可能な入力要件を要求する。動作４１０では、この要求に応答して、グラフィックス・デバイス・ドライバ３２０が任意の適用可能な入力要件をレンダラー３１０に提供する。

【００５６】

動作４１２で、レンダラー３１０はビデオ・データ用のストリームを開く。動作４１４で、レンダラー３１０とグラフィックス・デバイス３２２は、ビデオを処理し、ストリーミングし、最終的に表示する。動作４１６で、レンダラー３１０は、動作４１２で開いたストリームを閉じる。

【００５７】

典型的なインターフェース実装

装置３００は、ビデオ・データのインターレース解除、フレーム・レート変換および／または他の処理、を提供するように構成することができるビデオ処理システムの一例である。さらなる例として、本発明のある種の典型的な実装によって、Ｍｉｃｒｏｓｏｆｔ（登録商標）ＤｉｒｅｃｔＸ（登録商標）ＶＡは、レンダリングされ表示されるべき画像データの処理に関連付けられたインターレース解除およびフレーム・レート変換をサポートするように拡張されるか、または他の方法で強化することができる。さらなる関連情報を、Ｍｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗｓ（登録商標）Ｐｌａｔｆｏｒｍ Ｄｅｓｉｇｎ Ｎｏｔｅ ｅｎｔｉｔｌｅｄ ＤｉｒｅｃｔＸ（登録商標）ＶＡ： Ｖｉｄｅｏ Ａｃｃｅｌｅｒａｔｉｏｎ ＡＰＩ／ＤＤＩ, ｄａｔｅｄ Ｊａｎｕａｒｙ ２３, ２００１、に見つけることができ、これらを参照により本明細書に組み込む。

【００５８】
以下では、グラフィックス・デバイス・ドライバにおいてビデオ・コンテンツに関するインターレース解除、フレーム・レート変換および／または他の処理能力をサポートするようにＭｉｃｒｏｓｏｆｔ（登録商標）ＤｉｒｅｃｔＸ（登録商標）ＶＡを有利に拡張するＡＰＩを説明する。この説明は、ドライバ開発者および他の当業者にたいして、典型的なＡＰＩと本明細書に記載の方法および装置について明確な理解を提供する。
【００５９】
本明細書では、方法と装置が、Ｍｉｃｒｏｓｏｆｔ（登録商標）Ｗｉｎｄｏｗ（登録商標）のＰＣ用オペレーティング・システムの現在におけるバージョンに適用可能なＡＰＩの見地から、説明しているが、この方法および装置は他のオペレーティング・システムおよび／または他のデバイスにも適用可能であることを理解されたい。
【００６０】
インターレースされたビデオは、例えば、ビデオ用ＤｉｒｅｃｔＸ（登録商標）Ｖｉｄｅｏ　ＡｃｃｅｌｅｒａｔｉｏｎおよびＤｉｒｅｃｔ３Ｄ（登録商標）９　ＡＰＩを使用してインターレース解除することができる。グラフィックス・エンジン（例えばＧＰＵ）およびハードウェア・オーバーレイは、存在するならば、最小限のＢＯＢをサポートし、インターレース解除機能を組み入れているはずである。
【００６１】
インターレース解除またはフレーム・レート変換処理の出力はプログレッシブ・フレームである。本明細書に記載のインターレース解除の、およびフレーム・レート変換のインターフェースはすべてのビデオ表示機構から独立している。以下の例では、目標ＤｉｒｅｃｔＤｒａｗ（登録商標）サーフェスにインターレース解除された出力が提供される。これを行うことによって、従来のハードウェア・オーバレイ・ソリューションの必要性が排除される。
【００６２】
本明細書で使用するインターレース解除という用語は、所与の時間ｔの１つのフィールドに欠けているラインを再構成することによって、時間ｔのプログレッシブなビデオ・フレームを生成することを意味する。本明細書で使用するフレーム・レート変換という用語は、フィールドまたはフレームのシーケンスから任意の時間ｔの新しいプログレッシブなビデオ・フレームを生成することを意味する。
【００６３】
ビデオ・サンプルは、例えば１つまたは２つのビデオ・フィールドを収容するＤｉｒｅｃｔＤｒａｗ（登録商標）サーフェスであってよい。そのサーフェスに収容されるフィールド数とそのフィールドの時間順序を示すために、フラグがサンプルに関連付けられる。ビデオは、多くの実現可能な経路、例えばＤＸ－ＶＡビデオ・デコーダ・プロセスの出力、ビデオ・ポートの出力、またはホスト・ベースのビデオ・デコーダの出力、等を介してこのサーフェスに達することができる。
【００６４】
単一フィールドのインターレース解除の例
ＳｔｒｅｔｃｈＢｌｔを実行することのできる大部分のグラフィックス・アダプタは、簡単なＢＯＢスタイルのインターレース解除を実行することもできる。これは、基本的に、ＤｉｒｅｃｔＤｒａｗ（登録商標）サーフェスのビデオのメモリ・レイアウトを再解釈する問題である。例えば２つのフィールドを分離するためにサーフェスが再解釈された後では、１つのフィールド内の各ラインは（例えば、中間ラインを生成するための補間によって）２倍になる。インターレース解除されたビデオが垂直ジッタを表示することを防止するために、単一のフレーム・ライン・オフセットが必要となる場合がある。
【００６５】
ビデオ・サーフェスが、例えば図２に示すように２つのインターレースされたフィールドを収容する場合、各フィールドを分離するためにそのサーフェスのメモリ・レイアウトを再解釈することができる。これは、例えば、図示するように元のサーフェスのストライ

ドを２倍にし、そのサーフェスの高さを半分に分割することによって達成することができる。このようにしてこの２つのフィールドが分離されると、これらのフィールドは、個々のフィールドを正確なフレームの高さにまで伸ばすことによって簡単にインターレース解除することができる。付加的な水平の伸張または縮小を、ビデオ画像のピクセルに対してアスペクト比を補正するために適用することもできる。

【００６６】

グラフィックス・デバイス・ドライバ３２０（図３）は、以下で詳述する「ＱｕｅｒｙＣａｐａｂｉｌｉｔｉｅｓ」ＡＰＩ（例えばＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅＣａｐｓ）を介してレンダラー３１０（例えばＤｉｒｅｃｔＸ（登録商標）ＶｉｄｅｏＭｉｘｉｎｇ　Ｒｅｎｄｅｒｅｒ（ＶＭＲ）等）に対して、これを行うグラフィックス・デバイスの機能を識別するように構成することができる。

【００６７】

個々のフィールドの高さは、例えばライン・レプリケーションによって、またはフィルタしてからストレッチする（filtered stretch）好適な方法によって垂直に伸ばすことができる。ライン・レプリケーションの方法が使用されると、その結果得られる画像は「濃淡のむらのある（blocky）」外観を有するか可能性がある。フィルタしてからストレッチする方法が使用される場合、その結果得られる画像はわずかに「ぼやけた（fuzzy）」外観を有する可能性がある。

【００６８】

ＡＰＩ記述のインターレース解除およびフレーム・レート変換

本明細書に記載の典型的なＡＰＩ３１６は、メソッドの２つの機能グループに分割することができる。第１のグループには、グラフィックス・デバイスのインターレース解除能力を決定するために使用することができる一組のメソッドと装置が含まれる。第２のグループには、インターレース解除されたストリーム・オブジェクトを作成し使用するために用いられる一組のメソッドと装置が含まれる。

【００６９】

ＡＰＩ３１６は、グラフィック・インターフェース３１２をサポートし、グラフィックス・デバイス・ドライバ３２０とインターフェースするデバイス・ドライバ・インターフェース３１４の一部をなすように設計されている。本発明のある種の実装によれば、ＡＰＩ３１６は、アプリケーションがアクセスすることのできるユーザ・モードのＡＰＩである必要はない。しかし、他の実装ではそうでなければならない場合がある。

【００７０】

使用可能なインターレース解除モードの決定

「ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓ」

ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓのメソッドは、特定の入力ビデオ形式に関して使用可能なインターレース解除、フレーム・レート変換モードおよび／または他の機能／モード（能力）を照会するために使用することができる。ここで、例えば、グラフィックス・デバイスによって戻された各モードに対してグローバル一意識別子（ＧＵＩＤ：globally unique identifier）または他の適切な識別子が提供される。ＧＵＩＤは、例えば、品質の降順（または昇順）などのある種の特定の順番で戻すことができる。したがって例えば、最高品質のモードは、戻されるＧＵＩＤアレイの第１の要素（first element）を占有することができる。したがって、例として、

```
HRESULT
DeinterlaceQueryAvailableModes(
      [in] LPDXVA_VideoDesc lpVideoDescription,
      [in out] LPDWORD lpdwNumModesSupported,
      [in out] LPGUID pGuidsDeinterlaceModes
      );
```

を検討されたい。

【００７１】

この例では、ｌｐＶｉｄｅｏＤｅｓｃｒｉｐｔｉｏｎパラメータは、インターレース解除されるか、レート変換されるか、かつ／または他の方法で処理されるかするビデオのタイプの「記述」である。ｌｐＶｉｄｅｏＤｅｓｃｒｉｐｔｉｏｎパラメータは、ソース・ビデオの解像度と形式をサポートするようにグラフィックス・デバイスを適合させるため、グラフィックス・デバイス・ドライバに伝達される。例えば、グラフィックス・デバイスは、４８０ｉコンテンツの３フィールド適応インターレース解除を実行することができる可能性があるが、しかし、ＢＯＢ１０８０ｉコンテンツに対して可能であるだけかも知れない。

【００７２】

ある種の実装では、グラフィックス・デバイスは、既存のＢｌｔ'ｔｅｒハードウェアを使用するなどして、少なくともＢＯＢをサポートすることができる筈である。したがって以下の例を検討する。

```
typedef enum_DXVA_SampleFormat {
    DXVA_SamplePreviousOutputFrame = 1,
    DXVA_SampleProgressiveFrame = 2,
    DXVA_SampleFieldInterleavedEvenFirst = 3,
    DXVA_SampleFieldInterleavedOddFirst = 4,
    DXVA_SampleFieldSingleEven = 5,
    DXVA_SampleFieldSingleOdd = 6,
} DXVA_SampleFormat;

typedef struct_DXVA_Frequency {
    DWORD Numerator;
    DWORD Denominator;
} DXVA_Frequency;

typedef struct_DXVA_VideoDesc {
    DWORD Size;
    DWORD SampleWidth;
    DWORD SampleHeight;
    DXVA_SampleFormat;
    D3DFORMAT d3dFormat;
    DXVA_Frequency InputSampleFreq;
    DXVA_Frequency OutputFrameFreq;
} DXVA_VideoDesc, * LPDXVA_VideoDesc;
```

【００７３】

ＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃデータ構造は、以下の例に示されるように、インターレース解除またはフレーム・レート変換を実行しようとする意図を、ドライバに伝達する。

例：７２０ｘ４８０ｉコンテンツのインターレース解除

２９．９７Ｈｚの周波数で１つのサンプルあたり２つのフィールドとして供給される７２０ｘ４８０ｉコンテンツをインターレース解除するために、ＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃデータ構造は以下のものを収容することができる。

```
SampleWidth = 720;
SampleHeight = 480;
SampleFormat = DXVA_SampleFieldInterleavedOddFirst;
d3dFormat = D3DFMT_YUV2;
InputSampleFreq.Numerator = 30000; // 29.97
InputSampleFreq.Denominator = 1001;
OutputFrameFreq.Numerator = 60000; // 59.94;
OutputFrameFreq.Denominator = 1001;
```

【００７４】
インターレース解除をし、フレーム・レート変換を実行することを意図する場合、ＯｕｔｐｕｔＦｒａｍｅＦｒｅｑフィールドは以下の通りであってよい。

```
OutputFrameFreq.Numerator = 85; // 85 Hz monitor Frequency
OutputFrameFreq.Denominator = 1;
```

後でＭＰＥＧ符号化するために単一フィールドをプログレッシブ・フレームにインターレース解除することだけを意図する場合、ＯｕｔｐｕｔＦｒａｍｅＦｒｅｑフィールドは以下の通りであってよい。

```
OutputFrameFreq.Numerator = 30000; // 29.97
OutputFrameFreq.Denominator = 1001;
```

【００７５】
例：４８０ｐコンテンツのフレーム・レート変換
例えばモニタ表示の周波数を合わせるために、４８０ｐコンテンツ上でフレーム・レート変換を実行する場合、ＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃ構造は以下のものを収容することができる。

```
SampleWidth = 720;
SampleHeight = 480;
SampleFormat = DXVA_SampleProgressiveFrame;
d3dFormat = D3DFMT_YUV2;
InputSampleFreq.Numerator = 60; // 60 Hz
InputSampleFreq.Denominator = 1;
OutputFrameFreq.Numerator = 85; // 85 Hz monitor Frequency
OutputFrameFreq.Denominator = 1;
```

【００７６】
インターレース解除およびフレーム・レート変換は、上記の典型的な構造を使用して一緒に実行することもできることに留意されたい。
【００７７】
典型的なインターレース解除モード
グラフィックス・デバイスは、例えば以下の可能なインターレース解除モードを報告することができる。
例えばＢｌｔ'ｔｅｒを使用したＢＯＢ（ライン倍加）。ある種の実装では、このモードは常に使用可能である筈である。
単純な切替適応。ここでは、例えば、そのフィールドに関して少ない動き（low motion）が検出された場合は２つの隣接フィールドのブレンドであり、大きな動き（high motion）が検出された場合はＢＯＢである。
拡張型３Ｄ適応。ここでは、例えば、欠けているラインが、そのグラフィックス・デバイスに固有であるようなある適応プロセスによって生成される。このプロセスは、例えば、欠けているラインの生成を支援するためにいくつかの参照サンプルを使用するかも知れない。この基準サンプルは、時間的に過去であっても未来であってもよい。例えば、３次元線形フィルタリングはこのカテゴリーに分類されることになる。
運動ベクトル・ステアード（Motion Vector Steered）。ここでは、補間が行われる前に、シーン内の個々のオブジェクトの運動ベクトルが使用され、個々の移動が時間軸で整列させられる。
【００７８】
当業者ならば、同様に本発明の方法および装置によってサポートすることができる他の実現可能なインターレース解除の能力を認識するだろう。
【００７９】
典型的なフレーム・レート変換モード
ある種の典型的なグラフィックス・デバイスは、以下の実現可能なフレーム・レート変換モードを報告することができる。

フレーム反復／ドロップ。これは、選択されたソース・サンプルを宛先サーフェスにコピーすることによって必要以上のメモリ帯域幅を消費する傾向があるので、推奨されないかもしれない。

線形時相補間。ここでは、新しいフレームを作成するために未来の参照フィールドと以前の参照フィールドがＡｌｐｈａブレンドされる。

運動ベクトル・ステアード。補間が行われる前に、シーン内の個々のオブジェクトの運動ベクトルが使用され、個々の移動が時間軸で整列させられる。

【００８０】

当業者ならば、同様に本発明の方法および装置によってサポートすることができる他の実現可能なフレーム・レート変換および／または他の処理能力を認識するだろう。

【００８１】

入力要件の決定：

「ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅＣａｐｓ」

レンダラー３１０は、特定のビデオ形式に使用可能なインターレース解除モードを決定した後で、特定のインターレース解除モードおよび／または選択された他の適用可能なビデオ処理の入力要件に関するより詳細な情報を決定するために再度グラフィックス・デバイス・ドライバ３２０に照会する。したがって、以下の例を検討されたい。

```
HRESULT
DeinterlaceQueryModeCaps(
      [in] LPGUID pGuidDeinterlaceMode,
      [in] LPDXVA_VideoDesc lpVideoDescription,
      [out] DXVA_DeinterlaceCaps* lpDeinterlaceCaps
      );
```

【００８２】

さらなる例として、グラフィックス・デバイス・ドライバ３２０は、ｌｐＤｅｉｎｔｅｒｌａｃｅＣａｐｓのために出力されたＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＣａｐｓ構造で選択されたモードに対してＧＰＵ３２２の能力を報告するように構成することができる。例えば以下のものを検討されたい。

```
typedef struct_DXVA_DeinterlaceCaps{
     DWORD Size;
     DWORD NumPreviousOutputFrames;
     DWORD InputPool;
     DWORD NumForwardRefSamples;
     DWORD NumBackwardRefSamples;
     D3DFORMAT OutputFrameFormat;
     DWORD VideoProcessingCaps;
     DWORD DeinterlaceTechnology;
} DXVA_DeinterlaceCaps;
```

【００８３】

ここで、ＮｕｍＰｒｅｖｉｏｕｓＯｕｔｐｕｔＦｒａｍｅｓフィールドは、インターレース解除アルゴリズムによって既に出力済みの要求されたフレーム数を示す。このパラメータは、反復的インターレース解除アルゴリズムによって使用することができる。

【００８４】

ＩｎｐｕｔＰｏｏｌフィールドは、インターレースされたソース・サーフェスがそこから割り当てられるべきメモリ・プールを示す。例えば、有効なメモリ・プール・ロケーションの記述に関しては、上記で参照したテキストのＤｉｒｅｃｔ３Ｄ（登録商標）およびＤｉｒｅｃｔＤｒａｗ（登録商標）の資料を参照されたい。

【００８５】

ＮｕｍＦｏｒｗａｒｄＲｅｆＳａｍｐｌｅｓフィールドは、このインターレース解除モードに関して、時間的に未来の、要求された追加の基準サンプル数を示し、この数は、Ｂ

ＯＢおよびライン・ブレンディングに関しては存在せず、適応インターレース解除およびフレーム・レート変換に関しては、恐らく、複数である。

【００８６】

ＮｕｍＢａｃｋｗａｒｄＲｅｆＳａｍｐｌｅｓフィールドは、このインターレース解除モードに関して時間的に過去の、要求された追加の基準サンプル数を示し、その数は、ＢＯＢに関しては存在せず、ライン・ブレンディングに関しては１つであり、適応インターレース解除およびフレーム・レート変換に関しては恐らくは複数である。

【００８７】

ＯｕｔｐｕｔＦｒａｍｅＦｏｒｍａｔフィールドは、出力フレームのＤｉｒｅｃｔ３Ｄサーフェス形式を示す。通常、多分、インターレース解除アルゴリズムは、入力サンプル形式と一致するサーフェス形式でフレームを出力する。このフィールドは、レンダラー３１０が、インターレース解除するハードウェア／ロジックに正確な出力フレーム・サーフェスを確実に供給することができるようにする。ＤＸＶＡ＿Ｄｅｉｎｔｅｒｌａｃｅ＿ＹＵＶ２ＲＧＢフラグがＶｉｄｅｏＰｒｏｃｅｓｓｉｎｇＣａｐｓフィールドに戻された場合、レンダラー３１０は、ＲＧＢ３２形式に加えて、このフィールドによって有効な出力形式が指定されたと見なすことができることに留意されたい。

【００８８】

ＶｉｄｅｏＰｒｏｃｅｓｓｉｎｇＣａｐｓフィールドは、要求されたインターレース解除と同時に実行することができる他の操作を識別する。以下のフラグは、いくつかの典型的な動作を識別する。

【００８９】

ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＳｕｂＲｅｃｔｓ。インターレース解除ハードウェア／ロジックは、ビデオ画像の副長方形領域上で動作することができる。これは、例えば出力フレームのサイズが縮小される際にビデオ画像がさらに処理される前にトリミングされなければならない場合に有用である。

【００９０】

ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＹＵＶ２ＲＧＢ。インターレース解除ハードウェア／ロジックは、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。ある種の実装では、使用されるＲＧＢ形式は、各色成分に対して少なくとも８ビットの精度を有することになる。これが可能ならば、レンダラー３１０内のバッファー・コピーを回避することができる。

【００９１】

ある種の実装では、グラフィックス・デバイス・ドライバ３２０はＢＯＢインターレース解除モードに対してこの動作をサポートすることができる筈であることに留意されたい。色空間変換は、以下のフラグのどれか（理想的にはすべて）と結合することができる場合には、レンダラー３１０内で特に有用である。

DXVA_VideoProcess_StretchX
DXVA_VideoProcess_StretchY
DXVA_VideoProcess_AlphaBlend

【００９２】

この典型的な実装では、ＲＧＢ色空間からＹＵＶ色空間に変換することは必須ではないことに留意されたい。

【００９３】

ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＳｔｒｅｔｃｈＸ。デインターレーサーが水平に縮小することができる場合、ビデオがインターレース解除されるのと同時にアスペクト比補正を実行することができる。このフラグは、ＢＯＢインターレース解除モードに関してもサポートすることができる。

【００９４】

ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＳｔｒｅｔｃｈＹ。時として、アスペクト比調整は一般的な図形サイズ変更操作と組み合わされ、アプリケーションによって規定された

合成空間内で、ビデオ画像を大きさ変更する。これは恐らくは非常に稀であり、ある種の実装では基本的な機能ではない。このフラグは、ＢＯＢインターレース解除モードに関してサポートすることができる。通常、アプリケーション/ウィンドウにビデオを適合させるために要求される大きさ変更を、インターレース解除に要求される大きさ変更と同時に行うことができる場合、これはさらに有効である。例えば、これを行うことによって累積的なアーティファクトが回避される。

【００９５】

ＤＸＶＡ＿ＶｉｄｅｏＰｒｏｃｅｓｓ＿ＡｌｐｈａＢｌｅｎｄ。ここでもまた、レンダラー３１０によるバッファー・コピーを回避することができる。通常、アプリケーションがビデオ・ストリームに関連付けられた定数アルファ値を変更することは非常に稀であるので、ある種の実装ではこれはシステムに対する優先順位の「低い」機能と見なすことができる。

【００９６】

ある種の実装では、他の能力も識別することができる。例えば他の色／ガンマ補正情報を識別することができる。上記技術を使用して、回転、シアリング（sheering）および／または他の類似の処理を識別することもできる。

【００９７】

ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈｎｏｌｏｇｙフィールドは、この特定のインターレース解除アルゴリズムを実装するために使用される基本的な技術を示す。以下のフラグはいくつかの典型的な動作を識別する。これらのフラグは、アルゴリズムの実装に最も厳密に一致するように必要に応じて組み合わせることができる。

【００９８】

ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿Ｕｎｋｎｏｗｎは、そのアルゴリズムが未知であるか、またはハードウェア製造業者に固有のものであることを示す。

【００９９】

ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＢＯＢＬｉｎｅＲｅｐｌｉｃａｔｅは、そのアルゴリズムが、ラインをその上下どちらかで反復することによって欠けているラインを作成することを示す。この方法は、非常にぎざぎざ（jaggy）に見えることになり、したがって一部のユーザには適切でない場合がある。

【０１００】

ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＢＯＢＶｅｒｔｉｃａｌＳｔｒｅｔｃｈ。このアルゴリズムは、各ビデオ・フィールドを、例えば２つのラインを平均化するか、または［－１、９、９、－１］／１６のフィルタを４つのライン全体に対して使用することによって、垂直方向に２倍に伸ばして、欠けているラインを作成する。この結果生じる画像が上下に揺れないようにするために僅かに垂直な調整を施すことができる。

【０１０１】

ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＭｅｄｉａｎＦｉｌｔｅｒｉｎｇ。欠けているライン内のピクセルはメジアン・フィルタリング操作（median filtering operation）によって再現される。

【０１０２】

ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＥｄｇｅＦｉｌｔｅｒｉｎｇ。欠けているライン内のピクセルはエッジ・フィルタ等によって再生される。この処理では、例えば図形コンテンツのエッジの配向を決定するために空間方向フィルタを適用することができ、（検出されたエッジを横切るのではなく）検出されたエッジに沿ってフィルタリングすることによって、欠けているピクセルが作成される。

【０１０３】

ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＦｉｅｌｄＡｄａｐｔｉｖｅ。例えば動き量によって空間補間（spatial interpolation）と時相補間（temporal interpolation）のうちどちらを使用するかをフィールド・ベースによりフィールドを切り替えることによって、欠けているライン内のピクセルを再現することができる。

【０１０４】
ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＰｉｘｅｌＡｄａｐｔｉｖｅ。例えば動き量によって、空間補間と時相補間の使用のうちどちらを使用するかをピクセル・ベースでピクセルを切り替えることによって、欠けているライン内のピクセルを再現することができる。
【０１０５】
ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＴｅｃｈ＿ＭｏｔｉｏｎＶｅｃｔｏｒＳｔｅｅｒｅｄ。運動ベクトル・ステアリングは、ビデオ・フィールドのシーケンス内のオブジェクトを識別する。場面の個々のオブジェクトの移動軸が時間軸と平行になるように最初に移動軸を調整した後で、欠けているピクセルが再現される。
【０１０６】
ストリームの作成：
例：「ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍ」オブジェクト
適切なインターレース解除モードＧＵＩＤが見つかった後で、ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍオブジェクトを作成することができる。ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍ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３２０が、要求されたインターレース解除または他の選択された動作を実行するために必要となる任意のハードウェア資源（例えばＧＰＵ３２２等に関連付けられたもの）を確保することが可能になる。
【０１０７】
オープンストリームの作成：「ＤｅｉｎｔｅｒｌａｃｅＯｐｅｎＳｔｒｅａｍ」
ＤｅｉｎｔｅｒｌａｃｅＯｐｅｎＳｔｒｅａｍメソッドは、ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍオブジェクトを作成する。例えば以下のものを検討されたい。

```
HRESULT
DeinterlaceOpenStream(
      [in] LPGUID pGuidDeinterlaceMode,
      [in] LPDXVA_VideoDesc lpVideoDescription,
      [out] HDXVA_DeinterlaceStream* lphDiStrm
      );
```

【０１０８】
ここで、ＨＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍ出力パラメータは、ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍオブジェクトへのハンドルであり、それ以降に発生するすべてのコールでストリームを識別するために使用することができる。したがって、以下の例を検討されたい。

```
typedef struct_DXVA_VideoSample {
     REFERENCE_TIME rtStart;
     REFERENCE TIME rtEnd;

     DXVA_SampleFormat SampleFormat;
     LPVOTD lpDDSSrcSurface;
} DXVA_VideoSample, *LPDXVA_VideoSample;
```

ここで、ビデオ・サンプルが２つのインターリーブされたフィールドを収容している場合、
DXVA_SampleFieldInterleavedEvenFirst, or
DXVA_SampleFieldInterleavedOddFirst,
また、第２のフィールドの開始時間は以下の要領で計算することができる。

```
rtStartSecondField = (rtStart + rtEnd) / 2;
```

【０１０９】
上記の典型的なケースでは、第１のフィールドの終了時間が第２のフィールドの開始時間であることに留意されたい。

【０１１０】
インターレース解除の例：「ＤｅｉｎｔｅｒｌａｃｅＢｌｔ」
ＤｅｉｎｔｅｒｌａｃｅＢｌｔメソッドは、出力を宛先サーフェスに書き込むことによってインターレース解除またはフレーム・レート変換動作を実行する。したがって、以下のものを検討されたい。

```
HRESULT
DeinterlaceBlt(
[in] HDXVA_DeinterlaceStream hDiStrm
     [in] REFERENCE_TIME rtTargetFrame,
     [in] LPRECT lprcDstRect,
     [in] LPDDSURFACE lpDDSDstSurface,
     [in] LPRECT lprcSrcRect,
     [in] LPDXVA_VideoSample lpDDSrcSurfaces,
     [in] DWORD dwNumSurfaces,
     [in] FLOAT fAlpha /*0.0F transparent, 1.0F opaque */
     );
```

【０１１１】
ＤｅｉｎｔｅｒｌａｃｅＢｌｔメソッドでは、ｒｔＴａｒｇｅｔＦｒａｍｅパラメータは、入力フレームのシーケンス内における出力フレームの位置を識別する。インターレース解除が実行されているだけであれば、目標時間は参照サンプルのｒｔＳｔａｒｔ時間の１つと一致することになる。フレーム・レート変換が要求されている場合、ｒｔＴａｒｇｅｔＦｒａｍｅ時間は基準サンプルのｒｔＳｔａｒｔ時間のどれとも異なる場合がある。
【０１１２】
ここで、副長方形のインターレース解除または伸張には、ソース長方形および宛先長方形が必要となる。伸張をサポートすることは任意選択である（また、例えばＣＡＰｓフラグによって報告することができる）。副長方形をサポートすることは、ある種の実装では必要でない場合がある。
【０１１３】
宛先サーフェスは、Ｄｉｒｅｃｔ３Ｄ（登録商標）画面外平面、Ｄｉｒｅｃｔ３Ｄ（登録商標）レンダー・ターゲット、Ｄｉｒｅｃｔ３Ｄ（登録商標）テクスチャ等であってよく、これらはレンダーの目標でもある。ある種の典型的な実装では、宛先サーフェスはローカル・ビデオ・メモリ内に割り当てられることになる。
【０１１４】
ＹＵＶ－ＲＧＢ色空間変換がインターレース解除手続きの一部として実行されない限り、宛先サーフェスのピクセル形式は、ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＣａｐｓ構造内で示されるものの１つとなる。この典型的なケースでは、宛先サーフェス形式は、各色成分に対して少なくとも８ビットの精度を有するＲＧＢ形式であってよい。
【０１１５】
ストリームを閉じる：「ＤｅｉｎｔｅｒｌａｃｅＣｌｏｓｅＳｔｒｅａｍ」
ＤｅｉｎｔｅｒｌａｃｅＣｌｏｓｅＳｔｒｅａｍメソッドは、ＤｅｉｎｔｅｒｌａｃｅＳｔｒｅａｍオブジェクトを閉じ、このストリームに関連付けられたハードウェア資源をどれも解放するようデバイス・ドライバに指示する。例えば、以下のものを検討されたい。

```
HRESULT
DeinterlaceCloseStream(
     HDXVA_DeinterlaceStream hDiStrm
     );
```

【０１１６】
インターレース解除インターフェースに対するデバイス・ドライバ・インターフェース（ＤＤＩ）のマッピング

Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システムに関してＤＤＩインフラストラクチャとの互換性のために、本明細書で前述し、提案されたＡＰＩおよびそれらに類似の他のインターフェースは、ＤｉｒｅｃｔＤｒａｗ（登録商標）およびＤｉｒｅｃｔＸ（登録商標）ＶＡに対する既存のＤＤＩに「マッピング」することができる。本節では、既存のＤｉｒｅｃｔＤｒａｗ（登録商標）およびＤＸ－ＶＡ　ＤＤＩに対する典型的なインターレース解除インターフェースのマッピングを説明する。

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離する。

ＤＸ－ＶＡコンテナＤＤＩグループの目的は、ディスプレイ・ハードウェアに収容されている様々なＤＸ－ＶＡデバイスの数と能力を決定することである。したがって、ＤＸ－ＶＡドライバは、複数のＤＸ－ＶＡデバイスをサポートしながらも、単一コンテナを有することのみを必要とする。

ＤＸ－ＶＡコンテナ・グループのどのＤＤＩエントリーポイントに対してもインターレース解除デバイス「照会」コールをマッピングすることはできない。何故ならば、それ以外のＤＸ－ＶＡと異なり、このコンテナ・メソッドは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。したがって提案されたインターレース解除インターフェースをそのグループのメソッドにマッピングすることが可能である。以下の本節では、本明細書に記載の新しいインターフェースがＤＸ－ＶＡデバイスＤＤＩにどのようにマッピングされ得るか、を説明する。

【０１１７】

インターレース解除コンテナ・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。したがってこのメソッドは多くの異なる目的のため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デバイスのコンテキストにおいて使用することのみが可能である。したがって特別な「インターレース解除コンテナ・デバイス」を最初に定義し、作成することが必要となる。本明細書で使用されるように、ＤＸ－ＶＡインターレース解除コンテナ・デバイスはソフトウェア構成のみであり、一般的なデバイスに収容されるいかなる機能的なハードウェアも表すものではない。本明細書で後述するインターレース解除サンプル・デバイス・ドライバ・コードは、このコンテナ・デバイスがどのようにしてドライバによって実装され得るか、を示している。

【０１１８】

ユーザ・モード構成要素からのＤＤＩの呼び出し

レンダラー３１０などのユーザ・モード構成要素からＤＤＩ３１４を使用するためのステップのシーケンスは、以下の通りである。

グラフィックス・デバイス・ドライバ３２０によってサポートされるＤＸ－ＶＡデバイスのリストを入手するためにＧｅｔＭｏＣｏｍｐＧｕｉｄｓを呼び出す。

「インターレース解除コンテナ・デバイス」ＧＵＩＤが存在する場合、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、例えば以下のように定義することができる。

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

【０１１９】

ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓ動作を識別するｄｗＦｕｎｃｔｉｏｎパラメータよって、ＲｅｎｄｅｒＭｏｃｏｍｐを呼び出す。グラフィックス・デバイス・ドライバ３２０に入力パラメータを伝えるためにｌｐＩｎｐｕｔＤａｔａパラメータを使用することができ、グラフィックス・デバイス・ドライバ３２０は、その出力をｌｐＯｕｔｐｕｔＤａｔａパラメータによって戻す。

【０１２０】

ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅＣａｐｓ動作を識別するｄｗＦｕｎｃｔｉ

ｏｎパラメータによって、ＲｅｎｄｅｒＭｏｃｏｍｐを呼び出す。ここでもまた、グラフィックス・デバイス・ドライバ３２０に入力パラメータを伝えるためにｌｐＩｎｐｕｔＤａｔａパラメータを使用することができ、グラフィックス・デバイス・ドライバ３２０は、その出力をｌｐＯｕｔｐｕｔＤａｔａパラメータによって戻す。

【０１２１】

レンダラー３１０は、所望のインターレース解除デバイスを選択した後で、ＣｒｅａｔｅＭｏｃｏｍｐを呼び出し、このインターレース解除デバイスのインスタンスを作成する。

【０１２２】

次いでレンダラー３１０は、各インターレース解除動作に対してＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＢｌｔＦｎＣｏｄｅの機能パラメータによってインターレース解除デバイスのＲｅｎｄｅｒＭｏｃｏｍｐを呼び出す。

【０１２３】

レンダラー３１０は、これ以上インターレース解除動作の実行を必要としなくなるとＤｅｓｔｒｏｙＭｏｃｏｍｐを呼び出す。

【０１２４】

グラフィックス・デバイス・ドライバ３２０は、インターレース解除デバイスによって使用されているいかなる資源も解放する。

【０１２５】

ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓ

この典型的メソッドは、インターレース解除コンテナ・デバイスのＲｅｎｄｅｒＭｏＣｏｍｐメソッドへのコールに対して直接的にマッピングする。ＤＤ＿ＲＥＮＤＥＲＭＯＣＯＭＰＤＡＴＡ構造は以下の要領で完成することができる。

ｄｗＮｕｍＢｕｆｆｅｒｓはゼロ。

ｌｐＢｕｆｆｅｒＩｎｆｏはＮＵＬＬ。

ｄｗＦｕｎｃｔｉｏｎは、ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＡｖａｉｌａｂｌｅＭｏｄｅｓＦｎＣｏｄｅとして定義される

ｌｐＩｎｐｕｔＤａｔａは、完全なＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃ構造を指し示すことになる。

ｌｐＯｕｔｐｕｔＤａｔａは、以下の構造を指し示す。

```
#define MAX_DEINTERLACE_DEVICE_GUIDS 32
typedef struct_DXVA_DeinterlaceQueryAvailableModes {
    DWORDSize;
    DWORD NumGuids;
    GUID Guids[MAX_DEINTERLACE_DEVICE_GUIDS];
} DXVA_DeinterlaceQueryAvailableModes;
```

【０１２６】

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

【０１２７】

ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅＣａｐｓ

この典型的なメソッドは、インターレース解除コンテナ・デバイスのＲｅｎｄｅｒＭｏＣｏｍｐメソッドへのコールに対して直接的にマッピングする。ＤＤ＿ＲＥＮＤＥＲＭＯＣＯＭＰＤＡＴＡ構造は以下の要領で完成することができる。　ｄｗＮｕｍＢｕｆｆｅｒｓはゼロ。

ｌｐＢｕｆｆｅｒＩｎｆｏはＮＵＬＬ。

ｄｗＦｕｎｃｔｉｏｎは、ＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏｄｅＣａｐｓＦｎＣｏｄｅとして定義される。

ｌｐＩｎｐｕｔＤａｔａは、以下のＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＱｕｅｒｙＭｏ

ｄｅＣａｐｓ構造を指し示すことになる。

```
typedef struct _DXVA_DeinterlaceQueryModeCaps {
    DWORD Size;
    GUID Guid;
    DXVA_VideoDesc VideoDesc;
} DXVA_DeinterlaceQueryModeCaps;
```

【０１２８】

ここで、ｌｐＯｕｔｐｕｔＤａｔａはＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＣａｐｓ構造を指し示すことになる。ＤＸ－ＶＡコンテナ・デバイスのＲｅｎｄｅｒＭｏｃｏｍｐメソッドは、ＢｅｇｉｎＭｏＣｏｍｐＦｒａｍｅまたはＥｎｄＭｏＣｏｍｐＦｒａｍｅを最初に呼び出さずに呼び出すことができるということに留意されたい。

【０１２９】

ＤｅｉｎｔｅｒｌａｃｅＯｐｅｎＳｔｒｅａｍ

この典型的なメソッドは、ＧＵＩＤが要求されたインターレース解除のタイプであり、ｐＵｎｃｏｍｐＤａｔａがデータを収容していない（すべてがゼロの）構造を指し示しており、ｐＤａｔａがＤＸＶＡ＿ＶｉｄｅｏＤｅｓｃ構造を指し示している場合、ＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＣｒｅａｔｅＭｏＣｏｍｐメソッドに直接的にマッピングする。

【０１３０】

ドライバが、圧縮されたビデオの加速式復号（accelerated decoding）をサポートする場合、レンダラー３１０は２つのＤＸ－ＶＡデバイスを作成するためにドライバを呼び出すことができる。この２つのデバイスの１つは、ＤｉｒｅｃｔＸ（登録商標）ＶＡビデオ復号基準によって規定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のものであり、もう１つは、インターレース解除に使用されるべきものである。

【０１３１】

例：ＤｅｉｎｔｅｒｌａｃｅＯｐｅｎＳｔｒｅａｍへのＣｒｅａｔｅＭｏＣｏｍｐのマッピング

以下の典型的なサンプル・コードは、ドライバが、ＣｒｅａｔｅＭｏＣｏｍｐＤＤＩコールをＤｅｉｎｔｅｒｌａｃｅＯｐｅｎＳｔｒｅａｍへのコールにどのようにマッピングすることができるかを示している。このサンプル・コードは、インターレース解除にＣｒｅａｔｅＭｏｃＣｏｍｐ機能がどのように使用されるかだけを示している。このドライバが、ＭＰＥＧ－２ビデオ・ストリームの復号などの他のＤＸ－ＶＡ機能もサポートしている場合、以下のサンプル・コードは追加のＤＸ－ＶＡ　ＧＵＩＤの処理を含めるように拡張することができる。

```
DWORD APIENTRY CreateMoComp(PDD_CREATEMOCOMPDATA lpData)
{
// DXVA_DeinterlaceStream is data structure defined by the driver
// to store any data required for the driver
// to de-interlace this particular video data
//
   LPDXVA_DeinterlaceStream pDXVA_State = NULL;

   // Make sure it's a guid we like.
   if (FAILED(ValidDXVAGuid(lpData->lpGuid))) {
      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->IpDriverReserved l =

    (LPVOID)(DXVA_DeviceBaseClass*)lpDev;
      return DDHAL_DRIVER_HANDLED;
   }
   // Look for the deinterlace BOB device GUID
   if (*lpData->lpGuid == DXVA_DeinterlaceBobDevice) {
      DXVA_DeinterlaceBobDeviceClass* IpDev =
        new DXVA_DeinterlaceBobDeviceClass(*lpData->lpGuid,

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

   lpVideoDescription);
         if (lpData->ddRVal != DD_OK) {
            delete lpDev;
            lpDev = NULL;
         }
      }
      else lpData->ddRVal = E_OUTOFMEMORY;
      lpData->lpMoComp->lpDriverReserved 1 =

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

【０１３２】

例：「ＧｅｔＭｏＣｏｍｐＧｕｉｄｓ」の実装

ＣｒｅａｔｅＭｏＣｏｍｐＤＤＩ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。以下の典型的なサンプル・コードは、ドライバ内にこの機能を実装する１つの可能な方法を示している。

```
   // This is the list of all DV-VA device GUIDs supported by
   // the driver - this list will include decoder, de-interlacing and
   // the de-interlacing container device. There is no significance to
```

```
   // the order of the GUIDs on the list.
   //
   DWORD g_dwDXVANumSupportedGUlDs = 4;
   const GUID* g_DXVASupportedGUIDs[4] = {
      &DXVA_DeinterlaceContainerDevice,
      &DXVA_Deinterlace3Samples,
      &DXVA_Deinterlace2Samples,
      &DXVA_DeinterlaceBobDevice
   };

   //
   // This is the list of de-interlacing devices, in order of
   // visual quality.
   //
   DWORD g_dwDXVADeinterlaceNumSupportedGUlDs = 3;
   const GUID* g_DXVADe-interlacedSupportedGUlDs[3] = {
   // replace this GUID with the own implementation
      &DXVA_Deinterlace3Samples,
   // replace this GUID with the own implementation
      &DXVA_Deinterlace2Samples,
   // everyone has to support BOB
      &DXVA_DeinterlaceBobDevice
   };

   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_dwDXVANumSupportedGUlDs,
                        lpData->dwNumGuids);
      for (DWORD i = 0; i < dwNumToCopy; i++) {
         lpData->lpGuids[i] = *g_DxVASupportedGUIDs[i];
      }
   }
   else {
      dwNumToCopy = g_dwDXVANumSupportedGUlDs;
   }

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

【０１３３】

ＤｅｉｎｔｅｒｌａｃｅＢｌｔ

この典型的なメソッドは、ＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＲｅｎｄｅｒＭｏＣｏｍｐメソッドに直接的にマッピングする。ここで、

ｄｗＮｕｍＢｕｆｆｅｒはソース・サーフェスの数＋１である。

ｌｐＢｕｆｆｅｒＩｎｆｏはサーフェスのアレイを指し示す。第１のサーフェスは宛先サーフェスであり、残りのサーフェスはソース・サーフェスである。

ｄｗＦｕｎｃｔｉｏｎはＤＸＶＡ＿ＤｅｉｎｔｅｒｌａｃｅＢｌｔＦｎＣｏｄｅとして定義されている。

ｌｐＩｎｐｕｔＤａｔａは以下の構造を指し示すことになる。

```
#define MAX_DEINTEREACE_INPUT_SURFACES 32
typedef struct _DXVA_DeinterlaceBlt
{
     DWORD Size;
     REFERENCE_TIME rtTargetFrame;
     RECT DstRect;
     RECT SrcRect;
     DWORD NumSourceSurfaces;
     FLOAT fAlpha;
     DXVA_VideoSample
Source[MAX_DEINTERLACE_INPUT_SURFACES];
} DXVA_DeinterlaceBlt;
lpOutoutData is NULL.
```

【０１３４】

インターレース解除のために使用されるＤＸ－ＶＡデバイスの場合、ＢｅｇｉｎＭｏＣｏｍｐＦｒａｍｅまたはＥｎｄＭｏｃｏｍｐＦｒａｍｅを呼び出さずにＲｅｎｄｅｒＭｏＣｏｍｐが呼び出されることになることに留意されたい。

【０１３５】

例：ＤｅｉｎｔｅｒｌａｃｅＢｌｔへのＲｅｎｄｅｒＭｏｃｏｍｐのマッピング

以下の典型的なサンプル・コードは、ドライバが、ＲｅｎｄｅｒＭｏＣｏｍｐＤＤＩコールをＤｅｉｎｔｅｒｌａｃｅＢｌｔへのコールにどのよう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、インターレース解除にＲｅｎｄｅｒＭｏＣｏｍｐ機能がどのように使用されるかだけを示している。このドライバが、ＭＰＥＧ－２ビデオ・ストリームの復号などの他のＤＸ－ＶＡ機能もサポートしている場合、サンプル・コードは追加のＤＸ－ＶＡ　ＧＵＩＤの処理を含めるために拡張することができる。

```
DWORD APIENTRY
RenderMoComp(
   PDD_RENDERMOCOMPDATA lpData
   )
{
   LPDXVA_DeinterlaceStream pDXVAState =
      (LPDXVA_DeinterlaceStream)lpData->lpMoComp
>lpDriverReserved1;

   DXVA_DeinterlaceBlt* lpBlt =
                     (DXVA_DeinterlaceBlt*)lpData->lpInputData;
   LPDDMOCOMPBUFFERINFO lpBuffInfo = lpData->BufferInfo;

for (DWORD i =0; i <lpBlt->NumSourceSurfaces; i++) {
   lpBlt->Source[i].lpDDSSrcSurface =
                           lpBuffInfo[l + i].lpCompSurface;
}
```

```
lpData->ddRVal = DeinterlaceBlt(pDXVAState,
                 lpBlt->rtTarget,
                 &lpBlt->DstRect,
                 lpBuffInfo[0].lpCompSurface,
                 &lpBlt->SrcRect,
                 &lpBlt->Source,
                 lpBlt->NumSourceSurfaces,
                 lpBlt->Alpha);

   return DDHAL_DRIVER_HANDLED;
}
```

【０１３６】

ＤｅｉｎｔｅｒｌａｃｅＣｌｏｓｅＳｔｒｅａｍ

この典型的なメソッドは、ＤＤ＿ＭＯＴＩＯＮＣＯＭＰＣＡＬＬＢＡＣＫＳ構造のＤｅｓｔｒｏｙＭｏＣｏｍｐメソッドに直接的にマッピングする。

【０１３７】

例：ＤｅｉｎｔｅｒｌａｃｅＣｌｏｓｅＳｔｒｅａｍへのＤｅｓｔｒｏｙＭｏＣｏｍｐのマッピング

以下の典型的なサンプル・コードは、ドライバが、ＤｅｓｔｒｏｙＭｏＣｏｍｐＤＤＩコールをＤｅｉｎｔｅｒｌａｃｅＣｌｏｓｅＳｔｒｅａｍへのコールにどのようにマッピングすることができるかを示している。このサンプル・コードは、インターレース解除にＤｅｓｔｒｏｙＭｏＣｏｍｐ機能がどのように使用されるかだけを示している。このドライバが、ＭＰＥＧ－２ビデオ・ストリームの復号などの他のＤＸ－ＶＡ機能もサポートしている場合、以下のサンプル・コードは追加のＤＸ－ＶＡ　ＧＵＩＤの処理を含めるために拡張することができる。DWORD APIENTRY

```
DestroyMoComp(
   PDD_DESTROYMOCOMPDATA lpData
   )
{
   LPDXVA_DeinterlaceStream pDXVAState =
      (LPDXVA_DeinterlaceStream)lpData->lpMoComp-
>lpDriverReserved 1;

   lpData->ddRVal = DeinterlaceCloseStream(pDXVAState);
   lpData->lpMoComp->lpDnverReserved 1 = NULL;

   return DDHAL_DRIVER_HANDLED;
}
```

【０１３８】

結論

本明細書に記載の様々な典型的実装により、本発明は、コンピュータのモニタまたは他の類似の表示装置上に正確に表示することができるように、ビデオ画像データのインターレース解除の問題点に対処する。指摘したように、従来型インターレース解除技術は、通常、複数の制約および制限を有するグラフィックス・プロセッサによる「グラフィックス・オーバーレイ・デバイス」の使用を必要とする。本明細書に記載の様々な方法および装置を使用することによって、例えば、コンピュータのモニタ上に表示することのできる単一ビデオ・フレームを作成するためにビデオ・フィールドをどのようにインターレース解除するかに関してグラフィックス・プロセッサに指示することができ、その結果、インターレースされたビデオがリアルタイムで正確に表示される。さらなる例として様々なＡＰ

Ｉも示したが、その一部は、グラフィックス・デバイス・ドライバにおいてビデオ・コンテンツ用のインターレース解除および／またはフレーム・レート変換をサポートするために特にＭｉｃｒｏｓｏｆｔ（登録商標）ＤｉｒｅｃｔＸ（登録商標）ＶＡを拡張する。

【０１３９】

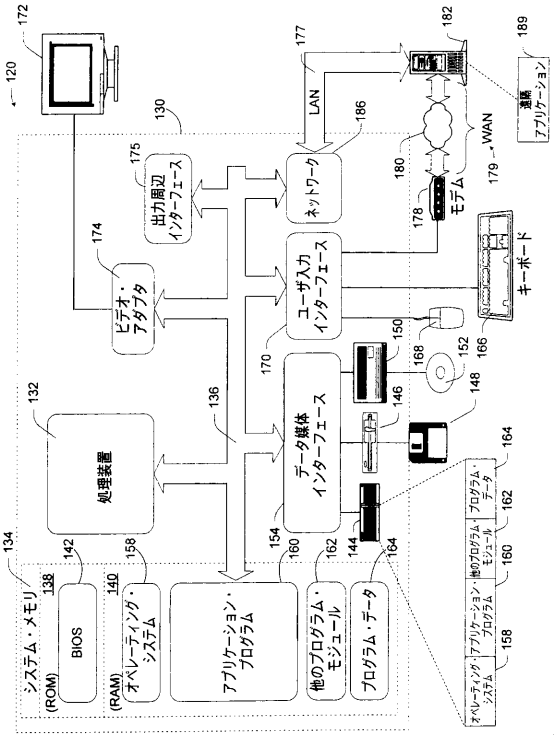
上記の説明では構造上の特徴および／または方法論的動作（methodlogical acts）に特有の用語を使用しているが、頭記の特許請求の範囲で規定された本発明は、記載された特有の特徴および動作に限定されるものではないということが理解されよう。そうではなく、この特有の特徴および動作は、本発明を実装する上での典型的な形式として開示されたものである。

【符号の説明】

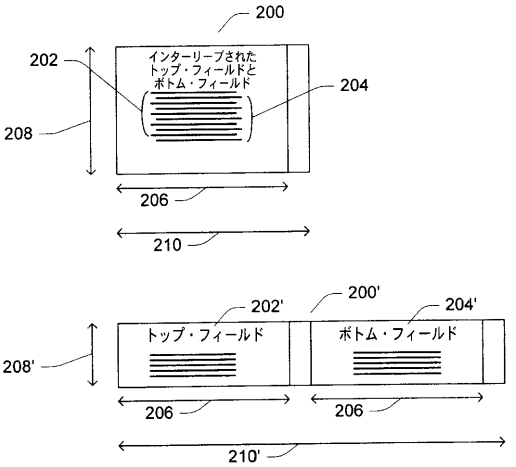
【０１４０】

１７２　ディスプレイ・デバイス
３００　本発明のある種の典型的な装置
３０２　変換ロジック
３０４　１つのソース
３０６　インターネット
３０８　遠隔ソース
３１０　レンダラー
３１２　グラフィックス・インターフェース・ロジック
３１４　デバイス・ドライバ・インターフェース
３３２　グラフィックス・デバイス

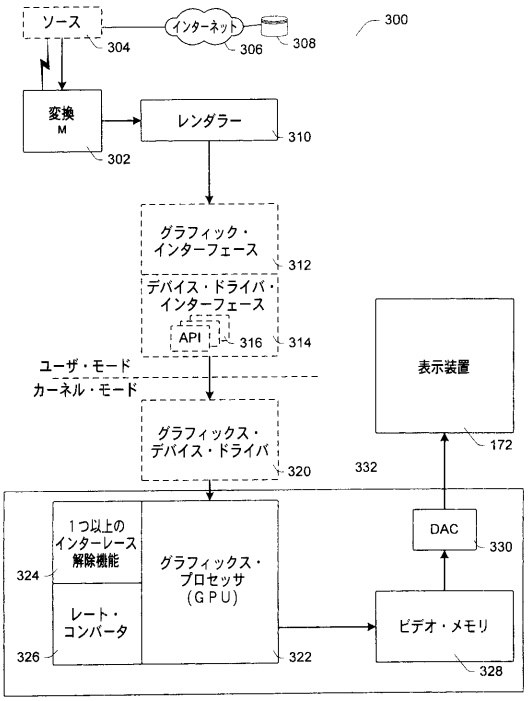
【図１】



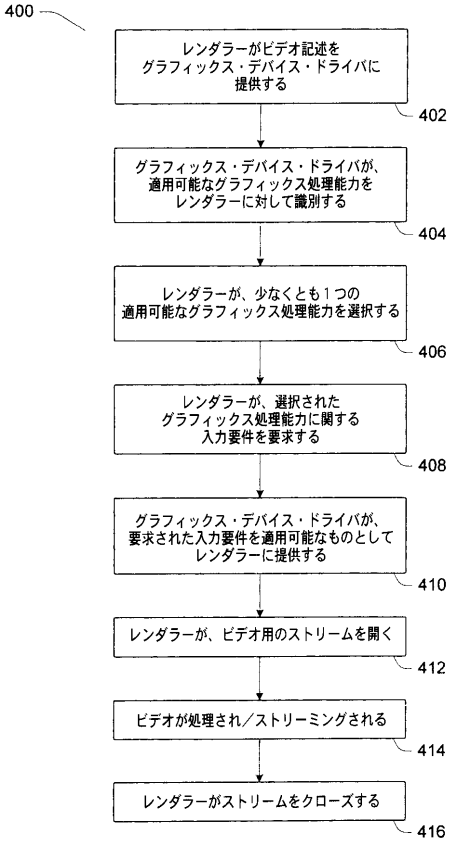
【図２】



【図３】


ソース
304
インターネット
306
308
300
変換
M
302
レンダラー
310
グラフィック・
インターフェース
312
デバイス・ドライバ・
インターフェース
API
316
314
ユーザ・モード
カーネル・モード
表示装置
172
グラフィックス・
デバイス・ドライバ
320
332
1つ以上の
インターレース
解除機能
324
レート・
コンバータ
326
グラフィックス・
プロセッサ
(GPU)
322
DAC
330
ビデオ・メモリ
328

【図４】


400
レンダラーがビデオ記述を
グラフィックス・デバイス・ドライバに
提供する
402
グラフィックス・デバイス・ドライバが、
適用可能なグラフィックス処理能力を
レンダラーに対して識別する
404
レンダラーが、少なくとも1つの
適用可能なグラフィックス処理能力を選択する
406
レンダラーが、選択された
グラフィックス処理能力に関する
入力要件を要求する
408
グラフィックス・デバイス・ドライバが、
要求された入力要件を適用可能なものとして
レンダラーに提供する
410
レンダラーが、ビデオ用のストリームを開く
412
ビデオが処理され／ストリーミングされる
414
レンダラーがストリームをクローズする
416

フロントページの続き

(56)参考文献 特開２０００－０１０５３５（ＪＰ，Ａ）
特開２０００－３３１１５０（ＪＰ，Ａ）
特開２０００－３１１２４０（ＪＰ，Ａ）
特開２０００－２９８５６５（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｈ０４Ｎ　　７／０１
Ｇ０９Ｇ　　３／００
Ｇ０９Ｇ　　５／００