

(19) 日本国特許庁(JP)

(12) 特許公報(B1)

(11) 特許番号

特許第6073533号
(P6073533)

(45) 発行日 平成29年2月1日(2017.2.1)

(24) 登録日 平成29年1月13日(2017.1.13)

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

請求項の数 22 (全 32 頁)

(21) 出願番号	特願2016-543051 (P2016-543051)	(73) 特許権者	595020643
(86) (22) 出願日	平成26年12月4日 (2014.12.4)		クアルコム・インコーポレイテッド
(86) 国際出願番号	PCT/US2014/068573		QUALCOMM INCORPORATED
(87) 国際公開番号	W02015/099970		ED
(87) 国際公開日	平成27年7月2日 (2015.7.2)		アメリカ合衆国、カリフォルニア州 92
審査請求日	平成28年8月2日 (2016.8.2)		121-1714、サン・ディエゴ、モア
(31) 優先権主張番号	61/921, 145		ハウス・ドライブ 5775
(32) 優先日	平成25年12月27日 (2013.12.27)	(74) 代理人	100108855
(33) 優先権主張国	米国 (US)		弁理士 蔵田 昌俊
(31) 優先権主張番号	14/154, 996	(74) 代理人	100109830
(32) 優先日	平成26年1月14日 (2014.1.14)		弁理士 福原 淑弘
(33) 優先権主張国	米国 (US)	(74) 代理人	100158805
早期審査対象出願			弁理士 井関 守三
		(74) 代理人	100112807
			弁理士 岡田 貴志

最終頁に続く

(54) 【発明の名称】 タイルベースのアーキテクチャ上での最適化されたマルチパスレンダリング

(57) 【特許請求の範囲】

【請求項 1】

タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするための方法において、

グラフィック処理ユニット (GPU) によってクエリパスを実行することと、

フラッシュ動作を実行することなく、前記 GPU によって前記クエリパスに基づいて条件真パスを実行することと、

フラッシュ動作を実行することなく、前記 GPU によって前記クエリパスに基づいて条件偽パスを実行することと、

前記条件真パスおよび前記条件偽パスを実行することに対応して、前記 GPU によってフラッシュ動作を実行することを含む方法。

10

【請求項 2】

前記クエリパスは、第 1 のクエリパスを含み、

前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含む請求項 1 記載の方法。

【請求項 3】

前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む請求項 2 記載の方法。

【請求項 4】

前記条件偽パスを実行することは、前記条件偽パスの終了を示すグラフィックコマンド

20

を実行することをさらに含む請求項 1 記載の方法。

【請求項 5】

直接レンダリングを実行することをさらに含む請求項 1 記載の方法。

【請求項 6】

前記条件真パスを実行することは、第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む請求項 5 記載の方法。

【請求項 7】

前記条件偽パスを実行することは、前記条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含む請求項 1 記載の方法。

【請求項 8】

10
 タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするデバイスにおいて、

メモリと、

G P U とを具備し、

前記 G P U は、

クエリパスを実行するようにと、

フラッシュ動作を実行することなく、前記クエリパスに基づいて条件真パスを実行するようにと、

フラッシュ動作を実行することなく、前記クエリパスに基づいて条件偽パスを実行するようにと、

20
 前記条件真パスおよび前記条件偽パスを実行することに対応して、フラッシュ動作を実行し、前記クエリパス、前記条件真パス、または前記条件偽パスのうちの少なくとも 1 つの結果を前記メモリに書き込むように構成されているデバイス。

【請求項 9】

前記クエリパスは、第 1 のクエリパスを含み、

前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含む請求項 8 記載のデバイス。

【請求項 10】

前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む請求項 9 記載のデバイス。

【請求項 11】

前記条件偽パスを実行することは、前記条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含む請求項 8 記載のデバイス。

【請求項 12】

直接レンダリングを実行することをさらに含む請求項 8 記載のデバイス。

【請求項 13】

前記条件真パスを実行することは、第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む請求項 12 記載のデバイス。

【請求項 14】

40
 前記条件偽パスを実行することは、前記条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含む請求項 8 記載のデバイス。

【請求項 15】

50
 タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするデバイスにおいて、

クエリパスを実行する手段と、

フラッシュ動作を実行することなく、前記クエリパスに基づいて条件真パスを実行する手段と、

フラッシュ動作を実行することなく、前記クエリパスに基づいて条件偽パスを実行する手段と、

前記条件真パスおよび前記条件偽パスを実行することに対応して、フラッシュ動作を実

行する手段とを具備するデバイス。

【請求項 16】

前記クエリパスは、第1のクエリパスを含み、

前記クエリパスを実行する手段は、前記第1のクエリパスを実行する手段を備え、前記第1のクエリパスを実行する手段は、前記第1のクエリパスの開始を示すグラフィックレンダリングコマンドを実行する手段を含む請求項15記載のデバイス。

【請求項 17】

前記第1のクエリパスを実行する手段は、前記第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行する手段をさらに含む請求項16記載のデバイス。

【請求項 18】

前記条件偽パスを実行する手段は、前記条件偽パスの終了を示すグラフィックコマンドを実行する手段をさらに備える請求項15記載のデバイス。

【請求項 19】

直接レンダリングを実行する手段をさらに具備する請求項15記載のデバイス。

【請求項 20】

前記条件真パスを実行する手段は、第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行する手段をさらに備える請求項19記載のデバイス。

【請求項 21】

前記条件偽パスを実行する手段は、前記条件偽パスの開始を示すグラフィックコマンドを実行する手段をさらに備える請求項15記載のデバイス。

【請求項 22】

タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするための命令をその上に記憶した非一時的コンピュータ読取可能記憶媒体において、

前記命令は、実行される際に、1つ以上のグラフィック処理ユニット(GPU)に、クエリパスを実行させ、

フラッシュ動作を実行させることなく、前記クエリパスに基づいて条件真パスを実行させ、

フラッシュ動作を実行させることなく、前記クエリパスに基づいて条件偽パスを実行させ、

前記条件真パスおよび前記条件偽パスを実行させることに応答して、フラッシュ動作を実行させる非一時的コンピュータ読取可能記憶媒体。

【発明の詳細な説明】

【優先権の主張】

【0001】

[0001]

本出願は、2013年12月27日に出願され、そのすべての内容が参照によりここに組み込まれている米国仮出願第61/921,145号の利益を主張する。

【技術分野】

【0002】

[0002]

本開示は、グラフィック処理のための技術に関連し、より具体的には、グラフィック処理においてプリミティブをレンダリングするための技術に関連している。

【背景】

【0003】

[0003]

グラフィカル処理ユニット(GPU)は、タイルベースのレンダリングを実行してもよく、3次元シーンをレンダリングするために使用してもよい。このような3次元シーンのレンダリングは、メモリ帯域幅を非常にたくさん使うことがあることから、特化されたグラフィックメモリ(GMEM)をGPUコアの近くに位置付けてもよい。GPUコアは一般的に、GMEMを使用してシーンをレンダリングする。GPUまたは中央処理ユニット

10

20

30

40

50

(CPU)はその後、シーンを含む、GMEMのコンテンツをシステムメモリにリゾルブしてもよい。言い換えると、シーンを表すデータは、GMEMからシステムメモリに転送されてもよい。移動体環境中のGMEMのサイズは、物理的エリア制約およびメモリ帯域幅により制限されるかもしれないことから、GPUはレンダリングされることになるシーンをより小さなパーツに分け、したがって、これらのより小さなパーツを個々にレンダリングしてもよい。特に、GMEMにレンダリングできる部分にシーンを分割し、シーンの各部分をGMEMに個々にレンダリングすることにより、GPUはシーンをレンダリングしてもよい。

【概要】

【0004】

10

[0004]

一般的に、本開示は、タイルベースのグラフィック処理ユニット(GPU)アーキテクチャに対してグラフィックレンダリングを最適化するための技術を説明している。レンダリング命令を実行するときに、中央処理ユニット(CPU)とGPUとの間の通信およびデータ転送を減らすことにより、タイルベースのGPUは、グラフィカルオブジェクトおよびシーンをレンダリングする際に、その性能を改善することができる。特に、タイルベースのレンダリングをするように構成されているGPUは、CPU対話を待つ必要なく、GPU自体の上で、より多くの量のグラフィカルシーンのレンダリングを実行でき、このことは、GPUのレンダリング性能を改善する。

【0005】

20

[0005]

1つの例において、グラフィック処理ユニット(GPU)によってクエリパスを実行することと、フラッシュ動作を実行することなく、GPUによってクエリパスに基づいて条件真パスを実行することと、フラッシュ動作を実行することなく、GPUによってクエリパスに基づいて条件偽パスを実行することと、条件真パスおよび条件偽パスを実行することに対応して、GPUによってフラッシュ動作を実行することを含む方法を本開示は説明する。

【0006】

[0006]

別の例において、クエリパスを実行するようにと、フラッシュ動作を実行することなく、クエリパスに基づいて条件真パスを実行するようにと、フラッシュ動作を実行することなく、クエリパスに基づいて条件偽パスを実行するようにと、条件真パスおよび条件偽パスを実行することに対応して、フラッシュ動作を実行するように構成されているGPUを含むデバイスを本開示は説明する。

30

【0007】

[0007]

別の例において、グラフィック処理ユニット(GPU)によってクエリパスを実行する手段と、フラッシュ動作を実行することなく、GPUによってクエリパスに基づいて条件真パスを実行する手段と、フラッシュ動作を実行することなく、GPUによってクエリパスに基づいて条件偽パスを実行する手段と、条件真パスおよび条件偽パスを実行することに対応して、GPUによってフラッシュ動作を実行する手段とを含むデバイスを本開示は説明する。

40

【0008】

[0008]

別の例において、コンピュータ読取可能記憶媒体を本開示は説明する。命令をその上に記憶したコンピュータ読取可能記憶媒体において、命令は、実行される際に、1つ以上のプロセッサに、クエリパスを実行させ、フラッシュ動作を実行させることなく、クエリパスに基づいて条件真パスを実行させ、フラッシュ動作を実行させることなく、クエリパスに基づいて条件偽パスを実行させ、条件真パスおよび条件偽パスを実行させることに応答して、フラッシュ動作を実行させる。

50

【図面の簡単な説明】

【0009】

【図1】[0009] 図1は、本開示のいくつかの態様にしたがって、3次元シーンをレンダリングするために使用される、プロセッサ、グラフィック処理ユニット、およびシステムメモリを図示しているブロックダイアグラムである。

【図2】[0010] 図2は、本開示のいくつかの態様にしたがって、タイルベースのレンダリングアーキテクチャ中のシーンのタイルを図示している概念のダイアグラムである。

【図3】[0011] 図3は、本開示の態様にしたがって、ピン間に分割されたプリミティブを示している概念のダイアグラムである。

10

【図4】[0012] 図4は、本開示の技術にしたがって、マルチパスレンダリングを実行するための技術を図示している概念のダイアグラムである。

【図5】[0013] 図5は、本開示中で説明する1つ以上の例にしたがって、ハードウェアによって実行することができる機能の例を図示している概念のダイアグラムである。

【図6】[0014] 図6は、本開示中で説明する1つ以上の例にしたがって、タイルベースのアーキテクチャ上でのマルチパスレンダリングのための例示的な方法を図示しているフローチャートである。

【図7】[0015] 図7は、本開示の1つ以上の態様を実現するように構成されていてもよい、デバイスの例を図示しているブロックダイアグラムである。

20

【詳細な説明】

【0010】

[0016]

グラフィック処理ユニット(GPU)は、3次元(3D)シーンをレンダリングするために使用してもよい。3Dシーンのこのようなレンダリングはメモリ帯域幅を非常にたくさん使うことがあることから、特化されたグラフィックメモリ(GMEM)を使用してもよい。GMEMが非常に広いメモリ帯域幅(すなわち、GMEMへの読み出しおよび書き込みアクセスが比較的速い)を有するように、GMEMはGPUのグラフィック処理コアの近くに位置付けられていてもよい。GPUのグラフィック処理コアによって、シーンをGMEMにレンダリングすることができ、その後、シーンをディスプレイデバイスに表示できるように、シーンをGMEMからメモリ(例えば、フレームバッファ)にリゾルブすることができる。しかしながら、物理的エリア制約のために、GMEMのサイズは、制限されるかもしれないことから、GMEMは、3次元シーン全体(例えばフレーム全体)を含むのに十分なメモリ容量を有していないかもしれない。

30

【0011】

[0017]

いくつかの例において、GPUおよび他の処理デバイスは、3Dシーンをタイルに分けるように構成され、シーンを構成する各タイルがGMEMに適合するようにしてもよい。これは、タイルベースのレンダリングまたは「ピニング」として呼ばれている。例として、GMEMが512kBのデータを記憶することができる場合、各タイル中に含まれるピクセルデータが512kBより少なくまたは等しくなるように、シーンをタイルに分割するかもしれない。このような方法で、GMEMにレンダリングすることができるタイルにシーンを分割し、シーンの各タイルをGMEMに個々にレンダリングすることと、レンダリングされるタイルをGMEMからフレームバッファに記憶させることと、およびシーンの各タイルに対して、レンダリングと記憶を繰り返すこととによって、GPUまたは他のプロセッサは、シーンをレンダリングすることができる。したがって、GPUまたは他のプロセッサは、複数のレンダリングパスを使用して、シーンをタイル毎にレンダリングして、シーンの各タイルをレンダリングすることができる。

40

【0012】

[0018]

50

いくつかの例において、タイルベースのレンダリングを、いくつかのステップにおいて実行してもよい。例えば、タイルベースのアーキテクチャを実現するGPUは、ピニングパスの間に、シーン全体を最初に処理する、または、事前に処理して、「タイル」として呼ばれる多数のピンを規定してもよい。ピニングパスに、一連のレンダリングパスが続き、一連のレンダリングパスの間に、規定されたタイルのそれぞれがレンダリングされる。いくつかの例において、レンダリングパスのそれぞれは、3つのステージ、(1)クリア/アンリゾルブ、(2)レンダリング、(3)リゾルブ、で完了する。クリア/アンリゾルブステージの間、GPUは、新たなタイルのためにGMEMを初期化して、外部メモリから読み込まれる値をGMEMに記憶させてもよい。レンダリングの間、GPUは、現在のタイルに関係付けられている多角形を再生成させるとともに、ピクセル値を発生させて、現在のタイルを完成してもよく、したがって、タイルをディスプレイ上に表示できる。リゾルブステップは、完成したシーンを表示する際にディスプレイによって使用されるバッファのような、GPUの外部のメモリに、オンチップメモリ(GMEM)のコンテンツをGPUがコピーすることを伴ってもよい。

【0013】

[0019]

ピニングパスの間、GPUは、シーンを構成する多角形(例えば、三角形)を発生させ、多角形を複数の「ピン」にソートしてもよい。ここで説明するように、ピニングパスの間に規定されるピンは、ディスプレイ上に提示される最終シーンのタイル(例えば、「スクリーンタイル」として呼ばれることがある)と同意語である。例えば、各ピンは、最終シーンの、一部分またはタイル(例えば、ビデオデータ、コンピュータ発生グラフィック画像、静止画像、またはこれらに類するもののフレームの予め規定された部分)を表す。したがって、用語「ピン」および「タイル」は、ここでは交換可能に使用されるかもしれない。シーンを構成するタイルは、各それぞれのタイル中に含まれるプリミティブを記憶するメモリ中のピンにそれぞれ関係付けることができる。ピンは、メモリの一部、例えば、ピクチャまたはフレームのタイル中のプリミティブである、ピクチャまたはフレームの一部である。シーンのタイルをGMEM中にレンダリングすることは、コマンドを実行して、関係付けられているピン中のプリミティブをGMEM中にレンダリングすることを含んでもよい。GPUのピニングパスは、シーンを構成するプリミティブを適切なピンにソートすることができる。GPUのピニングパスは、最終のレンダリングされるシーン中で、ピン中の何らかのプリミティブが見えるであろうか否かを示す、各ピンに対する可視ストリームも生成させてもよい。可視ストリームは、レンダリングされる時、プリミティブが各タイル中に見えるか否かを示すビットのストリームである。

【0014】

[0020]

ピン中のプリミティブをレンダリングするためのコマンドは、間接バッファ中にロードされてもよい。間接バッファは、例えば、GMEM、フレームバッファ、または他のメモリの一部であってもよい。しかしながら、一般的に、間接バッファはシステムメモリの一部であってもよい。GPUは、間接バッファ中に記憶されているコマンドを実行して、ピン中に含まれるプリミティブをレンダリングしてもよい。ピンに対する可視ストリームが、ピンが何らかの可視プリミティブを含んでいないと示す場合(すなわち、ピン中のプリミティブのすべてが、最終のレンダリングされるシーンにおいて見えないだろう)、ピンに関係付けられている間接バッファ中の命令の実行をスキップすることによって、GPUがピン中のプリミティブをレンダリングしない場合に、性能を向上させることができる。

【0015】

[0021]

マルチパスレンダリングのいくつかの例において、シーンおよび関係付けられているオブジェクトを複数回レンダリングしてもよい。オブジェクトが描かれる毎に、オブジェクトの外観の追加の態様が、計算され、以前の結果と結合されてもよい。一般的に、これは、粗い初期のレンダリングと第1の粗いパスのクエリ結果に基づく詳細な第2のレンダリ

10

20

30

40

50

ングパスとを伴うかもしれない。クエリパス結果は、データ、例えば、ビニングパスが実行されるべきか否かを示すカウンタ値またはヒューリスティクスを含んでいてもよい。例えば、レンダリングされることになるオブジェクトが（相対的に言うと）単純である場合、レンダリングパスのみが続くクエリパスを実行することが有利であるかもしれない。代替的に、レンダリングされることになるオブジェクトが（相対的に言うと）複雑である場合、ビニングパスとレンダリングパスとが続くクエリパスを実行することが有利であるかもしれない。

【 0 0 1 6 】

[0 0 2 2]

いくつかの例において、深度テストを実行して多角形が他の多角形をおおい隠しているか否かを決定するような、ビニングパスの間に動作を実行して多角形のうちのどれがシーン中で見えるかを決定するようにもGPUは構成されていてもよい。どの多角形がシーン中で見えるかを決定すると、GPUは、「可視ストリーム」として呼ばれるデータのストリームを発生させることができる。可視ストリームは、シーンの多角形のそれぞれに対する値を含んでいてもよく、値は多角形が見えるか否かを表してもよい（例えば、「1」の値は多角形が見えると示してもよく、「0」の値は多角形が見えないと示してもよい）。

10

【 0 0 1 7 】

[0 0 2 3]

ビニングパスの後、GPUは、タイルのそれぞれを再度処理することによって、タイルのそれぞれを別々にレンダリングしてもよい。いくつかの例において、GPUはビニングの間に発生させた可視ストリームを使用して、不可視多角形のレンダリングを省略またはスキップする。したがって、可視多角形のみ、すなわち、最終シーンに寄与する多角形のみが処理され、レンダリングされる。GPUは3つのステージ、（1）クリア/アンリゾルブ、（2）レンダリング、（3）リゾルブにおいて、タイルのそれぞれ上でレンダリングプロセスを実行してもよい。

20

【 0 0 1 8 】

[0 0 2 4]

クリア/アンリゾルブステージの間、GPUは、新たなタイルのために、ローカル（例えば、GPUに対してローカル、または、GMEMとしても呼ばれるかもしれない、オンチップGPUメモリに対してローカル）メモリリソースを初期化してもよい。いくつかの例では、クリアプロセスを実行してGMEMをクリアすることにより、GPUはGMEMを初期化する。他の例では、アンリゾルブプロセスを実行することにより、GPUはGMEMを初期化してもよい。アンリゾルブプロセスの間、GPUは、外部メモリからGMEMに値を読み込んでよい。シーンの一部分が新たなデータにより更新されるときのみ、GPUはアンリゾルブプロセスを実現してもよい。例えば、1つのシーンから別のシーンで変化しなかったピクセルデータは、アンリゾルブプロセスを使用して、1つより多くのシーン（例えば、グラフィックデータのうちの1つより多くのフレーム）に渡って保持されてもよい。

30

【 0 0 1 9 】

[0 0 2 5]

レンダリングの間、GPUは現在のタイルに関係付けられている多角形を再生成させるとともに、ピクセル値を発生させ、現在のタイルを完成してもよく、したがって、ディスプレイ上にタイルを表示できる。例えば、レンダリングステージの間、GPUは適切なピクセル値を発生させてもよく、したがって、表示されるピクセルデータはシーンを正確に表す。いくつかの例において、GPUは、（GPUに対して）ローカルに、または、オンチップメモリ、すなわちGMEMに最終ピクセル値を記憶させてもよい。

40

【 0 0 2 0 】

[0 0 2 6]

レンダリングの後、完成したシーンを表示する際にディスプレイによって使用されるバッファのような、GPUの外部のメモリに、オンチップメモリのコンテンツをコピーする

50

ことによって、GPUは現在のタイルをリゾルブしてもよい。GPUは典型的に、ピクセルデータがレンダリングを完成するまで、ピクセルデータをリゾルブするのを待たなければならない。例えば、ピクセルが完全にレンダリングされる前に、GMEMから外部メモリに、GPUがピクセルデータをリゾルブまたはコピーする場合、表示されるときに、結果のシーンは意図されるシーンの適切な属性を示さないだろう。

【0021】

[0027]

いくつかの例において、タイル全体がレンダリングを完成するまで、GPUはタイルをリゾルブするのを待ってもよい。例えば、GMEMから外部メモリにタイルをコピーする前に、タイル全体が表示に対して準備完了となるまで、GPUは待ってもよい。次のタイルに対するGMEMをクリア/アンリゾルブし、次のタイルをレンダリングし、次のタイルをリゾルブすることにより、シーン全体が完成するまで、GPUはプロセスを繰り返す。

10

【0022】

[0028]

タイルベースのアーキテクチャにおいて、ピニングを両方のパス上で実行し、ピニング関連データが両方のパスに対して発生され、これは、余分なフラッシュポイントと中間シーンリゾルブとを伴うかもしれない。単一パスシナリオに渡るこの挙動によって、アプリケーションの性能が増加しないときでさえ、これはその通りであるかもしれない。したがって、いくつかのケースでは、マルチパスレンダリングは、第1のパスレンダリング、クエリ、第2のパスレンダリングを実行することを含んでいるかもしれない。クエリは、アプリケーションによってトリガされる、情報に対する任意の要求であってもよい。例えば、アプリケーションはAPI側で何かをレンダリングし、レンダリングされたオブジェクトをグラフィックカードにフラッシュしてもよい。これは特定のレンダリングを完了させる。その後クエリが送られてもよい。クエリは、APIからグラフィックカードにレンダリングがフラッシュされたときに渡されるピクセルの数であってもよい。クエリは、ハードウェアによってサポートされる、最後のレンダリングジョブが、または、現在のレンダリングジョブが実行される状態についての情報に対する何らかの要求であることがある。いくつかのケースにおいて、アプリケーションは、何かをレンダリングし、クエリをトリガし、クエリ結果に基づいて、データのみを送ってもよい。本願のいくつかの例にしたがうと、アプリケーションは、クエリと複数のレベル2間接バッファ(IB2)とを送ってもよい。IB2はレンダリングパイプラインのさまざまな状態に対するコマンドを含んでいる。例えば、GPUの静的状態を初期化し、GPUの初期レンダリング状態を設定するコマンドのような、GPUによって実行可能なプリアンブルコマンドを、IB2は含んでもよい。GPUのレンダリング状態は、特定のアプリケーションに基づいて変化するかもしれないGPU設定を含んでもよい。IB2は、一連の状態コマンドと、ロードされたピン中に三角形を描画するための描画コマンドとを含んでもよい。各描画コマンドは、グラフィック処理パイプラインにしたがって、三角形を描画するようにGPUに命令してもよい。IB2 68は、GPUによって実行されるグラフィック処理パイプラインの挙動に影響を及ぼすかもしれない。例えば、状態コマンドは、色、多角形モード(例えば、立体または線の代わりに点)、ブレンド化(オン/オフ)、深度テスト(オン/オフ)、テクスチャ化(オン/オフ)、選別、クリップ、および他の論理動作を変えてもよい。IB2状態コマンドは、三角形毎(またはプリミティブ毎)ベースで発行されてもよい。

20

30

40

【0023】

[0029]

例において、アプリケーションは各可能性あるクエリ結果に対して1つのIB2を送ってもよい。したがって、各可能性あるクエリ結果に対するレンダリングパイプラインのさまざまな状態に対するコマンドを送ってもよい。1つの例において、2つのクエリ結果は、「真」および「偽」である可能性があるかもしれない。したがって、アプリケーション

50

は2つのIB2を、「真」のクエリ結果に対する1つのIB2と、「偽」のクエリ結果に対する1つのIB2とを送ってもよい。したがって、「真」のクエリ結果に対するレンダリングパイプラインのさまざまな態様に対するコマンドを送ってもよく、「偽」のクエリ結果に対するレンダリングパイプラインのさまざまな態様に対するコマンドを送ってもよい。アプリケーションは両方の可能性を、「真」のクエリ結果に対するIB2と、「偽」のクエリ結果に対するIB2とを送ってもよい。

【0024】

[0030]

「真」のクエリ結果に対するIB2と、「偽」のクエリ結果に対するIB2との両方をアプリケーションは送ることから、アプリケーションは、クエリ結果に基づいて、データを送るのを待つ必要はない。むしろ、GPUは、アプリケーションによって送られた、「真」のクエリ結果に対するIB2と、「偽」のクエリ結果に対するIB2との両方を有しており、GPUがクエリの結果を待ってもよい。したがって、アプリケーションを待たせるよりもむしろ、GPUがクエリ結果を待ち、その後、GPUがクエリパスを実行してもよい。クエリパスの結果が「真」である場合、GPUは条件真パスを実行してもよい。さらに、フラッシュ動作を実行することなく、条件真パスを実行してもよい。クエリパスの結果が「偽」である場合、GPUは条件偽パスを実行してもよい。フラッシュ動作を実行することなく、条件偽パスも実行してもよい。クエリパスの結果に基づいて、条件真パスまたは条件偽パスのうちの1つを条件付きで実行することに対応して、その後、GPUはフラッシュ動作を実行してもよい。

【0025】

[0031]

したがって、本開示のいくつかの態様では、クエリ結果を待つことは、アプリケーションからGPUに移される。言い換えると、アプリケーションよりもむしろ、GPUがクエリの結果を待ってもよい。いくつかの例では、条件「真」のケースに対するIB2と、条件「偽」のケースに対するIB2との両方をアプリケーションがGPUに送ることができることから、これは可能である。したがって、「真」のケースと「偽」のケースの両方に対して、アプリケーションがレンダリングするであろうものと、アプリケーションが実行したいものとのGPUは有している。なぜなら、IB2のそれぞれは、そのそれぞれのケースに対する、例えば、「真」クエリ結果と「偽」クエリ結果とのそれぞれに対する条件「真」のケースと条件「偽」のケースとに対するレンダリングパイプラインのさまざまな態様に対するコマンドを含んでいるからである。

【0026】

[0032]

パスレンダリング上の追加のクエリおよび続くものは次の通りである。タイルシステムに対して、これは以下のシーケンスに対応していてもよい。(1)粗いパスまたはクエリパスとも呼ばれる、第1のパスレンダリング。第1のパスレンダリングでは、タイルシステムが、可視ストリームを発生させるピンングパスを実行し、第1のパスのレンダリングパスに対して、ロードを取り扱い、レンダリングし、記憶する。(2)(アプリケーションによるかもしれない)クエリチェック。クエリチェックは、クエリパス(第1のパス)の結果がチェックされる時間である。(3)第2のパス。第2のパスは、第1のパスのクエリ結果に基づいてなされるレンダリングのすべてを含んでいてもよい。このような例において、タイルベースのシステムは、ピンングパスを実行し、可視ストリームを発生させ、この第2のパス(第2のパスは、アプリケーションの挙動に基づいて詳細なジオメトリの異なるセットを有しているかもしれない、これは、そのケースである可能性が最も高い)のレンダリングパスに対して、ロード、レンダリング、記憶を実行してもよい。したがって、タイルベースのシステムでは、両方のパスに対するピン化されたレンダリングをトリガするかもしれないバスアクセスにより、ボトルネックが生じるかもしれない。したがって、可視ストリームの使用により得られる何らかの最適化は、最小化されるかもしれない。なぜなら、これらのステップを実現するアプリケーションが、クエリパスを実行して、

データを、例えば、ピニングパスを実行すべきか否かを示すカウンタ値またはヒューリスティックス、を決定できるようにするために、余分なフラッシュを生じさせるかもしれないためである。これらのカウンタ値またはヒューリスティックスは、クエリパスレンダリング統計としても呼ばれることがある。さらに、一般的にデータを、例えば、ピニングパスを実行すべきか否かを示すカウンタ値またはヒューリスティックスを、初期パスの一部として、発生させ、または、決定してもよい。

【 0 0 2 7 】

[0 0 3 3]

一般的に、いくつかの例において、グラフィックアプリケーションは、クエリをトリガし、粗いパス（第1のパス）をレンダリングし、その後クエリを終了する。グラフィックアプリケーションは、クエリ値（すなわち、詳細なレンダリングが要求されるか否かを検出するために渡されるピクセルの数）をチェックしてもよい。クエリ結果に基づいて（グラフィックアプリケーションが第2のパスをトリガしてもよい）。クエリが真であるとき、詳細なシーンをレンダリングしてもよく、クエリが偽であるとき、シーンをまったくレンダリングしなくてもよいが、または、粗くてもよいが、あるいは、しかしながらカラーパイプラインシーンをレンダリングしてもよい。したがって、第2のパスはレンダリングのすべてを含んでいてもよく、これは、第1のパスのクエリ結果に基づいて、実行されてもよく、または、実行されなくてもよい。

10

【 0 0 2 8 】

[0 0 3 4]

いくつかの例は、マルチパスレンダリングに対するそれらの利点においてタイルベースのアーキテクチャを使用するために、アプリケーションの挙動を修正してもよい。いくつかの例は、新たな実行ポイントを規定して、完全な制御をGPUとグラフィックドライバに移してもよい。例えば、いくつかの例は、（1）導入されたクエリ条件を有する第1のパス中の、開始__クエリ__パスを呼び出し、（2）クエリ__パスに対するレンダリングを呼び出しを実行依頼し、（3）クエリ__パスを終了し（クエリ__パス__終了を呼び出し）、（4）開始__条件__真__パスを呼び出し、（5）条件__真__パスに対するレンダリングを呼び出しを実行依頼し、（6）終了__条件__真__パスを呼び出し、（7）開始__条件__偽__パスを呼び出し、（8）条件__偽__パスに対するレンダリングを呼び出しを実行依頼し、（9）終了__条件__偽__パスを呼び出ししてもよい。

20

30

【 0 0 2 9 】

[0 0 3 5]

1つの例では、間接バッファ1（IB1）は、クエリパス、ピニングパス、またはレンダリングパスを呼び出ししてもよい。クエリパス、ピニングパス、およびレンダリングパスは、間接バッファ2（IB2）の一部であってもよい。IB1およびIB2は、バッファ、例えば、マルチレベルバッファである。最上レイヤバッファ（IB1）中のコマンドを使用して、より低いレベルのバッファ（IB2）中のコマンドのセット全体を呼び出ししてもよい。例において、クエリパスに対するすべてのコマンドを含んでいてもよいクエリパスIB2を呼び出す、IB1中のコマンドによって、クエリパスを実行してもよい。別のIB2は、ピニングに対するすべてのコマンドを含んでいてもよく、さらに別のIB2は、レンダリング等に対するすべてのコマンドを含んでいてもよい。例えば、クエリパス、ピニングパス、およびレンダリングパスは、それぞれ別々のIB2、すなわちIB2レベルバッファ内の別々のエンティティであってもよい。クエリパスはピニングパスの前に実行してもよい。ピニングパスはレンダリングパスの前に実行してもよい。いくつかのケースでは、ピニングパスをスキップしてもよく、したがって、レンダリングパスはクエリパスの直後に実行される。いくつかのケースでは、ピニングパスもレンダリングパスもクエリパスの後に実行されない。

40

【 0 0 3 0 】

[0 0 3 6]

例えば、クエリパスは、データを、例えば、ピニングパスを実行すべきか否かを示す、

50

カウンタ値またはヒューリスティクスを戻してもよい。例において、条件付き実行決定ユニットは、ビニングパスまたはレンダリングパスを実行すべきか否かを決定してもよい。ビニングパスを実行する際の決定は、レンダリングされることになるオブジェクトの複雑性に基づいていてもよい。例えば、単純なオブジェクトに対しては、ビニングパスをスキップしてもよい。逆に、より複雑なオブジェクトに対しては、ビニングパスを実行してもよい。したがって、より単純なオブジェクトに対しては、ビニングパスをスキップして、クエリパスの直後にレンダリングパスを実行してもよい。さらに、単純なオブジェクトに対しては、単一の繰り返しにおいて、レンダリングを実行してもよい。例えば、スクリーン全体が、一連のブロック中よりもむしろ単一のパス中で書かれてもよい。これは、とても単純な、反復スクリーンレンダリングに対して可能であるかもしれない。

10

【 0 0 3 1 】

[0 0 3 7]

図 1 は、本開示のいくつかの態様にしたがって、3次元(3D)シーンをレンダリングするために使用される、プロセッサ 102、グラフィック処理ユニット(GPU) 120、およびシステムメモリ 118を図示しているブロックダイヤグラムである。プロセッサ 102は、ソフトウェアアプリケーション 112、オペレーティングシステム(OS) 114、およびグラフィックドライバ 116を実行してもよい。プリミティブをレンダリングするためのコマンドストリームとともに、GPU 120によって実行されることになる2次コマンドを記憶する間接バッファを、システムメモリ 118は含んでいてもよい。GPU 120は、GMEM 122を含んでいてもよい。GMEM 122は、上記で説明した

20

【 0 0 3 2 】

[0 0 3 8]

図 1 の例において、プロセッサ 102、システムメモリ 118、およびGPU 120は、デバイスの一部であってもよい。デバイスの例は、ビデオデバイスや、メディアプレーヤーや、セットトップボックスや、移動体電話機およびいわゆるスマートフォンのようなワイヤレスハンドセットや、パーソナルデジタルアシスタント(PDA)や、デスクトップ

30

【 0 0 3 3 】

[0 0 3 9]

プロセッサ 102は、中央処理ユニット(CPU)であってもよい。GPU 120は、ディスプレイでの表示のためのグラフィックデータを発生させて、出力するような、グラフィック関連機能を実行するとともに、GPU 120によって提供される大量な並列処理を活用する、非グラフィック関連機能を実行するように構成されている処理ユニットであってもよい。例えば、GPU 120は、グラフィックアプリケーションおよび非グラフィックアプリケーションの両方を実行してもよい。GPU 120はグラフィック処理能力に加えて、汎用処理能力を提供することができることから、GPU 120は、汎用GPU(GP-GPU)として呼ばれることがある。

40

【 0 0 3 4 】

[0 0 4 0]

プロセッサ 102およびGPU 120の例は、デジタル信号プロセッサ(DSP)、汎用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論理アレイ(FPGA)、または他の同等な集積されたまたはディスクリットな論理回路を含むが、これらに限定されない。いくつかの例では、GPU 120は、グラフィックを処理するために大量な並列処理を提供するとともに、非グラフィック関連アプリケーショ

50

ンを実行するような、特定の使用法に対して設計されたマイクロプロセッサであってもよい。さらに、プロセッサ102およびGPU120は、別々のコンポーネントとして図示されているが、本開示の態様は、このように限定されない。例えば、プロセッサ102およびGPU120は、共通集積回路(IC)中に存在してもよい。

【0035】

[0041]

プロセッサ102上で実行するソフトウェアアプリケーション112は、プロセッサ102に命令して、グラフィックのデータを(示されていない)ディスプレイにレンダリングさせる、1つ以上のグラフィックレンダリング命令を含んでいてもよい。いくつかの例において、グラフィックレンダリング命令は、例えば、オープングラフィックライブラリ(オープンGL(登録商標))アプリケーションプログラミングインターフェース(API)、オープングラフィックライブラリ組み込みシステム(OpenGL ES)API、ダイレクト3D API、X3D API、レンダーマンAPI、WebGL API、または他の何らかの公開またはプロプライエタリな標準グラフィックAPIのような、グラフィックAPIに準拠する、ソフトウェア命令を含んでいてもよい。グラフィックレンダリング命令を処理するために、プロセッサ102は、1つ以上のグラフィックレンダリングコマンドを(例えば、グラフィックドライバ116を通して)GPU120に発行して、GPU120に、グラフィックデータのレンダリングのうちのいくつかのまたはすべてを実行させてもよい。いくつかの例において、レンダリングされることになるグラフィックデータは、グラフィックプリミティブのリストを、例えば、点、線、三角形、四角形、三角形ストリップ等を含んでいてもよい。

10

20

【0036】

[0042]

GPU120は、グラフィック動作を実行して、1つ以上のグラフィックプリミティブをディスプレイにレンダリングするように構成されていてもよい。したがって、プロセッサ102上で実行するソフトウェアアプリケーションのうちの1つがグラフィック処理を要求するとき、プロセッサ102は、ディスプレイへのレンダリングのために、グラフィックコマンドおよびグラフィックデータをGPU120に提供してもよい。グラフィックデータは、例えば、描画コマンド、状態情報、プリミティブ情報、テクスチャ情報等を含んでいてもよい。GPU120は、いくつかの事例において、プロセッサ102よりも、複雑なグラフィック関連動作のより効率的な処理を提供する、高度に並列な構造で構築されていてもよい。例えば、GPU120は、並列方法で、複数の頂点またはピクセルを操作するように構成されている、複数の処理エレメントを含んでいてもよい。プロセッサ102の高度に並列する性質は、いくつかの事例において、プロセッサ102を使用してシーンをディスプレイに直接描画するよりも、GPU120がグラフィック画像(例えば、GUIならびに2次元(2D)および/または3次元(3D)グラフィックシーン)をディスプレイ上により速く描画することができるようにする。

30

【0037】

[0043]

GPU120は、GMEM122に直接結合されていてもよい。言い換えると、GPU120は、オフチップメモリの代わりに、ローカル記憶装置を使用して、データをローカルに処理してもよい。これは、多量のバストラフィックを経験するかもしれない、例えば、共有バスを介してGPU120がデータを読み書きする必要をなくすことにより、GPU120が、より効率的な方法で動作できるようにする。しかしながら、いくつかの事例において、GPU120は、別のメモリを含んでいないかもしれないが、代わりに、システムメモリ118を利用していてもよい。GMEM112は、例えば、ランダムアクセスメモリ(RAM)、静的RAM(SRAM)、動的RAM(DRAM)、および1つ以上のレジスタのような、1つ以上の揮発性または不揮発性メモリあるいは記憶デバイスを含んでいてもよい。

40

【0038】

50

[0 0 4 4]

プロセッサ 1 0 2 および / または G P U 1 2 0 は、レンダリングされる画像データをフレームバッファ 1 2 4 中に記憶させてもよい。フレームバッファ 1 2 4 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 1 1 8 内に割り振られてもよい。(示されていない) ディスプレイプロセッサは、レンダリングされる画像データをフレームバッファ 1 2 4 から取り出して、レンダリングされる画像データをディスプレイ上に表示させてもよい。

【 0 0 3 9 】

[0 0 4 5]

システムメモリ 1 1 8 は、デバイスのシステムメモリであってもよく、プロセッサ 1 0 2 と G P U 1 2 0 との外部に存在し、すなわち、プロセッサ 1 0 2 に関してオフチップであり、G P U 1 2 0 に関してオフチップであってもよい。システムメモリ 1 1 8 は、プロセッサ 1 0 2 と G P U 1 2 0 によって実行されるアプリケーションを記憶してもよい。さらに、システムメモリ 1 1 8 は、実行されるアプリケーションが操作するデータとともに、アプリケーションから結果的に生じるデータを記憶してもよい。しかしながら、すべての例において、このようなデータのすべてを、システムメモリ 1 1 8 中に記憶させる必要はない。いくつかの事例において、データは、プロセッサ 1 0 2 または G P U 1 2 0 上にローカルに記憶させてもよい。例えば、データのうちのいくつかまたはデータのすべてを、オンチップ G P U メモリ (例えば、グラフィックメモリ、G M E M 1 2 2) 内にローカルに記憶させてもよい。

【 0 0 4 0 】

[0 0 4 6]

システムメモリ 1 1 8 は、プロセッサ 1 0 2 による実行のためにアクセス可能であるプログラムモジュール、命令、または両方、プロセッサ 1 0 2 上で実行するプログラムによる使用のためのデータ、あるいは、これらのうちの 2 つ以上を記憶してもよい。例えば、システムメモリ 1 1 8 は、プロセッサ 1 0 2 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、ディスプレイ上にグラフィカルユーザインターフェース (G U I) を提示してもよい。加えて、システムメモリ 1 1 8 は、ユーザアプリケーションと、アプリケーションに関係付けられている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 1 1 8 は、G P U 1 2 0 のためのデバイスメモリとして機能してもよく、G P U 1 2 0 によって操作されることになるデータとともに、G P U 1 2 0 によって実行される操作から結果的に生じるデータを記憶してもよい。例えば、システムメモリ 1 1 8 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、またはこれらに類するものの任意の組み合わせを記憶してもよい。

【 0 0 4 1 】

[0 0 4 7]

システムメモリ 1 1 8 は、コンピュータ読取可能記憶媒体の例であってもよい。例えば、システムメモリ 1 1 8 は、プロセッサ 1 0 2 および G P U 1 2 0 に本開示中のそれぞれに与えられた機能を実行させる命令を記憶してもよい。システムメモリ 1 1 8 は、1 つ以上のプロセッサ (例えば、プロセッサ 1 0 2 または G P U 1 2 0) にさまざまな機能を実行させる命令を含むコンピュータ読取可能記憶媒体として考えてもよい。

【 0 0 4 2 】

[0 0 4 8]

システムメモリ 1 1 8 の例は、ランダムアクセスメモリ (R A M)、リードオンリーメモリ (R O M)、または、電氣的消去可能プログラマブルリードオンリーメモリ (E E P R O M (登録商標))、あるいは、命令またはデータ構造の形態で所望のプログラムコードを伝えるまたは記憶させるために使用することができ、コンピュータまたはプロセッサによってアクセスできる他の何らかの媒体を含むが、これらに限定されない。1 つの例として、システムメモリ 1 1 8 をデバイスから取り除き、別のデバイスへ移動してもよい。

10

20

30

40

50

別の例として、システムメモリ 118 と実質的に類似する記憶デバイスを、デバイスに挿入してもよい。

【0043】

[0049]

本開示の技術にしたがうと、いくつかの例は、マルチパスレンダリングに対するそれらの利点においてタイルベースのアーキテクチャを使用するために、アプリケーションの挙動を修正してもよい。いくつかの例において、アプリケーションは新たな実行ポイント/バックドア/拡張を使用してもよく、GPUおよびグラフィックドライバに完全な制御を移す。

【0044】

[0050]

例えば、何がアプリケーションに実行依頼されているかの表示をGPUがドライバに提供できるように、アプリケーションに「エクスポーズアウト」するさまざまなエントリポイントをGPUは呼び出してもよい。「エクスポーズアウト」は、それらが必要とされるときに、コードブロックをトリガできる機能的ポイントを提供する、コードのブロックへのエントリポイントを指す。一般的に、アプリケーションが、何らかの開始手順を、すなわち、開始XXXを呼び出す場合、呼び出しはドライバ中で終わり、いくつかの例では、続くレンダリング/状態呼び出しは、終了呼び出しが、すなわち終了XXXが呼び出されるまでの、XXXパスに対するものである。結果として、関係付けられているパスの開始と終了の間の、すべてのレンダリング/状態呼び出しは、蓄積され、これらのパスに対する間接バッファを構築するために使用されてもよい。例示的な呼び出しは、導入されたクエリ条件を有する開始__クエリ__パス(第1のパス)を含んでいる。GPUは、通常通りクエリ__パス__に対するレンダリング呼び出しを実行依頼してもよく、クエリ__パス__終了を呼び出してもよい。GPUは、開始__条件__真__パスと設定クエリも呼び出してもよい。GPUは、条件__真__パスに対するレンダリング呼び出しを実行依頼してもよい。GPUはまた、終了__条件__真__パスと開始__条件__偽__パスを呼び出してもよい。GPUはまた、条件__偽__パスに対するレンダリング呼び出しを実行依頼し、終了__条件__偽__パスを呼び出ししてもよい。したがって、例において、1つのフラッシュのみが必要とされるかもしれない。フラッシュは、すべての蓄積されたレンダリングコマンドをオペレーティングシステムに実行依頼または送ることである。グラフィックアプリケーションがレンダリングコマンドをトリガするとき、コマンドはハードウェアに、例えばスクリーンに直接送られない。むしろ、グラフィックドライバにより、コマンドは蓄積される(必要に応じて変換される)。フラッシュ呼び出しは境界であり、そこでは、レンダリングが取り扱われることが要求され、ドライバが、オペレーティングシステムカーネルを通して、すべての蓄積されたコマンド/バッファをハードウェアに送る/実行依頼する。

【0045】

[0051]

いくつかの例では、すべてのデータが送られるまで、アプリケーションはフラッシュする必要がないかもしれない。さらに、アプリケーションは、クエリデータを明示的にチェックする必要がないかもしれない。例では、ハードウェアは、クエリパス、ピニングパス、およびレンダリングパスを実行してもよい。結果として、ピニングパスは、クエリ結果に基づいて、マッチングするレンダリングパスに対して、一度トリガされてもよい。一般的に、よく書かれたアプリケーションに対して、このパスは、他のパスと比較して、実行するためにより多くの処理サイクルを要求するかもしれない。したがって、ハードウェアリソースのよりよい利用により、単一のフラッシュポイントにおいて、動作を完了してもよい。したがって、さまざまな例は、不必要なロード/記憶(リゾルブ/アンリゾルブ)およびフラッシュポイントをなくすかもしれない。

【0046】

[0052]

図2は、タイルベースのレンダリングアーキテクチャ中のシーンのタイルを図示してい

10

20

30

40

50

る概念のダイアグラムである。図 2 中に示すように、図 1 中で示す GPU 120 のような GPU によってレンダリングされることになる 3D グラフィカルオブジェクト 206 は、プリミティブ 208 のようなプリミティブから構成されていてもよい。図 2 中で示す例において、プリミティブは、3つの頂点を含む三角形であってもよい。他の例において、プリミティブは、点、線、およびこれらに類するものであってもよい。グラフィカルオブジェクト 206 を含んでいる 3D シーン 202 は、タイル 204 のような、タイルに分割できる。シーン 202 のタイル 204 のような、各タイルのサイズは、GMEM のサイズに少なくとも部分的に基づいて決定してもよい。例えば、タイル中に含まれているシーン 202 の一部分が、図 1 中に示す G MEM 122 のような、グラフィックメモリ中に完全にレンダリングされるように、シーン 202 の各タイルは、サイズが決められてもよい。シーン 202 の各タイルは、そのタイル中に三角形を含むピンであると考えてもよい。1つの例において、ピンの幅および高さは、32ピクセルで整列させてもよい。シーン 202 は、5×5グリッドのタイルに分割されることから、図 2 中に示されているシーン 202 に対して合計 25 タイルがある。

10

【 0047】

[0053]

GPU 120 は、三角形をレンダリングするためのコマンドを実行することによって、三角形をレンダリングしてもよい。したがって、グラフィカルオブジェクト 206 を構成する三角形のそれぞれをレンダリングするためのコマンドを実行することにより、GPU 120 は、グラフィックオブジェクト 206 をレンダリングしてもよい。各ピンが、ピン中に含まれる三角形をレンダリングするためのコマンドストリーム(1組のコマンド)を含むように、GPU 120 は、シーンの三角形をピンにソートしてもよい。シーン 202 に対して合計 25 タイルがあることから、シーン 202 に対して 25 の対応するピンがあってもよい。図 1 中に示すシステムメモリ 108 のような、メモリ中の間接バッファ中に、各ピンに対するコマンドストリームを記憶させてもよい。各ピンのコマンドストリームを実行して、ピンのそれぞれにおける三角形を G MEM 122 上にレンダリングすることにより、GPU 120 は、グラフィカルオブジェクト 206 をレンダリングする。

20

【 0048】

[0054]

いくつかの例では、シーンをレンダリングするために、GPU 120 は第 1 の粗いパスを実行する。GPU はその後第 2 の細かいパスを実行してもよい。第 1 の粗いパスの間、GPU 120 は、ピンのそれぞれの中の三角形が見えるか否かを決定してもよい。以前の技術では、GPU が第 1 のパスを完了した後、CPU はフラッシュ動作を実行する。フラッシュ動作は第 1 のパスの結果を記憶して、結果を CPU に戻す。結果は、例えば、どの三角形が見え、どの三角形が見えない等を含んでいてもよい。

30

【 0049】

[0055]

クエリ動作の結果に、例えば、どの三角形が見え、どの三角形が見えないか、に基づいて、CPU は第 2 のパスに対するパラメータを発生させる。第 2 のパスの間、GPU は第 2 のピンングパスを実行する。GPU はまた、第 2 のパスの間に、可視ストリームを発生させる。GPU は、第 2 のパスの間に、新たな可視ストリームを発生させてもよい。この新たな可視ストリームを使用して、GPU は第 2 のレンダリングパスを実行する。第 2 のパスの後、GPU は別のフラッシュ動作を実行する。第 2 のフラッシュ動作中で、GMEM のコンテンツは、グラフィックバッファまたはシステムメモリ 118 に書き込まれてもよい。

40

【 0050】

[0056]

GPU 120 が、ピン中に含まれるシーン 202 の一部分を G MEM 122 上にレンダリングした後、シーン 202 のそのレンダリングされた部分は、G MEM 112 から、図 1 中に示すフレームバッファ 124 のようなメモリにロードしてもよい。GPU 120 は

50

、コマンドストリームを実行し、ピンの三角形を G M E M 1 2 2 上にレンダリングし、シーン 2 0 2 のレンダリングされた部分を G M E M 1 2 2 からフレームバッファ 1 2 4 にロードするプロセスを、各ピンに対して繰り返して、シーン 2 0 2 を完全にレンダリングしてもよい。

【 0 0 5 1 】

[0 0 5 7]

ここで説明するように、「ピニング」または「タイルベースのレンダリング」は 3 D シーンをより小さなパーツでレンダリングするための方法である。3 D レンダリングは、メモリ帯域幅上で非常に重いことから、3 D コアに近く、高い帯域幅を有する、特化されたグラフィックメモリ G M E M を使用することが有用である。しかしながら、移動体環境では、G M E M のサイズは、エリア制約により制限されている。したがって、それぞれを別々にレンダリングすることができるように、シーンをより小さなパーツに分ける必要があるかもしれない。

10

【 0 0 5 2 】

[0 0 5 8]

別の例では、フェイスネスストリームを使用してもよいが、それは、各ピンに対して別々に適用される。他の例では、フェイスネスストリームは、三角形が正面を向いているかまたは背面を向いているかを示す、三角形毎のビットのデータを含んでもよい。このような例では、これは、可視ストリームに拡張してもよく、可視ストリームでは、所定のピンに対して三角形が少しでも見えるか否かを各ビットは示す。各ピンに対して、1つの可視ストリームがあり、そのピンに対して見える三角形をリストアップする。可視値は、複数のファクター、(1) 三角形がバックフェースカリングされているか否か、(2) 三角形が (Z 方向を含む) ピンエリアにヒットするか否か、(3) 低解像度 Z チェックによって、三角形が妨げられるか否か、を使用して計算してもよい。

20

【 0 0 5 3 】

[0 0 5 9]

例において、ピニングパスの間、各ピンに対して1つの可視ストリームである、複数の可視ストリームが生成される。レンダリングパスの間、(現在のピンに対して1つの) ただ1つの可視ストリームが読み取られる。さらに、可視ストリームは圧縮される。これは結果として、より小さなメモリ消費となる。また、レンダリングステージの間、見えない三角形の高速スキップも可能にする。

30

【 0 0 5 4 】

[0 0 6 0]

例において、ピニングパスの間に可視ストリームを発生させてもよい。これは、シーン全体のコマンドストリームを処理することを伴うかもしれない。しかしながら、一般的に、ピクセルシェーディングはされない。生成は以下のステージ、(1) ピンシェーダーによる頂点シェーディング、(2)、低解像度ラスタ化 (3)、低解像度 Z テスト、および (4) 可視ストリーム圧縮を含んでもよい。

【 0 0 5 5 】

[0 0 6 1]

両方の例において、ピニングパスは特定のピニングシェーダーを要求する。これは、頂点シェーダの修正バージョンであってもよく、唯一出力されるのは、頂点ポジションである。ピニングシェーダーからとともに、それに関連する任意の計算から、すべてのパラメータのエクスポートを取り除いてもよい。しかしながら、いくつかの例において、何らかの特定のピン関連シェーダーコードを追加する必要はない (その上、ドライバ開発の初期ステージの間、通常の頂点シェーダーをピニングシェーダーとして使用することも可能であるかもしれない。このケースにおいて、使用する適切なピクセルシェーダーも同様であるべきであるが、それは、何らかのピクセルをただ受信するのではない) 。

40

【 0 0 5 6 】

[0 0 6 2]

50

シェーディングされた頂点から、ラスタライザは三角形の低解像度表現を発生させ、各ピクセルは最終画像における4×4ピクセルエリアと等しい。発生させた低解像度ピクセルは、部分的におおい隠されている、または、完全におおい隠されている、2つの値をとることができる。ラスタライザは、通常のラスタライザである同じ選別ルール（フェイスネス、錐台等）を使用し、したがって、実際にシーンに寄与する三角形のみを生成させる。

【0057】

[0063]

プロセス中の第3のステージは、低解像度Zテストである。GMEMは、ピニングパスの間、Zバッファとしても使用することができる。レンダリングは4×4ピクセルブロックでされることから、GMEM中のZ-バッファもこの解像度中にある。さらに、GMEM中にカラーバッファは必要ない。これは、低解像度Z-バッファ（LRZ-バッファ）が、フル解像度と比較して、とても広いオンスクリーンエリアをカバーできることを意味する。LRZ-バッファはフル解像度で動作しないことから、LRZ-処理は保守的にする必要はある。LRZ-バッファへの書き込みは、三角形によって完全におおい隠されるピクセルに対してのみなされ、部分的におおい隠されるピクセルは、Z書き込みに寄与しない。三角形のエッジにギャップがあることがあるので、これもLRZバッファが完全には正確でないことを意味する。ピニングパスの終りにおいて、LRZバッファを外部メモリに書き出すことができ、その後、これはレンダリングパスの間にZ-バッファを初期化するために使用できる。これは、レンダリングの間、改善されたZテストを提供する。

【0058】

[0064]

図3は、本開示の態様にしたがって、ピン間で分割されたプリミティブを示している概念のダイアグラムである。図3中に示すように、それぞれが4×4グリッドのピクセルを含む、ピン302、304、306、および308が、複数のピクセル310を含むようにレンダリング/ラスタライザされる。一般的に、ビデオレンダリングは、既存のオブジェクトまたはモデルに基づいて、画像を発生させるプロセスである。ラスタライザは、一般的に、ビデオディスプレイまたはプリンタ上の出力のために、あるいは、ビットマップファイルフォーマットでの記憶のために、ベクトルグラフィックフォーマット（例えば、形状）で記載された画像を取り、それをラスタライザ画像（例えば、ピクセルまたはドット）に変換することである。

【0059】

[0065]

1つ以上のグラフィックプリミティブが、各ピン中に見えるかもしれない。例えば、三角形A（三角形A）の一部分は、ピン302およびピン306の両方の中に見える。三角形B（三角形B）の一部分は、ピン302、ピン304、ピン306、およびピン308のそれぞれのおいて見える。三角形C（三角形C）は、ピン304中のおいてのみ見える。レンダリングパスの間、GPU120は、シーンをピンに分けて、三角形をピンに割り当ててもよい。三角形が1つのピンより多くで見える場合、ピン302、304、306、308のそれぞれがレンダリングされるときに、三角形が複数回レンダリングされないように、GPU120は、三角形が見える、ピンのうちのただ1つに三角形を割り当ててもよい。

【0060】

[0066]

GPU120は、ピン中のどの三角形が、最終のレンダリングされるシーンのおいて実際に見えるかも決定してもよい。例えば、いくつかの三角形は、1つ以上の他の三角形の後ろにあるかもしれない、最終のレンダリングされるシーンのおいて見えないだろう。このような方法で、見えない三角形は、そのピンに対してレンダリングする必要はない。

【0061】

[0067]

特定のレンダリングパスを実行している間に、その特定のレンダリングパスに関係付け

10

20

30

40

50

られている、ピンに対するピクセルデータは、図 1 中で示す G M E M 1 2 2 のような、(時には、ピンバッファと呼ばれる)グラフィックメモリ中に記憶させてもよい。レンダリングパスを実行した後、G P U 1 2 0 は、G M E M 1 2 2 のコンテンツをフレームバッファ 1 2 4 に転送してもよい。いくつかのケースにおいて、G P U 1 2 0 は、フレームバッファ 1 2 4 中のデータの一部を、G M E M 1 2 2 中に記憶されているデータで上書きしてもよい。他のケースにおいて、G P U 1 2 0 は、G M E M 1 2 2 中に記憶されているデータと、フレームバッファ 1 2 4 中のデータとを合成または組み合わせてもよい。G M E M 1 2 2 のコンテンツをフレームバッファ 1 2 4 に転送した後、G P U 1 2 0 は、G M E M 1 2 2 をデフォルト値に初期化して、異なるピンに関して、後続するレンダリングパスを開始してもよい。

10

【 0 0 6 2 】

[0 0 6 8]

図 4 は、本開示の技術にしたがって、マルチパスレンダリングを実行するための技術を図示している概念のダイアグラムである。一般的に、これは、フラッシュを一度のみ実行することを可能にする。機能が図 4 で方向付けられているように、機能は、「上から下」および「左から右」へ実行することができる。より具体的には、図 4 で図示されているように、クエリパス 4 0 0 が実行された後に、条件真パス 4 0 4、条件偽パス 4 0 6 が続き、3つのパス、クエリパス 4 0 0、条件真 4 0 4、条件偽 4 0 6 がいったん完了すると、フラッシュ動作 4 0 8 が実行される。したがって、いくつかの例では、本開示の技術は、20
 タイルベースの G P U アーキテクチャを利用するアプリケーションの挙動を修正することにより、G P U 1 2 0 がクエリパスを完了した後に生じるフラッシュ動作 4 0 8 をなくすことができる。特に、いくつかの例では、本開示の技術は、新たな実行ポイント、バックドアコマンド、および/または拡張を含んでもよい。これらの新たな実行ポイント、バックドアコマンド、および/または拡張は、G P U 1 2 0 およびグラフィックドライバが上記で説明した第 2 のクエリ動作をなくすことができるようにする。いくつかの例では、本開示の技術は、G P U 1 2 0 がフラッシュコマンドをなくすことができるようにする、追加のレンダリングコマンドを含むように、D i r e c t X 1 1 グラフィック A P I を修正する。

20

【 0 0 6 3 】

[0 0 6 9]

マルチパススタイルベースのレンダリングを実行するとき、G P U 1 2 0 が一般的にクエリパス 4 0 0 とクエリチェック 4 0 2 を実行した後に、条件真パス 4 0 4 と条件偽パス 4 0 6 が続く。クエリチェック 4 0 2 は、アプリケーションによってトリガされる情報に対する何らかの要求であってもよい。例えば、アプリケーションは、A P I 側で何かをレンダリングし、レンダリングされたオブジェクトをグラフィックカードにフラッシュしてもよい。これは、特定のレンダリングを完了させる。その後、クエリが送られてもよい。クエリは、A P I からグラフィックカードにレンダリングがフラッシュされたときに渡されるピクセルの数であってもよい。クエリは、ハードウェアによってサポートされる、最後のレンダリングジョブが、または、現在のレンダリングジョブが実行される状態の情報について30
 の何らかの要求であることがある。いくつかのケースにおいて、アプリケーションは、何かをレンダリングし、クエリをトリガし、クエリ結果に基づいて、データのみを送ってもよい。

30

40

【 0 0 6 4 】

[0 0 7 0]

本願のいくつかの例にしたがうと、アプリケーションは、クエリと複数の I B 2 とを送ってもよい。上記で説明したように、I B 2 はレンダリングパイプラインのさまざまな態様に対するコマンドを含んでいる。

【 0 0 6 5 】

[0 0 7 1]

1つの例において、2つのクエリ結果は、「真」および「偽」である可能性があるかも

50

しれない。したがって、アプリケーションは2つのIB2を、「真」のクエリ結果に対する1つのIB2と、「偽」のクエリ結果に対する1つのIB2とを送ってもよい。したがって、「真」のクエリ結果に対するレンダリングパイプラインのさまざまな態様に対するコマンドを送ってもよく、「偽」のクエリ結果に対するレンダリングパイプラインのさまざまな態様に対するコマンドを送ってもよい。アプリケーションは両方の可能性を、「真」のクエリ結果に対するIB2と、「偽」のクエリ結果に対するIB2とを送る。

【0066】

[0072]

「真」のクエリ結果に対するIB2と、「偽」のクエリ結果に対するIB2との両方をアプリケーションは送ることから、アプリケーションは、クエリ結果に基づいて、データを送るのを待つ必要はない。むしろ、GPUは、アプリケーションによって送られた、「真」のクエリ結果に対するIB2と、「偽」のクエリ結果に対するIB2との両方を有しており、GPUがクエリの結果を待ってもよい。したがって、アプリケーションを待たせるよりもむしろ、GPUがクエリ結果を待ち、その後、GPUがクエリパスを実行してもよい。クエリパスの結果が「真」である場合、GPUは条件真パスを実行してもよい。さらに、フラッシュ動作408を実行することなく、条件真パスを実行してもよい。クエリパスの結果が「偽」である場合、GPUは条件偽パスを実行してもよい。フラッシュ動作408を実行することなく、条件偽パスも実行してもよい。クエリパスの結果に基づいて、条件真パスまたは条件偽パスのうちの1つを条件付きで実行することに対応して、その後、GPUはフラッシュ動作408を実行してもよい。

【0067】

[0073]

したがって、本開示のいくつかの態様では、クエリ結果を待つことは、アプリケーションからGPUに移る。言い換えると、アプリケーションよりもむしろ、GPUがクエリの結果を待ってもよい。いくつかの例では、条件「真」パスのケースに対するIB2と、条件「偽」パスのケースに対するIB2との両方をアプリケーションがGPUに送ることができることから、これは可能である。したがって、「真」のケースと「偽」のケースの両方に対して、アプリケーションがレンダリングするであろうものと、アプリケーションが実行したいものとをGPUは有している。なぜなら、IB2のそれぞれは、そのそれぞれのケースに対する、例えば、「真」クエリ結果と「偽」クエリ結果とのそれぞれに対する条件「真」のケースと条件「偽」のケースとに対するレンダリングパイプラインのさまざまな態様に対するコマンドを含んでいるからである。

【0068】

[0074]

クエリパス400は、三角形が見えるか否かを決定し、条件真パス404と条件偽パス406とに対する条件を確立する。いったんGPU120がクエリパス400を完了すると、GPU120は条件真パス404を実行し、その後、条件偽パス406を実行する。条件真パスおよび条件偽パスは、異なるデータと、アプリケーションレンダリングシーケンスに基づいて、それらの中で蓄積されたレンダリングコマンドとを有している。

【0069】

[0075]

本開示の技術は、GPU120が実行するレンダリングパスの開始と終了を特定するレンダリング命令を含んでもよい。したがって、本開示のいくつかの例示的な技術は、例えば、クエリパス400、条件真パス404、および条件偽パス406の開始および終了を特定するレンダリング命令を含んでいる。特に、本開示のいくつかの例示的な技術は、開始__クエリ__パスコマンド、終了__クエリ__パスコマンド、開始__条件__真__パス、終了__条件__真__パス、および終了__条件__偽__パスコマンドのような、エントリポイントを含んでいる。これらは、「エクスポーズアウト」されて、異なるコードのサブルーチンにアクセスできるようにするエントリポイントである。再度説明すると、「エクスポーズアウト」は、必要とされるときに、コードブロックをトリガすることができる機能的ボ

10

20

30

40

50

インタを提供する、コードのブロックへのエントリポイントを指す。ここで説明するように、これらはアプリケーションにエクスポートアウトされるエントリポイントであり、したがって、アプリケーションによりドライバに実行依頼されるものの表示をアプリケーションがドライバに提供できる。

【 0 0 7 0 】

[0 0 7 6]

例では、対応するコマンドの各ペア間で、例えば、クエリパス開始とクエリパス終了、条件真開始と条件真終了、条件偽開始と条件偽終了間で、グラフィックドライバまたはアプリケーションは、GPU 120がそのレンダリングパスの間に実行するレンダリングコマンドを特定する。いったんパスのすべてが完了すると、GPU 120はフラッシュコマンドを実行する。フラッシュコマンドは3つのパス、クエリパス400、条件真404、および条件偽406の結果をシステムメモリ118に書き込んでよい。

10

【 0 0 7 1 】

[0 0 7 7]

ここで説明するように、クエリパス400はピニングパスの前に実行してもよい。ピニングパスはレンダリングパス、例えば、条件真パス404に対するレンダリング、または、条件偽パス406に対するレンダリングの前に実行してもよい(ピニングパスは図4中に図示されていない。)。いくつかのケースでは、ピニングパスをスキップしてもよく、したがって、レンダリングパス404または406は、クエリパス400の直後に実行される。いくつかのケースでは、ピニングパスもレンダリングパスも、例えば、条件真パス404に対するレンダリングも条件偽パス406に対するレンダリングも、クエリパス400の後に実行されない。例えば、常にこのケースが起こるわけではないが、条件偽パスはピニングパスおよびレンダリングパスを実行させなくてもよい。しかしながら、いくつかの例では、ピニングパスおよびレンダリングパスは、条件偽パスに対して実行される。ピニングパスおよびレンダリングパスは、条件真パスに対して実行されてもよい。条件真パスと条件偽パスとの間の主な差は、異なるデータと、アプリケーションレンダリングシーケンスに基づいて、それらの中で蓄積された、レンダリングコマンドとを有することである。

20

【 0 0 7 2 】

[0 0 7 8]

クエリパス400は、ピニングパスを実行すべきであるか否かを示すデータを戻してもよい。例において、条件付き実行決定ユニットは、ピニングパスまたはレンダリングパスを、例えば条件真パス404に対するレンダリング、または、条件偽パス406に対するレンダリングを実行すべきか否かを決定してもよい。ピニングパスを実行する際の決定は、レンダリングされることになるオブジェクトの複雑性に基づいていてもよい。例えば、単純なオブジェクトに対して、ピニングパスをスキップしてもよい。逆に、より複雑なオブジェクトに対して、ピニングパスを実行してもよい。ここで説明したように、条件真レンダリングと条件偽レンダリングとを生じさせてもよい。クエリパスと条件真レンダリングと条件偽は、フラッシュ408の前に生じてもよい。

30

【 0 0 7 3 】

[0 0 7 9]

ここで説明するように、いくつかの例は、すべてのデータがキューされるまでフラッシュしない。例えば、クエリパス400、レンダリングパス404、およびレンダリングパス406は、クエリパス400、レンダリングパス404、およびレンダリングパス406のそれぞれの後にフラッシュを実行するよりもむしろ、フラッシュ408のために、それぞれがキューされてもよい。したがって、単一のフラッシュ408を実行することができる。これは、条件付きであり、クエリに基づいていてもよい。ここで説明するように、いくつかの例では、GPU 120は、クエリパス400、レンダリングパス404、およびレンダリングパス406とともに、フラッシュ408を完了する。フラッシュ408は、オペレーティングシステム114による使用のために、データをプロセッサ102に送

40

50

ってもよい。データは、蓄積されたレンダリングコマンドからのデータであってもよい。ここで説明するように、いくつかの例では、フラッシュは、すべての蓄積されたレンダリングコマンドをオペレーティングシステム 1 1 4 に実行依頼するまたは送ることである。グラフィックアプリケーションがレンダリングコマンドをトリガするとき、グラフィックドライバは、蓄積されたレンダリングコマンドをハードウェアに直接送らない。レンダリングコマンドは、グラフィックドライバによって蓄積される（必要に応じて変換される）。

【 0 0 7 4 】

[0 0 8 0]

さらに、いくつかの例では、クエリ値が何であるかは問題ではない。したがって、例えば、メモリロケーションが上書きされるか否かは、問題にならないかもしれないことから、ロックされたメモリロケーションに値を書き込むことができないように、メモリロケーションをロックする必要はないかもしれない。コールバックを得ること等は必要ないかもしれない。いくつかの例では、メモリドライバは、可視ストリームに寄与しない、事前ビニングパス/クエリパスを実行してもよい。いくつかの例では、メモリドライバは、ビニングパスを実行してもよい。ビニングパスは、条件付きで実行される。クエリ結果は、クエリチェック 4 0 2 から戻されてもよい。クエリチェック 4 0 2 は、「真」または「偽」の値に戻してもよい。クエリの真または偽の結果に基づいて、シーンをレンダリングしてもよい。言い換えると、ビニングパスは、真または偽のクエリ結果に基づいて、条件付きで実行される。真のクエリ結果は、結果としてレンダリングパス 4 0 4 になるかもしれない一方で、偽のクエリ結果は、結果としてレンダリングパス 4 0 6 になるかもしれない。

【 0 0 7 5 】

[0 0 8 1]

ビニングパスによって戻される条件または値が真であるとき、条件真レンダリング 4 0 4 は、可視ストリームに寄与するかもしれない。代替的に、ビニングパスによって戻される条件または値が偽である場合、条件偽レンダリング 4 0 6 は、レンダリング I B 2 として可視ストリームに寄与する。レンダリングパス 4 0 4 および 4 0 6 は、条件付きで実行される。補正レンダリングパスに対して、ビニングパスを一度トリガしてもよい。その後、補正可視ストリームおよび最適化を実行してもよい。クエリパス上ではなく、補正ジオメトリ上のみで、レンダリングパスを実行してもよい。単一のフラッシュポイントで、動作を完了してもよい。

【 0 0 7 6 】

[0 0 8 2]

例えば、いくつかのデバイスは、タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングのための方法を実行してもよい。このようなデバイスは、クエリチェック 4 0 2 と、フラッシュ動作 4 0 8 を実行することなく、クエリパスに基づく条件真パス 4 0 4 と、フラッシュ動作 4 0 8 を実行することなく、クエリパスに基づく条件偽パス 4 0 6 とを、実行する G P U を含んでもよい。一般的に、条件真パス 4 0 4 または条件偽パス 4 0 6 のいずれかが、所定のクエリチェック 4 0 2 の結果に基づいて実行される。条件真パスおよび条件偽パスを実行することに対応して、G P U はフラッシュ動作 4 0 8 を実行してもよい。いくつかの例では、条件真または条件偽のいずれかが、結果として、可視ストリームを発生させるビニングパスとなってもよい。代替的に、これらのいずれか、条件真または条件偽は、シーンをレンダリングする際に、直接レンダリングパスを使用してもよい。

【 0 0 7 7 】

[0 0 8 3]

いくつかの例では、クエリパス、例えばクエリチェック 4 0 2 は、第 1 のクエリパスを含んでもよい。第 1 のクエリパスを実行することは、第 1 のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含んでもよい。さらに、いくつかの例では、第 1 のクエリパスを実行することは、第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することを含んでもよい。

10

20

30

40

50

ティックレンダリングコマンドを実行することをさらに含んでいる。いくつかの例では、条件偽パスを実行することは、条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含んでいる。いくつかの例では、開始条件パスを実行することは、開始条件パスの終了を示すグラフィックレンダリングコマンドをさらに含んでいる。いくつかの例では、条件真パスを実行することは、第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含んでいる。いくつかの例では、条件偽パスを実行することは、条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含んでいる。

【0078】

[0084]

図5は、本開示で説明する1つ以上の例にしたがって、ハードウェアによって実行できる機能の例を図示している概念のダイアグラムである。図5に図示している例において、クエリパス500は、ハードウェア中で実行してもよい。ピニングパス502も、ハードウェア中で実行してもよい。さらに、レンダリングパス504を、ハードウェア中で実行してもよい。いくつかの例では、ハードウェアは、GPU120または他の処理ハードウェアを含んでいてもよい。クエリパス500、ピニングパス502、およびレンダリングパス504は、オペレーティングシステム506によって制御してもよい。オペレーティングシステム506は、クエリパス500を開始してもよい。いくつかの例では、レンダリングパスは、クエリ結果の結果に基づく、条件真パスと条件偽パスのうちの1つの実行を含んでいる。結果は、第2のパスのピニングとレンダリングの両方を制御してもよい。

【0079】

[0085]

予め定められたメモリまたは予め定められたレジスタにクエリ結果を戻すクエリブロック508において、クエリパス500を実行してもよい。オペレーティングシステム506は、予め定められたメモリまたは予め定められたレジスタ中にクエリ結果を記憶させてもよい。さらに、予め定められたメモリまたは予め定められたレジスタ中に記憶されてもよいクエリ結果は、ピニングパス502、レンダリングパス504、または両方によって使用してもよい。例えば、シーンおよび関係付けられているオブジェクトをマルチパスレンダリングすることに関連して、クエリ結果を使用してもよい。マルチパスレンダリングでは、シーンおよび関係付けられているオブジェクトは、複数回レンダリングしてもよい。オブジェクトが描かれる毎に、オブジェクトの外観の追加の態様を計算して、以前の結果と組み合わせてもよい。一般的に、これは、粗い最初のレンダリングと、第1の粗いパスのクエリ結果に基づく、詳細な第2のレンダリングパスとを伴ってもよい。クエリチェックの間、クエリ結果をチェックしてもよく、クエリ結果は、結果として、条件真のクエリ結果または条件偽のクエリ結果となる。上記で説明したように、クエリは、アプリケーションによってトリガされる情報に対する何らかの要求であってもよい。クエリ結果は、その後、条件真キューを実行することにより、結果として条件真グラフィックレンダリング404となってもよく、または、クエリ結果はその後、条件偽キューを実行することにより、結果として、条件偽グラフィックレンダリング406となってもよい。

【0080】

[0086]

ピニングパス502は条件付きである。ピニングパス502の間、GPUは、シーンを構成する多角形（例えば、三角形）を発生させ、多角形を複数の「ピン」にソートしてもよい。ここで説明するように、ピニングパス502の間に規定されたピンは、ディスプレイ（例えば、「スクリーンタイル」として呼ばれることがある）上に提示される最終のシーンのタイルに直接関連していてもよい。例えば、各ピンは、最終のシーンの一部分またはタイル（例えば、ビデオデータ、コンピュータ発生グラフィック画像、静止画像、またはこれらに類するもののフレームの予め定められた一部分）を表す。したがって、用語「ピン」および「タイル」はここでは交換可能に使用されるかもしれない。

【0081】

10

20

30

40

50

[0 0 8 7]

いくつかの例では、ビニングパス502の間、GPUは、多角形が別の多角形をおおい隠すか否かを決定するために深度テストを実行するような動作も実行して、多角形のどれがシーン中で見えるかを決定する。シーン中でどの多角形が見えるかを決定すると、GPUは「可視ストリーム」として呼ばれるデータのストリームを発生させることができる。可視ストリームは、シーンの多角形のそれぞれに対する値を含んでいてもよく、値は、多角形が見えるか否かを表してもよい(例えば、「1」の値は多角形が見えることを示し、「0」の値は多角形が見えないことを示してもよい)。

【 0 0 8 2 】

[0 0 8 8]

レンダリングパス504も条件付きである。レンダリングパス504の間、規定されたタイルのそれぞれがレンダリングされる。いくつかの例では、レンダリングパスのそれぞれは、3つのステージ、(1)クリア/アンリゾルブ、(2)レンダリング、(3)リゾルブ、で完了してもよい。

【 0 0 8 3 】

[0 0 8 9]

いくつかの例では、メモリドライバは、可視ストリームに寄与しない事前ビニングパスを実行してもよい。いくつかの例では、メモリドライバは、ビニングパス502を実行してもよい。ビニングパス502は条件付きで実行される。例えば、直接レンダリングに対して、ビニングパス502をスキップしてもよい。いくつかの例では、ビニングパス502が条件付きで実行されるとき、ビニングパス502が可視ストリームに寄与するか否かを示す値をビニングパス502が戻してもよい。代替的に、ビニングパス502によって戻された条件または値が偽である場合、ビニングパス502は、レンダリングIB2として、可視ストリームに寄与する。レンダリングパス504も、条件付きで実行される。条件が真のとき、レンダリングパス504は可視ストリームに寄与する。レンダリングパス504が条件付きで実行されるとき、レンダリングパス504は「真」または「偽」の値も戻してもよい。条件が偽のとき、レンダリングパス504は、レンダリングIB2として、可視ストリームに寄与する。補正レンダリングパス504に対して、ビニングパス502を一度トリガしてもよい。その後、補正可視ストリームおよび最適化を実行してもよい。クエリパス500上ではなく、補正ジオメトリ上のみで、レンダリングパス504を実行してもよい。単一のフラッシュポイントで、動作を完了してもよい。

【 0 0 8 4 】

[0 0 9 0]

1つの例では、間接バッファ1(IB1)はクエリパス500、ビニングパス502、またはレンダリングパス504を呼び出してもよい。クエリパス500、ビニングパス502、およびレンダリングパス504は、間接バッファ2(IB2)の一部であってもよい。例えば、クエリパス500、ビニングパス502、およびレンダリングパス504は、それぞれ別々のIB2、すなわち、IB2レベルバッファ内の別々のエンティティであってもよい。クエリパス500は、ビニングパス502の前に実行してもよい。ビニングパス502は、レンダリング504の前に実行してもよい。いくつかのケースでは、ビニングパス502をスキップしてもよく、したがって、レンダリング504をクエリパス500の直後に実行する。いくつかのケースでは、ビニングパス502もレンダリング504もクエリパス500の後に実行されない。

【 0 0 8 5 】

[0 0 9 1]

例えば、クエリパス500は、データを、例えば、ビニングパス502を実行すべきか否かを示す、カウンタ値またはヒューリスティクスを戻してもよい。例において、条件付き実行決定ユニットは、ビニングパス502またはレンダリングパス504を実行すべきか否かを決定してもよい。ビニングパス502を実行する際の決定は、レンダリングされることになるオブジェクトの複雑性に基づいていてもよい。例えば、単純なオブジェクト

10

20

30

40

50

に対しては、ピニングパス502をスキップしてもよい。逆に、より複雑なオブジェクトに対しては、ピニングパス502を実行してもよい。したがって、より単純なオブジェクトに対しては、ピニングパス502をスキップして、クエリパス500の直後にレンダリングパス504を実行してもよい。さらに、単純なオブジェクトに対しては、単一の繰り返しにおいて、レンダリングパス504を実行してもよい。例えば、スクリーン全体が一連のブロック中よりもむしろ単一のパス中で書かれてもよい。これは、例えば、とても単純な、反復スクリーンレンダリングに対して可能であるかもしれない。

【0086】

[0092]

ここで説明するように、いくつかの例は、すべてのデータが送られるまでフラッシュしない。さらに、いくつかの例では、クエリ値が何であるかは問題ではない。したがって、例えば、メモリロケーションが上書きされるか否かは、問題にはならないかもしれないことから、メモリロケーションをロックする必要はないかもしれない。コールバックを得ること等は必要ないかもしれない。いくつかの例では、メモリドライバは、可視ストリームに寄与しない、事前ピニングパス/クエリパスを実行してもよい。いくつかの例では、メモリドライバは、ピニングパス502を実行してもよい。ピニングパス502は条件付きで実行されてもよい。いくつかの例では、ピニングパス502が条件付きで実行される時、可視ストリームに寄与することを示す値を戻してもよい。代替的に、ピニングパス502によって戻される条件または値が偽である場合、ピニングパス502は、レンダリングIB2として可視ストリームに寄与する。レンダリングパス504も条件付きで実行される。条件が真であるとき、レンダリングパス504は可視ストリームに寄与する。レンダリングパス504が条件付きで実行される時、レンダリングパス504は、「真」または「偽」の値も戻してもよい。条件が偽であるとき、レンダリングパス504は、レンダリングIB2として可視ストリームに寄与する。補正レンダリングパスに対して、ピニングパスを一度トリガしてもよい。その後、補正可視ストリームおよび最適化を実行してもよい。クエリパス500上ではなく、補正ジオメトリ上のみで、レンダリングパス504を実行してもよい。単一のフラッシュポイントで、動作を完了してもよい。

【0087】

[0093]

例えば、いくつかのデバイスは、タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするための方法を実行してもよい。このようなデバイスは、クエリパスを実行し、フラッシュ動作を実行することなく、クエリパス500に基づいて条件真パスを実行し、フラッシュ動作を実行することなく、クエリパス500に基づいて条件偽パスを実行するGPUを含んでいてもよく、条件真パスおよび条件偽パスを実行することに対応して、GPUはフラッシュ動作を実行してもよい。

【0088】

[0094]

いくつかの例では、クエリパス500は、第1のクエリパスを含んでいてもよい。第1のクエリパスを実行することは、第1のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含んでいてもよい。さらに、いくつかの例では、第1のクエリパスを実行することは、第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含んでいる。いくつかの例では、条件偽パスを実行することは、条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含んでいる。いくつかの例では、開始条件パスを実行することは、開始条件パスの終了を示すグラフィックレンダリングコマンドをさらに含んでいる。いくつかの例では、条件真パスを実行することは、第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含んでいる。いくつかの例では、条件偽パスを実行することは、条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含んでいる。

【0089】

[0095]

10

20

30

40

50

図6は、本開示で説明した1つ以上の例にしたがうタイルベースアーキテクチャ上で、マルチパスグラフィックレンダリングをするための例示的な方法を図示しているフローチャートである。GPU120は、クエリパスを発生させる(600)。クエリパス500は、第1のクエリパスをさらに含んでいてもよい。さらに、第1のクエリパスを実行することは、第1のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含んでいてもよい。いくつかの例では、第1のクエリパスを実行することは、第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含んでいる。

【0090】

[0096]

GPU120は、フラッシュ動作を実行することなく、クエリパス500に基づいて、条件真パスを発生させる(602)。いくつかの例では、開始条件パスを実行することは、開始条件パスの終了を示すグラフィックレンダリングコマンドをさらに含んでいる。条件真パスを実行することは、第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することもさらに含んでいる。

【0091】

[0097]

GPU120は、フラッシュ動作を実行することなく、クエリパス500に基づいて、条件偽パスを発生させる(604)。いくつかの例では、条件偽パスを実行することは、条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含んでいる。いくつかの例では、条件偽パスを実行することは、条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含んでいる。

【0092】

[0098]

GPU120は、条件真パスおよび条件偽パスを実行することに対応して、フラッシュ動作を実行する(606)。3つのパス、クエリパス400と、条件真404と、条件偽406とがいったん完了すると、フラッシュ動作を実行してもよい。一般的にこれは、単一のフラッシュ動作の実行を可能にする。フラッシュコマンド408は、3つのパス、クエリパス400と、条件真404と、条件偽406との結果を、システムメモリ118に書き出してもよい。

【0093】

[0099]

図7は、本開示の1つ以上の態様を実現するように構成されているかもしれない、デバイスの例を図示しているブロックダイヤグラムである。例えば、図7は、デバイス702を図示している。デバイス702の例は、ビデオデバイス、メディアプレーヤー、セットトップボックス、移動体電話機およびいわゆるスマートフォンのようなワイヤレスハンドセット、パーソナルデジタルアシスタント(PDA)、デスクトップコンピュータ、ラップトップコンピュータ、ゲームコンソール、テレビ会議ユニット、タブレットコンピューティングデバイス、およびこれらに類するものを含むが、これらに限定されない。

【0094】

[0100]

図7の例において、デバイス702は、プロセッサ102、システムメモリ118、およびGPU120を含んでいてもよい。これらのコンポーネントは、図1に関して以前に説明しているので、簡潔さの目的で、プロセッサ102、システムメモリ118、およびGPU120は、図7に関してさらには記述しない。デバイス702は、ディスプレイプロセッサ724、トランシーバモジュール726、ユーザインターフェース728、およびディスプレイ730も含んでいてもよい。トランシーバモジュール726およびディスプレイプロセッサ724は、両方とも、プロセッサ102および/またはGPU120と同じ集積回路(IC)の一部であってもよい。別の例では、トランシーバモジュール726およびディスプレイプロセッサ724は、両方とも、プロセッサ120および/または

10

20

30

40

50

GPU120を含んでいるICの外部にあってもよい。さらに別の例では、トランシーバモジュール726およびディスプレイプロセッサ724は、プロセッサ120および/またはGPU120を含んでいるICの外部にあるIC中に形成してもよい。

【0095】

[0101]

デバイス702は、明確さの目的で、図7中に示されていない追加のモジュールまたはユニットを含んでいてもよい。例えば、デバイス702は、いずれも図7中に示されていない、スピーカーおよびマイクロフォンを含んでいてもよい。デバイス702が移動体ワイヤレス電話機である例では、電話通信を実現するために、スピーカーおよびマイクロフォンを使用してもよい。デバイス702がメディアプレーヤーであるとき、デバイス702は、サウンド出力を提供するためのスピーカーを含んでいてもよく、または、デバイス702は、出力ジャックを含んでいてもよい。デバイス702は、ビデオカメラも含んでいてもよい。さらに、デバイス702中に示されているさまざまなモジュールおよびユニットは、デバイス702のすべての例において必要ではないかもしれない。例えば、デバイス702がデスクトップコンピュータである、あるいは、外部のユーザインターフェースまたはディスプレイとインターフェースするように装備されている他のデバイスである例において、ユーザインターフェース728およびディスプレイ730は、デバイス702の外部にあってもよい。

【0096】

[0102]

ユーザインターフェース728の例は、タッチスクリーン、トラックボール、マウス、キーボード、および他のタイプの入力デバイスを含むが、これらに限定されない。ユーザインターフェース728はまた、タッチスクリーンであってもよく、ディスプレイ730の一部として組み込まれていてもよい。トランシーバモジュール726は、デバイス702と他のデバイスまたはネットワークと間のワイヤレスまたはワイヤード通信を可能にする回路を含んでいてもよい。トランシーバモジュール726は、変調器、復調器、増幅器、およびワイヤードまたはワイヤレス通信のための他のこのような回路を含んでいてもよい。

【0097】

[0103]

いくつかの例において、GPU120は、システムメモリ118中に、完全に形成された画像を記憶させてもよい。ディスプレイプロセッサ724は、システムメモリ118から画像を取り出して、ディスプレイ730のピクセルを照光させて、画像を表示させる値を出力してもよい。ディスプレイ730は、GPU120によって発生させた画像コンテンツを表示する、デバイス702のディスプレイであってもよい。ディスプレイ730は、液晶ディスプレイ(LCD)、有機発光ダイオードディスプレイ(OLED)、ブラウン管(CRT)ディスプレイ、プラズマディスプレイ、または他のタイプのディスプレイデバイスであってもよい。

【0098】

[0104]

1つ以上の例において、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、またはこれらの任意の組み合わせで実現してもよい。ソフトウェアで実現する場合、機能は、1つ以上の命令またはコードとしてコンピュータ読取可能媒体上に記憶されていてもよく、あるいは、1つ以上の命令またはコードとしてコンピュータ読取可能媒体上で送信されてもよい。コンピュータ読取可能媒体は、1つの場所から別の場所へのコンピュータプログラムの転送を促進する任意の媒体を含む、コンピュータデータ記憶媒体または通信媒体を含んでいてもよい。データ記憶媒体は、本開示で説明した技術を実現するための命令、コード、および/またはデータ構造を取り出すために、1つ以上のコンピュータまたは1つ以上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、このようなコンピュータ読取可能媒体は、RA

10

20

30

40

50

M、ROM、EEPROM（登録商標）、CD-ROMまたは他の光ディスク記憶装置、磁気ディスク記憶装置または他の磁気記憶デバイスを備えることができる。ここで使用するような、ディスク（disk）およびディスク（disc）は、コンパクトディスク（CD）、レーザーディスク（登録商標）、光ディスク、デジタル汎用ディスク（DVD）、フロッピー（登録商標）ディスク、およびブルーレイ（登録商標）ディスクを含み、ここでディスク（disk）は通常、磁氣的にデータを再生し、ディスク（disc）は、レーザーを用いて光学的にデータを再生する。上記の組み合わせもまた、コンピュータ読取可能媒体の範囲内に含まれるべきである。

【0099】

[0105]

コードは、1つ以上のデジタル信号プロセッサ（DSP）、汎用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論理アレイ（FPGA）、あるいは他の同等な集積またはディスクリートの論理回路のような、1つ以上のプロセッサによって実行してもよい。したがって、ここで使用されるように、用語「プロセッサ」は、前述の構造、またはここで説明した技術のインプリメンテーションに適した他の何らかの構造のいずれかを指してもよい。加えて、いくつかの態様では、ここで説明した機能性は、専用のハードウェア、ソフトウェアモジュール、または、エンコードおよびデコードするように構成されたこれらの組み合わせ内に提供することができ、あるいは、組み合わせられたコーデック中に組み込むことができる。さらに、技術は、1つ以上の回路または論理エレメントにおいて十分に実現することができる。

【0100】

[0106]

本開示の技術は、ワイヤレスハンドセット、集積回路（IC）またはICのセット（すなわち、チップセット）を含む、幅広いさまざまなデバイスまたは装置で実現してもよい。さまざまなコンポーネント、モジュール、またはユニットは、開示された技術を実行するように構成されているデバイスの機能的な態様を強調するためにここ説明されているが、それらは、異なるハードウェアユニットによる実現を必ずしも要求するわけではない。むしろ、上記で説明したように、さまざまなユニットは、コーデックハードウェアユニットにおいて組み合わせられるか、あるいは、適切なソフトウェア、ファームウェア、または両方とともに、上記で説明したような1つ以上のプロセッサを含む、相互動作可能ハードウェアユニットの集合によって提供してもよい。

【0101】

[0107]

さまざまな例を説明してきた。これらおよび他の例は、以下の特許請求の範囲の範囲内にある。

以下に、本願出願の当初の特許請求の範囲に記載された発明を付記する。

[1] タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするための方法において、

グラフィック処理ユニット（GPU）によってクエリパスを実行することと、

フラッシュ動作を実行することなく、前記GPUによって前記クエリパスに基づいて条件真パスを実行することと、

フラッシュ動作を実行することなく、前記GPUによって前記クエリパスに基づいて条件偽パスを実行することと、

前記条件真パスおよび前記条件偽パスを実行することに対応して、前記GPUによってフラッシュ動作を実行することを含む方法。

[2] 前記クエリパスは、第1のクエリパスを含み、

前記第1のクエリパスを実行することは、前記第1のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含む[1]記載の方法。

[3] 前記第1のクエリパスを実行することは、前記第1のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む[2]記載の方法。

10

20

30

40

50

[4] 前記条件偽パスを実行することは、前記条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含む [1] 記載の方法。

[5] 直接レンダリングを実行することをさらに含む [1] 記載の方法。

[6] 前記条件真パスを実行することは、第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む [5] 記載の方法。

[7] 前記条件偽パスを実行することは、前記条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含む [1] 記載の方法。

[8] タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするデバイスにおいて、

クエリパスを実行するようにと、

フラッシュ動作を実行することなく、前記クエリパスに基づいて条件真パスを実行するようにと、

フラッシュ動作を実行することなく、前記クエリパスに基づいて条件偽パスを実行するようにと、

前記条件真パスおよび前記条件偽パスを実行することに対応して、フラッシュ動作を実行するように構成されている GPU を具備するデバイス。

[9] 前記クエリパスは、第 1 のクエリパスを含み、

前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含む [8] 記載のデバイス。

[10] 前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む [9] 記載のデバイス

。
[11] 前記条件偽パスを実行することは、前記条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含む [8] 記載のデバイス。

[12] 直接レンダリングを実行することをさらに含む [8] 記載のデバイス。

[13] 前記条件真パスを実行することは、第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む [12] 記載のデバイス。

[14] 前記条件偽パスを実行することは、前記条件偽パスの開始を示すグラフィックコマンドを実行することをさらに含む [8] 記載のデバイス。

[15] タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするデバイスにおいて、

グラフィック処理ユニット (GPU) によってクエリパスを実行する手段と、

フラッシュ動作を実行することなく、前記 GPU によって前記クエリパスに基づいて条件真パスを実行する手段と、

フラッシュ動作を実行することなく、前記 GPU によって前記クエリパスに基づいて条件偽パスを実行する手段と、

前記条件真パスおよび前記条件偽パスを実行することに対応して、前記 GPU によってフラッシュ動作を実行する手段とを具備するデバイス。

[16] 前記クエリパスは、第 1 のクエリパスを含み、

前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの開始を示すグラフィックレンダリングコマンドを実行することを含む [15] 記載のデバイス。

[17] 前記第 1 のクエリパスを実行することは、前記第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行する手段をさらに備える [16] 記載のデバイス。

[18] 前記条件偽パスを実行することは、前記条件偽パスの終了を示すグラフィックコマンドを実行することをさらに含む [15] 記載のデバイス。

[19] 直接レンダリングを実行することをさらに含む [15] 記載のデバイス。

[20] 前記条件真パスを実行することは、第 1 のクエリパスの終了を示すグラフィックレンダリングコマンドを実行することをさらに含む [19] 記載のデバイス。

[21] 前記条件偽パスを実行することは、前記条件偽パスの開始を示すグラフィッ

10

20

30

40

50

クコマンドを実行することをさらに含む [1 5] 記載のデバイス。

[2 2] タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするための命令をその上に記憶したコンピュータ読取可能記憶媒体において、

前記命令は、実行される際に、1つ以上のグラフィック処理ユニット (G P U) に、クエリパスを実行させ、

フラッシュ動作を実行させることなく、前記クエリパスに基づいて条件真パスを実行させ、

フラッシュ動作を実行させることなく、前記クエリパスに基づいて条件偽パスを実行させ、

前記条件真パスおよび前記条件偽パスを実行させることに応答して、フラッシュ動作を実行させるコンピュータ読取可能記憶媒体。

10

【要約】

本開示は、グラフィック処理ユニット (G P U) によってクエリパスを実行することと、フラッシュ動作を実行することなく、G P U によってクエリパスに基づいて条件真パスを実行することと、フラッシュ動作を実行することなく、G P U によってクエリパスに基づいて条件偽パスを実行することと、条件真パスおよび条件偽パスを実行することに応答して、G P U によってフラッシュ動作を実行することを含む、タイルベースのアーキテクチャ上でマルチパスグラフィックレンダリングをするためのシステムおよび方法を提供する。

【選択図】 図 6

20

【図 1】

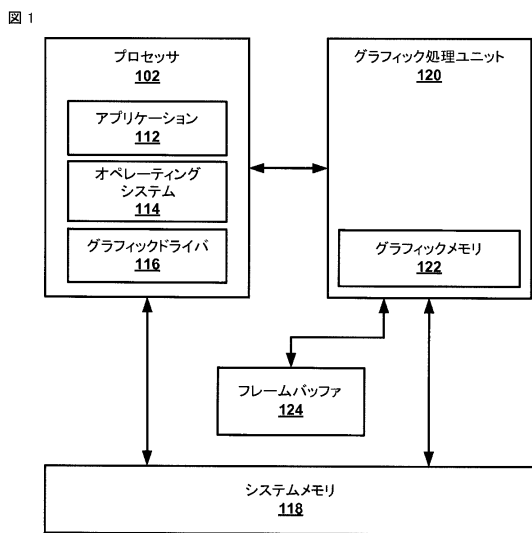


FIG. 1

【図 2】

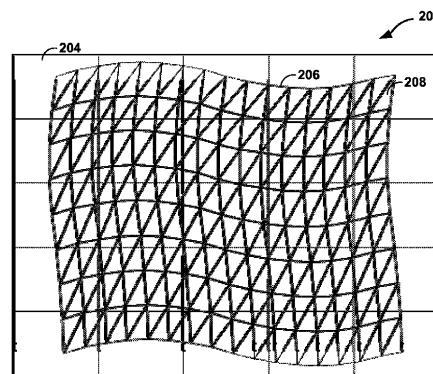


FIG. 2

【図3】

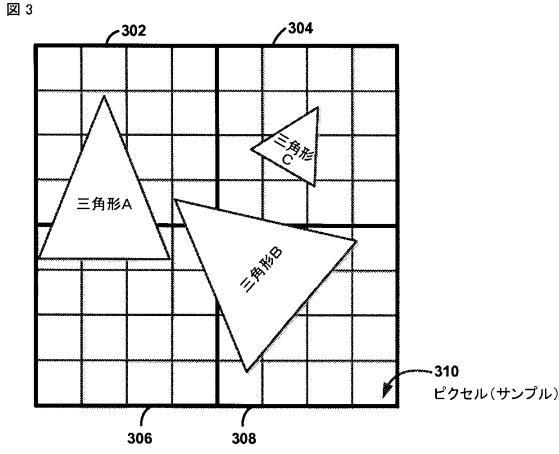


FIG. 3

【図4】

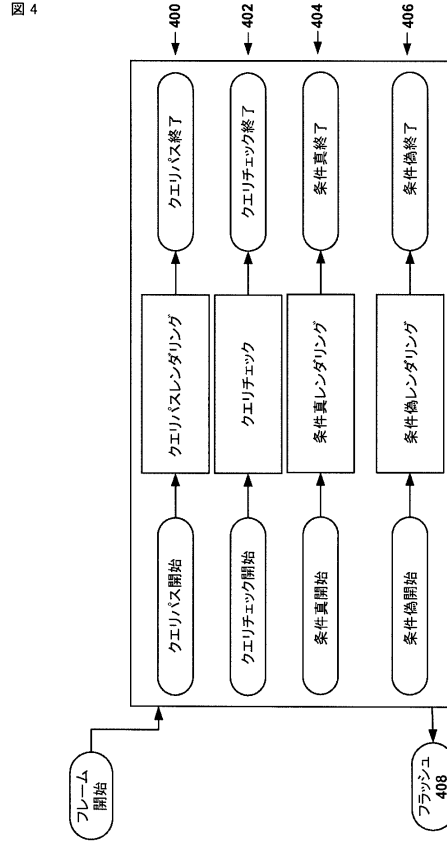


FIG. 4

【図5】

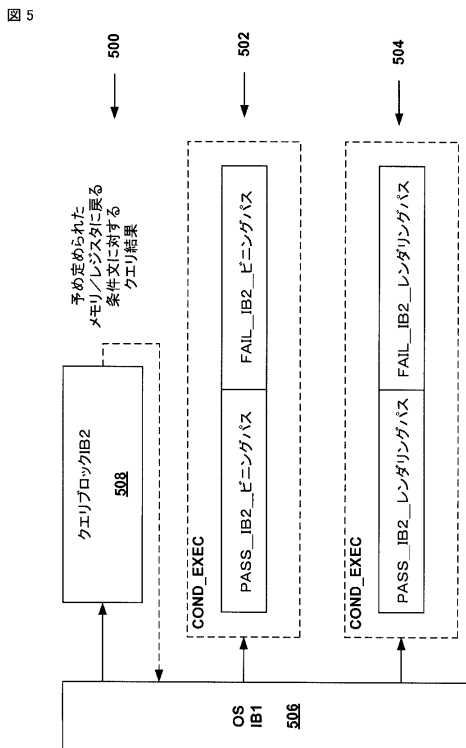


FIG. 5

【図6】

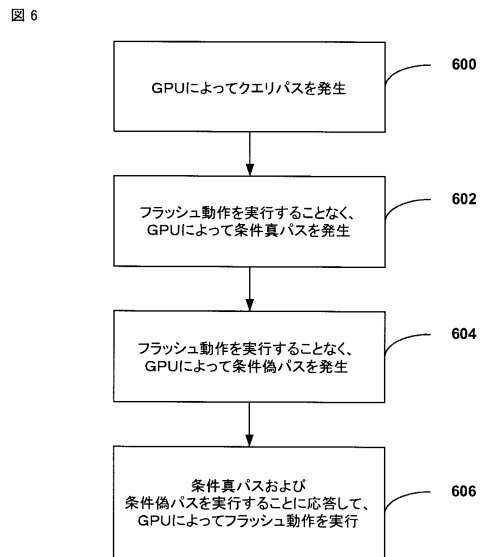


FIG. 6

【 図 7 】

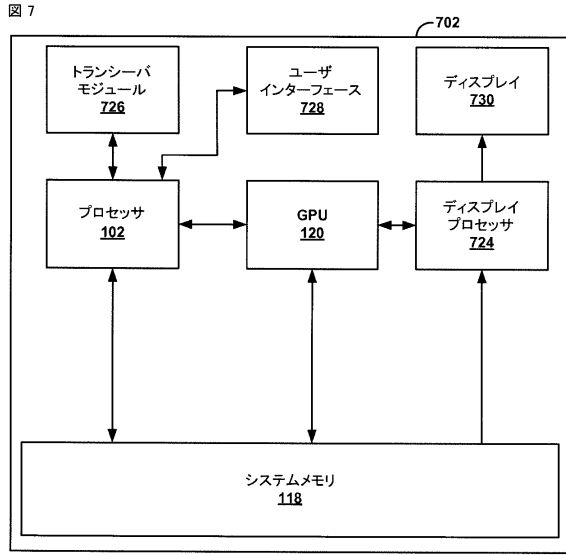


FIG. 7

フロントページの続き

- (72)発明者 バルシー、ムラート
アメリカ合衆国、カリフォルニア州 9 2 1 2 1 - 1 7 1 4、サン・ディエゴ、モアハウス・ドライブ 5 7 7 5
- (72)発明者 フラスカティ、クリストファー・ポール
アメリカ合衆国、カリフォルニア州 9 2 1 2 1 - 1 7 1 4、サン・ディエゴ、モアハウス・ドライブ 5 7 7 5
- (72)発明者 セーサラマイアー、アピナシュ
アメリカ合衆国、カリフォルニア州 9 2 1 2 1 - 1 7 1 4、サン・ディエゴ、モアハウス・ドライブ 5 7 7 5

審査官 真木 健彦

- (56)参考文献 国際公開第2013/081788(WO, A1)
国際公開第2013/081789(WO, A1)
国際公開第2013/151748(WO, A2)
国際公開第2013/151750(WO, A2)
国際公開第2013/151751(WO, A2)
特開2009-295166(JP, A)
米国特許第5778432(US, A)
米国特許出願公開第2005/0225670(US, A1)
米国特許出願公開第2011/0320771(US, A1)

(58)調査した分野(Int.Cl., DB名)

G 0 6 T 1 5 / 0 0 - 1 5 / 8 7
G 0 6 T 1 7 / 0 0 , 1 7 / 1 0 - 1 7 / 3 0
G 0 6 T 1 / 2 0