(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6541685号
(P6541685)
(45)発行日　令和1年7月10日(2019.7.10)　(24)登録日　令和1年6月21日(2019.6.21)

(51)Int.Cl.　F I
G06T 15/00 (2011.01)　G06T 15/00 501

請求項の数 13 (全 27 頁)

(21)出願番号	特願2016-562910 (P2016-562910)
(86)(22)出願日	平成27年4月21日 (2015.4.21)
(65)公表番号	特表2017-516207 (P2017-516207A)
(43)公表日	平成29年6月15日 (2017.6.15)
(86)国際出願番号	PCT/US2015/026910
(87)国際公開番号	WO2015/164397
(87)国際公開日	平成27年10月29日 (2015.10.29)
審査請求日	平成30年4月5日 (2018.4.5)
(31)優先権主張番号	61/982,147
(32)優先日	平成26年4月21日 (2014.4.21)
(33)優先権主張国	米国(US)
(31)優先権主張番号	14/691,358
(32)優先日	平成27年4月20日 (2015.4.20)
(33)優先権主張国	米国(US)

(73)特許権者　507364838
クアルコム，インコーポレイテッド
アメリカ合衆国　カリフォルニア　９２１２１　サン　ディエゴ　モアハウス　ドライブ　５７７５
(74)代理人　100108453
弁理士　村山　靖彦
(74)代理人　100163522
弁理士　黒田　晋平
(72)発明者　ムラット・バルチ
アメリカ合衆国・カリフォルニア・９２１２１－１７１４・サン・ディエゴ・モアハウス・ドライヴ・５７７５

最終頁に続く

(54)【発明の名称】グラフィックス処理におけるレンダーターゲットに基づいたフレックスレンダリング

(57)【特許請求の範囲】
【請求項１】
グラフィックス処理の方法であって、
グラフィックス処理ユニット(GPU)によって、前記GPUがフレームの少なくとも一部分をレンダリングするためにダイレクトレンダリングモードおよびビニングレンダリングモードからレンダリングモードを選択してもよいことを示すGPUコマンドパケットを受信するステップと、
前記GPUによって、前記受信されたGPUコマンドパケット内の情報に基づいて、前記フレームの少なくとも前記一部分をレンダリングするために前記レンダリングモードを選択するステップであって、前記受信されたGPUコマンドパケット内で受信された前記情報は、1つまたは複数の第2のレベルの間接バッファへのポインタを含むシーン記述子を備える、ステップと、
前記GPUによって、前記シーン記述子によって参照される1つまたは複数の第2のレベルの間接バッファへの前記ポインタを使用して、実行順序を生成するステップと、
前記GPUによって、前記選択されたレンダリングモードに対する一連の実行コマンドを使用して前記フレームの少なくとも前記一部分をレンダリングするステップと
を含む、方法。
【請求項２】
前記GPUによって、前記受信されたGPUコマンドパケット内の前記情報および前記GPUの状態に基づいて、前記フレームの少なくとも前記一部分をレンダリングするために前記複

数のレンダリングモードから前記レンダリングモードを選択するステップであって、前記GPUの前記状態が、前記GPUの電力消費量、前記GPUの処理負荷、前記GPUのメモリ使用量、または前記GPUの利用負荷のうちの少なくとも1つに対応する情報を含む、ステップをさらに含む、請求項1に記載の方法。

【請求項３】

前記GPUの電力消費量がしきい値電力消費値を超えることを示す前記GPUの状態に基づいて、ビニングレンダリングモードを選択するステップ

をさらに含む、請求項1に記載の方法。

【請求項４】

前記GPUの利用負荷が利用値を超えることを示す前記GPUの状態に基づいて、ビニングレンダリングモードを選択するステップ

をさらに含む、請求項1に記載の方法。

【請求項５】

前記シーン記述子が、構成ペアを含む、請求項1に記載の方法。

【請求項６】

前記シーン記述子が、前記フレームの少なくとも前記一部分についての解像度情報を含む、請求項1に記載の方法。

【請求項７】

前記受信されたGPUコマンドパケット内の前記情報が、前記フレームの少なくとも前記一部分をレンダリングするために必要とされるすべてのビンを含む、請求項1に記載の方法。

【請求項８】

前記方法が、

前記フレームの少なくとも前記一部分をレンダリングするときに生じるオーバードローの量を決定するステップと、

生じる前記オーバードローの量に基づいて前記レンダリングモードを選択するステップと

をさらに含む、請求項1に記載の方法。

【請求項９】

グラフィックス処理ユニット(GPU)を備えるデバイスであって、前記GPUが、

前記GPUがフレームの少なくとも一部分をレンダリングするために複数のレンダリングモードからレンダリングモードを選択してもよいことを示すGPUコマンドパケットを受信するための手段であって、前記複数のレンダリングモードが、ダイレクトレンダリングモードおよびビニングレンダリングモードを含む、手段と、

前記受信されたGPUコマンドパケット内の情報に基づいて、前記フレームの少なくとも前記一部分をレンダリングするために前記複数のレンダリングモードから前記レンダリングモードを選択するための手段であって、前記受信されたGPUコマンドパケット内で受信された前記情報は、1つまたは複数の第2のレベルの間接バッファへのポインタを含むシーン記述子を備える、手段と、

前記選択されたレンダリングモードを使用して前記フレームの少なくとも前記一部分をレンダリングするための手段と

をさらに備える、デバイス。

【請求項１０】

前記GPUの電力消費量がしきい値電力消費値を超えることを示す前記GPUの状態に基づいて、ビニングレンダリングモードを選択するための手段

をさらに備える、請求項9に記載のデバイス。

【請求項１１】

前記GPUの利用負荷が利用値を超えることを示す前記GPUの状態に基づいて、ビニングレンダリングモードを選択するための手段

をさらに備える、請求項9に記載のデバイス。

【請求項１２】
前記受信されたGPUコマンドパケット内の前記情報および前記GPUの状態に基づいて、前記フレームの少なくとも前記一部分をレンダリングするために前記複数のレンダリングモードから前記レンダリングモードを選択するための手段をさらに備え、前記GPUの前記状態が、前記GPUの電力消費量、前記GPUの処理負荷、前記GPUのメモリ使用量、または前記GPUの利用負荷のうちの少なくとも1つに対応する情報を含む、請求項9に記載のデバイス。
【請求項１３】
実行されると、GPUの少なくとも1つのプロセッサに、請求項1～8のいずれか一項に記載の方法を行わせる命令を記憶する、非一時的コンピュータ可読記憶媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本出願は、その内容全体が参照により本明細書に組み込まれる、2014年4月21日に出願された米国仮出願第61/982,147号の優先権を主張する。
【０００２】
本開示は、グラフィックス処理のための技法に関する。
【背景技術】
【０００３】
グラフィカルユーザインターフェースおよびビデオゲーム用のコンテンツなど、ディスプレイ用の視覚コンテンツは、グラフィックス処理ユニット(GPU)によって生成される場合がある。GPUは、2次元または3次元(3D)オブジェクトを、表示される場合がある2次元(2D)ピクセル表現に変換する場合がある。3Dオブジェクトについての情報を表示できるビットマップに変換することは、ピクセルレンダリングとして知られており、相当なメモリおよび処理能力を必要とする。これまでは、3Dグラフィックス能力は高性能のワークステーションでのみ利用可能であった。しかしながら、今では、3Dグラフィックスアクセラレータは、パーソナルコンピュータ(PC)において、ならびにスマートフォン、タブレットコンピュータ、ポータブルメディアプレーヤ、ポータブルビデオゲームコンソールなどの組込みデバイスにおいてよく見られる。典型的には、組込みデバイスは、従来のPCと比較して、より少ない計算能力およびメモリ容量を有する。したがって、3Dグラフィックスレンダリング技法における複雑さの増大は、そのような技法を組込みシステム上で実装するときの問題点を提示する。
【発明の概要】
【課題を解決するための手段】
【０００４】
一般に、本開示は、GPUがグラフィックス処理において直接レンダリングとビニングレンダリングとの間で切り替えることを可能にする技法、GPUがレンダリングモードを決定することを可能にするための技法、およびGPUでレンダリングループを実行するための技法について説明する。
【０００５】
本開示の一例では、グラフィックス処理の方法は、グラフィックス処理ユニット(GPU)によって、GPUがGPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示すGPUコマンドパケットを受信するステップと、GPUによって、受信されたコマンドパケット内の情報またはGPUの状態のうちの少なくとも1つに基づいて、GPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定するステップと、GPUによって、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングするステップとを含む。
【０００６】
本開示の技法による別の例は、グラフィックス処理ユニット(GPU)を備えるデバイスに

ついて説明する。GPUは、メモリと、少なくとも1つのプロセッサとを含む。少なくとも1つのプロセッサは、GPUがGPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示すGPUコマンドパケットを受信し、受信されたコマンドパケット内の情報またはGPUの状態のうちの少なくとも1つに基づいて、GPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定し、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングするように構成されてもよい。

【０００７】

本開示の技法による別の例は、デバイスについて説明する。デバイスは、GPUがGPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示すGPUコマンドパケットを受信するための手段と、受信されたコマンドパケット内の情報またはGPUの状態のうちの少なくとも1つに基づいて、GPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定するための手段と、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングするための手段とを備える。

【０００８】

本開示の技法による別の例は、その上に記憶された命令を含む非一時的コンピュータ可読記憶媒体について説明する。命令は、実行されると、少なくとも1つのプロセッサに、GPUがGPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示すGPUコマンドパケットを受信させ、受信されたコマンドパケット内の情報またはGPUの状態のうちの少なくとも1つに基づいて、GPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定させ、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングさせてもよい。

【図面の簡単な説明】

【０００９】

【図１】本開示の技法による、特定のレンダリングモードを使用するかどうかを決定するために使用されてもよい例示的なコンピューティングデバイスを示すブロック図である。

【図２】図1のCPU6、GPU12、およびシステムメモリ10の例示的な実装形態をさらに詳細に示すブロック図である。

【図３】ビニングレンダリングモードで使用される場合のフレームのビンを示す概念図である。

【図４】ビニングレンダリングモードで使用される場合のフレームのビンをより詳細に示す概念図である。

【図５】「ソフトウェア」ビニングを使用するビニングレンダリングモードのコマンドバッファを示す概念図である。

【図６】「ハードウェア」ビニングを使用するビニングレンダリングモードのコマンドバッファを示す概念図である。

【図７】直接レンダリングモードのコマンドバッファを示す概念図である。

【図８】異なるレンダリングモードのコマンドバッファを示す概念図である。

【図９】本開示の技法による、直接レンダリングモードまたはビニングレンダリングモードを使用してシーンをレンダリングするための例示的なコマンド構造を示す概念図である。

【図１０】本開示の一例による方法を示すフローチャートである。

【発明を実施するための形態】

【００１０】

本開示は、グラフィックス処理のための技法に関し、より詳細には、グラフィックス処

理システムにおいてGPUがレンダリングモードを決定し、レンダリングモード間で切り替えることを可能にする技法に関する。

【００１１】

現在のグラフィックスレンダリングシステムは、典型的には、シーンをレンダリングするためにビニングレンダリングモード(タイルベースのレンダリングと呼ばれることがある)または直接レンダリングモードを利用する。ビニングレンダリングでは、2Dまたは3Dシーンの1つのフレームは、フレームをより小さい部分(たとえば、矩形のビンまたはタイル)に分け、これらのビンの各々を別々にレンダリングすることによって、レンダリングされる。ビニングレンダリングは、モバイルアプリケーションなど、専用の高速グラフィックスメモリ(GMEM)がほとんど利用できないアプリケーションに有用であり、ならびにメモリ帯域幅が制限されている事例において有用である。タイルのサイズは、GMEMにおいて利用可能なデータの量を表すように構成することが可能である。たとえば、GMEMが512kBを記憶することができる場合、タイルのサイズは、そのタイルに含まれるそのピクセルデータが512kB以下になるように構成されてもよい。

【００１２】

一方、直接レンダリングモードにおけるグラフィックス処理は、フレームをより小さいビンに分けない。代わりに、フレームの全体が一度にレンダリングされる。いくつかのグラフィックス処理システム(たとえば、モバイルデバイス上のグラフィックス処理システム)では、ピクセルデータのフレーム全体を保持するのに十分なGMEMがない。代わりに、直接レンダリングモードでは、ダイナミックランダムアクセスメモリ(DRAM)などのより遅いシステムメモリが、フレームをレンダリングするために使用されてもよい。

【００１３】

以前のGPUコマンドパケットはそれぞれ、単一の間接バッファ(IB)のみを含んでいた。シーンをレンダリングするために、GPUドライバは複数のコマンドパケットを生成し、シーンをレンダリングするために間接バッファごとに1つのコマンドパケットを必要とした。「スモールバッチ」問題と呼ばれる、レンダリングされるべき少数の三角形を各々が含む複数のコマンドパケットを生成するCPUおよびメモリのオーバーヘッドは、非常に高くなる可能性がある。一例として、アプリケーションがドローコール間で最小量の状態しか変更していないとき、GPUドライバは不相応に大量の作業を行う場合がある。本開示の技法は、シーンをレンダリングするために必要なすべてのIBを多数の小さいパケットにではなく単一のコマンドパケットに含めることによって、また、レンダリング負荷の一部をCPU上で実行されるGPUドライバからGPU自体に移行することによって、CPUおよびメモリのオーバーヘッドを低減する場合がある。

【００１４】

加えて、CPUベースのGPUドライバは、CPUおよびGPUに、シーンをレンダリングするために必要なすべてのコマンドパケットを生成するためにレンダリングループを何度もループさせ、そのシーンをレンダリングさせる場合がある。レンダリングループは、レンダリングされるべきフレームのための各個々のビンをレンダリングするためにGPUドライバが実行するループである。本開示の技法は、追加のレンダリング情報、たとえば、コマンドパケットに組み込まれるビンにより、また、レンダリング論理をGPUドライバからGPUに移行することによって、レンダリングループを低減または削減する場合がある。

【００１５】

本開示の技法はまた、GPUがシーンのためのレンダリングモードを決定するために使用する場合がある、よりきめ細かいリアルタイムのGPU状態情報(たとえば、ヒューリスティック)を考慮する。CPUベースのGPUドライバは、GPUからのGPU処理負荷などのGPU状態を要求することができるが、CPU上で実行されるドライバは、ポーリング要求および/またはGPUからの統計値の蓄積を使用してしか、そうすることができない。したがって、CPUドライバがアクセスすることができる任意のGPU状態は正確ではない場合があり、最新ではない場合がある。したがって、GPU状態を決定することに関する本開示の技法は、より正確で最新のGPU状態に基づいたレンダリングモードのより正確な決定を考慮する。

【００１６】
加えて、本開示の技法は、コマンドパケットに、レンダーターゲット情報、および/またはレンダリングされるべきシーンの解像度などのCPUが蓄積したヒューリスティックを含む。GPUがレンダーターゲット情報にアクセスできるので、GPU状態を考慮することなしにGPUが使用するためのレンダリングモードをCPUに決定させるのではなく、GPU自体が、コマンドパケット、ならびにGPU負荷および電力消費量などの他のGPU状態ヒューリスティックに少なくとも部分的に基づいて、直接モードまたはビニングモードを使用してシーンをレンダリングするかどうかを決定してもよい。直接レンダリングモードを使用することは、レンダリングされるべきジオメトリの量が少ないときにレンダリング性能を改善する場合がある一方で、ビニングレンダリングモードは、直接レンダリングモードと比べて電力を節約する場合がある。したがって、GPUが直接レンダリングおよび間接レンダリングから選択することを可能にすることは、電力、性能、および他の制約に基づいてGPUがレンダリングを最適化するための柔軟性をもたらす。
【００１７】
一般に、直接レンダリングモードは、ビニングレンダリングモードよりも遅い場合がある。しかしながら、ビニングレンダリングモードは、いくらかの固有のオーバーヘッドを有する。しかしながら、直接レンダリングモードの負荷が十分に軽い場合、直接レンダリングモードを使用することはビニングレンダリングモードよりも好ましい場合がある。オーバードローの量が少ない、したがって、オーバーヘッドがより少ないとき、ビニングレンダリングモードが好ましい場合がある。オーバードローは、たとえば、重複する三角形により、単一のピクセルがフレームごとにレンダリングされる/更新される回数に対応する。
【００１８】
高いオーバードローは、高い帯域幅利用率および直接レンダリングモード用の(比較的遅い)システムメモリとの対話を引き起こす場合がある。本明細書で説明するヒューリスティックは、非限定的な例として、レンダリングされるピクセル数、オーバードロー、クロック速度、GPUアーキテクチャなどの要因に基づいて、直接レンダリングモードがビニングレンダリングモードよりも速いポイントをGPUが決定することを可能にする場合がある。
【００１９】
本開示の一例では、グラフィックス処理の方法は、GPUによって、GPUがGPUによってレンダリングされるべきフレームの一部分をレンダリングするために直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示すコマンドパケットを受信するステップを含む。GPUは、受信されたコマンドパケットまたはGPU状態のうちの少なくとも1つに基づいて、GPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定してもよい。GPUは、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングしてもよい。
【００２０】
図1は、本開示の技法による、特定のレンダリングモードを使用するかどうかを決定するために使用されてもよい例示的なコンピューティングデバイスを示すブロック図である。コンピューティングデバイス2は、たとえば、パーソナルコンピュータ、デスクトップコンピュータ、ラップトップコンピュータ、タブレットコンピュータ、コンピュータワークステーション、ビデオゲームプラットフォームもしくはコンソール、たとえば、セルラー電話もしくは衛星電話などの携帯電話、固定電話、スマートフォン、ポータブルビデオゲームデバイスもしくは携帯情報端末(PDA)などのハンドヘルドデバイス、パーソナル音楽プレーヤ、ビデオプレーヤ、ディスプレイデバイス、テレビジョン、テレビジョンセットトップボックス、サーバ、中間ネットワークデバイス、メインフレームコンピュータ、任意のモバイルデバイス、またはグラフィカルデータを処理および/もしくは表示する任意の他のタイプのデバイスを含んでもよい。

【００２１】
図1の例に示すように、コンピューティングデバイス2は、ユーザ入力インターフェース4、中央処理ユニット(CPU)6、メモリコントローラ8、システムメモリ10、グラフィックス処理ユニット(GPU)12、グラフィックスメモリ14、ディスプレイインターフェース16、ディスプレイ18、ならびにバス20および22を含んでもよい。いくつかの例では、グラフィックスメモリ14はGPU12に「オンチップ」であってもよいことに留意されたい。場合によっては、図1に示すすべてのハードウェア要素は、たとえば、システムオンチップ(SoC)設計において、オンチップであってもよい。ユーザ入力インターフェース4、CPU6、メモリコントローラ8、GPU12およびディスプレイインターフェース16は、バス20を使用して互いに通信してもよい。メモリコントローラ8およびシステムメモリ10はまた、バス22を使用して互いに通信してもよい。バス20、22は、第3世代バス(たとえば、HyperTransportバスまたはInfiniBandバス)、第2世代バス(たとえば、アドバンストグラフィックスポートバス、 Peripheral Component Interconnect(PCI) Expressバス、または Advanced eXentisible Interface(AXI)バス)または別のタイプのバスもしくはデバイス相互接続などの様々なバス構造のいずれかであってもよい。図1に示す異なる構成要素間のバスおよび通信インターフェースの特定の構成は例にすぎず、同じまたは異なる構成要素を有するコンピューティングデバイスおよび/または他のグラフィックス処理システムの他の構成が本開示の技法を実装するために使用されてもよいことに留意されたい。

【００２２】
CPU6は、コンピューティングデバイス2の動作を制御する汎用または専用プロセッサを備えてもよい。ユーザは、CPU6に1つまたは複数のソフトウェアアプリケーションを実行させるために、入力をコンピューティングデバイス2に与える場合がある。CPU6上で実行されるソフトウェアアプリケーションは、たとえば、オペレーティングシステム、ワードプロセッサアプリケーション、電子メールアプリケーション、スプレッドシートアプリケーション、メディアプレーヤアプリケーション、ビデオゲームアプリケーション、グラフィカルユーザインターフェースアプリケーションまたは別のプログラムを含んでもよい。加えて、CPU6は、GPU12の動作を制御するためのGPUドライバ7を実行する場合がある。ユーザは、ユーザ入力インターフェース4を介してコンピューティングデバイス2に結合されたキーボード、マウス、マイクロフォン、タッチパッドまたは別の入力デバイスなどの1つまたは複数の入力デバイス(図示せず)を介して、入力をコンピューティングデバイス2に与える場合がある。

【００２３】
CPU6上で実行されるソフトウェアアプリケーションは、ディスプレイ18へのグラフィックスデータのレンダリングを行わせるようCPU6に命令する1つまたは複数のグラフィックスレンダリング命令を含んでもよい。いくつかの例では、ソフトウェア命令は、たとえば、オープングラフィックスライブラリ(OpenGL(登録商標))API、オープングラフィックスライブラリ組込みシステム(OpenGL ES)API、Direct3D API、X3D API、RenderMan API、WebGL API、または任意の他の公的もしくは独占的な(proprietary)規格グラフィックスAPIなどのグラフィックスアプリケーションプログラミングインターフェース(API)に準拠する場合がある。命令はまた、様々な例では、OpenCLなどのいわゆるヘテロジニアスコンピューティングライブラリおよび/またはDirectComputeに準拠する場合がある。グラフィックスレンダリング命令を処理するために、CPU6は、GPU12にグラフィックスデータのレンダリングの一部またはすべてを実行させるために、1つまたは複数のグラフィックスレンダリングコマンドをGPU12に(たとえば、GPUドライバ7を通じて)発行する場合がある。いくつかの例では、レンダリングされるべきグラフィックスデータは、グラフィックスプリミティブ、たとえば、点、線、三角形、四角形、三角形ストリップなどのリストを含んでもよい。

【００２４】
メモリコントローラ8は、システムメモリ10を出入りするデータの転送を容易にする。たとえば、メモリコントローラ8は、メモリ読取りコマンドおよびメモリ書込みコマンド

を受信し、メモリサービスをコンピューティングデバイス2内の構成要素に提供するためにメモリシステム10に対してそのようなコマンドをサービスする場合がある。メモリコントローラ8は、メモリバス22を介してシステムメモリ10に通信可能に結合される。メモリコントローラ8は、CPU6とシステムメモリ10の両方とは別の処理モジュールであるものとして図1に示されているが、他の例では、メモリコントローラ8の機能の一部またはすべては、CPU6およびシステムメモリ10の一方または両方で実装される場合がある。

【００２５】

システムメモリ10は、CPU6が実行するためにアクセス可能なプログラムモジュールおよび/もしくは命令ならびに/またはCPU6上で実行されるプログラムが使用するためのデータを記憶してもよい。たとえば、システムメモリ10は、ディスプレイ18上にグラフィカルユーザインターフェース(GUI)を提示するためにCPU6によって使用されるウィンドウマネージャアプリケーションを記憶してもよい。加えて、システムメモリ10は、ユーザアプリケーションと、アプリケーションに関連付けられたアプリケーションサーフェスデータとを記憶してもよい。システムメモリ10は加えて、コンピューティングデバイス2の他の構成要素が使用するためのおよび/またはそれらの構成要素によって生成される情報を記憶してもよい。たとえば、システムメモリ10は、GPU12用のデバイスメモリとして働いてもよく、GPU12による操作を受けるべきデータならびにGPU12によって実行された動作から生じたデータを記憶してもよい。たとえば、システムメモリ10は、テクスチャバッファ、深度バッファ、ステンシルバッファ、頂点バッファ、フレームバッファなどの任意の組合せを記憶してもよい。システムメモリ10は、たとえば、ランダムアクセスメモリ(RAM)、スタティックRAM(SRAM)、ダイナミックRAM(DRAM)、読取り専用メモリ(ROM)、消去可能プログラマブルROM(EPROM)、電気的消去可能プログラマブルROM(EEPROM)、フラッシュメモリ、磁気データ媒体または光学記憶媒体など、1つまたは複数の揮発性もしくは不揮発性メモリまたはストレージデバイスを含んでもよい。

【００２６】

GPU12は、1つまたは複数のグラフィックスプリミティブをディスプレイ18にレンダリングするためのグラフィックス演算を実行するように構成されてもよい。したがって、CPU6上で実行されるソフトウェアアプリケーションのうちの1つがグラフィックス処理を必要とするとき、CPU6はディスプレイ18にレンダリングするためにグラフィックスコマンドおよびグラフィックスデータをGPU12に与えてもよい。グラフィックスデータは、たとえば、描画コマンド、状態情報、プリミティブ情報、テクスチャ情報などを含んでもよい。GPU12は、ある事例では、CPU6よりも効率的な、複雑なグラフィック関連動作の処理を実現する高度並列構造で構築されてもよい。たとえば、GPU12は、複数の頂点またはピクセル上で並行して動作するように構成される複数の処理要素を含んでもよい。GPU12の高度並列の性質は、ある事例では、CPU6を使用してシーンを直接ディスプレイ18に描画するよりも速く、GPU12がグラフィックス画像(たとえば、GUIならびに2次元(2D)および/または3次元(3D)グラフィックスシーン)をディスプレイ18上で描画することを可能にする場合がある。

【００２７】

GPU12は、ある事例では、コンピューティングデバイス2のマザーボードに統合される場合がある。他の事例では、GPU12は、コンピューティングデバイス2のマザーボード中のポートにインストールされたグラフィックスカード上に存在する場合があるか、そうでなければコンピューティングデバイス2と相互動作するように構成される周辺デバイス内に組み込まれる場合がある。GPU12は、1つまたは複数のマイクロプロセッサ、特定用途向け集積回路(ASIC)、フィールドプログラマブルゲートアレイ(FPGA)、デジタル信号プロセッサ(DSP)、または他の等価の集積論理回路もしくはディスクリート論理回路などの1つまたは複数のプロセッサを含んでもよい。

【００２８】

GPU12は、グラフィックスメモリ14に直接結合されてもよい。したがって、GPU12は、バス20を使用することなしに、グラフィックスメモリ14からデータを読み取り、グラフィッ

クスメモリ14にデータを書き込んでもよい。言い換えれば、GPU12は、オフチップメモリの代わりにローカルストレージを使用して、データをローカルに処理してもよい。これは、重いバストラフィックを経験する場合がある、GPU12がバス20を介してデータの読取りおよび書込みを行う必要をなくすことによって、GPU12がより効率的に動作することを可能にする。しかしながら、ある事例では、GPU12は別個のメモリを含まないが、代わりにバス20を介してシステムメモリ10を利用する場合がある。グラフィックスメモリ14は、たとえば、ランダムアクセスメモリ(RAM)、スタティックRAM(SRAM)、ダイナミックRAM(DRAM)、消去可能プログラマブルROM(EPROM)、電気的消去可能プログラマブルROM(EEPROM)、フラッシュメモリ、磁気データ媒体または光学記憶媒体など、1つまたは複数の揮発性もしくは不揮発性メモリまたはストレージデバイスを含んでもよい。

【００２９】

CPU6および/またはGPU12は、レンダリングされた画像データをフレームバッファ15に記憶してもよい。フレームバッファ15は、独立したメモリであってもよく、またはシステムメモリ10内で割り振られてもよい。ディスプレイインターフェース16は、フレームバッファ15からデータを取り出し、レンダリングされた画像データによって表される画像を表示するようにディスプレイ18を構成してもよい。いくつかの例では、ディスプレイインターフェース16は、フレームバッファから取り出されたデジタル値をディスプレイ18が消費できるアナログ信号にコンバートするように構成されるデジタルアナログコンバータ(DAC)を含んでもよい。他の例では、ディスプレイインターフェース16は、処理のためにデジタル値を直接ディスプレイ18に渡してもよい。ディスプレイ18は、モニタ、テレビジョン、投影デバイス、液晶ディスプレイ(LCD)、プラズマディスプレイパネル、有機LED(OLED)ディスプレイなどの発光ダイオード(LED)アレイ、陰極線管(CRT)ディスプレイ、電子ペーパー、表面伝導電子放出ディスプレイ(SED)、レーザーテレビジョンディスプレイ、ナノ結晶ディスプレイまたは別のタイプのディスプレイユニットを含んでもよい。ディスプレイ18は、コンピューティングデバイス2内に統合されてもよい。たとえば、ディスプレイ18は、携帯電話のスクリーンであってもよい。代替的に、ディスプレイ18は、ワイヤードまたはワイヤレス通信リンクを介してコンピューティングデバイス2に結合されたスタンドアロンデバイスであってもよい。たとえば、ディスプレイ18は、ケーブルまたはワイヤレスリンクを介してパーソナルコンピュータに接続されたコンピュータモニタまたはフラットパネルディスプレイであってもよい。

【００３０】

本開示の一例によれば、CPU6および/またはGPUドライバ7は、GPU12が直接レンダリングモードおよびビニングレンダリングモードから選択してもよいことを示すGPUコマンドパケットを生成するように構成されてもよい。GPUコマンドパケットは、本明細書でより詳細に説明するように、様々な例ではPM4(「プログラミングモデル4」)フォーマットを有するGPUコマンドパケットを含んでもよい。

【００３１】

コマンドパケットは、レンダリングシーン記述子、間接バッファ情報などのフレームをレンダリングするためのコマンド、ならびに構成データ(たとえば、構成ペア)、およびレンダリングされるべきシーンの解像度(たとえば、ピクセル単位の高さおよび幅)などのレンダリングされるべきシーンについての情報を含んでもよい。GPU12は、受信されたコマンドパケットに含まれる情報に少なくとも部分的に基づいて、グラフィカルシーンをレンダリングするために直接モードレンダリングを使用するかまたはビニングモードレンダリングを使用するかを決定してもよい。しかしながら、コマンドパケットは好ましいレンダリングモードを直接的に示さない場合がある。

【００３２】

レンダーターゲット記述子は、GPU12が実行する場合があるレンダリングモードごとの構成情報、プリアンブル、レンダリング、ビニングおよびコマンドバッファポインタを含む。いくつかの例では、レンダーターゲット記述子はまた、GPU12が現在のシーンのためのレンダリングモードを決定するために使用する場合がある情報を含んでもよい。たとえ

ば、レンダーターゲット記述子はまた、ビニングレンダリングモードを使用してレンダリングするために使用されるべきビンの高さおよび幅を示す場合がある。GPU12は、構成情報を調べ、レンダーターゲット記述子の情報に基づいて、現在のシーンのためのレンダリングモードを決定してもよい。決定されたレンダリングモードに基づいて、GPU12は、必要に応じて、ターゲットレンダリングモードに対応するコマンドバッファにおいて命令を実行する。

【００３３】

コマンドパケットに含まれる間接バッファ情報に関して、コマンドパケットは、1つまたは複数の間接バッファ、色アンリゾルブ(unresolve)バッファ、および深度ステンシルアンリゾルブバッファへのポインタを含んでもよい。コマンドパケットはまた、最上位ビットと、これらの間接バッファの長さとを含んでもよい。PM4コマンドのタイプは、MSBに含まれる。GPU12はアンリゾルブバッファを使用してもよく、GPU12はシステムメモリ10などの外部メモリからGPUメモリに値を読み込んでもよい。GPU12は、たとえば、2つ以上のシーンにわたって変化しないピクセルデータを保持するために、シーンの一部分のみが新しいデータで更新されているとき、アンリゾルブプロセスを実行してもよい。

【００３４】

コマンドパケットはまた、様々な「クリア」(「破棄」とも呼ばれる)矩形を含んでもよい。クリア矩形は、あるフレームから次のフレームに保持されない、フレームバッファ、たとえばUIの領域を示す場合がある。GPU12は、クリア矩形によって指定された領域からピクセルまたはジオメトリデータをロードしない場合がある。

【００３５】

GPU12はまた、GPU状態ヒューリスティックに基づいて、直接レンダリングを使用するかまたはビニングレンダリングを使用するかを決定してもよい。GPUヒューリスティックの例は、GPU処理負荷、GPU電力消費量、および/またはGPUメモリ使用量もしくはGPU12の任意の他の状態パラメータを含んでもよい。直接レンダリングを使用するかまたはビニングレンダリングを使用するかを決定することの一例として、GPU12は、GPU電力消費量がしきい値電力消費量を超えるときに電力を節約するために、直接レンダリングモードから切り替えてもよい。GPU12は、同様に、電力消費量またはGPU負荷が低いとき、ビニングレンダリングモードから直接レンダリングモードに切り替えてもよい。

【００３６】

本開示の技法によれば、GPU12は、GPUがフレームの一部分をレンダリングするために直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示すレンダーターゲット記述子を受信し、受信されたコマンドパケットまたはGPU状態のうちの少なくとも1つに基づいて、GPUによってレンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定し、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングするように構成されてもよい。

【００３７】

図2は、図1のCPU6、GPU12、およびシステムメモリ10の例示的な実装形態をさらに詳細に示すブロック図である。CPU6は、少なくとも1つのソフトウェアアプリケーション24、グラフィックスAPI26、およびGPUドライバ7を含んでもよく、これらの各々は、CPU6上で実行される1つまたは複数のソフトウェアアプリケーションまたはサービスであってもよい。GPU12は、グラフィックス処理コマンドを実行するために一緒に動作する複数のグラフィックス処理ステージを含むグラフィックス処理パイプライン30を含んでもよい。GPU12は、ビニングレンダリングモードおよび直接レンダリングモードを含む様々なレンダリングモードでグラフィックス処理パイプライン30を実行するように構成されてもよい。図2に示すように、グラフィックス処理パイプライン30は、コマンドエンジン32、ジオメトリ処理ステージ34、ラスタ化ステージ36、およびピクセル処理パイプライン38を含んでもよい。グラフィックス処理パイプライン30中の構成要素の各々は、固定機能構成要素、(たとえば、プログラマブルシェーダユニット上で実行されるシェーダプログラムの一部と

しての)プログラマブル構成要素として、または固定機能構成要素とプログラマブル構成要素の組合せとして実装されてもよい。CPU6およびGPU12が利用可能なメモリは、システムメモリ10およびフレームバッファ15を含んでもよい。フレームバッファ15は、システムメモリ10の一部であってもよく、またはシステムメモリ10とは別であってもよい。フレームバッファ15は、レンダリングされた画像データを記憶してもよい。

【００３８】

ソフトウェアアプリケーション24は、GPU12の機能を利用する任意のアプリケーションであってもよい。たとえば、ソフトウェアアプリケーション24は、GUIアプリケーション、オペレーティングシステム、ポータブルマッピングアプリケーション、エンジニアリングもしくは美術アプリケーション用のコンピュータ支援設計プログラム、ビデオゲームアプリケーション、または2Dもしくは3Dグラフィックスを使用する別のタイプのソフトウェアアプリケーションであってもよい。

【００３９】

ソフトウェアアプリケーション24は、グラフィカルユーザインターフェース(GUI)および/またはグラフィックスシーンをレンダリングするようGPU12に命令する1つまたは複数の描画命令を含んでもよい。たとえば、描画命令は、GPU12によってレンダリングされるべき1つまたは複数のグラフィックスプリミティブのセットを定義する命令を含んでもよい。いくつかの例では、描画命令は、GUIで使用される複数のウィンドウ処理サーフェスのすべてまたは一部をまとめて定義してもよい。追加の例では、描画命令は、アプリケーションによって定義されたモデルスペースまたはワールドスペース内の1つまたは複数のグラフィックスオブジェクトを含むグラフィックスシーンのすべてまたは一部をまとめて定義してもよい。

【００４０】

ソフトウェアアプリケーション24は、1つまたは複数のグラフィックスプリミティブを表示可能なグラフィックス画像にレンダリングするための1つまたは複数のコマンドをGPU12に発行するために、グラフィックスAPI26を介してGPUドライバ7を呼び出す場合がある。様々な例では、コマンドは、コマンドならびにシーン(すなわち、1つまたは複数のフレーム)をレンダリングすることに関する他の情報を含んでもよいデータのいわゆる「パケット」を含んでもよい。たとえば、ソフトウェアアプリケーション24は、プリミティブ定義をGPU12に与えるために、グラフィックスAPI26を介してGPUドライバ7を呼び出す場合がある。ある事例では、プリミティブ定義は、描画プリミティブ、たとえば、三角形、矩形、三角形ファン、三角形ストリップなどのリストの形でGPU12に与えられる場合がある。プリミティブ定義は、レンダリングされるべきプリミティブに関連付けられた1つまたは複数の頂点を指定する頂点仕様を含んでもよい。頂点仕様は、頂点ごとの位置座標と、ある事例では、たとえば、色座標、法線ベクトル、およびテクスチャ座標など、頂点に関連付けられた他の属性とを含んでもよい。プリミティブ定義はまた、プリミティブタイプ情報(たとえば、三角形、矩形、三角形ファン、三角形ストリップなど)、スケーリング情報、回転情報などを含んでもよい。ソフトウェアアプリケーション24によってGPUドライバ7に発行された命令に基づいて、GPUドライバ7は、プリミティブをレンダリングするためにGPU12が実行する1つまたは複数の動作を指定する1つまたは複数のコマンドを公式化する場合がある。GPU12がCPU6からコマンドを受信すると、グラフィックス処理パイプライン30はコマンドを復号し、コマンドにおいて指定された動作を実行するようにグラフィックス処理パイプライン30内の1つまたは複数の処理要素を構成する。指定された動作を実行した後、グラフィックス処理パイプライン30は、レンダリングされたデータをディスプレイデバイスに関連付けられたフレームバッファ15に出力する。グラフィックス処理パイプライン30は、ビニングレンダリングモードおよび直接レンダリングモードを含む複数の異なるレンダリングモードのうちの1つで実行するように構成されてもよい。ビニングレンダリングモードおよび直接レンダリングモードの動作について、以下でより詳細に説明する。

【００４１】

GPUドライバ7は、1つまたは複数のシェーダプログラムをコンパイルし、コンパイルされたシェーダプログラムをGPU12内に含まれる1つまたは複数のプログラマブルシェーダユニットにダウンロードするようにさらに構成されてもよい。シェーダプログラムは、たとえば、OpenGL Shading Language(GLSL)、High Level Shading Language(HLSL)、C for Graphics(Cg)シェーディング言語などの高レベルシェーディング言語で書かれる場合がある。コンパイルされたシェーダプログラムは、GPU12内のプログラマブルシェーダユニットの動作を制御する1つまたは複数の命令を含んでもよい。たとえば、シェーダプログラムは、頂点シェーダプログラムおよび/またはピクセルシェーダプログラムを含んでもよい。頂点シェーダプログラムは、プログラマブル頂点シェーダユニットまたはユニファイドシェーダユニットの実行を制御し、1つまたは複数の頂点ごとの動作を指定する命令を含んでもよい。ピクセルシェーダプログラムは、プログラマブルピクセルシェーダユニットまたはユニファイドシェーダユニットの実行を制御するピクセルシェーダプログラムを含み、1つまたは複数のピクセルごとの動作を指定する命令を含んでもよい。

【００４２】

GPUドライバ7はまた、フレームまたはフレームの一部分をレンダリングするときにGPU12が使用すべきレンダリングモードを定義するコマンド、たとえば、1つまたは複数のコマンド、ヘッダ、レンダーシーン記述子、バッファなどからなるコマンドパケットをGPU12に送信してもよい。たとえば、GPUドライバ7は、特定のフレームが直接モードレンダリングを使用してレンダリングされるべきであり、別のフレームがビニングモードレンダリングを使用してレンダリングされるべきであることを決定する場合がある。GPUドライバ7はまた、GPUコマンドパケットをGPU12に送信することによって、フレームの一部分をレンダリングするときにどのレンダリングモードを使用するかを決定するようGPU12に命令する場合がある。

【００４３】

本開示の技法は、CPU6上で実行されるGPUドライバ7などの別の処理ユニットの介入なしに、GPU12が直接モードレンダリングおよびビニングモードレンダリングから選択することを可能にする。直接レンダリングモードおよびビニングレンダリングモードから選択するための以前の技法は、特定のフレームまたはフレームの部分をレンダリングするときにどのレンダリングモードを使用するかを決定し、それをGPU12に命令するために、GPUドライバ7および/またはCPU6がヒューリスティックに関連付けられた多数の計算を実行することを必要とする可能性がある。ドライバが実行したヒューリスティックベースの計算は大量のCPU使用率を引き起こす可能性があり、このことは、アプリケーション性能、および場合によっては、コンピューティングデバイス2の全体的な応答性に影響を及ぼす可能性がある。

【００４４】

上記で示したように、GPU12は、受信されたコマンドパケットからの情報に基づいておよび/またはGPU状態に基づいて決定されたモードのための待ち行列に入れられたレンダリングコマンドの実行シーケンスおよび順序を制御する。GPU状態は、たとえば、電力消費量、メモリ消費量、作業負荷情報、および/または他の状態もしくはプロファイリング情報に関する、GPU12が維持するリアルタイム状態および統計値を含んでもよい。

【００４５】

GPU12が直接レンダリングモードおよびビニングレンダリングモードから選択することを可能にするために、CPU6はGPUコマンドパケット(たとえば、レンダーシーン記述子)を生成し、GPUコマンドパケットをGPU12に送信する場合がある。様々な例では、コマンドパケットはPM4パケットを含んでもよい。上記で説明したように、PM4パケットは、1つまたは複数のIBへのポインタと、スクリーン解像度、(たとえば、16ピクセル倍数での)ビン幅、リゾルブバッファ、クリア領域、および他の情報を含むレンダーターゲット情報とを含んでもよい。

【００４６】

グラフィックス処理パイプライン30は、GPUドライバ7を介してCPU6から1つまたは複数

のグラフィックス処理コマンドを受信し、グラフィックス処理コマンドを実行して表示可能なグラフィックス画像を生成するように構成されてもよい。上記で説明したように、グラフィックス処理パイプライン30は、グラフィックス処理コマンドを実行するために一緒に動作する複数のステージを含む。しかしながら、そのようなステージは必ずしも別個のハードウェアブロックで実施される必要はないことに留意されたい。たとえば、ジオメトリ処理ステージ34およびピクセル処理パイプライン38の部分は、ユニファイドシェーダユニットの一部として実施されてもよい。さらにまた、グラフィックス処理パイプライン30は、CPU6が実行するように構成される記述子パケットをGPUドライバ7から受信したことに応答して、ビニングレンダリングモードおよび直接レンダリングモードを含む複数の異なるレンダリングモードのうちの1つで実行するように構成されてもよい。

【００４７】

コマンドエンジン32は、グラフィックス処理コマンドを受信し、グラフィックス処理コマンドを実施するための様々な動作を実行するようにグラフィックス処理パイプライン30内の残りの処理ステージを構成してもよい。コマンドエンジン32は、GPUドライバ7からコマンドパケットを受信してもよい。グラフィックス処理コマンドは、たとえば、描画コマンドおよびグラフィックス状態コマンドを含んでもよい。描画コマンドは、1つまたは複数の頂点の位置座標、および、ある事例では、たとえば、色座標、法線ベクトル、テクスチャ座標およびかぶり座標など、頂点の各々に関連付けられた他の属性値を指定する頂点仕様コマンドを含んでもよい。グラフィックス状態コマンドは、プリミティブタイプコマンド、変換コマンド、照明コマンドなどを含んでもよい。プリミティブタイプコマンドは、レンダリングされるべきプリミティブのタイプおよび/またはプリミティブを形成するために頂点がどのように組み合わされるかを指定してもよい。変換コマンドは、頂点に対して実行すべき変換のタイプを指定してもよい。照明コマンドは、グラフィックスシーン内の異なる照明のタイプ、方向および/または配置を指定してもよい。コマンドエンジン32は、ジオメトリ処理ステージ34に、1つまたは複数の受信されたコマンドに関連付けられた頂点および/またはプリミティブに対してジオメトリ処理を実行させる場合がある。

【００４８】

GPU12はコマンドパケットを受信し、GPUコマンドパケットを受信したことに応答して、CPU6からのさらなる介入を必要とせずに直接モードレンダリングおよびビニングモードレンダリングから選択してもよい。GPU12は、GPU状態ヒューリスティックおよびコマンドパケットのデータのうちの少なくとも1つに基づいて、直接モードレンダリングを使用するかまたはビニングモードレンダリングを使用するかを決定してもよい。別の例として、GPU12は、直接レンダリングモードがビニングレンダリングモードよりも速くシーンをレンダリングするのはいつかを決定するために、GPU状態を使用してもよい。

【００４９】

一例として、GPU12は、直接レンダリングモードおよびビニングレンダリングモードから選択するために、GPU状態、たとえば、GPU処理負荷、電力消費量、メモリ使用量、またはGPU12の1つもしくは複数の他の性能プロファイリング特性を使用してもよい。ヒューリスティックに基づいて直接レンダリングとビニングレンダリングとの間で切り替えることの一例として、GPU12は、GPU電力消費量がしきい値電力消費量を超えるときに電力を節約するために、直接レンダリングモードから切り替えてもよい。GPU12は、同様に、電力消費量またはGPU負荷が低いとき、ビニングレンダリングモードから直接レンダリングモードに切り替えてもよい。

【００５０】

一例として、高いGPU利用率(たとえば、80%超)は、GPUがビジーである時間が多いこと、およびGPUのバスがデータ転送で飽和していることを示すことができる。GPU12は、測定された転送スループットをGPU12の理論上の最大スループットと比較することによって、発生しているデータ転送の量を測定する場合がある。GPU12はまた、GPU12の「常時オン」カウンタと比較して、個々のGPUハードウェアブロックのビジーカウンタの値を読み取る場合がある(たとえば、測定期間の間、GPU12によって使用された1000サイクルのうちの90

0サイクルでシェーダプロセッサがビジーであった場合)。

【００５１】

ジオメトリ処理ステージ34は、ラスタ化ステージ36のためのプリミティブデータを生成するために、1つまたは複数の頂点に対して頂点ごとの動作および/またはプリミティブセットアップ動作を実行してもよい。各頂点は、たとえば、位置座標、色値、法線ベクトル、およびテクスチャ座標などの属性のセットに関連付けられてもよい。ジオメトリ処理ステージ34は、様々な頂点ごとの動作に従って、これらの属性のうちの1つまたは複数を修正する。たとえば、ジオメトリ処理ステージ34は、修正された頂点位置座標を生成するために、頂点位置座標に対して1つまたは複数の変換を実行してもよい。ジオメトリ処理ステージ34は、修正された頂点位置座標を生成するために、たとえば、モデリング変換、ビューイング変換、投影変換、モデルビュー(ModelView)変換、モデルビュー投影(ModelViewProjection)変換、ビューポート変換および深度範囲スケーリング変換のうちの1つまたは複数を頂点位置座標に適用してもよい。ある事例では、頂点位置座標はモデル空間座標であってもよく、修正された頂点位置座標はスクリーン空間座標であってもよい。スクリーン空間座標は、モデリング変換、ビューイング変換、投影変換およびビューポート変換の適用の後で取得されてもよい。ある事例では、ジオメトリ処理ステージ34はまた、頂点の修正された色座標を生成するために、頂点に対して頂点ごとの照明動作を実行してもよい。ジオメトリ処理ステージ34はまた、たとえば、正規変換、ノーマル正規化動作(normal normalization operations)、ビューボリュームクリッピング、同次除算動作および/またはバックフェースカリング(culling)動作を含む他の動作を実行してもよい。

【００５２】

ジオメトリ処理ステージ34は、ラスタ化されるべきプリミティブを定義する1つまたは複数の修正された頂点のセットを含むプリミティブデータ、ならびにプリミティブを形成するために頂点がどのように組み合わされるかを指定するデータを生成してもよい。修正された頂点の各々は、たとえば、修正された頂点位置座標と、頂点に関連付けられた処理された頂点属性値とを含んでもよい。プリミティブデータはまとめて、グラフィックス処理パイプライン30のさらなるステージによってラスタ化されるべきプリミティブに対応する場合がある。概念的には、各頂点は、プリミティブの2つの辺がぶつかるプリミティブの角に対応する場合がある。ジオメトリ処理ステージ34は、さらなる処理のためにプリミティブデータをラスタ化ステージ36に与えてもよい。

【００５３】

いくつかの例では、ジオメトリ処理ステージ34のすべてまたは一部は、1つまたは複数のシェーダユニット上で実行される1つまたは複数のシェーダプログラムによって実施される場合がある。たとえば、ジオメトリ処理ステージ34は、そのような例では、頂点シェーダ、ジオメトリシェーダまたはそれらの任意の組合せによって実施される場合がある。他の例では、ジオメトリ処理ステージ34は、固定機能ハードウェア処理パイプラインとして、または固定機能ハードウェアと1つもしくは複数のシェーダユニット上で実行される1つもしくは複数のシェーダプログラムの組合せとして実施される場合がある。

【００５４】

ラスタ化ステージ36は、ジオメトリ処理ステージ34から、ラスタ化されるべきプリミティブを表すプリミティブデータを受信し、プリミティブをラスタ化して、ラスタ化されたプリミティブに対応する複数のソースピクセルを生成するように構成される。いくつかの例では、ラスタ化ステージ36は、どのスクリーンピクセルロケーションがラスタ化されるべきプリミティブによってカバーされるかを決定し、プリミティブによってカバーされると決定されたスクリーンピクセルロケーションごとのソースピクセルを生成してもよい。ラスタ化ステージ36は、たとえば、エッジウォーキング(edge-walking)技法、エッジ方程式評価(evaluating edge equations)などの、当業者に知られている技法を使用することによって、どのスクリーンピクセルロケーションがプリミティブによってカバーされるかを決定してもよい。ラスタ化ステージ36は、さらなる処理のために得られたソースピクセルをピクセル処理パイプライン38に与えてもよい。

【００５５】
ラスタ化ステージ36によって生成されたソースピクセルは、スクリーンピクセルロケーション、たとえば、宛先ピクセルに対応し、1つまたは複数の色属性に関連付けられる場合がある。特定のラスタ化されたプリミティブのために生成されたソースピクセルのすべては、ラスタ化されたプリミティブに関連付けられていると言える。プリミティブによってカバーされるべき、ラスタ化ステージ36によって決定されたピクセルは、概念的には、プリミティブの頂点を表すピクセル、プリミティブの辺を表すピクセル、およびプリミティブの内部を表すピクセルを含んでもよい。
【００５６】
ピクセル処理パイプライン38は、ラスタ化されたプリミティブに関連付けられたソースピクセルを受信し、ソースピクセルに対して1つまたは複数のピクセルごとの動作を実行するように構成される。ピクセル処理パイプライン38によって実行される場合があるピクセルごとの動作は、たとえば、アルファテスト、テクスチャマッピング、色計算、ピクセルシェーディング、ピクセルごとの照明、かぶり処理、ブレンディング、ピクセルオーナーシップテスト、ソースアルファテスト、ステンシルテスト、深度テスト、シザーテストおよび/またはスティップリング動作を含む。加えて、ピクセル処理パイプライン38は、1つまたは複数のピクセルごとの動作を実行するための1つまたは複数のピクセルシェーダプログラムを実行してもよい。ピクセル処理パイプライン38によって生成された得られたデータは、本明細書では宛先ピクセルデータと呼ばれ、フレームバッファ15に記憶される場合がある。宛先ピクセルデータは、処理されたソースピクセルと同じ表示ロケーションを有する、フレームバッファ15内の宛先ピクセルに関連付けられる場合がある。宛先ピクセルデータは、たとえば、色値、宛先アルファ値、深度値などのデータを含んでもよい。
【００５７】
フレームバッファ15は、GPU12のための宛先ピクセルを記憶する。各宛先ピクセルは、一意のスクリーンピクセルロケーションに関連付けられる場合がある。いくつかの例では、フレームバッファ15は、宛先ピクセルごとの色成分および宛先アルファ値を記憶してもよい。たとえば、フレームバッファ15はピクセルごとの赤(Red)、緑(Green)、青(Blue)、アルファ(Alpha)(RGBA)成分を記憶してもよく、"RGB"成分は色値に対応し、"A"成分は宛先アルファ値に対応する。フレームバッファ15およびシステムメモリ10は別個のメモリユニットであるものとして示されているが、他の例では、フレームバッファ15はシステムメモリ10の一部であってもよい。
【００５８】
上記で説明したように、グラフィックス処理パイプライン30は、ビニングレンダリングモードおよび直接レンダリングモードを含む特定のレンダリングモードに従って、グラフィックス画像をレンダリングしてもよい。ビニングレンダリングモードに従ってレンダリングするとき、グラフィックス処理パイプライン30は、プリミティブのバッチ(すなわち、1つまたは複数のプリミティブ)を受信して、得られるグラフィックス画像にレンダリングしてもよい。プリミティブのバッチをレンダリングするために、得られるグラフィックス画像は、複数のより小さい部分(たとえば、ピクセルのタイルまたはビン)に再分割されてもよく、グラフィックス処理パイプライン30は、グラフィックス画像の各部分を別個のレンダリングパスとしてレンダリングしてもよい。
【００５９】
図3は、ビニングレンダリングモード用のビンに分割されたフレームを示す概念図である。フレーム40は、ビン42などの複数のビンに分割されてもよい。典型的には、グラフィックスハードウェアは、データの少なくとも1つのビンを保持するのに十分なサイズである高速メモリ(たとえば、図2のグラフィックスメモリ14)を含む。フレームの特定の部分のための単一のレンダリングパスの一部として、グラフィックス処理パイプライン30は、フレームの宛先ピクセルの特定のサブセット(たとえば、宛先ピクセルの特定のビン)に対して、プリミティブのバッチのすべてまたはサブセットをレンダリングしてもよい。第1のビンに対して第1のレンダリングパスを実行した後、グラフィックス処理パイプライン3

0は第2のビンに対して第2のレンダリングパスを実行してもよく、以下同様である。グラフィックス処理パイプライン30は、あらゆるビンに関連付けられたプリミティブがレンダリングされるまで、徐々にビンを横断してもよい。

【００６０】

図4は、ビニングレンダリングモードで使用されるビンをより詳細に示す概念図である。ビン44、46、48および50は、複数のピクセル52を含むようにレンダリング/ラスタ化される。1つまたは複数のグラフィックスプリミティブは、各ビンにおいて可視であってもよい。たとえば、三角形A(Tri A)の部分は、ビン44とビン48の両方において可視である。三角形B(Tri B)の部分は、ビン44、ビン46、ビン48、およびビン50の各々において可視である。三角形C(Tri C)はビン46のみにおいて可視である。レンダリングパスの間、ビニングレンダリングモードの一例では、シーンはビンに分割され、ビン内にあるすべての三角形はレンダリングされる(これはソフトウェアビニングと呼ばれることがある)。ビニングレンダリングモードの別の例では、最終的なレンダリングされたシーンにおいてビン内のどの三角形が実際に可視であるかを決定するために、レンダリングする前に追加のステップがとられる(これはハードウェアビニングと呼ばれることがある)。たとえば、いくつかの三角形は1つまたは複数の他の三角形の後ろにあってもよく、最終的なレンダリングされたシーンにおいて可視ではなくなる。このようにして、可視ではない三角形は、そのビンについてレンダリングされる必要がない。

【００６１】

特定のレンダリングパスを実行している間、その特定のレンダリングパスに関連付けられたビンのピクセルデータは、グラフィックスメモリ14(ビンバッファと呼ばれることがある)に記憶されてもよい。レンダリングパスを実行した後、グラフィックス処理パイプライン30は、グラフィックスメモリ14のコンテンツをフレームバッファ15に転送してもよい。場合によっては、グラフィックス処理パイプライン30は、フレームバッファ15内のデータの一部分をグラフィックスメモリ14に記憶されたデータで上書きしてもよい。他の場合には、グラフィックス処理パイプライン30は、フレームバッファ15内のデータをグラフィックスメモリ14に記憶されたデータと合成するかまたは組み合わせてもよい。グラフィックスメモリ14のコンテンツをフレームバッファ15に転送した後、グラフィックス処理パイプライン30は、グラフィックスメモリ14をデフォルト値に初期化し、異なるビンに対して後続のレンダリングパスを開始してもよい。

【００６２】

図5は、「ソフトウェア」ビニングを使用するビニングレンダリングモードを使用してシーンをレンダリングするための例示的なコマンド構造を示す概念図である。レベル1間接バッファ(IB1)60は、グラフィックス処理パイプライン30の様々なステップを実行するようGPU12に命令するための一連の実行コマンドを含む。GPU12が受信するPM4コマンドパケットは、IB1 60へのポインタである。IB1 60内の各実行コマンドは、本質的には、レンダリングパイプラインの様々な態様のためのコマンドを含む1つまたは複数のレベル2間接バッファ(IB2)へのポインタである。このようにして、グラフィックスレンダリングパイプラインを実行するための2つ以上のレベル構造が確立される。GPU12は、IB1 60内の各実行コマンドを順次1ステップずつ実行してもよく、IB1 60内の各実行コマンドは、IB2に記憶されたコマンドの特定のスタックを指す。IB1およびIB2は、GPU12に搭載のメモリであってもよく、またはシステムメモリ10などのGPU12の外部にあるメモリであってもよい。

【００６３】

IB1 60内のプリアンブル実行コマンドは、GPU12によって実行可能であるプリアンブルコマンドを含むプリアンブルIB2 62を指す。たとえば、プリアンブルIB2 62は、GPU12のその静的状態を初期化し、GPU12の初期レンダリング状態を設定するコマンドを含んでもよい。GPUの静的状態は、特定のアプリケーションに基づいて変化しない設定を含む。一方、レンダリング状態は、特定のアプリケーション(たとえば、OpenGLアプリケーション対Direct Xアプリケーション)に基づいて変化する場合があるGPU設定を含む。プリアンブルIB2内のコマンドが完了した後、制御は次の実行コマンドを実行するためにIB1 60に戻

る。

【００６４】

IB1 60内の次の実行コマンドは、利用されているレンダリングモードのためのレンダーパスを構成する。やはり、図5の例では、レンダリングモードはソフトウェアビニングを使用するビニングレンダリングモードである。次に、IB1内のロードビン実行コマンドは、BLT(ブロック転送(Block Transfer)、すなわち、データコピー)IB2 64内の非同期ブリッティングコマンドを指す。ブリッティングコマンドは、いくつかの例では、ラスタ演算子を使用して、いくつかのビットマップを1つに組み合わせてもよい。図5の例では、ブリッティングは、プリアンブルにおいて生成されたグラフィックスコマンドまたは状態を組み合わせるかまたは修正してもよい。次に、IB1 60内のロードビン実行コマンドは、ロードIB2 66内のコマンドを指す。ソフトウェアビニングの場合、特定のビンのためのデータはGMEM14にロードされる(Load 2 GMEM)。次いで、制御はIB1 60に戻され、レンダリングビン実行コマンドはレンダリングIB2内のコマンドを指す。レンダリングIB2 68は、一連の状態コマンドおよびロードされたビンにおいて三角形を描画するための描画コマンドからなる。各描画コマンドは、コマンドおよび/またはGPUハードウェアによって確立された(たとえば、ジオメトリ処理ステージ34、ラスタ化ステージ36、および/またはピクセル処理パイプライン38を含む)グラフィックス処理パイプライン30に従って三角形を描画するようGPU12に命令する。レンダリングIB2 68に示すように、描画コマンドの各々は、特定の三角形がビンにおいて実際に可視であるかどうかを決定するために可視性ストリームが使用されないことを示す。可視性ストリームは、「ハードウェア」ビニングを使用するビニングレンダリングモードで生成され、これについては図6を参照しながらより詳細に説明する。レンダリングIB2 68内の状態コマンドは、GPU12によって実行されるグラフィックス処理パイプラインの挙動に影響を及ぼす。たとえば、状態コマンドは、色、ポリゴンモード(たとえば、立体または線ではなく点)、ブレンディング(オン/オフ)、深度テスト(オン/オフ)、テクスチャリング(オン/オフ)、カリング、クリッピング、および他の論理演算を変更する場合がある。レンダリングIB2 68に示すように、状態コマンドは三角形ごと(またはプリミティブごと)に発行される場合がある。すなわち、コマンド"State Tri A"は三角形Aを描画するときにGPU12の挙動に影響を及ぼす場合があり、一方、"State Tri B1"コマンドおよび"State Tri B2"コマンドは三角形Bを描画するときにGPU12の挙動に影響を及ぼす場合がある。"State Tri B1"コマンドおよび"State Tri B2"コマンドは、複数の状態コマンドが三角形ごとに実行される場合があることを示すにすぎない。

【００６５】

すべてのコマンドがレンダリングIB2 68において実行された後(たとえば、すべての三角形が描画された後)、制御はIB1 60に戻る。記憶ビン実行コマンドは、GMEM14からのレンダリングされたビンをメモリ(たとえば、フレームバッファ15)に記憶するためのコマンドを含む記憶IB2 70へのポインタを含んでもよい。次いで、レンダーパス(たとえば、IB1 60に示すようなレンダーパスの構成からビンの記憶までの実行コマンド)が1つまたは複数のフレームについてビン72ごとに繰り返される。

【００６６】

図6は、「ハードウェア」ビニングを使用するビニングレンダリングモードを使用してシーンをレンダリングするための例示的なコマンド構造を示す概念図である。IB1 61内の実行コマンドは、「ビニング」パスに関するコマンドを除いて、図6のIB1 60の実行コマンドと類似している。「ビニング」パスは、ビン内の特定の三角形が最終的なレンダリングされたシーンにおいて実際に可視であるかどうかを示す可視性ストリームを生成するために使用される。たとえば、いくつかの三角形はシーン内の別の三角形の後ろにあってもよく、いくつかのシナリオでは可視ではなくなる(たとえば、前方の三角形が不透明であるときまたはブレンディングが使用されないとき)。ビン72をレンダリングする前に、IB1 61は、ビニングIB2 74内のコマンドを指すビニングパス実行コマンドを含んでもよい。ビニングIB2 74は、GPU12にグラフィックスパイプラインの簡略化されたバージョン(たとえば、レンダリングIB2 69の簡略化されたバージョン)を実行させるコマンドを含むが、

三角形が最終的なレンダリングされたシーンにおいて可視であるかどうかを決定する深度テスト(Zテスト)に基づいてビン内の三角形ごとに可視性ストリームを更新するステップを追加する。

【００６７】

ビニングパスの目標は、現在のビンと交わる三角形を識別することである。したがって、三角形が特定のビンと交わるかどうかを識別するために、三角形の頂点の位置のみを決定する必要がある。ビニングパスは、頂点の位置に影響を及ぼす命令のみを含む簡略化された頂点シェーダを利用する。たとえば、三角形の頂点の位置に影響を及ぼさない色の命令、テクスチャ座標および他の命令は、ビニングパスに使用するための簡略化された頂点シェーダから除去されてもよい。ビニングパスはまた、各三角形のおおよその深度を決定するために、細かいラスタ化ではなく粗いラスタ化を使用する。粗いラスタ化は、細かいラスタ化よりも低い精度で(たとえば、より少ないビット数を使用して)深度値を計算する。三角形がビンにおいて可視であるかどうかを決定するためには、おおよその深度値のみが必要である。ピクセルシェーダはビニングパスにおいて使用されない。

【００６８】

次いで、ビニングパスは、三角形がビン内の他の三角形に対してビンにおいて可視であるかどうかを決定するために、粗い深度値に対して深度テストを利用する。この深度テストに基づいて、可視性ストリームが更新される。可視性ストリームは、レンダリングされたビン内の特定の三角形が可視であるかどうかを示すビットのストリングであってもよい(たとえば、1は三角形が可視であることを示し、0は三角形が可視ではないことを示す)。

【００６９】

レンダリングIB2 69内のコマンドは、図5のレンダリングIB 68のコマンドと類似しているが、可視性ストリーム用である。レンダリングIB2 69内の描画コマンド(たとえば、Draw Tri A、Draw Tri B、Draw Tri Cなど)は、特定の三角形を描画する必要があるかどうかを決定するために、ビニングパスによって生成された可視性ストリームを使用してもよい。たとえば、可視性ストリームによって可視ではないものとして示された三角形については描画がスキップされる場合がある。

【００７０】

ビニングレンダリングモードの場合のように、ビンごとにフレームをレンダリングすることとは対照的に、直接レンダリングは、グラフィックスパイプラインを通じた1つのパスにおいてフレーム全体をレンダリングする。直接レンダリングは、典型的には、グラフィックスメモリの量が制限されたビニングベースのアーキテクチャで実行されるとき、より遅いシステムメモリを利用する。

【００７１】

図7は、直接レンダリングモードを使用してシーンをレンダリングするための例示的なコマンド構造を示す概念図である。直接レンダリングモードのためのコマンドは、図5のソフトウェアビニングのためのコマンドと類似しているが、ビンごとにレンダリングする代わりに、フレーム全体が1つのパスにおいてレンダリングされる。たとえば、IB1 80内の実行コマンドはIB1 60内の実行コマンドと類似しているが、ビンをロードし、レンダリングし、記憶するのではなく、IB1 80内の実行コマンドは記憶フレームをレンダリングするそれぞれのIB2内のコマンドを指す。IB1 80の非同期ブリッティングコマンドは、BLT IB2 84内の非同期ブリッティングコマンドを指す。より詳細には、IB1 80内のコマンドは、プリアンブル、プリアンブル/復元、レンダーパスの構成、およびフレームのレンダリングを含んでもよい。IB1 80内のレンダリングフレーム実行コマンドは、GPU12にフレームにおいてプリミティブを描画させる命令を含むレンダリングIB2 88を指す。ソフトウェアビニングのように、直接レンダリングモードにおけるレンダリングは、ハードウェアビニングパスまたは可視性ストリームを利用しない。レンダリングIB2 88内の任意の描画コマンド(たとえば、Draw Tri A、Draw Tri B、Draw Tri C)は、レンダリングされた三角形を記憶するためにGMEMメモリを利用するであろう。

【００７２】

IB1 80内のプリアンブル実行コマンドは、GPU12の静的状態および初期レンダリング状態を確立するための命令を含むプリアンブルIB2 82を指す。これらのコマンドは、図6のプリアンブルIB2 62内のコマンドと同様に働くが、ビニングレンダリングモードではなく直接レンダリングモードのためのレンダリング状態をセットアップする。

【００７３】

図5～図7に示すように、直接モードレンダリングとビニングモードレンダリングの両方はIB2を利用する。レンダリングパスまたはレンダリングループの間、GPU12は、レンダリングされるべきシーンについての情報に基づいて、ビニングおよび動的パント(punting)論理のためにレンダリングIB2にわたってループする。以前の技法では、GPU12以外のプロセッサ上で実行されるドライバ(たとえば、CPU6上で実行されるGPUドライバ7)は、フレームの一部分をレンダリングするときに直接レンダリングモードを利用するかまたはビニングレンダリングモードを利用するかを決定することを受け持っていた。シーンをレンダリングするために直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定するためにCPU6を利用するとき、このプロセスは非常にCPUバウンドである場合がある(すなわち、CPU6の比較的高い利用率を有する場合がある)。GPU12が直接モードレンダリングを利用するかまたはビニングモードレンダリングを利用するかを決定するとき、その結果は直接モードレンダリングとビニングモードレンダリングとの間の能力格差にも依存する場合がある。

【００７４】

本開示の技法によれば、GPU12は、たとえば、CPU6上で実行されるGPUドライバ7から、GPUコマンドパケットを受信してもよい。コマンドパケットは、GPU12がGPU12によってレンダリングされるべきフレームの一部分をレンダリングするために直接モードレンダリングまたはビニングモードレンダリングのいずれかを利用してもよいことを示す場合がある。加えて、コマンドパケットは、構成情報をカプセル化する場合がある。コマンドパケットはまた、直接モードレンダリングならびに/またはハードウェアおよび/もしくはビニングレンダリングのための、いくつかの例としてIB2 82、84、86、88、および90などの1つまたは複数のIB2を参照する(または指す)場合がある。

【００７５】

アプリケーション(たとえば、図2のソフトウェアアプリケーション24)がシーンのレンダリングを開始する前に、GPU12は、コマンドパケットに含まれるシーン情報に基づいて、レンダリングモードを決定してもよい(たとえば、ビニングレンダリングおよび直接レンダリングの間での決定)。GPU12はまた、GPU状態ヒューリスティックに基づいて、直接レンダリングを使用するかまたはビニングレンダリングを使用するかを決定してもよい。いくつかの技法では、GPU12はまた、以前のレンダリングパスからのデータに基づいて、レンダリングモードを決定してもよい。しかしながら、アプリケーションが新しいシーンをレンダリングする前にレンダリング技法を切り替える場合があるので、この技法は必ずしもレンダリングモードを決定するための最適な方法であるとは限らないことがある。したがって、この技法は、適切なレンダリングモードに切り替えるのに十分な新しいデータが蓄積されるまで、非効率的なレンダリングをもたらす場合がある。

【００７６】

加えて、過去のレンダリングに関するヒューリスティックデータは、現在のシーンが以前のシーンとは異なる場合、必ずしも最も最適なレンダリングモードを現在のシーンに提供するとは限らないことがある。たとえば、急速に変化するレンダリングモードを最適に使用するアプリケーションは、多くの誤った予測をもたらす場合がある。本開示の技法によれば、GPU12はGPU12の電力使用量を決定することができ、電力使用量に基づいてレンダリングモードを決定してもよい。本開示の技法による別の例では、GPU12は、GPU12の現在または将来の利用率を決定してもよく、GPU12の現在または将来の利用率に基づいて、フレームの少なくとも一部分をレンダリングするために使用すべきレンダリングモードを決定してもよい。本開示の技法によれば、GPU12は、本明細書では詳細に説明しない他のヒューリスティックを使用してもよく、GPU12は、CPU6からの介入または干渉なしに、本明

細書で説明する任意のヒューリスティックを実行してもよい。

【００７７】

以下の技法は、任意のグラフィックスアプリケーションプログラムインターフェース(API)を使用するグラフィックス処理システムに適用可能であり、特に、ビニングレンダリングを利用するグラフィックスAPIに適している。そのようなAPIの例として、MicrosoftのDirectX9(DX9)、DX10、およびDX11、DX12、ならびにOpenGLおよびOpenGL ESなどのオープンソースグラフィックスAPI、OpenCLおよびDirectComputeなどのコンピューティングAPIが含まれる。

【００７８】

図8は、異なるレンダリングモードのコマンドバッファを示す概念図である。図8は、IB1 100、IB1 102、およびIB1 104を含む。IB1 100、102、および104の各々は、特定のレンダリングモードに対応するIB1バッファを含んでもよい。各IB1は、ハードウェアビニングのためのコマンドバッファに対応する場合がある。IB1 102はソフトウェアビニングのためのバッファに対応する場合があり、IB1は直接モードレンダリングのためのバッファに対応する場合がある。図8に示す技法によれば、異なるレンダリングモードを利用するために、GPU12は異なるIB1から読み取らなければならず、このことは、GPUドライバ7が複数のIB1をGPU12に転送することを必要とする場合がある。加えて、GPU12は複数のIB1を記憶することが必要とされる場合があり、このことは、GPU12の追加のメモリを消費する。本開示の技法は、図8に関して示す技法とは異なり、複数のIB1がGPU12に転送されることを必要としない。

【００７９】

図9は、本開示の技法による、直接レンダリングモードまたはビニングレンダリングモードを使用してシーンをレンダリングするための例示的なコマンド構造を示す概念図である。図9の例では、GPUドライバ7またはGPU12はIB1 120を生成または受信してもよい。IB1 120は、図5～図7に関して説明する他のIB1と類似している。しかしながら、本開示の技法によれば、IB1 120は、PM4シーン記述子を含んでもよいコマンドパケットを含む。

【００８０】

図9の例では、IB1 120はコールビニングIB2 132、ならびにプリアンブルIB2 122、BLT IB2 124、およびレンダリングIB2 128を参照し(または指し)、IB2 130を記憶する。GPUコマンドパケット、たとえば、IB1 120のPM4シーン記述子は、GPU12が様々なヒューリスティックに基づいてハードウェアビニング、ソフトウェアビニング、または直接モードレンダリングを実行することができるように、上述のIB2をカプセル化する場合がある。IB1 120のPM4シーン記述子は、IB2 122、124、126、128、130、および132のメモリアドレスを参照することによって、これらのIB2を参照する。しかしながら、PM4シーン記述子は、必ずしもIB2 122、124、126、128、130、および132を含むとは限らない。GPU12は、実行順序、たとえば、図5～図7に示すレンダリングモードの実行順序に類似した実行順序を生成するためにPM4シーン記述子が参照するIB2を使用してもよい。

【００８１】

コマンドパケットにシーン記述子を含め、GPU12がフレームの部分のためのレンダリングモードを決定することを可能にすることによって、本開示の技法は複数の利点を提供する。直接モードレンダリングとビニングモードレンダリングとの間で切り替えるための以前の技術は、複数のIB2の生成を必要とし、ドライバベースのヒューリスティックに基づいたビニングおよび動的パント論理のためのレンダリングIB2にわたるルーピングを必要とした(たとえば、「フック」ベースの関数呼出しではなく、グラフィックスAPIが提供する関数呼出しに依拠する)。生成、反復、およびドライバベースのヒューリスティック、ならびにフック論理はすべて、非常にCPUバウンドである(すなわち、かなりのCPUリソースを消費する)。

【００８２】

図9に示すPM4シーン記述子は、GPU12が異なるレンダリングモードを条件付きでトリガすることを可能にし、GPU12が余分のGPU状態固有のヒューリスティックをレンダリングモ

ード決定機構に統合することを可能にする。一例では、CPU6は、PM4記述子にシーン記述子ヒューリスティックを含んでもよい。GPU12は、レンダリングモードを決定するために、PM4記述子に含まれるヒューリスティックを評価してもよい。別の例では、GPU12はGPUハードウェア状態分析に基づいてレンダリングモードを決定してもよく、GPUハードウェア状態分析は、いくつかの例では、電力状態、メモリ使用量、および/またはレンダリング負荷を含んでもよい。別の例では、ヒューリスティックは、GPU12のハードウェア能力を決定することを含んでもよく、その結果、異なるハードウェア能力を有するGPUは、シーンを異なるようにレンダリングしてもよい。加えて、決定されたヒューリスティックは、他のGPU性能クエリ、ならびに本明細書では明示的に開示しない実行時の状態決定を含んでもよい。本開示の技法はまた、パント(すなわち、いつビニングレンダリングモードから直接レンダリングモードに切り替えるかを決定すること)およびレンダリング論理をCPU6および/またはGPUドライバ7からGPU12に移動する。

【００８３】

加えて、GPUレンダリングモード決定をGPU12に移動することは、CPU6からのオーバーヘッドをなくし、メモリフットプリントを低減し、待ち行列に入れられたグラフィックスコマンドをサブミットするために使用されるIB1コマンド空間を節約する。加えて、GPUベースまたはGPU支援のヒューリスティックは、動的パントプロセスを用いて、CPU6および/またはGPUドライバ7からの対話なしに、GPU12上で実行されるかまたは埋め込まれることが可能である。本開示の技法はまた、CPU集約的である場合がある、多くの小さいレンダリングコールを一括処理することによって引き起こされる、いわゆる「スモールバッチ」問題に関連付けられた性能ペナルティを低減する。

【００８４】

図10は、本開示の一例による方法を示すフローチャートである。図10の方法は、コンピューティングデバイス2上に存在する、GPU12、および/もしくは任意の他のハードウェア、ファームウェア、ソフトウェア要素、またはそれらの組合せによって実行されてもよい。CPU6は、IB1などの、フレームをレンダリングするためのレンダリングコマンドを生成するように構成されてもよい。CPU6は、GPUコマンドパケットをGPU12に送信してもよい。様々な例では、コマンドパケットはPM4フォーマットのコマンドパケットを含んでもよい。

【００８５】

GPU12は、CPU6からコマンドパケットを受信してもよい。コマンドパケットは、GPU12がGPU12によってレンダリングされるべきフレームの一部分について直接レンダリングモードまたはビニングレンダリングモードから選択してもよいことを示す場合がある(160)。GPU12は、受信されたコマンドパケット内の情報またはGPU12の状態のうちの少なくとも1つに基づいて、レンダリングされるべきフレームの一部分について直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定し(162)、決定された直接レンダリングモードまたはビニングレンダリングモードを使用してフレームの一部分をレンダリングしてもよい(164)。様々な例では、フレームの一部分をレンダリングするために、GPU12はレンダリングループを反復する、および/または動的パントを実行する場合がある。

【００８６】

様々な例では、直接レンダリングモードを利用するかまたはビニングレンダリングモードを利用するかを決定するために、GPU12は、レンダリングされるべきフレームの一部分について直接レンダリングモードを利用するかまたはビニングレンダリングモードを利用するかを決定するヒューリスティックを実行してもよい。いくつかの例では、GPU12は、GPU12の電力消費量に基づいて、直接レンダリングモードを利用するかまたはビニングレンダリングモードを利用するかを決定してもよい。GPU12はまた、GPU12のいくらかのリソース利用率に基づいて、直接レンダリングモードを利用するかまたはビニングレンダリングモードを利用するかを決定してもよい。

【００８７】

様々な例では、コマンドパケットは、間接バッファ(IB2)情報および/または構成ペアなどの、レンダリングされるべきフレームをレンダリングするために必要とされる情報をカプセル化するシーン記述子パケットを含んでもよい。シーン記述子のカプセル化された情報は、フレームが属するシーンについての解像度情報をさらに含んでもよい。

【００８８】

いくつかの例では、GPU状態は、GPUの電力消費量を含んでもよく、GPU12は、GPUの電力消費量がしきい値電力消費値を超えるときに、フレームの一部分をレンダリングするためにビニングレンダリングモードを使用するとさらに決定してもよい。GPU状態はまた、GPUの利用負荷を含んでもよく、GPU12は、GPUの利用負荷が利用値を超えるときに、ビニングレンダリングモードを使用すると決定してもよい。GPU状態はまた、GPUのメモリ使用量を含んでもよい。

【００８９】

様々な例では、受信されたコマンドパケットは、レンダリングされるべきフレームをレンダリングするために必要とされる情報をカプセル化するシーン記述子パケットを含んでもよい。シーン記述子のカプセル化された情報は、いくつかの例では、間接バッファ情報および構成ペアを含んでもよい。シーン記述子のカプセル化された情報はまた、レンダリングされるべきフレームについての解像度情報を含んでもよい。

【００９０】

様々な例では、受信されたコマンドパケット内の情報は、レンダリングされるべきフレームの一部分におけるジオメトリの量を含んでもよい。受信されたコマンドパケットはまた、GPUによってレンダリングされるべきフレームをレンダリングするために必要とされるすべてのビンを含んでもよい。いくつかの例では、GPU12は、GPUによって以前にレンダリングされたフレームからのレンダリング情報に基づいて、直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定してもよい。

【００９１】

いくつかの例では、GPU12は、フレームをレンダリングするときに生じるオーバードローの量を決定するようにさらに構成されてもよい。GPU12は、生じるオーバードローの量に基づいて、直接レンダリングモードを使用するかまたはビニングレンダリングモードを使用するかを決定してもよい。

【００９２】

1つまたは複数の例では、上記で説明した機能は、ハードウェア、ソフトウェア、ファームウェア、またはそれらの任意の組合せで実装されてもよい。ソフトウェアで実装される場合、機能は、非一時的コンピュータ可読媒体を備える製造品上の1つまたは複数の命令またはコードとして記憶されてもよい。コンピュータ可読媒体は、コンピュータデータ記憶媒体を含んでもよい。データ記憶媒体は、本開示で説明する技法を実装するための命令、コードおよび/またはデータ構造を取り出すために1つもしくは複数のコンピュータまたは1つもしくは複数のプロセッサによってアクセスすることができる任意の利用可能な媒体であってもよい。限定ではなく例として、そのようなコンピュータ可読媒体は、RAM、ROM、EEPROM、CD-ROMもしくは他の光ディスクストレージ、磁気ディスクストレージもしくは他の磁気ストレージデバイス、フラッシュメモリ、または、命令もしくはデータ構造の形態の所望のプログラムコードを搬送もしくは記憶するために使用することができ、コンピュータによってアクセスすることができる、任意の他の媒体を備えることができる。本明細書で使用するディスク(disk)およびディスク(disc)は、コンパクトディスク(disc)(CD)、レーザーディスク(登録商標)(disc)、光ディスク(disc)、デジタル多用途ディスク(disc)(DVD)、フロッピーディスク(disk)およびブルーレイディスク(disc)を含み、ディスク(disk)は通常、データを磁気的に再生するが、ディスク(disc)はレーザーを用いてデータを光学的に再生する。上記の組合せもコンピュータ可読媒体の範囲内に含まれるべきである。

【００９３】

コードは、1つまたは複数のDSP、汎用マイクロプロセッサ、ASIC、FPGA、または他の等

価の集積論理回路もしくはディスクリート論理回路などの、1つまたは複数のプロセッサによって実行されてもよい。加えて、いくつかの態様では、本明細書で説明する機能は、専用のハードウェアモジュールおよび/またはソフトウェアモジュール内で提供されてもよい。また、本技法は、1つまたは複数の回路または論理要素において完全に実装することができる。

【００９４】

本開示の技法は、ワイヤレスハンドセット、集積回路(IC)またはICのセット(たとえば、チップセット)を含む、多種多様なデバイスまたは装置において実装されてもよい。開示する技法を実行するように構成されるデバイスの機能的態様を強調するために、様々な構成要素、モジュール、またはユニットについて本開示で説明したが、これらの構成要素、モジュール、またはユニットは、必ずしも異なるハードウェアユニットによる実現を必要とするとは限らない。むしろ、上記で説明したように、様々なユニットは、コーデックハードウェアユニットにおいて組み合わされてもよく、または適切なソフトウェアおよび/もしくはファームウェアとともに、上記で説明したような1つもしくは複数のプロセッサを含む、相互動作可能なハードウェアユニットの集合によって提供されてもよい。

【００９５】

様々な例について説明した。これらおよび他の例は以下の特許請求の範囲内に入る。

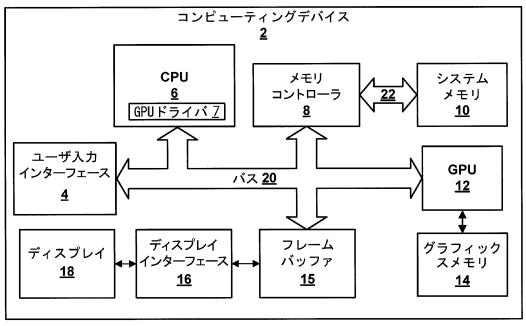
【符号の説明】

【００９６】

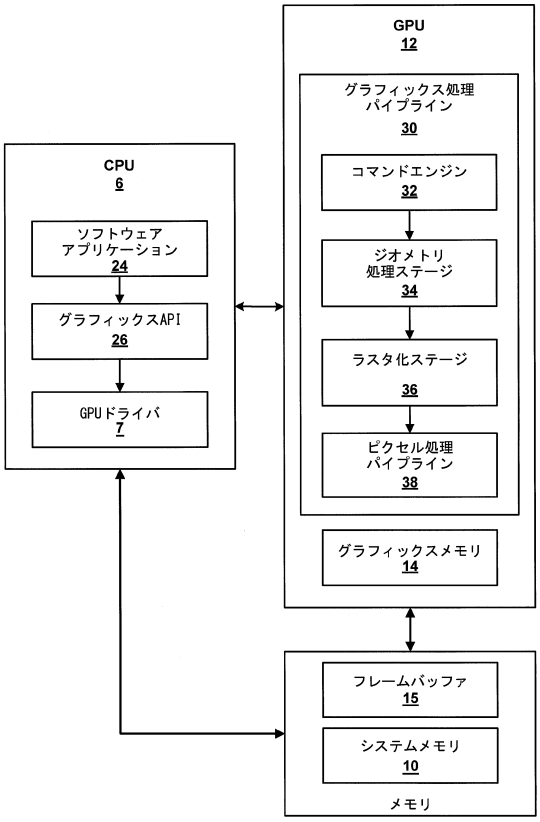
2 コンピューティングデバイス
4 ユーザ入力インターフェース
6 中央処理ユニット、CPU
7 GPUドライバ
8 メモリコントローラ
10 システムメモリ
12 グラフィックス処理ユニット、GPU
14 グラフィックスメモリ
15 フレームバッファ
16 ディスプレイインターフェース
18 ディスプレイ
20 バス
22 バス
24 ソフトウェアアプリケーション
26 グラフィックスAPI
30 グラフィックス処理パイプライン
32 コマンドエンジン
34 ジオメトリ処理ステージ
36 ラスタ化ステージ
38 ピクセル処理パイプライン
40 フレーム
42 ビン
44 ビン
46 ビン
48 ビン
50 ビン
52 ピクセル
60 レベル1間接バッファ、IB1
62 プリアンブルIB2
64 BLT IB2
66 ロードIB2

68　レンダリングIB2
69　レンダリングIB2
70　記憶IB2
72　ビン
74　ビニングIB2
80　IB1
82　プリアンブルIB2
84　BLT IB2、IB2
86　IB2
88　レンダリングIB2、IB2
90　IB2
100　IB1
102　IB1
104　IB1
120　IB1
122　プリアンブルIB2、IB2
124　IB2
126　IB2
128　IB2
130　IB2
132　IB2

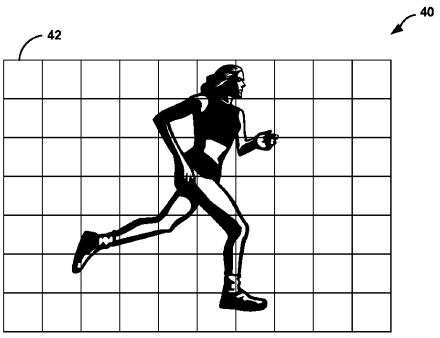
【図１】



【図２】



【図３】



【図４】

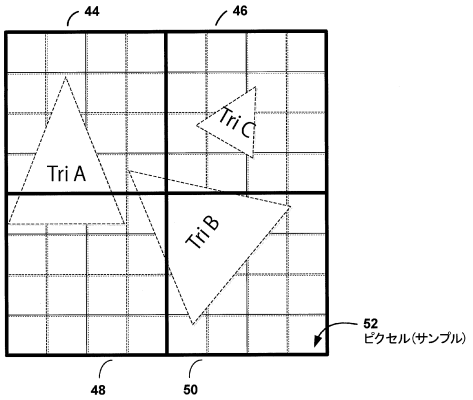
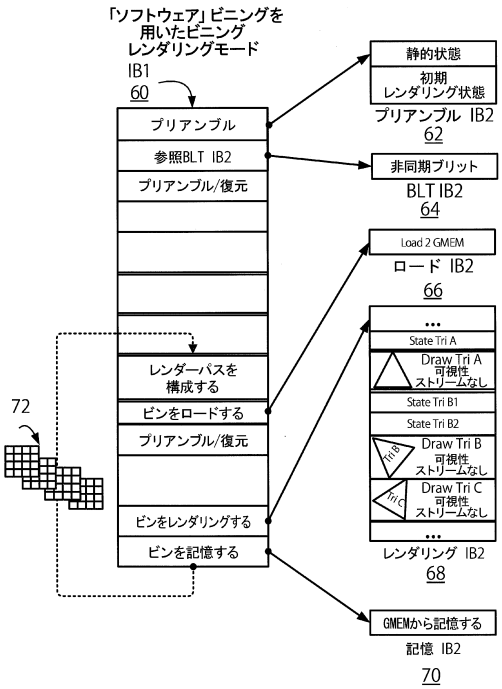
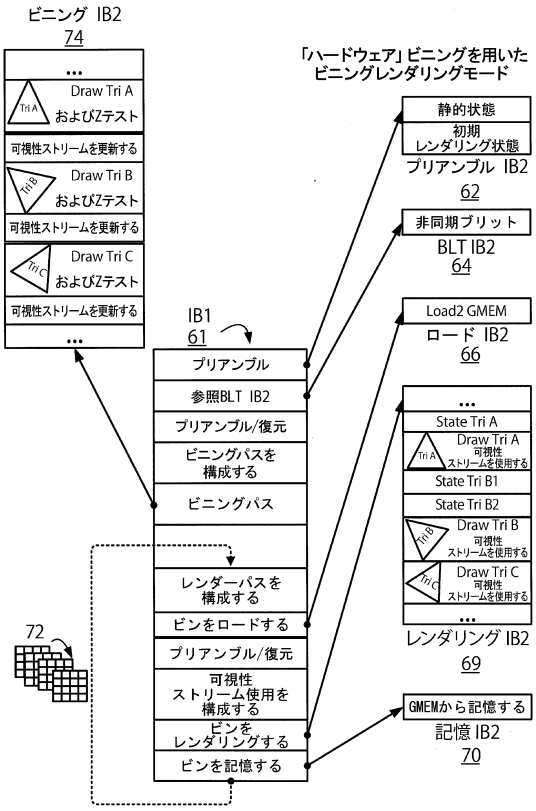


FIG. 3

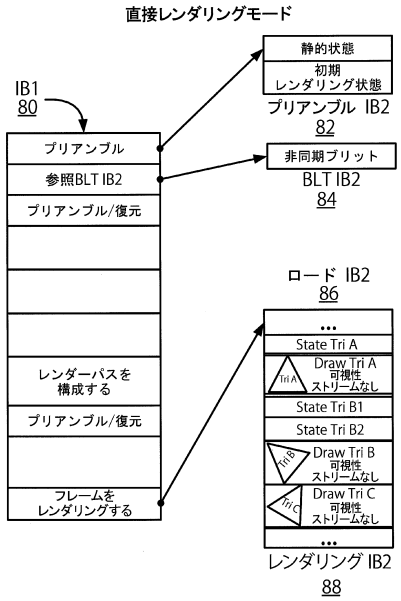
【図５】



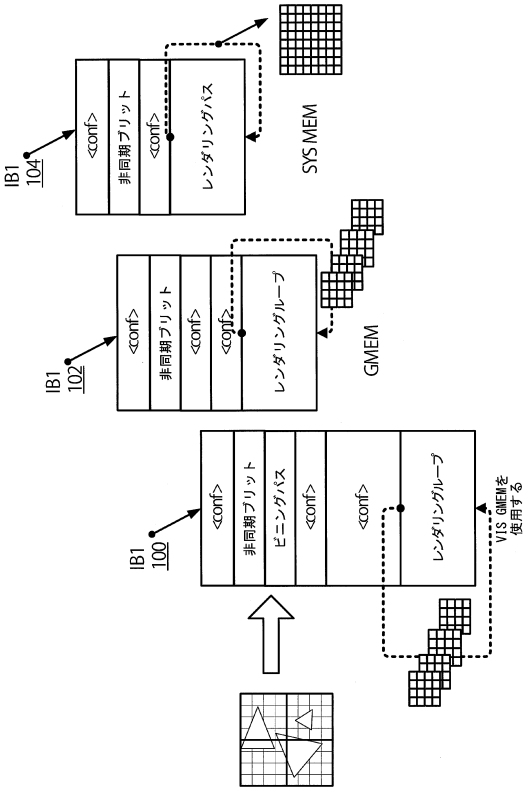
【図６】



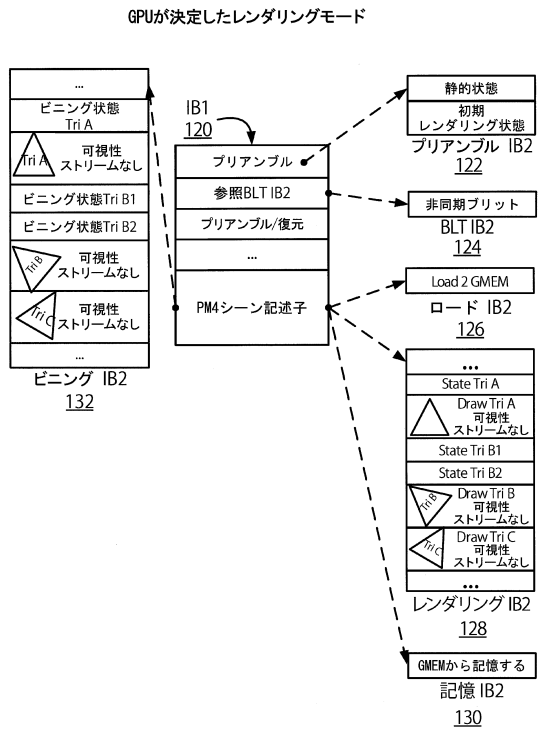
【図７】


直接レンダリングモード
IB1
80
プリアンブル
参照BLT IB2
プリアンブル/復元
レンダーパスを
構成する
プリアンブル/復元
フレームを
レンダリングする
静的状態
初期
レンダリング状態
プリアンブル IB2
82
非同期プリット
BLT IB2
84
ロード IB2
86
State Tri A
Draw Tri A
可視性
ストリームなし
State Tri B1
State Tri B2
Draw Tri B
可視性
ストリームなし
Draw Tri C
可視性
ストリームなし
レンダリング IB2
88

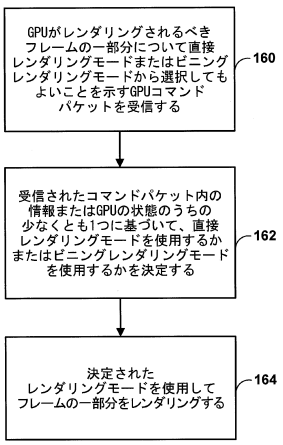
【図８】


IB1
104
<conf>
非同期プリット
<conf>
レンダリングパス
SYS MEM
IB1
102
<conf>
非同期プリット
<conf>
<conf>
レンダリングループ
GMEM
IB1
100
<conf>
非同期プリット
ビニングパス
<conf>
<conf>
レンダリングループ
VIS GMEMを
使用する

【図９】


GPUが決定したレンダリングモード
...
ビニング状態
Tri A
Tri A
可視性
ストリームなし
ビニング状態Tri B1
ビニング状態Tri B2
Tri B
可視性
ストリームなし
Tri C
可視性
ストリームなし
...
ビニング IB2
132
IB1
120
プリアンブル
参照BLT IB2
プリアンブル/復元
...
PM4シーン記述子
静的状態
初期
レンダリング状態
プリアンブル IB2
122
非同期プリット
BLT IB2
124
Load 2 GMEM
ロード IB2
126
State Tri A
Draw Tri A
可視性
ストリームなし
State Tri B1
State Tri B2
Draw Tri B
可視性
ストリームなし
Draw Tri C
可視性
ストリームなし
レンダリング IB2
128
GMEMから記憶する
記憶 IB2
130

【図１０】


GPUがレンダリングされるべき
フレームの一部分について直接
レンダリングモードまたはビニング
レンダリングモードから選択しても
よいことを示すGPUコマンド
パケットを受信する
160
受信されたコマンドパケット内の
情報またはGPUの状態のうちの
少なくとも1つに基づいて、直接
レンダリングモードを使用するか
またはビニングレンダリングモード
を使用するかを決定する
162
決定された
レンダリングモードを使用して
フレームの一部分をレンダリングする
164

フロントページの続き

(72)発明者　アヴィナシュ・セータラマイア
アメリカ合衆国・カリフォルニア・９２１２１－１７１４・サン・ディエゴ・モアハウス・ドライヴ・５７７５
(72)発明者　クリストファー・ポール・フラスカティ
アメリカ合衆国・カリフォルニア・９２１２１－１７１４・サン・ディエゴ・モアハウス・ドライヴ・５７７５
(72)発明者　ジョナラ・ガッダ・ナゲンドラ・クマール
アメリカ合衆国・カリフォルニア・９２１２１－１７１４・サン・ディエゴ・モアハウス・ドライヴ・５７７５
(72)発明者　コリン・クリストファー・シャープ
アメリカ合衆国・カリフォルニア・９２１２１－１７１４・サン・ディエゴ・モアハウス・ドライヴ・５７７５
(72)発明者　デイヴィッド・リゲール・ガルシア・ガルシア
アメリカ合衆国・カリフォルニア・９２１２１－１７１４・サン・ディエゴ・モアハウス・ドライヴ・５７７５

審査官　千葉　久博

(56)参考文献　特開２００９－２９５１６６（ＪＰ，Ａ）
国際公開第２０１３／０８１７８９（WO，Ａ１）
国際公開第２０１３／０８１７８８（WO，Ａ１）
国際公開第２０１２／１５８８１７（WO，Ａ１）
“クアルコム、実使用ベンチマークの重要性を強調”，[online]，２０１２年　７月２６日，[平成３１年４月２４日検索]，インターネット＜ＵＲＬ：https://k-tai.watch.impress.co.jp/docs/news/549294.html＞

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｔ　　１５／００－１５／８７