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

特許第6310151号
(P6310151)

(45)発行日　平成30年4月11日(2018.4.11)　(24)登録日　平成30年3月23日(2018.3.23)

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

請求項の数 30 (全 35 頁)

(21)出願番号　特願2017-508972(P2017-508972)
(86)(22)出願日　平成27年8月3日(2015.8.3)
(65)公表番号　特表2017-530444(P2017-530444A)
(43)公表日　平成29年10月12日(2017.10.12)
(86)国際出願番号　PCT/US2015/043466
(87)国際公開番号　WO2016/028482
(87)国際公開日　平成28年2月25日(2016.2.25)
審査請求日　平成29年12月18日(2017.12.18)
(31)優先権主張番号　14/465,371
(32)優先日　平成26年8月21日(2014.8.21)
(33)優先権主張国　米国(US)

早期審査対象出願

(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　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

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リング対象コマンドの並べ替え

(57)【特許請求の範囲】
【請求項１】
グラフィックスデータをレンダリングするための方法であって、
画像をレンダリングするための、複数のレンダリング対象と関連付けられる複数のコマンドを受信するステップであって、前記複数のコマンドのうちのコマンドが初期の順序で受信され、前記複数のレンダリング対象のうちの各レンダリング対象が、それぞれのバッファであり、前記画像が、前記複数のレンダリング対象からのコンテンツで構成される、ステップと、
前記初期の順序と異なる順序である、前記コマンドの実行順序を決定するステップであって、前記実行順序を決定するステップが、前記初期の順序の前記コマンドのレンダリング対象間のデータ依存関係に基づいて、前記複数のレンダリング対象の各レンダリング対象のためのそれぞれのレンダリング対象ごとのコマンドリストを生成するステップを備える、ステップと、
前記レンダリング対象ごとのコマンドリストを前記決定された実行順序でコマンドバッファに追加するステップと、
前記決定された実行順序で前記コマンドバッファからの前記複数のコマンドを実行するステップとを備える、方法。
【請求項２】
各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するステップが、前記複数のコマンドのそれぞれのコマンドに対して、

前記それぞれのコマンドがデータ依存関係を有しないとき、前記それぞれのコマンドと関連付けられるレンダリング対象に基づき、レンダリング対象ごとのコマンドリストに前記それぞれのコマンドを追加するステップを備え、

前記複数のコマンドを実行するステップが、前記それぞれのレンダリング対象ごとのコマンドリストの前記複数のコマンドを実行するステップと備える、請求項1に記載の方法。

【請求項３】

各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するステップが、前記複数のコマンドのそれぞれのコマンドに対して、

前記それぞれのコマンドが以前のコマンドに依存するとき、前記初期の順序において前記それぞれのコマンドに後続するコマンドを並べ替えることなく、前記実行順序を決定するステップを備える、請求項1に記載の方法。

【請求項４】

前記実行順序を決定するステップが、前記複数のコマンドのそれぞれのコマンドに対して、

前記それぞれのコマンドの同じレンダリング対象の前記初期の順序において、前のコマンドと異なる関連するレンダリングモードを前記それぞれのコマンドが有するかどうかを決定するステップと、

前記それぞれのコマンドが異なる関連するレンダリングモードを有するとき、前記初期の順序において前記それぞれのコマンドに後続する前記複数のコマンドを並べ替えることなく、前記実行順序を決定するステップとを備える、請求項1に記載の方法。

【請求項５】

前記レンダリング対象ごとのコマンドリストの各々にレンダリングモードを割り当てるステップをさらに備える、請求項1に記載の方法。

【請求項６】

前記複数のコマンドと関連付けられる1つまたは複数のタイムスタンプに基づいて、前記レンダリング対象間の前記データ依存関係を決定するステップをさらに備える、請求項1に記載の方法。

【請求項７】

前記複数のコマンドを受信するステップが、第1のレンダリング対象の第1のコマンドを受信するステップと、第2のレンダリング対象の第2のコマンドを受信するステップと、前記第1のレンダリング対象の第3のコマンドを受信するステップとを備え、

前記データ依存関係に基づいてそれぞれのレンダリング対象ごとのコマンドリストを生成するステップが、前記第1のレンダリング対象の前記第3のコマンドが前記第2のレンダリング対象の前記第2のコマンドの結果に依存するかどうかを決定するステップを備え、

前記第3のコマンドが前記第2のコマンドに依存しないとき、前記実行順序を決定するステップが、前記コマンドを実行するステップが前記第2のコマンドの前に前記第3のコマンドを実行するステップを備えるように、前記第3のコマンドおよび前記第2のコマンドを並べ替えるステップを備える、請求項1に記載の方法。

【請求項８】

前記それぞれのコマンドリストを生成するステップが、コマンドごとに、前記それぞれのレンダリング対象ごとのコマンドリストを生成するステップを備える、請求項１に記載の方法。

【請求項９】

前記それぞれのコマンドリストを生成するステップが、前記初期の順序で、前記それぞれのレンダリング対象のそれぞれのコマンドを前記それぞれのレンダリング対象ごとのコマンドリストに追加するステップを備える、請求項８に記載の方法。

【請求項１０】

前記複数のコマンドを受信するステップが、複数のアプリケーションプログラミングインターフェース(API)コマンドを受信するステップを備える、請求項１に記載の方法。

【請求項１１】
前記複数のコマンドが、コマンドストリーム内に含まれ、
前記それぞれのコマンドリストを生成するステップが、前記コマンドストリームの全体を分析することなく、前記それぞれのコマンドリストを生成するステップを備える、請求項１に記載の方法。
【請求項１２】
グラフィックスデータをレンダリングするためのデバイスであって、
画像をレンダリングするための、複数のレンダリング対象と関連付けられる複数のコマンドを記憶するように構成されるメモリであって、前記画像が、前記複数のレンダリング対象からのコンテンツで構成される、メモリと、
複数のバッファであって、前記複数のレンダリング対象のうちの各レンダリング対象が、前記複数のバッファのそれぞれのバッファである、複数のバッファと、
コマンドバッファと、
1つまたは複数のプロセッサとを備え、前記1つまたは複数のプロセッサが、
前記複数のレンダリング対象と関連付けられる前記複数のコマンドを受信することであって、前記複数のコマンドのうちのコマンドが初期の順序で受信される、ことと、
前記初期の順序と異なる順序である、前記コマンドの実行順序を決定することであって、前記実行順序を決定することが、前記初期の順序の前記コマンドのレンダリング対象間のデータ依存関係に基づいて、前記複数のレンダリング対象の各レンダリング対象のためのそれぞれのレンダリング対象ごとのコマンドリストを生成することを備える、ことと、
前記レンダリング対象ごとのコマンドリストを前記コマンドバッファに追加することと、
前記決定された実行順序で前記コマンドバッファからの前記複数のコマンドを実行することと
を行うように構成される、デバイス。
【請求項１３】
各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するために、前記1つまたは複数のプロセッサが、前記複数のコマンドのそれぞれのコマンドに対して、
前記それぞれのコマンドがデータ依存関係を有しないとき、前記それぞれのコマンドと関連付けられるレンダリング対象に基づき、レンダリング対象ごとのコマンドリストに前記それぞれのコマンドを追加するように構成され、
前記複数のコマンドを実行するために、前記1つまたは複数のプロセッサが、前記それぞれのレンダリング対象ごとのコマンドリストの前記複数のコマンドを実行するように構成される、請求項12に記載のデバイス。
【請求項１４】
各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するために、前記1つまたは複数のプロセッサが、前記複数のコマンドのそれぞれのコマンドに対して、
前記それぞれのコマンドが以前のコマンドに依存するとき、前記初期の順序において前記それぞれのコマンドに後続するコマンドを並べ替えることなく、前記実行順序を決定するように構成される、請求項12に記載のデバイス。
【請求項１５】
前記実行順序を決定するために、前記1つまたは複数のプロセッサが、前記複数のコマンドのそれぞれのコマンドに対して、
前記それぞれのコマンドの同じレンダリング対象の前記初期の順序において、前のコマンドと異なる関連するレンダリングモードを前記それぞれのコマンドが有するかどうかを決定し、
前記それぞれのコマンドが異なる関連するレンダリングモードを有するとき、前記初期

の順序において前記それぞれのコマンドに後続する前記複数のコマンドを並べ替えることなく、前記実行順序を決定するように構成される、請求項12に記載のデバイス。

【請求項１６】

前記1つまたは複数のプロセッサがさらに、前記レンダリング対象ごとのコマンドリストの各々にレンダリングモードを割り当てるように構成される、請求項12に記載のデバイス。

【請求項１７】

前記1つまたは複数のプロセッサがさらに、前記複数のコマンドと関連付けられる1つまたは複数のタイムスタンプに基づいて、前記レンダリング対象間の前記データ依存関係を決定するように構成される、請求項12に記載のデバイス。

【請求項１８】

前記複数のコマンドを受信するために、前記1つまたは複数のプロセッサが、第1のレンダリング対象の第1のコマンドを受信し、第2のレンダリング対象の第2のコマンドを受信し、前記第1のレンダリング対象の第3のコマンドを受信するように構成され、

前記データ依存関係に基づいてそれぞれのレンダリング対象ごとのコマンドリストを生成するために、前記1つまたは複数のプロセッサが、前記第1のレンダリング対象の前記第3のコマンドが前記第2のレンダリング対象の前記第2のコマンドの結果に依存するかどうかを決定するように構成され、

前記第3のコマンドが前記第2のコマンドに依存しないとき、前記実行順序を決定するために、前記1つまたは複数のプロセッサは、前記コマンドを実行するために前記1つまたは複数のプロセッサが前記第2のコマンドの前に前記第3のコマンドを実行するように構成されるように、前記第3のコマンドおよび前記第2のコマンドを並べ替えるように構成される、請求項12に記載のデバイス。

【請求項１９】

前記1つまたは複数のプロセッサがグラフィックス処理装置(GPU)に含まれる、請求項12に記載のデバイス。

【請求項２０】

グラフィックスデータをレンダリングするためのデバイスであって、

画像をレンダリングするための、複数のレンダリング対象と関連付けられる複数のコマンドを受信するための手段であって、前記複数のコマンドのうちのコマンドが初期の順序で受信され、前記複数のレンダリング対象のうちの各レンダリング対象が、それぞれのバッファであり、前記画像が、前記複数のレンダリング対象からのコンテンツで構成される、手段と、

前記初期の順序と異なる順序である、前記コマンドの実行順序を決定するための手段であって、前記実行順序を決定するための前記手段が、前記初期の順序の前記コマンドのレンダリング対象間のデータ依存関係に基づいて、前記複数のレンダリング対象の各レンダリング対象のためのそれぞれのレンダリング対象ごとのコマンドリストを生成するための手段を備える、手段と、

前記レンダリング対象ごとのコマンドリストをコマンドバッファに追加するための手段と、

前記決定された実行順序で前記コマンドバッファからの前記複数のコマンドを実行するための手段とを備える、デバイス。

【請求項２１】

各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するための前記手段が、前記複数のコマンドのそれぞれのコマンドに対して、

前記それぞれのコマンドがデータ依存関係を有しないとき、前記それぞれのコマンドと関連付けられるレンダリング対象に基づき、レンダリング対象ごとのコマンドリストに前記それぞれのコマンドを追加するための手段を備え、

前記複数のコマンドを実行するための手段が、前記それぞれのレンダリング対象ごとのコマンドリストの前記複数のコマンドを実行するための手段を備える、請求項20に記載の

デバイス。
【請求項２２】
各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するための前記手段が、前記複数のコマンドのそれぞれのコマンドに対して、
前記それぞれのコマンドが以前のコマンドに依存するとき、前記初期の順序において前記それぞれのコマンドに後続するコマンドを並べ替えることなく、前記実行順序を決定するための手段とを備える、請求項20に記載のデバイス。
【請求項２３】
前記実行順序を決定するための前記手段が、前記複数のコマンドのそれぞれのコマンドに対して、
前記それぞれのコマンドの同じレンダリング対象の前記初期の順序において、前のコマンドと異なる関連するレンダリングモードを前記それぞれのコマンドが有するかどうかを決定するための手段と、
前記それぞれのコマンドが異なる関連するレンダリングモードを有するとき、前記初期の順序において前記それぞれのコマンドに後続する前記複数のコマンドを並べ替えることなく、前記実行順序を決定するための手段とを備える、請求項20に記載のデバイス。
【請求項２４】
前記レンダリング対象ごとのコマンドリストの各々にレンダリングモードを割り当てるための手段をさらに備える、請求項20に記載のデバイス。
【請求項２５】
前記複数のコマンドと関連付けられる1つまたは複数のタイムスタンプに基づいて、前記レンダリング対象間の前記データ依存関係を決定するための手段をさらに備える、請求項20に記載のデバイス。
【請求項２６】
命令を記憶したコンピュータ可読記憶媒体であって、前記命令が、実行されると、1つまたは複数のプロセッサに、
画像をレンダリングするための、複数のレンダリング対象と関連付けられる複数のコマンドを受信することであって、前記複数のコマンドのうちのコマンドが初期の順序で受信され、前記複数のレンダリング対象のうちの各レンダリング対象が、それぞれのバッファであり、前記画像が、前記複数のレンダリング対象からのコンテンツで構成される、ことと、
前記初期の順序と異なる順序である、前記コマンドの実行順序を決定することであって、前記実行順序を決定することが、前記初期の順序の前記コマンドのレンダリング対象間のデータ依存関係に基づいて、前記複数のレンダリング対象の各レンダリング対象のためのそれぞれのレンダリング対象ごとのコマンドリストを生成することを含む、ことと、
前記レンダリング対象ごとのコマンドリストをコマンドバッファに追加することと、
前記決定された実行順序で前記コマンドバッファからの前記複数のコマンドを実行することと
を行わせる、コンピュータ可読記憶媒体。
【請求項２７】
各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するために、前記命令が、前記1つまたは複数のプロセッサに、前記複数のコマンドのそれぞれのコマンドに対して、
前記それぞれのコマンドがデータ依存関係を有しないとき、前記それぞれのコマンドと関連付けられるレンダリング対象に基づき、レンダリング対象ごとのコマンドリストへ前記それぞれのコマンドを追加させ、
前記複数のコマンドを実行するために、前記命令が、前記1つまたは複数のプロセッサに、前記それぞれのレンダリング対象ごとのコマンドリストの前記複数のコマンドを実行させる、請求項26に記載のコンピュータ可読記憶媒体。
【請求項２８】

各レンダリング対象のための前記レンダリング対象ごとのコマンドリストを生成するために、前記命令が、前記1つまたは複数のプロセッサに、前記複数のコマンドのそれぞれのコマンドに対して、

前記それぞれのコマンドが以前のコマンドに依存するとき、前記初期の順序において前記それぞれのコマンドに後続するコマンドを並べ替えることなく、前記実行順序を決定させる、請求項26に記載のコンピュータ可読記憶媒体。

【請求項２９】

前記実行順序を決定するために、前記命令が前記1つまたは複数のプロセッサに、前記複数のコマンドのそれぞれのコマンドに対して、

前記それぞれのコマンドの同じレンダリング対象の前記初期の順序において、前のコマンドと異なる関連するレンダリングモードを前記それぞれのコマンドが有するかどうかを決定させ、

前記それぞれのコマンドが異なる関連するレンダリングモードを有するとき、前記初期の順序において前記それぞれのコマンドに後続する前記複数のコマンドを並べ替えることなく、前記実行順序を決定させる、請求項26に記載のコンピュータ可読記憶媒体。

【請求項３０】

前記命令がさらに、前記1つまたは複数のプロセッサに、前記レンダリング対象ごとのコマンドリストの各々へレンダリングモードを割り当てさせる、請求項26に記載の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】

本開示は、グラフィックス処理をレンダリングすることに関する。

【背景技術】

【０００２】

電子ディスプレイ上での視覚的提示のためのコンテンツを提供するデバイスは、一般にグラフィックス処理装置(GPU)を含む。GPUは、ディスプレイ上にコンテンツを表現するピクセルをレンダリングする。GPUは、ディスプレイ上の各ピクセルに対して1つまたは複数のピクセル値を生成し、ディスプレイ上の各ピクセルのピクセル値に対してグラフィックス処理を実行して提示のための各ピクセルをレンダリングする。

【発明の概要】

【発明が解決しようとする課題】

【０００３】

本開示の技法は全般に、グラフィックスデータをレンダリングすることに関する。グラフィックス処理装置(GPU)は、レンダリングの間にレンダリング対象を変更することがある。レンダリング対象を変更することは、レンダリング対象の変更と関連付けられるロード動作(たとえば、GPUメモリへのデータのロード)および記憶動作(たとえば、GPUメモリから外部メモリへのデータの記憶)が原因で、相対的に計算集約的であり、かつ/または時間集約的であることがある。本開示の技法は、ロード動作および記憶動作を減らし得るような方法で、レンダリング対象と関連付けられるコマンドをコマンド間の依存関係に基づいて並べ替えることを含む。

【課題を解決するための手段】

【０００４】

ある例では、グラフィックスデータをレンダリングするための方法は、複数のレンダリング対象と関連付けられる複数のコマンドを受信するステップであって、複数のコマンドが初期の順序で受信される、ステップと、コマンド間のデータ依存関係に基づいて初期の順序と異なる順序で複数のコマンドの1つまたは複数を並べ替えることを含む、複数のコマンドの実行順序を決定するステップと、決定された実行順序で複数のコマンドを実行するステップとを含む。

【０００５】

別の例では、グラフィックスデータをレンダリングするためのデバイスは、複数のレンダリング対象と関連付けられる複数のコマンドを記憶するように構成されるメモリを含む。デバイスはまた、複数のレンダリング対象と関連付けられる複数のコマンドを受信し、ここで複数のコマンドが初期の順序で受信され、コマンド間のデータ依存関係に基づいて初期の順序と異なる順序で複数のコマンドの1つまたは複数を並べ替えることを含めて、複数のコマンドの実行順序を決定し、決定された実行順序で複数のコマンドを実行するように構成される、1つまたは複数のプロセッサを含む。

【０００６】

別の例では、グラフィックスデータをレンダリングするためのデバイスは、複数のレンダリング対象と関連付けられる複数のコマンドを受信するための手段であって、複数のコマンドが初期の順序で受信される、手段と、コマンド間のデータ依存関係に基づいて初期の順序と異なる順序で複数のコマンドの1つまたは複数を並べ替えることを含めて、複数のコマンドの実行順序を決定するための手段と、決定された実行順序で複数のコマンドを実行するための手段とを含む。

【０００７】

別の例では、非一時的コンピュータ可読媒体は、実行されると、1つまたは複数のプロセッサに、複数のレンダリング対象と関連付けられる複数のコマンドを受信させ、ここで複数のコマンドが初期の順序で受信され、コマンド間のデータ依存関係に基づいて初期の順序と異なる順序で複数のコマンドの1つまたは複数を並べ替えることを含めて、複数のコマンドの実行順序を決定させ、決定された実行順序で複数のコマンドを実行させる、命令を記憶している。

【０００８】

本開示の1つまたは複数の例の詳細が、添付の図面および以下の説明に記載される。他の特徴、目的、および利点は、その説明および図面、ならびに特許請求の範囲から明らかになろう。

【図面の簡単な説明】

【０００９】

【図１】本開示の態様を実装するように構成され得る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からのコンピューティングデバイスをより詳細に示すブロック図である。

【図３Ａ】本開示の技法による、例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図３Ｂ】本開示の技法による、例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図３Ｃ】本開示の技法による、例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図３Ｄ】本開示の技法による、例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図４】本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図５Ａ】本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図５Ｂ】本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図６】本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。

【図７】本開示の態様による、コマンドを並べ替えるための例示的なプロセスを示す流れ図である。

【発明を実施するための形態】

【００１０】

従来のグラフィックス処理装置(GPU)のアーキテクチャは、グラフィックスデータのフレーム(画像と呼ばれ得る)をレンダリングするときに比較的大量のデータがシステムメモリから読み取られること、およびシステムメモリに書き込まれることを必要とし得る。モバイルアーキテクチャ(すなわち、モバイルデバイス上のGPU)は、データのフレーム全体を処理するために必要とされるメモリ帯域幅の容量を欠いていることがある。したがって、画像を複数のタイルへと分割するタイルベースのアーキテクチャが開発されてきた。タイルは、比較的少量(たとえば、256kB)の高帯域幅のオンチップグラフィックスメモリ(グラフィックスメモリまたはGMEMと呼ばれることがある)を使用して処理され得るように、サイズ決定される。すなわち、各タイルのサイズは、利用可能なオンチップグラフィックスメモリの量に依存し得る。画像は次いで、各タイルを個々に処理した後で再構築される。

【００１１】

タイルベースのレンダリングは、いくつかの処理パスに関して説明され得る。たとえば、タイルベースのレンダリングを実行するとき、GPUはビニングパスおよびレンダリングパスを実行することができる。ビニングパスに関して、GPUは、画像全体を処理し、ラスタライズされたプリミティブ(三角形など)を、ビンと呼ばれるタイルサイズのエリアへと区分することができる。たとえば、GPUは、画像全体に対するコマンドストリームを処理し、画像のラスタライズされたプリミティブをビンに割り当てる。

【００１２】

いくつかの例では、GPUは、ビニングパスの間に1つまたは複数の可視性ストリームを生成する。可視性ストリームは、最終的な画像において可視であるプリミティブと、最終的な画像において不可視であるプリミティブとを示す。たとえば、あるプリミティブは、1つまたは複数の他のプリミティブによって遮られることでシェーディングされた完成画像において見えない場合、不可視であり得る。

【００１３】

可視性ストリームは、画像全体のために生成されることがあり、または、ビンごとに生成されることがある(たとえば、各ビンに対して1つの可視性ストリーム)。一般に、可視性ストリームは、1と0の列を含んでよく、各々の「1」または「0」がある特定のプリミティブと関連付けられる。各々の「1」は、プリミティブが最終的な画像において可視であることを示す。各々の「0」は、プリミティブが最終的な画像において不可視であることを示す。可視性ストリームは、(以下で説明される)レンダリングパスを制御することができる。たとえば、可視性ストリームは、不可視のプリミティブのレンダリングをスキップするために使用され得る。したがって、実際にビンに寄与する、すなわち、最終的な画像において可視であるプリミティブだけがレンダリングおよびシェーディングされ、それにより、レンダリングおよびシェーディングの動作を減らす。

【００１４】

他の例では、GPUは、異なるプロセス(たとえば、上で説明された可視性ストリーム以外の、またはそれに加えて)を使用して、特定のビンに位置するものとしてプリミティブを分類することができる。別の例では、GPUは、所与のビンの中に存在するプリミティブだけを表現する「インデックス」のビンごとに、別々のリストを出力することができる。たとえば、GPUは最初、1つのデータ構造の中のすべてのプリミティブ(すなわち、頂点)を含み得る。GPUは、各ビンにおいて可視であるプリミティブだけを指し示す、各ビンの構造の中へのポインタのセットを生成することができる。したがって、可視のインデックスに対するポインタだけが、ビンごとのインデックスリストに含まれる。そのようなポインタは、上で説明された可視性ストリームと同様の目的を果たすことができ、ポインタは、ある特定のビンにおいてどのプリミティブ(およびプリミティブと関連付けられるピクセル)が含まれ可視であるかを示す。

【００１５】

いずれの場合も、各レンダリングパスは、クリア/アンリゾルブ段階、レンダリング段階、およびリゾルブ段階を含み得る。クリア/アンリゾルブ段階の間、GPUは、レンダリン

グされるべき新たなタイルのためのオンチップメモリを初期化することができる。たとえば、GPUは、オンチップメモリをある値に初期化し(クリア)、または、外部メモリからオンチップメモリへと値を読み込むことができる(アンリゾルブ)。レンダリング段階の間、GPUは、タイルを処理し、処理されたタイルをオンチップメモリに記憶することができる。すなわち、GPUは、ピクセル値を決定してピクセル値をオンチップメモリに書き込むために、グラフィックス処理パイプラインを実装することができる。リゾルブ段階の間、GPUは、オンチップメモリからフレームバッファと呼ばれ得る外部メモリにタイルの完成したピクセル値を移送することができる。画像のタイルのすべてについて終了した後で、たとえば、画像のタイルのすべてをフレームバッファに記憶した後で、画像は出力(たとえば、表示)の準備ができる。

【００１６】

GPUは、1つまたは複数のレンダリング対象を使用して、グラフィックスデータをレンダリングすることができる。一般に、レンダリング対象は、レンダリングされている画像のためのピクセルをGPUがその中で描画する、バッファである。レンダリング対象を作成することは、メモリ中の特定の領域を描画のために確保することを伴い得る。いくつかの事例では、画像は、複数のレンダリング対象からのコンテンツから構成され得る。たとえば、GPUは、いくつかのレンダリング対象に対するコンテンツをレンダリングし(たとえば、オフスクリーンレンダリング)、コンテンツを組み立てて最終的な画像(シーンとも呼ばれる)を生成することができる。

【００１７】

レンダリング対象は、いくつかのコマンドと関連付けられ得る。たとえば、レンダリング対象は通常、ある幅とある高さを有する。レンダリング対象は表面フォーマットも有することがあり、これは、どれだけのビットが各ピクセルに割り振られるか、および、ビットが赤、緑、青、およびアルファ(または別のカラーフォーマット)の間でどのように分けられるかを記述する。レンダリング対象のコンテンツは、フラグメントシェーダと関連付けられるコマンドなどの、1つまたは複数のレンダリングコマンドによって修正され得る。レンダリング対象はまた、深度ステンシルバッファとともに機能し得る。

【００１８】

プログラム開発者は、Microsoft Incによって開発されたDirectXなどのアプリケーションプログラミングインターフェース(API)を使用して、レンダリング対象および関連するコマンドを定義することができる。いくつかの例では、レンダリング対象は、Open Graphics Library (「OpenGL」) APIにおいて定義されるような、Frame Buffer Object (FBO)と同義であり得る。本開示の技法は全般にレンダリング対象に関して説明されるが、本明細書で説明される技法は、FBOまたは任意の他の同様の構造に適用され得る。

【００１９】

いくつかの事例では、GPUは、レンダリングの間にレンダリング対象を変更することがある。たとえば、(たとえば開発者によって決定されるような)コマンドの初期の実行順序は、GPUに、画像のレンダリングの間にレンダリング対象を切り替えさせ得る。レンダリング対象を変更することは、相対的に計算集約的であり、かつ/または時間集約的であり得る。たとえば、上で述べられたように、各レンダリングパスは、クリア/アンリゾルブ段階、レンダリング段階、およびリゾルブ段階を含む、3つの付随する段階を有する。レンダリング対象を変更するとき、GPUは、画像のすべてのタイルに対して3つすべての段階を実行する。たとえば、GPUは、すべてのタイルを処理し、次のレンダリング対象に移る前に、現在のレンダリング対象のためのメモリにすべての関連するデータをフラッシュする。

【００２０】

説明を目的とする例では、GPUは、いくつかの定められたレンダリング対象および関連するコマンドを有する、コマンドストリームを受信することができる。GPUは、第1のレンダリング対象と関連付けられるレンダリングコマンドを実行することができる。GPUは次いで、第2のレンダリング対象に切り替え、第2のレンダリング対象と関連付けられるレン

ダリングコマンドを実行することができる。GPUは次いで、第1のレンダリング対象に戻り、第1のレンダリング対象と関連付けられる別のレンダリングコマンドを実行することができる。この例では、GPUは、第1のレンダリング対象と関連付けられるデータを、第2のレンダリング対象に切り替える前に、外部メモリにフラッシュすることができる。加えて、(たとえば、第2のレンダリング対象の後で)第1のレンダリング対象に戻るとき、GPUは、第1のレンダリング対象の適切な状態が維持されることを確実にするために、第1のレンダリング対象と関連付けられるデータをGPUメモリへ戻すようにロードすることができる。このデータのフラッシュおよびデータのリロードは、レンダリングプロセスにおけるボトルネックとなり得る。

【００２１】

本開示の技法は、上で説明された非効率性を減らし得るような方法で、レンダリング対象と関連付けられるコマンドをコマンド間の依存関係に基づいて並べ替えることを含む。たとえば、本開示の態様によれば、GPUは、データをレンダリングする前に、データをレンダリングすることと関連付けられるロード動作(たとえば、外部メモリからGPUメモリへのデータのロード)および記憶動作(たとえば、GPUメモリから外部メモリへのデータの記憶)の数を減らすために、レンダリング対象と関連付けられるコマンドの実行順序を決定することができる。GPUは、(たとえば、APIを介して)コマンドが受信および/または定義される初期の順序と異なる順序の実行順序で、コマンドを並べ替えることができる。

【００２２】

説明を目的とする例では、GPU(および/またはGPUドライバ)は、レンダリングの前にレンダリング対象ごとのコマンドリストを生成することができる。たとえば、GPUは、特定のレンダリング対象と関連付けられるものとしてコマンドを特定し、そのコマンドをその特定のレンダリング対象のためのコマンドリストに追加することができる。GPUは、コマンド間のデータ依存関係を特定するまで(たとえば、以前に処理されたコマンドが依存するコマンドを特定するまで)、コマンドリストの各々にコマンドを追加し続けることができる。データ依存関係を特定すると、GPUは、コマンドリストに含まれるコマンドを実行することができる。GPUは次いで、レンダリング対象ごとのコマンドリストを組み立てるプロセスを再開することができる。

【００２３】

本開示の技法は、GPU(および/またはGPUドライバ)が、コマンドを並べ替えてロード動作および記憶動作を除去することを可能にし得る。たとえば、上で与えられた例に関して、レンダリングの前に、GPUは、第1のコマンドおよび第3のコマンドを含む第1のレンダリング対象のための第1のコマンドリストと、第2のコマンドを含む第2のレンダリング対象のための第2のコマンドリストとを生成することができる。レンダリングの間、GPUは、第2のレンダリング対象に切り替える前に、第1のレンダリング対象と関連付けられる第1のコマンドと第3のコマンドの両方を実行することができる。この方法では、GPUは、第2のコマンドを実行した後で、第1のレンダリング対象に戻る必要がない。

【００２４】

以下でより詳細に説明されるように、実行順序は、GPU、GPUドライバ(たとえば、中央処理装置(CPU)によって実行されるような)、またはこれらの組合せによって決定され得る。一例として、GPUドライバは、コマンドストリーム、ならびに、コマンドストリームに基づくレンダリング対象ごとのコマンドリストを生成することができる。別の例では、GPUは、GPUドライバからコマンドストリームを受信し、コマンドストリームに基づいてレンダリング対象ごとのコマンドリストを生成することができる。さらに別の例では、GPUドライバおよびGPUは、コマンドの実行順序を決定する責任を共有し得る。

【００２５】

図1は、グラフィックスデータをレンダリングするための、本開示の技法を実装し得るコンピューティングデバイス30を示すブロック図である。コンピューティングデバイス30の例は、限定はされないが、ワイヤレスデバイス、いわゆるスマートフォンを含む携帯電話またはセルラー電話、携帯情報端末(PDA)、ビデオディスプレイを含むビデオゲームコ

ンソール、モバイルビデオゲームデバイス、モバイルビデオ会議ユニット、ラップトップコンピュータ、デスクトップコンピュータ、テレビのセットトップボックス、タブレットコンピューティングデバイス、電子書籍リーダ、固定式の、またはモバイルのメディアプレーヤーなどを含む。

【００２６】

図1の例では、コンピューティングデバイス30は、CPUメモリ34を有する中央処理装置(CPU)32と、GPUメモリ38および1つまたは複数のシェーディングユニット40を有するグラフィックス処理装置(GPU)36と、ディスプレイユニット42と、レンダリングされたデータ45(「ren.data」)を記憶するディスプレイバッファユニット44と、ユーザインターフェース46と、データ記憶ユニット48とを含む。加えて、記憶ユニット48は、コンパイラ54を有するGPUドライバ50と、GPUプログラム52と、ローカルにコンパイルされるGPUプログラム56とを記憶し得る。

【００２７】

CPU32の例は、限定はされないが、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)、または他の等価の集積論理回路もしくはディスクリート論理回路を含む。CPU32およびGPU36は図1の例では別々のユニットとして示されているが、いくつかの例では、CPU32およびGPU36は単一のユニットに統合され得る。CPU32は、1つまたは複数のアプリケーションを実行し得る。アプリケーションの例は、ウェブブラウザ、電子メールアプリケーション、スプレッドシート、ビデオゲーム、オーディオおよび/もしくはビデオキャプチャ、再生もしくは編集アプリケーション、ディスプレイユニット42を介して提示されるべき画像データの生成を開始する他のアプリケーションを含み得る。

【００２８】

図1に示される例では、CPU32はCPUメモリ34を含む。CPUメモリ34は、マシンコードまたはオブジェクトコードを実行する際に使用される、オンチップ記憶装置またはメモリを表し得る。CPUメモリ34は各々、一定の数のデジタルビットを記憶することが可能なハードウェアメモリレジスタを備え得る。CPU32は、たとえばシステムバスを通じてアクセスされ得る記憶ユニット48から値を読み取ること、またはそれに値を書き込むことよりも高速に、ローカルのCPUメモリ34から値を読取り、またはそれに値を書き込むことが可能であり得る。

【００２９】

GPU36は、グラフィカルな動作を実行するための1つまたは複数の専用プロセッサを表す。すなわち、たとえば、GPU36は、固定された機能を有する専用ハードウェア、および、グラフィックスをレンダリングしてGPUアプリケーションを実行するためのプログラム可能なコンポーネントであり得る。GPU36はまた、DSP、汎用マイクロプロセッサ、ASIC、FPGA、または他の等価な集積論理回路もしくはディスクリート論理回路を含み得る。

【００３０】

GPU36はまた、マシンコードまたはオブジェクトコードを実行する際に使用されるオンチップ記憶装置またはメモリを表し得る、GPUメモリ38を含む。GPUメモリ38は各々、一定の数のデジタルビットを記憶することが可能なハードウェアメモリレジスタを備え得る。GPU36は、たとえばシステムバスを通じてアクセスされ得る記憶ユニット48から値を読み取ること、またはそれに値を書き込むことよりも高速に、ローカルのGPUメモリ38から値を読取り、またはそれに値を書き込むことが可能であり得る。

【００３１】

ディスプレイユニット42は、ビデオデータ、画像、テキスト、または見る者が利用するための任意の他のタイプのデータを表示することが可能なユニットを表す。ディスプレイユニット42は、液晶ディスプレイ(LCD)、発光ダイオード(LED)ディスプレイ、有機LED(OLED)、アクティブマトリックスOLED(AMOLED)ディスプレイなどを含み得る。

【００３２】

ディスプレイバッファユニット44は、ディスプレイユニット42のための、コンピュータ

により生成されたグラフィックス、静止画像、ビデオフレームなど(レンダリングされたデータ45)のような、画像の提示のためのデータを記憶することに専用の、メモリまたは記憶デバイスを表す。ディスプレイバッファユニット44は、複数の記憶位置を含む2次元バッファを表し得る。ディスプレイバッファユニット44内の記憶位置の数は、ディスプレイユニット42上で表示されるべきピクセルの数と実質的に同様であり得る。たとえば、ディスプレイユニット42が640×480ピクセルを含むように構成される場合、ディスプレイバッファユニット44は、赤、緑、および青のピクセル値、または他の色値などの、ピクセルの色と強度の情報を記憶する、640×480の記憶位置を含み得る。

【００３３】

ディスプレイバッファユニット44は、GPU36によって処理されたピクセルの各々の最終的なピクセル値を記憶することができる。ディスプレイユニット42は、ディスプレイバッファユニット44から最終的なピクセル値を取り出し、ディスプレイバッファユニット44に記憶されているピクセル値に基づいて最終的な画像を表示することができる。

【００３４】

ユーザインターフェースユニット46は、ユーザが関わり得るユニット、またはそうでなければ、CPU32などのコンピューティングデバイス30の他のユニットと通信するためのインターフェースを表す。ユーザインターフェースユニット46の例は、限定はされないが、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。ユーザインターフェースユニット46はまた、タッチスクリーンであってよく、またはタッチスクリーンを含んでよく、タッチスクリーンは、ディスプレイユニット42の一部として組み込まれてよい。

【００３５】

記憶ユニット48は、1つまたは複数のコンピュータ可読記憶媒体を備え得る。記憶ユニット48の例は、限定はされないが、ランダムアクセスメモリ(RAM)、読取り専用メモリ(ROM)、電気的消去可能プログラマブル読取り専用メモリ(EEPROM)、CD-ROMもしくは他の光学ディスク記憶装置、磁気ディスク記憶装置もしくは他の磁気記憶デバイス、フラッシュメモリ、または、所望のプログラムコードを命令もしくはデータ構造の形で記憶するために使用され、かつコンピュータもしくはプロセッサによってアクセスされ得る任意の他の媒体を含む。

【００３６】

いくつかの例示的な実装形態では、記憶ユニット48は、CPU32および/またはGPU36に、本開示ではCPU32およびGPU36に帰する機能を実行させる命令を含み得る。記憶ユニット48は、いくつかの例では、非一時的記憶媒体と見なされ得る。「非一時的」という用語は、記憶媒体が搬送波または伝搬信号において具現化されないことを示し得る。しかしながら、「非一時的」という用語は、記憶ユニット48が可動ではないことを意味すると解釈されるべきではない。一例として、記憶ユニット48は、コンピューティングデバイス30から取り除かれてよく、別のデバイスに移されてよい。別の例として、記憶ユニット48と実質的に同様の記憶ユニットは、コンピューティングデバイス30に挿入され得る。いくつかの例では、非一時的記憶媒体は、経時的に変化し得るデータを(たとえば、RAM内に)記憶することができる。

【００３７】

記憶ユニット48は、GPUドライバ50およびコンパイラ54と、GPUプログラム52と、ローカルにコンパイルされるGPUプログラム56とを記憶する。GPUドライバ50は、GPU36にアクセスするためのインターフェースを提供するコンピュータプログラムまたは実行可能コードを表す。CPU32は、GPU36とインターフェースするために、GPUドライバ50またはその一部を実行し、この理由で、図1の例では、GPUドライバ50は、CPU32内の「GPUドライバ50」と名付けられた破線のボックスとして示されている。GPUドライバ50は、GPUプログラム52を含むCPU32によって実行されるプログラムまたは他の実行可能ファイルにアクセス可能である。

【００３８】

GPUプログラム52は、たとえばアプリケーションプログラミングインターフェース(API)を使用して、高水準(HL)プログラミング言語で書かれたコードを含み得る。APIの例は、Open-Computing Language (「OpenCL」)、Open Graphics Library (「OpenGL」)、およびMicrosoft Incによって開発されたようなDirectXを含む。一般に、APIは、関連するハードウェアによって実行される、所定の標準化されたコマンドのセットを含む。APIコマンドは、ハードウェアコンポーネントの仕様に関するユーザの知識を伴わずに、ユーザがGPUのハードウェアコンポーネントに対してコマンドを実行するように命令することを可能にする。

【００３９】

GPUプログラム52は、GPUドライバ50によって提供される1つまたは複数の機能を呼び出し、またはそうでなければ含み得る。CPU32は一般に、GPUプログラム52が埋め込まれるプログラムを実行し、GPUプログラム52に遭遇すると、GPUプログラム52をGPUドライバ50に渡す。CPU32は、この状況においてGPUドライバ50を実行して、GPUプログラム52を処理する。すなわち、たとえば、GPUドライバ50は、GPUプログラム52をGPU36によって実行可能なオブジェクトコードまたはマシンコードへとコンパイルすることによって、GPUプログラム52を処理することができる。このオブジェクトコードは、ローカルにコンパイルされるGPUプログラム56として、図1の例では示されている。

【００４０】

いくつかの例では、コンパイラ54は、GPUプログラム52が埋め込まれるプログラムの実行の間、GPUプログラム52をコンパイルするためにリアルタイムまたはほぼリアルタイムで動作することができる。たとえば、コンパイラ54は一般に、HLプログラミング言語に従って定義されたHL命令を低水準(LL)プログラミング言語のLL命令に変えるユニットを表す。コンパイルの後、これらのLL命令は、FPGA、ASICなど(たとえば、CPU32およびGPU36を含む)のような、特定のタイプのプロセッサまたは他のタイプのハードウェアによって実行されることが可能である。

【００４１】

図1の例では、コンパイラ54は、GPUプログラム52を含むHLコードを実行するとき、CPU32からGPUプログラム52を受信することができる。コンパイラ54は、GPUプログラム52をコンパイルして、LLプログラミング言語に適合するローカルにコンパイルされたGPUプログラム56を生成することができる。コンパイラ54は次いで、LL命令を含むローカルにコンパイルされたGPUプログラム56を出力する。

【００４２】

GPU36は一般に、ローカルにコンパイルされるGPUプログラム56(GPU36内で「ローカルにコンパイルされるGPUプログラム56」と名付けられた破線のボックスによって示されている)を受信し、その後、いくつかの事例では、GPU36は、1つまたは複数の画像をレンダリングし、レンダリングされた画像をディスプレイバッファユニット44に出力する。たとえば、GPU36は、ディスプレイユニット42において表示されるべきいくつかのプリミティブを生成することができる。プリミティブは、線(曲線、スプラインなどを含む)、点、円、楕円、多角形(ここで通常、多角形は1つまたは複数のプリミティブの集合として定義される)、または任意の他の2次元(2D)のプリミティブの1つまたは複数を含み得る。「プリミティブ」という用語はまた、立方体、円筒、球、円錐、三角錐、円環面などの、3次元(3D)のプリミティブを指すことがある。一般に、「プリミティブ」という用語は、ディスプレイユニット42を介して画像(またはビデオデータの文脈ではフレーム)として表示するためにGPU36によってレンダリングされることが可能な、任意の基本的な幾何学的な形状または要素を指す。

【００４３】

GPU36は、プリミティブと、プリミティブの(たとえば、色、テクスチャ、照明、カメラ構成、または他の態様を定義する)他の属性とを、(状態データにおいても規定され得る)1つまたは複数のモデル変換を適用することによって、いわゆる「ワールド空間」へと変換することができる。変換されると、GPU36は、アクティブカメラ(やはりカメラを定義する

状態データにおいても規定され得る)のためのビュー変換を適用して、プリミティブおよび照明の座標をカメラ空間または視点空間へと変換することができる。GPU36はまた、頂点シェーディングを実行して、任意のアクティブな照明のビューにおけるプリミティブの外観をレンダリングすることができる。GPU36は、上のモデル空間、ワールド空間、またはビュー空間の1つまたは複数において、頂点シェーディングを実行し得る(しかし、これは一般にワールド空間において実行される)。

【００４４】

プリミティブがシェーディングされると、GPU36は、投影を実行して、一例として(-1, -1, -1)および(1, 1, 1)にある極点を有する単位立方体へと画像を投影することができる。この単位立方体は一般に、標準ビューボリュームと呼ばれる。モデルを視点空間から標準ビューボリュームに変換した後で、GPU36は、クリッピングを実行して、ビューボリューム内に少なくとも部分的に存在しないあらゆるプリミティブを除去することができる。言い換えると、GPU36は、カメラのフレーム内にないあらゆるプリミティブを除去することができる。GPU36は次いで、プリミティブの座標をビューボリュームからスクリーン空間にマッピングすることができ、プリミティブの3D座標をスクリーンの2D座標へと実質的に変える。

【００４５】

関連するシェーディングデータを有するプリミティブを画定する変換され投影された頂点が与えられると、GPU36は次いで、プリミティブをラスタライズすることができる。ラスタライズの間、GPU36は、プリミティブと関連付けられる任意のテクスチャを適用することができる(ここでテクスチャは状態データを備え得る)。GPU36はまた、ラスタライズの間に、深度テストとも呼ばれるZバッファアルゴリズムを実行して、プリミティブおよび/またはオブジェクトのいずれかが任意の他のオブジェクトによって閉塞されるかどうかを決定することができる。Zバッファアルゴリズムは、各プリミティブをスクリーンに描画する順序をGPU36が知るように、深度に従ってプリミティブを分類する。(たとえば、タイルベースのレンダリングのための)ビニングのとき、シェーディングはラスタライズの間は実行されないことがある。しかしながら、プリミティブをレンダリングするとき、GPU36は、プリミティブによってカバーされるスクリーンのピクセルの色を計算して設定することができる。GPU36は次いで、レンダリングされたピクセルをディスプレイバッファユニット44に出力する。

【００４６】

ディスプレイバッファユニット44は、画像全体がレンダリングされるまで、レンダリングされた画像のレンダリングされたピクセルを一時的に記憶することができる。ディスプレイバッファユニット44は、この文脈では画像フレームバッファと見なされ得る。ディスプレイバッファユニット44は、ディスプレイユニット42上に表示されるべきレンダリングされた画像を送信することができる。別々に示され説明されているが、いくつかの事例では、ディスプレイバッファユニット44は、記憶ユニット48の一部分を形成し得る。

【００４７】

いくつかの例では、GPU36は、タイルベースのレンダリングを実施して画像をレンダリングすることができる。たとえば、GPU36は、画像をタイルと呼ばれる複数の部分に分割することによって画像をレンダリングする、タイルベースのアーキテクチャを実装し得る。タイルは、GPUメモリ38のサイズに基づいてサイズ決定され得る。

【００４８】

タイルベースのレンダリングを実施するとき、GPU36は、ビニングパスおよび1つまたは複数のレンダリングパスを実行し得る。たとえば、ビニングパスに関して、GPU36は、画像全体を処理して、ラスタライズされたプリミティブを(GPUドライバ50によって設定される)初期ビニング構成のビンへと区分することができる。GPU36はまた、ビニングパスの間に可視性ストリームを生成することができ、これはビンに従って分離され得る。たとえば、各ビンは、画像のための可視性ストリームの対応する部分を割り当てられ得る。GPUドライバ50は、可視性ストリームにアクセスし、各ビンをレンダリングするためのコマンド

ストリームを生成することができる。

【００４９】

各レンダリングパスに関して、GPU36は、クリア/アンリゾルブ段階、レンダリング段階、およびリゾルブ段階を実行し得る。クリア/アンリゾルブ段階の間、GPU36は、レンダリングされるべき新たなタイルのためのGPUメモリ38を初期化する。レンダリング段階の間、GPU36は、タイルをレンダリングし、レンダリングされたタイルをGPUメモリ38に記憶することができる。すなわち、GPU36は、ピクセルシェーディングおよび他の動作を実行して、タイルの各ピクセルのピクセル値を決定し、ピクセル値をGPUメモリ38に書き込むことができる。リゾルブ段階の間、GPU36は、タイルの完成したピクセル値をGPUメモリ38からディスプレイバッファユニット44(または記憶ユニット48)に移送することができる。GPU36がこのようにしてフレームと関連付けられるタイルのすべてをレンダリングした後で、ディスプレイバッファユニット44は、完成した画像をディスプレイユニット42に出力することができる。

【００５０】

いくつかの事例では、GPU36は、ピクセル値をGPUメモリ38に記憶するのではなく、レンダリングの後にピクセル値をディスプレイバッファユニット44(または記憶ユニット48)に記憶することによって、データを直接レンダリングすることができる。直接のレンダリングにより、GPUドライバ50は、最終的な画像において可視ではないプリミティブを特定してスキップするために、可視性ストリームを使用しない。むしろ、プリミティブが可視であるかどうかにかかわらず、コマンドストリームは、すべてのプリミティブをレンダリングするための命令を含む。したがって、記憶ユニット48および/またはディスプレイバッファユニット44における不可視のプリミティブは、最終的に、1つまたは複数の他のプリミティブと関連付けられるピクセル値によって上書きされ得る。

【００５１】

上で説明されたレンダリングの前に、GPUドライバ50は、GPUプログラム52を使用してコマンドストリームを生成する。たとえば、コマンドストリームは、GPUプログラム52からの画像をレンダリングするための命令を含み得る。GPUドライバ50は、命令をコマンドストリームに追加することができ、それらの命令は、それらがストリームにおいて現れる順序でGPU36によって実行される。コマンドストリームは、GPUプログラム52からの画像を構成するプリミティブを定義することができる。

【００５２】

上で述べられたように、コマンドストリームは、画像に対する1つまたは複数のレンダリング対象を定義する命令を含み得る。レンダリング対象を作成すると、メモリ中のある特定の領域(たとえば、ディスプレイバッファユニット44または記憶ユニット48など)が描画のために確保され得る。いくつかの事例では、画像は、複数のレンダリング対象からのコンテンツから構成され得る。各レンダリング対象は、レンダリングコマンドを含む複数の関連するコマンドを有し得る。レンダリング対象はまた、深度ステンシルバッファとともに機能し得る。

【００５３】

いくつかの事例では、GPU36は、レンダリングの間にレンダリング対象を変更することがある。たとえば、GPUプログラム52は、コマンドストリームを生成するときにGPUドライバ50によって維持される初期の実行順序でコマンドを含み得る。GPU36がローカルにコンパイルされるGPUプログラム56を実行するときにレンダリング対象を切り替えるように、異なるレンダリング対象のためのコマンドがコマンドストリームにおいてインターリーブされ得る。

【００５４】

上で述べられたように、レンダリング対象を変更することは、レンダリング対象と関連付けられるロード動作および記憶動作が原因で、相対的に計算集約的かつ/または時間集約的であることがある。たとえば、レンダリング対象と関連付けられるコマンドを実行する前に、GPU36は、レンダリングされている各タイルに対して、必要なデータを記憶ユニ

ット48またはディスプレイバッファユニット44からGPUメモリ38にロードすることができる。加えて、特定のレンダリング対象と関連付けられるコマンドを実行した後に、かつ新たなレンダリング対象に切り替える前に、GPU36は、レンダリングされている各タイルのために、その特定のレンダリング対象と関連付けられるデータをGPUメモリ38から記憶ユニット48またはディスプレイバッファユニット44にフラッシュすることができる。

【００５５】

本開示の態様によれば、GPU36(および/またはGPUドライバ50)は、複数の第1のコマンドを有する第1のレンダリング対象と複数の第2のコマンドを有する第2のレンダリング対象とを含む、複数のレンダリング対象と関連付けられるコマンドを受信することができる。GPU36(および/またはGPUドライバ50)は、コマンド間のデータ依存関係に基づいて、複数の第1のコマンドと複数の第2のコマンドとを含む複数のレンダリング対象のコマンドの実行順序を決定することができる。GPU36(および/またはGPUドライバ50)は次いで、決定された実行順序で複数のレンダリング対象のコマンドを実行することができる。いくつかの例では、決定された実行順序は、データをレンダリングすることと関連付けられるロード動作および記憶動作の数を減らすことができる。

【００５６】

いくつかの例では、GPU36(および/またはGPUドライバ50)は、コマンドが初期のコマンドストリームの初期順序と異なる順序となるように、実行順序を決定するときにコマンドを並べ替えることができる。説明を目的とする例では、コマンドストリームが第1のレンダリング対象Aおよび第2のレンダリング対象Bのための命令を含むと仮定する。加えて、コマンドストリームが、レンダリング対象Aと関連付けられる第1のコマンドと、レンダリング対象Bと関連付けられる第1のコマンドと、レンダリング対象Aと関連付けられる第2のコマンドと、レンダリング対象Bと関連付けられる第2のコマンドとを含む、コマンドの初期の実行順序を含むと仮定する。

【００５７】

上の例では、GPU36(および/またはGPUドライバ50)は、レンダリング対象Aおよびレンダリング対象Bのコマンドの実行順序を決定することができ、これは、コマンドストリームによって規定される初期の順序と異なり得る。たとえば、GPU36(および/またはGPUドライバ50)は、レンダリングの前にレンダリング対象ごとのコマンドリストを生成することができる。そのような例では、GPU36(および/またはGPUドライバ50)は、レンダリング対象Aのための第1のコマンドを用いてレンダリング対象コマンドリストAを構築し始めることができる。GPU36(および/またはGPUドライバ50)は次いで、レンダリング対象Bを特定し、レンダリング対象Bのための第1のコマンドを用いてレンダリング対象コマンドリストBを構築し始めることができる。GPU36(および/またはGPUドライバ50)は次いで、レンダリング対象Aのための第2のコマンドがレンダリング対象Bのためのコマンドに依存するかどうかを決定することができる。データの依存関係がない場合、GPU36(および/またはGPUドライバ50)は、レンダリング対象Aのための第2のコマンドをレンダリング対象コマンドリストAに追加することができる。同様に、データの依存関係がない場合、GPU36(および/またはGPUドライバ50)は、レンダリング対象Bのための第2のコマンドをレンダリング対象コマンドリストBに追加することができる。

【００５８】

したがって、この例では、レンダリング対象Aのためのコマンドを実行し、レンダリング対象Bに切り替え、レンダリング対象Bのためのコマンドを実行し、レンダリング対象Aに切り替えるのではなく、かつ、レンダリング対象Aのためのコマンドを実行し、レンダリング対象Bに切り替え、レンダリング対象Bのためのコマンドを実行するのではなく、GPU36は、決定された実行順序に従ってコマンドを実行することができる。たとえば、GPU36は、コマンドリストに基づいてコマンドを実行することができ、これにより、GPU36は、レンダリング対象コマンドリストAと関連付けられるすべてのコマンドを実行し、それに続いてレンダリング対象コマンドリストBと関連付けられるすべてのコマンドを実行することが可能になる。そうする際に、GPU36は、レンダリング対象Aとレンダリング対象Bと

の間の単一の遷移しか行わず、このことは、レンダリング対象を切り替えることと関連付けられるロード動作および記憶動作の数を減らすことができる。

【００５９】

コンピューティングデバイス30は単なる例として与えられ、本開示の技法を実行する他のコンピューティングデバイス30は異なるように構成され得ることを理解されたい。たとえば、ディスプレイバッファユニット44は記憶ユニット48とは別に示され記述されているが、他の例では、ディスプレイバッファユニット44および記憶ユニット48は、同じコンポーネントに組み込まれ得る。

【００６０】

その上、コンピューティングデバイス30は、明快にするために、図1には示されない追加のモジュールまたはユニットを含み得ることを理解されたい。たとえば、コンピューティングデバイス30は、データを送信して受信するための送受信機ユニットを含んでよく、コンピューティングデバイス30と別のデバイスまたはネットワークとの間のワイヤレス通信または有線通信を可能にするための回路を含んでよい。コンピューティングデバイス30はまた、コンピューティングデバイス30がスマートフォンなどのモバイルワイヤレス電話である例では電話通信を実現するために図1にはいずれも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よく、または、コンピューティングデバイス30がメディアプレーヤーもしくはタブレットコンピュータである場合にスピーカーおよび/もしくはマイクロフォンを含んでよい。いくつかの事例では、ユーザインターフェースユニット46およびディスプレイユニット42は、コンピューティングデバイス30が、デスクトップコンピュータ、または外部のユーザインターフェースもしくはディスプレイとインターフェースするために装備された他のデバイスである例では、コンピューティングデバイス30の外部にあることがある。

【００６１】

図2は、コンピューティングデバイス30の複数の部分をより詳細に示すブロック図である。図2の例では、GPU36は、GPUメモリ38と、コマンドプロセッサ60と、1つまたは複数の処理ユニット64と、ラスタライザ68と、可視性ストリーム72とを含む。加えて、CPU32は、CPUメモリ34と、GPUドライバ50と、コンパイラ54と、コマンド組立ユニット76とを含む。図2のいくつかのユニットは高度に集積され得るが、概念的な目的のために別々に示されていることを理解されたい。その上、いくつかのユニットは、概念的な目的で単一のユニットに関して説明され得るが、1つまたは複数の機能ユニットを含み得る。

【００６２】

図2は、グラフィックスデータをレンダリングするための、本開示の技法を利用できるGPUの単なる一例として与えられる。他の例では、グラフィックスデータをレンダリングするための技法は、他のコンポーネントを有する様々な他のGPUによって実行され得る。たとえば、GPU36はまた、入力組立ユニット、テクスチャユニット、スケジューリングユニット、算術論理装置(ALU)、または他の機能固定のもしくはプログラム可能なGPUコンポーネントなどの、画像を分析してレンダリングすることに関する様々な他のコンポーネントおよびユニットを含み得る。

【００６３】

GPU36のコンポーネントは、記憶ユニット48(図1)などの外部メモリにアクセスするよりも比較的小さなレイテンシで、GPUメモリ38にアクセスすることができる。たとえば、GPUメモリ38は、GPU36とともにオンチップでありGPUコンポーネントに比較的近接しているオンチップメモリであってよく、GPU36内の専用のメモリバスと関連付けられてよい。記憶ユニット48に記憶されているデータにアクセスするために、対照的に、GPU36は、コンピューティングデバイス30の他のコンポーネント(CPU32など)とメモリバスを共有しなければならない可能性があり、これにより、利用可能な帯域幅がより限られたものになり得る。

【００６４】

上で説明されたような、高帯域幅で低レイテンシのGPUメモリ38を利用するために、GPU

36は、タイルベースのレンダリングアーキテクチャを使用してグラフィックスをレンダリングすることができる。GPU36は、より小さな部分(たとえば、タイル)へと画像(シーンとも呼ばれ得る)を分割することができる。GPUメモリ38は、GPU36がタイルをレンダリングする間に、タイルと関連付けられるデータを記憶することができる。タイルをレンダリングした後で、GPU36は、メモリバスを介して、GPUメモリ38からディスプレイバッファユニット44に、レンダリングされたピクセルデータをリゾルブまたはコピーすることができる。

【００６５】

コマンドプロセッサ60は、GPUドライバ50からコマンドストリームを読み取ることを担い得る。たとえば、図1に関して上で説明されたように、GPUドライバ50は、コマンドストリームと呼ばれ得る、GPU36による実行のための命令を出すことができる。コマンドプロセッサ60は、コマンドストリームの命令を読み取り、かつ/または復号することができる。いくつかの例では、コマンドプロセッサ60は、コマンドストリームの命令を含むバッファから読み取ることができる。コマンドプロセッサ60はまた、GPU36において命令の実行を開始することができる。たとえば、コマンドプロセッサ60は、処理ユニット64によって実行されるべき命令をスケジューリングするスレッドスケジューラに命令を与えることができる。

【００６６】

処理ユニット64は、各々がプログラム可能な処理ユニットまたは機能固定の処理ユニットであり得る、1つまたは複数の処理ユニットを含み得る。いくつかの例では、プログラム可能なシェーダユニットは、並列に動作するように構成される複数の処理ユニット、たとえばSIMDパイプラインを含み得る。プログラム可能なシェーダユニットは、シェーダプログラム命令および実行状態レジスタ、たとえば、実行されているプログラム命令中の現在の命令またはフェッチされることになる次の命令を示すプログラムカウンタレジスタを記憶する、プログラムメモリを有し得る。処理ユニット64中のプログラム可能なシェーダユニットは、たとえば、頂点シェーダユニット、ピクセルシェーダユニット、ジオメトリシェーダユニット、ハルシェーダユニット、ドメインシェーダユニット、テッセレーション制御シェーダユニット、テッセレーション評価シェーダユニット、計算シェーダユニット、および/または統合シェーダユニットを含み得る。

【００６７】

処理ユニット64は、命令を実行することを担い得る。たとえば、処理ユニット64は、1つまたは複数のシェーダプログラムを実行することを担い得る。シェーダプログラムは、いくつかの例では、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水準シェーディング言語で書かれるプログラムのコンパイルされたバージョンであり得る。いくつかの例では、シェーダプログラムはシェーダカーネルと呼ばれ得る。一般に、カーネルは、GPU36によって実行されるべきタスクまたは機能を定義するプログラムコードを含み得る。

【００６８】

したがって、処理ユニット64は、頂点シェーディング動作、ジオメトリシェーディング動作、およびピクセルシェーディング動作を担う、プログラム可能なシェーディングユニットであり得る。たとえば、処理ユニット64の1つまたは複数は、シーンの三角形のメッシュを生成するために、シーンを構成するプリミティブ(たとえば、三角形)の頂点の位置を決定することを担い得る。加えて、処理ユニット64の1つまたは複数は、三角形のメッシュからプリミティブを生成すること、ならびに、ピクセル充填動作およびシェーディング動作を担い得る。

【００６９】

複数の処理ユニット64は、同一に構成されてよく、または、特定のタスクを実行するように個別に構成されてよい。たとえば、処理ユニット64の1つは、ビニング動作を担う「ビニングシェーダ」として指定され得るが、残りの処理ユニット64は、上で説明された頂点シェーディング動作、ジオメトリシェーディング動作、またはピクセルシェーディング

動作を実行することを担い得る。

【００７０】

ラスタライザ68は、いくつかの機能を実行するためにハードワイヤリングされる1つまたは複数の機能固定の処理ユニットを含み得る。機能固定のハードウェアは、たとえば1つまたは複数の制御信号を介して、異なる機能を実行するように構成可能であり得るが、機能固定のハードウェアは通常、ユーザによりコンパイルされるプログラムを受け入れることが可能なプログラムメモリを含まない。いくつかの例では、ラスタライザ68は、たとえば、深度テスト、シザーテスト、アルファブレンドなどの、ラスタ動作を実行するように構成され得る。

【００７１】

加えて、ラスタライザ68は、頂点情報を受信することができ、シーンのプリミティブの表現を生成することができる。いくつかの例では、ラスタライザ68は、受信された頂点情報に事前に定められた規則を適用して、どのプリミティブが最終的なシーンにおいて可視であるかを決定する。ラスタライザ68は、シーンの任意の不可視のプリミティブを間引き、または除去することができる。たとえば、ラスタライザ68は、zバッファリング(深度テストとも呼ばれ得る)を実行して、他のプリミティブによってカバーされるプリミティブを、したがって最終的なシーンにおいて可視ではないプリミティブを特定することができる。

【００７２】

可視性ユニット72は、機能固定のハードウェアコンポーネントおよび/またはプログラム可能な処理ユニットの任意の組合せを含み得る。可視性ユニット72は、ラスタライザ68からラスタライズされたデータを受信し、1つまたは複数の可視性ストリームを生成することができる。可視性ストリームを生成するために、可視性ユニット72は、ラスタライザ68によって決定されるように、可視のプリミティブの各々をビンに分配することができる。各ビンは、完成したシーンのタイルを表し得る。

【００７３】

いくつかの例では、可視性ユニット72は、ビンの各々のために別々の可視性ストリームを生成することができる。たとえば、可視性ユニット72は、ある特定のビンのための可視性ストリームを、その特定のビンのプリミティブのどのピクセルが可視であるかということと、その特定のビンのプリミティブのどのピクセルが不可視であるかということとを示すようにフラグを設定することによって、生成することができる。いくつかの態様によれば、可視性ユニット72は、最終的なシーンにおいてプリミティブが可視であることを示すために「1」というフラグ値を、および、最終的なシーンにおいてプリミティブが不可視であることを示すために「0」というフラグ値を設定することができる。いくつかの例では、可視性ユニット72は、画像の粗いラスタライズに従って動作し得る。すなわち、各ピクセルの可視性状態を示すのではなく、可視性ユニット72は、より粗い単位で(たとえば、4ピクセルのブロックに対して)可視性情報を決定することができる。

【００７４】

他の例では、可視性ユニット72は、特定のビンの中に位置しているものとしてプリミティブを分類するために、異なるプロセスを使用し得る。別の例では、可視性ユニット72は、所与のビンの中に存在するプリミティブだけを表現する「インデックス」のビンごとに、別々のリストを出力することができる。たとえば、可視性ユニット72は、最初、1つのデータ構造の中のすべてのプリミティブ(すなわち、頂点)を含み得る。可視性ユニット72は、各ビンにおいて可視であるプリミティブだけを指し示す、各ビンの構造の中へのポインタのセットを生成することができる。したがって、可視のインデックスに対するポインタだけが、ビンごとのインデックスリストに含まれる。

【００７５】

本開示の態様によれば、コマンド組立ユニット76は、コマンドストリームに含まれるコマンドの実行順序を決定することを担い得る。たとえば、コマンド組立ユニット76は、1つまたは複数のレンダリング対象と関連付けられる複数のコマンドを有するコマンドスト

リームを受信するように構成され得る。コマンドストリームは、初期の実行順序で複数のコマンドを含み得る。

【００７６】

コマンド組立ユニット76は、コマンド間にデータ依存関係が存在することまたは存在しないことに基づいて、コマンドストリームのコマンドの実行順序を決定することができる。いくつかの例では、コマンド組立ユニット76は、いくつかのレンダリング対象コマンドリストを使用して実行順序を決定することができ、レンダリング対象コマンドリストの各々が、それぞれのレンダリング対象と関連付けられるコマンドのリストを含む。コマンド組立ユニット76は、データ依存関係および/またはレンダリングモード(たとえば、直接レンダリングモード、ビニングされたレンダリングなど)の変化を特定するまで、コマンドリストにコマンドを追加することができる。

【００７７】

本開示の態様によれば、レンダリング対象コマンドリストはバケットと呼ばれ得る。たとえば、コマンド組立ユニット76は、レンダリング対象ごとにバケットを生成することができる。バケットは、FBO構成へのレンダリングの結果としての、単一のレンダリング対象のためのレンダリングコマンドの参照のチェーン(たとえば、FBO構成とも呼ばれる)として、表現され得る。このチェーンは、レンダリングのためにGPU36を準備するためのプリアンブルコマンド、実際のレンダリングコマンド(たとえば、描画)、および状態復元コマンド(たとえば、GPU36を描画のための特定の状態に復元するための)を含む、コマンドの混合物を含み得る。一般に、バケットは、特定のレンダリング対象のために出されたすべてのレンダリングコマンドを含む。バケットは、本明細書ではコマンドリストとも呼ばれ得る。

【００７８】

コマンド組立ユニット76は、レンダリングの前に2つ以上のバケットを生成することができる。GPU36は次いで、各バケットと関連付けられるコマンドを順番にレンダリングすることができる。一般に、GPU36は、バケットN+1のレンダリングコマンドの前に、バケットNのレンダリングコマンドを実行することができる。

【００７９】

いくつかの例では、以下でより詳細に説明されるように、GPU36およびコマンド組立ユニット76は、第1のコマンドバッファおよび第2のコマンドバッファを使用してレンダリングを制御することができる。たとえば、コマンド組立ユニット76は、第2のバッファを参照する第1のバッファのためのコマンドを決定することができる。第1のバッファは通常、描画コマンドまたはコピーコマンドなどのレンダリングコマンドを含まない。

【００８０】

第2のコマンドバッファは、レンダリングコマンド(たとえば、プリアンブルコマンド、描画コマンド、コピーコマンド、状態復元コマンドなどを含む)に対する参照を含み得る。この例では、コマンド組立ユニット76は、第2のコマンドバッファ中のレンダリングコマンドへの参照を組み立てる(および並べ替える)ことによって、実行順序を生成することができる。レンダリングすると、GPU36は第1のコマンドバッファを実行することができ、第1のコマンドバッファは、第2のコマンドバッファのコマンドへの参照を適切な順序で含む。

【００８１】

説明を目的としてGPUドライバ50とは別に示されているが、GPUドライバ50が本開示においてコマンド組立ユニット76に帰する技法を実行するように、コマンド組立ユニット76がGPUドライバ50と統合され得ることを理解されたい。しかしながら、上の技法はGPUドライバ50のコマンド組立ユニット76によって実行されるものとして説明されるが、コマンド組立ユニット76は、GPU36がコマンドをレンダリングするための実行順序を決定することを担うように、GPU36と統合され得ることを理解されたい。この例では、GPU36は、GPUドライバ50からコマンドストリームを受信し、コマンドストリームのコマンドをレンダリングする前に実行順序を決定することができる。

【００８２】
図3A～図3Dは、本開示の技法による、例示的なコマンド組立プロセスを示す流れ図である。たとえば、図3A～図3Dの一番左の列は、第1のレンダリング対象(レンダリング対象A)および第2のレンダリング対象(レンダリング対象B)と関連付けられるいくつかのレンダリングコマンドを示す。図3A～図3Dのレンダリングコマンドは、(丸で囲まれた数字によって示されるように)1から6と名付けられる。図3A～図3Dにおいて右にある列は、レンダリング対象Aおよびレンダリング対象Bと関連付けられるコマンドを実行順序で組み立てることを示す。すなわち、図3A～図3Dは、レンダリング対象ごとのバケットを生成することを示し、各バケットはコマンドへの参照のリストを含む。

【００８３】
たとえば、上で述べられたように、いくつかの事例では、GPU(GPU36など)またはGPUドライバ(GPUドライバ50など)は、IB1と呼ばれ得る第1のバッファと、IB2と呼ばれ得る第2のバッファとを使用して、実行するコマンドを順序付けることができる。IB1は間接バッファ1と呼ばれ得るが、IB2は間接バッファ2と呼ばれ得る。IB1およびIB2は階層的であり得る。たとえば、IB1の中のコマンドは、IB2の中のコマンドストリーム全体を呼び出し得る。したがって、IB1は通常、IB2への参照を含み、レンダリングの間に使用され得る。IB2は、レンダリングコマンド(プリアンブルコマンド、描画コマンド、コピーコマンド、状態復元コマンドなどを含む)に対する参照を含み得る。このようにして、GPU36は、IB2においてコマンドのリストを構築し、コマンドの決定されたリストを実行のためにIB1にフラッシュすることができる。

【００８４】
したがって、図3A～図3Dの例では、バケットは、FBO構成へのレンダリングの結果としての、単一のレンダリング対象のためのIB2参照のチェーン(たとえば、FBO構成)であり得る。このチェーンは、以下でより詳細に説明されるように、プリアンブルIB2、レンダリングIB2、および状態復元IB2(たとえば、プリアンブルIB2のサブセット)からのコマンドの混合物を含み得る。したがって、各バケットは、特定のレンダリング対象のために出されたレンダリングコマンドのすべてを含む。上で述べられたように、IB1に送られるのを待機している複数のバケットが存在し得る。一般に、GPU36は、バケットN+1のレンダリングコマンドの前に、バケットNの中のレンダリングコマンドを実行することができる。

【００８５】
GPU36が特定のレンダリング対象へとレンダリングするにつれて、GPU36はレンダリングIB2において蓄積し得る。あるレンダリング対象から別のレンダリング対象に切り替えるとき(またはIB1へのコマンドのフラッシュがあるとき)、GPU36は、現在のレンダリング対象のための既存のバケットへの蓄積されたレンダリングを保存することができ、または、新たなバケットエントリを生成することができる。各レンダリング対象は、レンダリングバケットIDに対応付けられてよく、レンダリングバケットIDはバケットの各々を識別する。

【００８６】
本開示の態様によれば、および以下でより詳細に説明されるように、GPU36は、レンダリング対象AおよびレンダリングBのコマンド間のデータ依存関係に基づいて、レンダリング対象AおよびレンダリングBのためのコマンドを初期の順序から並べ替えることができる。たとえば、動作の初期の順序は、レンダリング対象Aへのレンダリング、レンダリング対象Bへのレンダリング、レンダリング対象Aへのレンダリング、およびレンダリング対象Bへのレンダリングを含むと仮定する。本開示の態様によれば、レンダリング対象Aとレンダリング対象Bとの間にデータ依存関係がないとき、GPU36は、レンダリング対象Bからレンダリング対象Aへと戻るように切り替えて、GPU36は、レンダリング対象Aのための以前のコマンドが蓄積されたのと同じバケットに、レンダリング対象Aのためのコマンドを蓄積し続けることができる。同様に、GPU36は、レンダリング対象Bのためのコマンドを蓄積するために単一のバケットを使用することができる。GPU36は次いで、バケットをフラッシュすることができ、レンダリング対象Bのためのコマンドがレンダリング対象A

のためのコマンドに後続する。

【００８７】

本開示のいくつかの態様によれば、GPU36は、レンダリングコマンドと関連付けられるタイムスタンプに少なくとも部分的に基づいて、データ依存関係を決定することができる。たとえば、GPUドライバ50は、レンダリングコマンドが実行されることが意図される順序に基づいて、コマンドストリームの各レンダリングコマンドにタイムスタンプを発行することができる。したがって、GPU36は、現在のコマンドと関連付けられるいずれかのコマンドが現在のコマンドのタイムスタンプよりも早いタイムスタンプを有するかどうかを特定することによって、現在のコマンドに対するデータ依存関係を決定することができる。いくつかの例では、GPU36は加えて、または代替的に、GPU36がコマンドバッファにアクセスする方式に基づいて、データ依存関係を決定することができる。たとえば、GPU36は、ソースバッファ(データの読取り元)または宛先バッファ(データの書込み先)として、コマンドバッファを指定することができる。GPU36が以前のコマンドバッファの命令を実行するときのソースとして特定のコマンドバッファを参照した場合、GPU36は、後続のコマンドバッファにおいてその特定のコマンドバッファだけに書き込むことができる(たとえば、GPU36は読取りの前に書込みを並べ替えなくてよい)。同様に、GPU36がその特定のコマンドバッファを参照した場合、GPU36は、後続のコマンドバッファにおいてその特定のコマンドバッファだけから読み取ることができる(たとえば、GPU36は書込みの前に読取りを並べ替えなくてよい)。

【００８８】

上で説明された並べ替えは、レンダリング対象AおよびレンダリングBと関連付けられるロード動作および記憶動作の数を減らすのを助けることができ、それは、上で述べられたように、GPU36がレンダリング対象Aとレンダリング対象Bとの間で2回以上の遷移を行う必要がないからである。GPU36がレンダリングのバッチを終了するとき(たとえば、バケットに追加されるべき追加のコマンドがないとき)、GPU36は、バケットのためのレンダリングモードを決定することができる。例示的なレンダリングモードは、直接レンダリングモード、ソフトウェアを使用するビニングレンダリングモード、GPU36のハードウェアを使用するビニングレンダリングモード、または他のレンダリングモードを含む。GPU36は、バケットをIB1へとフラッシュするとき、決定されたレンダリングモードを使用することができる。いくつかの事例では、レンダリングモードは、バケットの一部であると見なされることがあり、バケットエントリが作成されるときに決定され、指定後は変化しないことがある。

【００８９】

バケットプロセスの例として、図3Aのステップ1に関して、GPU36は、レンダリング対象Aのためのカラーバッファを設定する命令を受信する(カラーバッファAを設定する)ことができ(90)、それに続いてレンダリング対象Aのための第1の描画コマンド(描画1-A)を受信することができる(92)。それに応答して、GPU36は、レンダリング対象Aのためのバケットを生成する(AのレンダリングバケットIDを得る)ことによってレンダリング対象Aのためのバケットを構築し(94)、レンダリングされている画像のビンレイアウトを更新する(ビンレイアウトを更新する)(96)ことを開始し得る。GPU36は次いで、プリアンブルコマンドをバケットに追加する(プリアンブルを生成する)(98)ことができ、このプリアンブルコマンドは、GPU36に、描画のために既知の状態(たとえば、GPUメモリ38の状態を含む)に入るように指示し得る。GPU36は次いで、レンダリングIB2コマンド(レンダリングIB2-0)を追加し(100)、描画コマンド(描画1-A)を追加することができる(102)。

【００９０】

ステップ2において、GPU36は、レンダリング対象Aのための第2の描画コマンド(描画2-A)を受信することができる(104)。GPU36は、ステップ1の間に以前に生成された第2の描画コマンドをIB2に追加することができる。すなわち、GPU36は、第2のコマンド(描画2-A)をレンダリング対象AのためのIB2の終わりに追加する(レンダリングIB2-0 (106)、描画1-A (108)、描画2-A (110))。

【００９１】
図3Bはステップ3において処理を続け、ステップ3において、GPU36は、レンダリング対象Bとして特定される新たなレンダリング対象を受信する。たとえば、GPU36は、レンダリング対象Bのためのカラーバッファを設定する(カラーバッファBを設定する)命令を受信し(112)、それに続いて、レンダリング対象Bのための第1の描画コマンド(描画1-B)を受信する(114)。それに応答して、GPU36は、Aのレンダリングを処理し(Aのレンダリングを処理し)(116)、フラッシュモードを決定し(Aのフラッシュモードを決定し)(118)、このフラッシュモードは、レンダリング対象Aのためのバケットのためのレンダリングモードとも呼ばれ得る。図3Bに示される例では、GPU36は、レンダリング対象Aのためのビニングモード(現在のバケットID(A-ビニング))を決定し(120)、これは、プリアンブルコマンド(プリアンブルIB2)(122)およびレンダリング対象Aのためのバケットの中に現在あるコマンド(レンダリングIB2-0 (124)、描画1-A (126)、描画2-A (128))と関連付けられる。
【００９２】
加えて、GPU36は、レンダリング対象Bのためのバケットを生成する(Bのレンダリングバケットを得る)ことによってレンダリング対象Bのための新たなバケットを構築し(130)、レンダリングされている画像のビンレイアウトを更新する(ビンレイアウトを更新する)(132)ことを開始し得る。GPU36はまた、レンダリングのための適切な状態へとGPU36を戻すために、復元状態を生成する(134)ためのコマンドを含み得る。たとえば、コマンドを並べ替えるとき、GPU36は、新たなレンダリングコマンドを実行する前に、予想される状態(たとえば、GPUメモリ38に記憶されている適切なデータを有すること)に戻される。GPU36は次いで、レンダリングIB2コマンド(レンダリングIB2-0)を追加することができ(136)、このレンダリングIB2コマンドは今、レンダリング対象Aのための第1の描画コマンド(描画1-A)(138)、レンダリング対象Aのための第2の描画コマンド(描画2-A)(140)、およびレンダリング対象Bのための第1の描画コマンド(描画1-B)(142)を含む。
【００９３】
図3Bはステップ4において処理を続け、ステップ4において、GPU36は、レンダリング対象Bのための第2の描画コマンド(描画2-B)を受信する(144)。GPU36は、ステップ3の間に以前に生成された第2の描画コマンドをIB2に追加することができる。すなわち、GPU36は、レンダリング対象Bのための第2のコマンド(描画2-B)をIB2の終わりに追加するので、IB2は今、レンダリング対象Aのための第1の描画コマンド(描画1-A)(148)、レンダリング対象Aのための第2の描画コマンド(描画2-A)(150)、レンダリング対象Bのための第1の描画コマンド(描画1-B)(152)、およびレンダリング対象Bのための第2の描画コマンド(描画2-B)(154)を含む。
【００９４】
図3Cはステップ5において処理を続け、ステップ5において、GPU36は、レンダリング対象Aのための第3の描画コマンド(描画3-A)(158)のためにレンダリング対象Aに戻すように切り替える(カラーバッファAを設定する)(156)。レンダリング対象を切り替えるとき、GPU36は、先行するレンダリング対象のためにまとめられたあらゆる作業を処理する。したがって、GPU36は、Bのレンダリングを処理し(Bのレンダリングを処理し)(160)、レンダリング対象Bのためのバケットのフラッシュモードを決定する(Bのフラッシュモードを決定する)(162)。図3Bに関して説明されたように、(レンダリング対象Aと関連付けられる)バケットID0に対して(164)、GPU36は、レンダリング対象Aのためのプリアンブルコマンド(プリアンブルIB2)(166)と、レンダリングIB2-0 (168)、描画1-A(170)、描画2-A(172)を含むレンダリング対象Aのためのバケット中に現在あるコマンドとを含む。(レンダリング対象Bと関連付けられる)バケットID1に対する新たな追加として(174)、GPU36は、状態復元コマンド(状態復元IB2)(176)と、レンダリングIB2-0 (178)、描画1-B (180)、および描画2-B (182)を含む、レンダリング対象Bのためのバケット中に現在あるコマンドとを含む。
【００９５】
加えて、図3Cの例では、GPU36は、レンダリング対象Aのための最初の2つの描画コマンドとは別のIB2に、レンダリング対象Aのための第3の描画コマンドを含める。たとえば、G

PU36は、レンダリング対象Aのための以前に生成されたバケットIDを得て(Aのレンダリングバケット IDを得て)(184)、レンダリングされている画像のビンレイアウトを更新する(ビンレイアウトを更新する)(186)。GPU36はまた、レンダリングのための適切な状態へとGPU36を戻すために復元状態を生成するためのコマンド(復元状態生成)を含み得る(188)。GPU36は次いで、レンダリングIB2コマンド(レンダリングIB2-1)を追加することができ(190)、レンダリングIB2コマンドは今、レンダリング対象Aのための第3の描画コマンド(描画3-A)を含む(192)。図3Cは別のIB2(IB2-1)に含まれるものとしてレンダリング対象Aのための第3の描画コマンドを示すが、他の例では、レンダリング対象Aのための第3の描画コマンドは、レンダリング対象Aのための第1の2つの描画コマンド(IB2-0)とともにIB2に追加され得る。

【００９６】

図3Dはステップ6において処理を続け、ステップ6において、GPU36は、レンダリング対象Cとして特定される新たなレンダリング対象を受信する。たとえば、GPU36は、レンダリング対象Cのためのカラーバッファを設定する(カラーバッファCを設定する)命令を受信し(194)、それに続いて、レンダリング対象Cのための第1の描画コマンド(描画1-C)を受信する(196)。レンダリング対象Cがレンダリング対象AまたはレンダリングBのいずれかとデータ依存関係を有すると、説明を目的として仮定する。この例では、GPU36は、レンダリングコマンドを、レンダリングのためのコマンドバッファ(たとえば、IB1バッファ)にフラッシュすることができる。

【００９７】

別の例では、たとえばディスプレイユニット42においてレンダリングされるべきデータを提示するために、アプリケーションはIB2をフラッシュすることができる(198)。すなわち、GPU36は、データが出力される準備ができているとき、後続のコマンドにかかわらずIB2のフラッシュを強いることができる。いずれの場合も、上で述べられたように、レンダリング対象を切り替える(またはフラッシュする)前に、GPU36は、先行するレンダリング対象のためにまとめられたあらゆる作業を処理する。

【００９８】

したがって、GPU36は、レンダリング対象Aのレンダリングコマンドを処理することができる(200)。たとえば、GPU36は、レンダリング対象Aのためのバケットのフラッシュモードを決定する(Aのフラッシュモードを決定する)(202)。上で説明されたように、(レンダリング対象Aと関連付けられる)バケットID0に対して(204)、GPU36は、レンダリング対象Aのためのプリアンブルコマンド(プリアンブルIB2)(206)と、レンダリングIB2-0 (208)、描画1-A(210)、描画2-A(212)を含むレンダリング対象Aのためのバケット中に現在あるコマンドとを含む。(レンダリング対象Aと関連付けられる)バケットID0に対する新たな追加として、GPU36はまた、状態復元コマンド(状態復元IB2)(214)と、第2のIB2(レンダリングIB2-1)(216)および描画3-A(218)のレンダリング対象Aのためのバケット中に現在あるコマンドとを含む。加えて、(レンダリング対象Bと関連付けられる)バケットID1に対して(220)、GPU36は、状態復元コマンド(状態復元IB2)(222)と、レンダリングIB2-0 (224)、描画1-B (226)、および描画2-B (228)を含む、レンダリング対象Bのためのバケット中に現在あるコマンドとを含む。

【００９９】

このようにして、GPU36は、コマンドの初期の順序と異なる順序でコマンドをレンダリングするための実行順序を決定することができる。たとえば、GPU36は、図3A～図3Dに関して説明された、順序付けられたIB2の参照を使用して、レンダリングのためのコマンドバッファを構築することができる。

【０１００】

図4は、本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。たとえば、本開示の態様によれば、GPU(GPU36など)は、「並べ替えオン」モードおよび「並べ替えオフ」モードを含む、2つの並べ替えモードで動作することができる。GPU36は、並べ替え機能が明示的に無効にされない限り、デフォルトで並べ替えオンモード

で動作することができる。

【０１０１】

並べ替えがオンにされるとき、各レンダリング対象がレンダリング対象へのレンダリングのためのバケットおよびコマンドと対応付けるGPU36が、バケットにおいて蓄積される。蓄積は、蓄積されたレンダリングコマンドのフラッシュモード(たとえば、直接レンダリングまたはビニングなどのレンダリングモード)が入来するレンダリングコマンドと一致する限り、以前に充填されたバケットにおいて継続し得る。図4に示される例では、レンダリング対象Aは、直接レンダリングモードおよびレンダリングコマンド240の第1のIB2チェーンと関連付けられる。レンダリング対象Bは、ハードウェアビニングレンダリングモードおよびレンダリングコマンド242の第2のIB2チェーンと関連付けられる。レンダリング対象Cは、直接レンダリングモードおよびレンダリングコマンド244の第3のIB2チェーンと関連付けられる。

【０１０２】

すでにバケットと関連付けられているコマンド(たとえば、チェーン240～244などのIB2チェーン)と異なるフラッシュモード(たとえば、レンダリングモード)と関連付けられるレンダリングコマンドに遭遇すると、GPU36は、並べ替えをオフにする(並べ替えオフモードに切り替える)ことができる。図4の例では、すでに定められているバケットと同じフラッシュモードを有しない例示的なコマンドは、ビニングモードを有するレンダリング対象Aのためのレンダリングコマンド、直接モードを有するレンダリング対象Bのためのレンダリングコマンド、ソフトウェアビニングモードを有するレンダリング対象Bのためのレンダリングコマンド、または、ビニングモードを有するレンダリング対象Cのためのレンダリングコマンドを含む。上で述べられたように、GPU36はまた、確立されているバケットのレンダリングコマンド間のデータ依存関係に遭遇するとき、並べ替えをオフにすることができる。データ依存関係の例は、レンダリング対象Aのためのテクスチャとして使用されているレンダリング対象Bのデータを用いて、レンダリング対象Aへとレンダリングすることであり得る。

【０１０３】

並べ替えがオフにされた後で、GPU36はレンダリング対象のために新たなバケットを生成することができる。すなわち、GPU36は、バケットがある特定のレンダリング対象のためにすでに確立されている場合であっても、レンダリング対象のための以前に定められたバケットのコマンドと、レンダリング対象のコマンドを組み合わせなくてよい。図4に示される例では、GPU36は、レンダリング対象A(直接レンダリングモード)、レンダリング対象B(ハードウェアビニングレンダリングモード)、およびレンダリング対象C(直接レンダリングモード)のための、新たなバケットを生成することができる。

【０１０４】

GPU36がレンダリングのためのハードウェアにレンダリングコマンドをフラッシュするとき、GPU36は、バケットと関連付けられるレンダリングモード(たとえば、直接レンダリングとビニングのいずれか)に従ってバケットの各々を処理し、バケットのコマンドをIB1に挿入し、IB1をレンダリングのために出す。各バケットエントリは、IB1に追加される作業の簡潔なバージョンとして概念化され得る。

【０１０５】

図4の例は、並べ替えがオンまたはオフのいずれかであることに関して説明されたが、他の例では、GPU36は、並べ替えがバケットごとにオンまたはオフへと切り替えられ得るようなさらなる精緻な手法を使用することができる。そのような精緻化は、依存関係をさらに追跡することと引き換えに実現され得る。

【０１０６】

図5A～図5Bは、本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。いくつかの事例では、図5Aおよび図5Bに示される例示的なコマンドは、上の図3A～図3Dに関して説明されたものと同様の方式の、実行順序の並べ替えであり得る。しかしながら、図5A～図5Bの例は、いくつかのブロック移送(BLT)コマンドを含む。

【０１０７】
一般に、BLTコマンドを実行することで、ソース位置から宛先位置にデータがコピーされる。ソース位置からコピーされるデータは、宛先位置のデータと組み合わされ得る。BLTコマンドは、現在のレンダリング対象と関連付けられることがあり、または関連付けられないことがある。BLTコマンドは、インラインで、または非同期的に実行され得る。たとえば、インラインBLTは、レンダリングコマンドとともに実行されてよく、BLTの宛先は、実行されているレンダリングコマンドのバッチのレンダリング対象と同じであり得る。非同期BLTは、レンダリング対象(またはソーステクスチャ)に対する依存関係を有しない。いくつかの例では、非同期BLTは、実行されているコマンドの第1のバッチへと並べ替えられ得る。

【０１０８】
本開示の態様によれば、BLTコマンドは、他のレンダリングコマンドと同じ方式で扱われ得る。たとえば、BLTコマンドに遭遇すると、GPU(GPU36など)は、BLTコマンドが他のレンダリング対象とのデータ依存関係を有するかどうかを決定することができる。BLTコマンドが1つまたは複数の以前のコマンドの結果に依存する場合、GPU36は、上で説明されたように、他のレンダリングコマンドと同じ方式でBLTを扱うことができる。たとえば、GPU36は、現在のバケットをフラッシュし、または(フラッシュすることなく)新たなバケットを作成してコマンドの並べ替えをオフに切り替えることができる。

【０１０９】
図5Aに示される例に関して、GPU36は、レンダリング対象Aのための第1のレンダリングコマンド(FBO A-描画0)と、レンダリング対象Aのための第2のレンダリングコマンド(FBO A-描画1)とを受信することができる。コマンドを受信したことに応答して、GPU36は、新たなバケット(バケットA)を生成し(260)、第1および第2のレンダリングコマンド(描画0、描画1)をバケットに追加することができる(262)。

【０１１０】
GPU36は次いで、Y(例として任意の位置)のコンテンツをZ(別の任意の位置)にコピーするBLTコマンド(非同期BLT)を受信する。BLTコマンドを受信したことに応答して、GPU36は、新たなBLTバケット(非同期BLT IB2チェーン)を生成し(264)、YからZ BLTを追加することができる(266)。いくつかの例では、GPU36は、他のレンダリングコマンドとは別に非同期BLTコマンドを実行することができる。上で述べられたように、非同期BLTは、レンダリング対象に対する依存関係を有せず、GPU36は、実行されているコマンドの第1のバッチへと非同期BLTを並べ替えることができる。GPU36は一般に、実行されているコマンドの第1のバッチにおいて非同期BLTを実行し得るが、いくつかの例では、GPU36は、新たなバケットを作成し、新たなバケットが新たなレンダリング対象への新たな描画と関連付けられているかのように、新たなバケットを処理することができる。

【０１１１】
GPU36は次いで、レンダリング対象Bのための第1のレンダリングコマンド(FBO B-描画0)を受信する。コマンドを受信したことに応答して、GPU36は、新たなバケット(バケットB)を生成し(268)、第1のレンダリングコマンド(描画0)をバケットに追加することができる(270)。

【０１１２】
GPU36は次いで、W(任意の位置)のコンテンツをX(別の任意の位置)にコピーする追加の非同期BLTコマンドを受信する。BLTコマンドを受信したことに応答して、GPU36は、BLTコマンドを非同期BLT IB2チェーンに追加することができる(272)。

【０１１３】
GPU36は次いで、レンダリング対象Aのための第3のレンダリングコマンド(FBO A-描画2)を受信する。コマンドを受信したことに応答して、GPU36は、レンダリング対象Aのための以前に生成されたバケットAにレンダリングコマンドを追加することができる(274)。いくつかの事例では、GPU36は、第3のレンダリングコマンド(描画2)が関連するデータ依存関係を有するかどうかを、コマンドをバケットAに追加する前に決定することができる。

【０１１４】
GPU36は次いで、C(任意の位置)のコンテンツをレンダリング対象AにコピーするインラインBLTコマンドを受信する。BLTコマンドを受信したことに応答して、GPU36は、レンダリング対象Aのための以前に生成されたバケットAにBLTコマンドを追加することができる(276)。やはり、GPU36は、BLTコマンド(CからA)が関連するデータ依存関係を有するかどうかを、コマンドをバケットAに追加する前に決定することができる。
【０１１５】
GPU36は次いで、レンダリング対象Bのための第2のレンダリングコマンド(FBO B-描画1)を受信する。コマンドを受信したことに応答して、GPU36は、レンダリング対象Bのための以前に生成されたバケットBに第2のレンダリングコマンドを追加することができる(278)。
【０１１６】
GPU36は次いで、レンダリング対象Aから位置DへのBLTを受信し、これはデータ依存関係を含む。この時点で、GPU36は、データ依存関係が原因で、生成されたコマンドリストをフラッシュすることができる(280)。別の例では、GPU36は、コマンドの並べ替えをオフにすることができ、新たなBLTバケット(バケットBLT)を生成し(282)、BLTコマンドを新たなBLTバケットに追加することができる(284)。
【０１１７】
並べ替えがオフにされると、GPU36は、入来するコマンドが以前に確立されたバケットを有するレンダリング対象と関連付けられる場合であっても、以前に生成されたバケットにコマンドを追加しなくてよい。したがって、GPU36がレンダリング対象Bのための第3のレンダリングコマンド(FBO-描画2)を受信するとき、GPU36は、新たなバケット(バケットB')を生成し、第3の並べ替えコマンドを新たなバケットに追加することができる(288)。他の例では、並べ替えはオフにされないことがある。たとえば、並べ替えをオフにすることは、追跡される依存関係の数を減らすのを助け得る。しかしながら、並べ替えがオフにされないと、レンダリング対象Bのための第3のレンダリングコマンド(FBO-描画2)がレンダリング対象Aまたは位置Dに対する依存関係を有しないとすれば、GPU36はレンダリング対象Bのための第3のレンダリングコマンド(FBO-描画2)を並べ替えることができる。この例では、GPU36は、レンダリング対象Bのための第2のレンダリングコマンド(描画1)に後続するように、レンダリング対象Bのための第3のレンダリングコマンド(FBO-描画2)を並べ替えることができる。
【０１１８】
GPU36は次いで、レンダリングされたコンテンツを外部メモリにフラッシュするためのコマンド(フラッシュ)を受信することができる。フラッシュコマンドを受信したことに応答して、GPU36は、並べ替えられたレンダリングコマンドを使用してコマンドバッファ(IB1)を構築することができる。すなわち、GPU36は、バケットA、バケットB、バケットBLT、およびバケットB'のための決定されたコマンドリストを、順番にコマンドバッファに追加することができる。このようにして、GPU36は、コマンドバッファのレンダリングコマンドを受信して柔軟に並べ替えるまで、コマンドバッファの構築を延期する。
【０１１９】
図6は、本開示の技法による、別の例示的なコマンド並べ替えのプロセスを示す流れ図である。具体的には、図6の例は、レンダリング対象の間に依存関係が存在する例を示す。たとえば、GPU36は、レンダリング対象Aのための第1のレンダリングコマンド(FBO A-描画0)と、レンダリング対象Aのための第2のレンダリングコマンド(FBO A-描画1)とを受信することができる。コマンドを受信したことに応答して、GPU36は、新たなバケット(バケットA)を生成し(300)、第1および第2のレンダリングコマンド(描画0、描画1)をバケットに追加することができる(302)。
【０１２０】
GPU36は次いで、レンダリング対象Bのための第1のレンダリングコマンド(FBO B-描画0)を受信することができる。コマンドを受信したことに応答して、GPU36は、新たなバケッ

ト(バケットB)を生成し(304)、第1のレンダリングコマンド(描画0)をバケットに追加することができる(306)。

【０１２１】

GPU36は次いで、テクスチャとしてレンダリング対象Bを使用するレンダリング対象Aのための第3のレンダリングコマンド(FBO A-描画2)を受信することができる。この例では、レンダリング対象Aのための第3のレンダリングコマンドはレンダリング対象Bに依存する。上で述べられたように、並べ替えの後で、現在のコマンドと関連付けられる読取りタイムスタンプが、現在のコマンドが依存するコマンドと関連付けられる書込みタイムスタンプよりも後にある場合、依存関係が特定され得る。

【０１２２】

説明を目的とする例では、レンダリング対象Aの第3のレンダリングコマンド(FBO A-描画2)を並べ替えることで、レンダリング対象Aの第3のレンダリングコマンドがバケットAに追加される。しかしながら、レンダリング対象Aの第3のレンダリングコマンド(FBO A-描画2)は、レンダリング対象Bに依存する。すなわち、レンダリング対象Aの第3のレンダリングコマンド(FBO A-描画2)と関連付けられる読取りタイムスタンプは、レンダリング対象Bと関連付けられる書込みタイムスタンプの後にあり、それは、レンダリング対象Aの第3のレンダリングコマンドが適切に計算されるには、レンダリング対象Bのコンテンツが利用可能でなければならないからである。したがって、GPU36は、タイムスタンプに基づいてこの依存関係を特定し、並べ替えをオフにすることができる。

【０１２３】

並べ替えをオフにした後で、GPU36は、コマンドリストをフラッシュし、または、レンダリング対象Aの第3のレンダリングコマンド(FBO A-描画2)(310)を含む新たなバケット(バケットA')を生成することができる(308)。

【０１２４】

図7は、本開示の態様による、コマンドを並べ替えるための例示的なプロセスを示す流れ図である。図7(および本開示の他の箇所)に示されるプロセスは、CPU32および/またはGPU36(図1および図2)によって実行されるものとして説明されているが、本技法は、様々な他の処理ユニットによって実施され得る。

【０１２５】

GPU36は、コマンドを初期の順序で受信する(320)。たとえば、GPU36は、コマンドがGPUプログラム52によって規定される順序で、GPUドライバ50からコマンドを受信することができる。コマンドは、複数のレンダリング対象のための複数のレンダリングコマンドを含み得る。

【０１２６】

本開示の態様によれば、GPU36は、コマンドの実行順序を決定することができる(322)。実行順序を決定する際、GPU36は、コマンド間のデータ依存関係に基づいて、初期の順序と異なる順序でコマンドの1つまたは複数を並べ替えることができる。たとえば、本明細書で説明されるように、GPU36は、レンダリング対象ごとのコマンドリストを生成することができ、追加されるコマンドが別のコマンドの結果に依存しないとすれば、レンダリングコマンドをリストに追加することができる。そのような独立のコマンド(依存関係のないコマンド)は、より複雑な依存関係の追跡を伴わずに不要なレンダリング対象の切替えをなくすために並べ替えられてよく、これにより、GPU36によって実行されるロード動作および記憶動作の数を減らす。

【０１２７】

GPU36は次いで、決定された実行順序を使用してコマンドを実行することができる(324)。たとえば、GPU36は、決定された実行順序で、生成されたレンダリングコマンドリストをコマンドバッファにフラッシュすることができる。GPU36は次いで、GPU36によって通常実行されるように、コマンドバッファからコマンドを実行することができる。

【０１２８】

例に応じて、本明細書で説明される方法のいずれかのいくつかの行為またはイベントは

、異なる順序で実行されてよく、一緒に追加され、統合され、または省略されてよい(たとえば、すべての説明された行為またはイベントが方法の実施のために必要であるとは限らない)ことも理解されたい。その上、いくつかの例では、行為またはイベントは、順次的にではなく、たとえばマルチスレッド処理、割り込み処理または複数のプロセッサを通じて同時に実行されてもよい。

【０１２９】

1つまたは複数の例において、説明された機能は、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得る。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に1つまたは複数の命令またはコードとして記憶され得る。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み得る。データ記憶媒体は、本開示で説明される技法の実装のための命令、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され得る任意の利用可能な媒体であり得る。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されコンピュータによってアクセスされ得る、任意の他の媒体を備え得る。ディスク(disk)およびディスク(disc)は、本明細書において使用されるとき、コンパクトディスク(disc)(CD)、レーザディスク(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し、一方、ディスク(disc)は、データをレーザで光学的に再生する。上記の組合せも、コンピュータ可読媒体の範囲内に同じく含まれるものとする。

【０１３０】

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得る。加えて、いくつかの態様では、本明細書で説明される機能は、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得る。また、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され得る。

【０１３１】

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得る。本開示では、開示される技法を実行するように構成されたデバイスの機能的態様を強調するために、様々なコンポーネント、モジュール、またはユニットが説明されたが、それらのコンポーネント、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、上で説明されたように、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、上で説明されたような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よい。

【０１３２】

様々な例が説明されてきた。これらの例および他の例は、以下の特許請求の範囲内に入る。

【符号の説明】

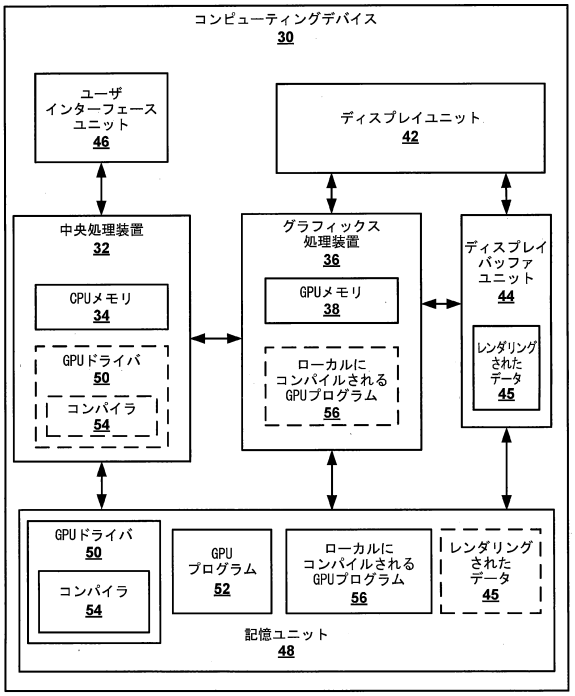
【０１３３】

30 コンピューティングデバイス
32 中央処理装置
34 CPUメモリ
36 グラフィックス処理装置
38 GPUメモリ
42 ディスプレイユニット

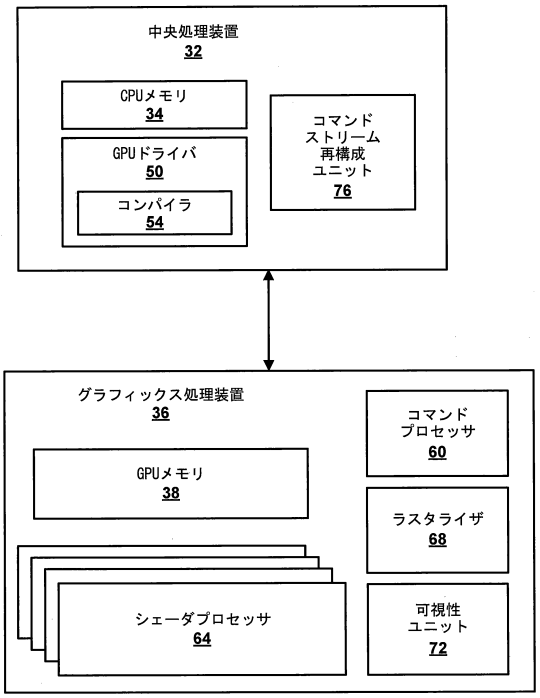
44 ディスプレイバッファユニット
45 レンダリングされたデータ
46 ユーザインターフェースユニット
48 記憶ユニット
50 GPUドライバ
52 GPUプログラム
54 コンパイラ
56 ローカルにコンパイルされるGPUプログラム
60 コマンドプロセッサ
64 シェーダプロセッサ
68 ラスタライザ
72 可視性ユニット
76 コマンドストリーム再構成ユニット
92 描画1-A
100 レンダリングIB2-0
102 描画1-A
104 描画2-A
106 レンダリングIB2-0
108 描画1-A
110 描画2-A
114 描画1-B
120 現在のバケットID(A-ビニング)
122 プリアンブルIB2
124 レンダリングIB2-0
126 描画1-A
128 描画2-A
134 復元状態生成
136 レンダリングIB2-0
138 描画1-A
140 描画2-A
142 描画1-B
144 描画2-B
146 レンダリングIB2-0
148 描画1-A
150 描画2-A
152 描画1-B
154 描画2-B
158 描画3-A
164 バケットID0(A-ビニング)
166 プリアンブルIB2
168 レンダリングIB2-0
170 描画1-A
172 描画2-A
174 バケットID1(B-直接)
176 状態復元IB2
178 レンダリングIB2-0
180 描画1-B
182 描画2-B
188 復元状態生成
190 レンダリングIB2-1

192 描画3-A
196 描画1-C
198 フラッシュ
204 バケットID0(A-ビニング)
206 プリアンブルIB2
208 レンダリングIB2-0
210 描画1-A
212 描画2-A
214 状態復元IB2
216 レンダリングIB2-1
218 描画3-A
220 バケットID1(B-直接)
222 状態復元IB2
224 レンダリングIB2-0
226 描画1-B
228 描画2-B
240 IB2チェーン
242 IB2チェーン
244 IB2チェーン
260 バケットA
262 描画0、描画1
264 同期BLT IB2チェーン
266 YからZ
268 バケットB
270 描画0
272 WからX
274 描画2
276 インラインBLT CからA
278 描画1
280 フラッシュ
282 バケットBLT
284 AからD
286 バケットB'
288 描画2
300 バケットA
302 描画0、描画1
304 バケットB
306 描画0
308 バケットA'
310 描画2

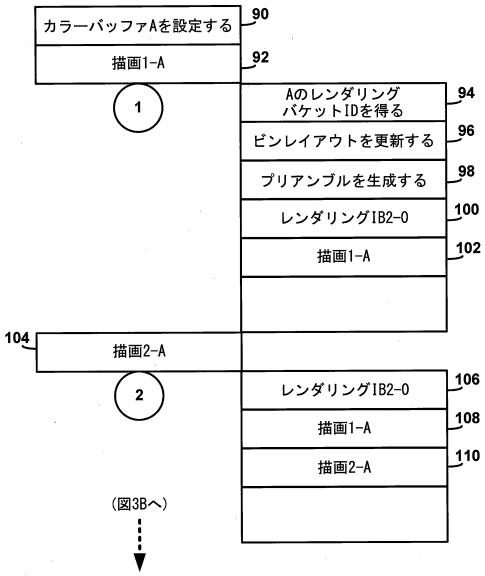
【図１】


コンピューティングデバイス
30
ユーザ
インターフェース
ユニット
46
ディスプレイユニット
42
中央処理装置
32
CPUメモリ
34
GPUドライバ
50
コンパイラ
54
グラフィックス
処理装置
36
GPUメモリ
38
ローカルに
コンパイルされる
GPUプログラム
56
ディスプレイ
バッファ
ユニット
44
レンダリング
された
データ
45
GPUドライバ
50
コンパイラ
54
GPU
プログラム
52
ローカルに
コンパイルされる
GPUプログラム
56
レンダリング
された
データ
45
記憶ユニット
48

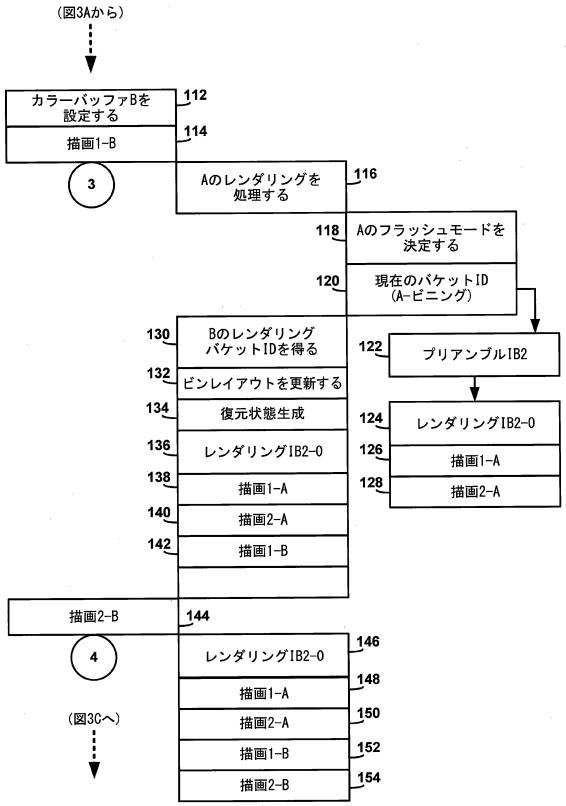
【図２】


中央処理装置
32
CPUメモリ
34
GPUドライバ
50
コンパイラ
54
コマンド
ストリーム
再構成
ユニット
76
グラフィックス処理装置
36
GPUメモリ
38
コマンド
プロセッサ
60
ラスタライザ
68
シェーダプロセッサ
64
可視性
ユニット
72

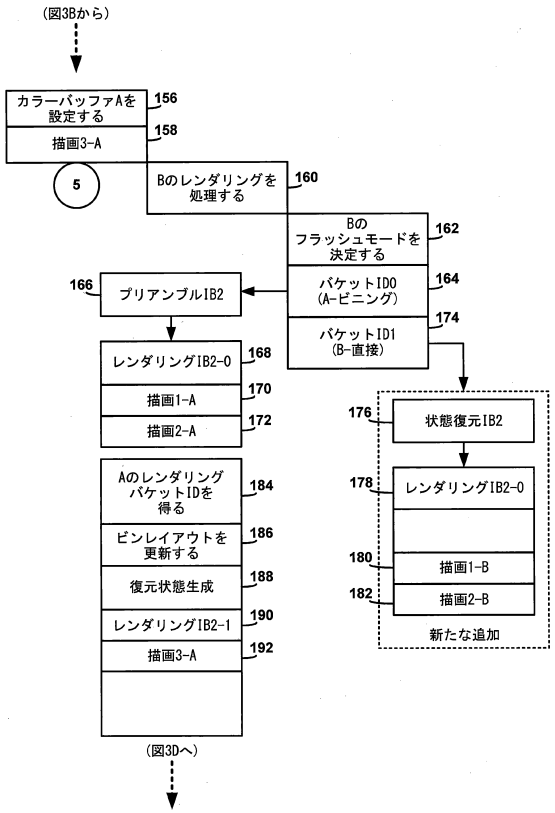
【図３Ａ】


カラーバッファAを設定する
90
描画1-A
92
1
Aのレンダリング
バケットIDを得る
94
ビンレイアウトを更新する
96
プリアンブルを生成する
98
レンダリングIB2-0
100
描画1-A
102
104
描画2-A
2
レンダリングIB2-0
106
描画1-A
108
描画2-A
110
(図3Bへ)

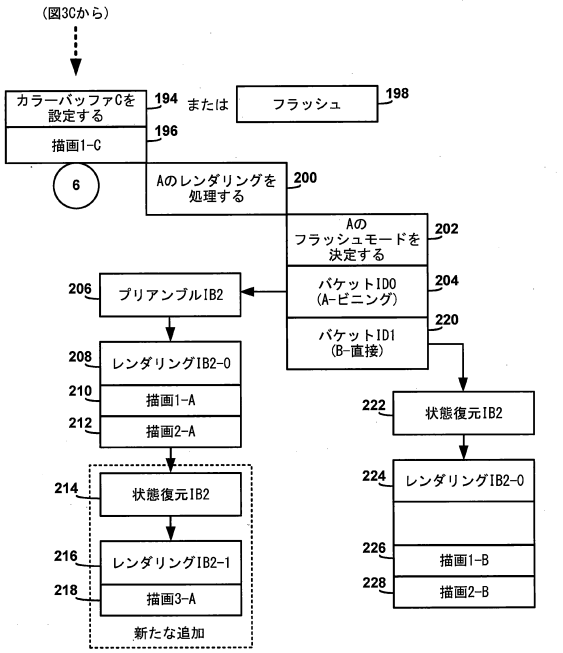
【図３Ｂ】


(図3Aから)
カラーバッファBを
設定する
112
描画1-B
114
3
Aのレンダリングを
処理する
116
118
Aのフラッシュモードを
決定する
120
現在のバケットID
(A-ビニング)
130
Bのレンダリング
バケットIDを得る
122
プリアンブルIB2
132
ビンレイアウトを更新する
134
復元状態生成
124
レンダリングIB2-0
136
レンダリングIB2-0
126
描画1-A
138
描画1-A
128
描画2-A
140
描画2-A
142
描画1-B
144
描画2-B
4
146
レンダリングIB2-0
148
描画1-A
150
描画2-A
152
描画1-B
154
描画2-B
(図3Cへ)

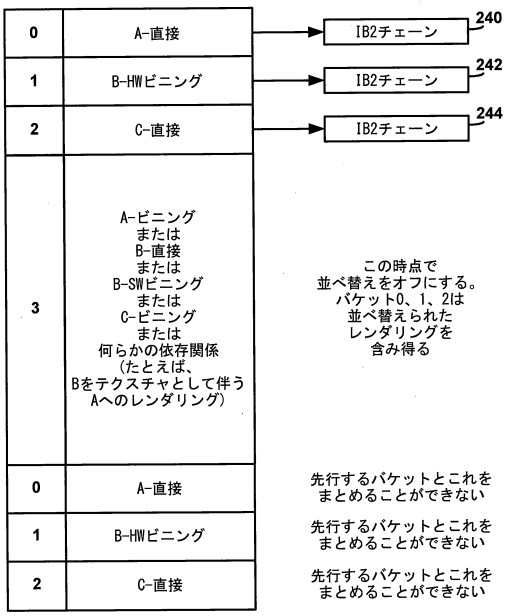
【図３Ｃ】



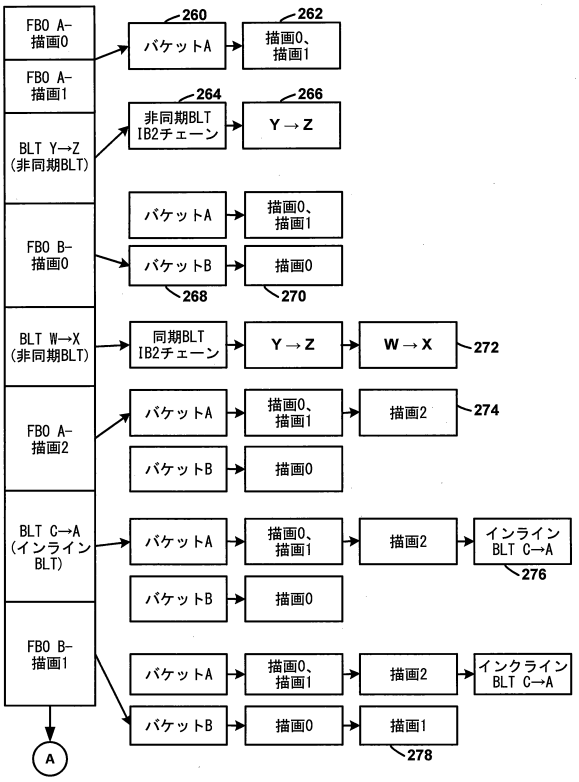
【図３Ｄ】



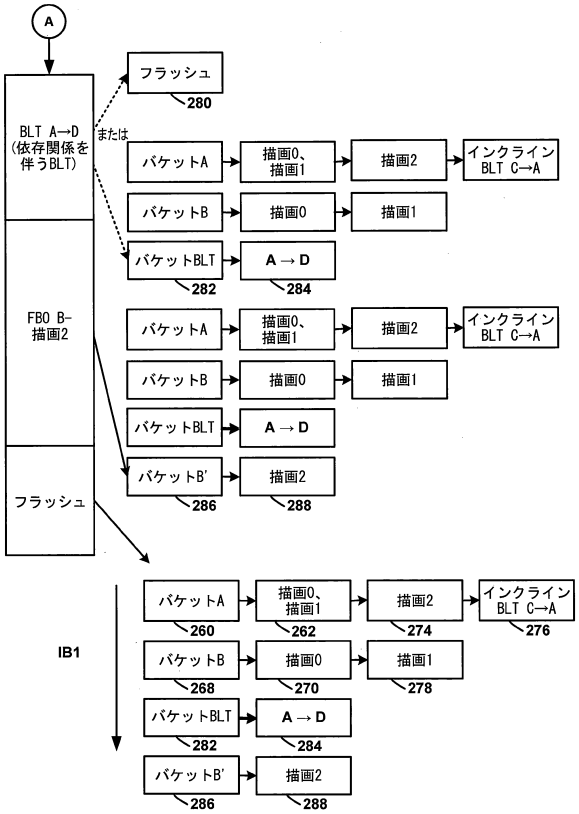
【図４】



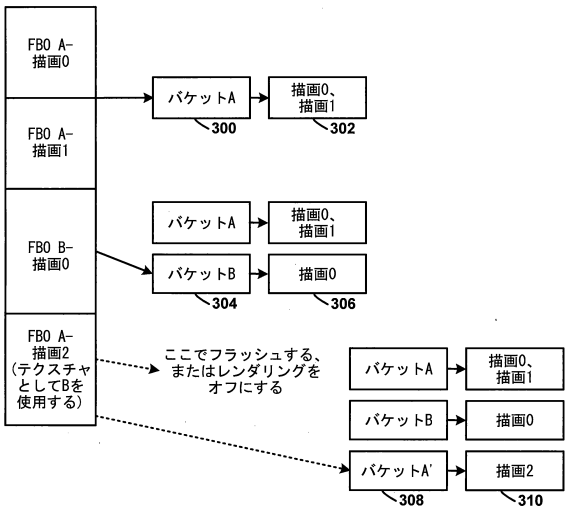
【図５Ａ】



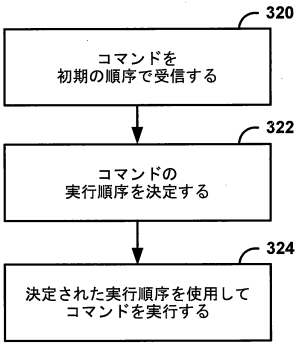
【図５Ｂ】


A
BLT A→D
(依存関係を
伴うBLT)
または
フラッシュ
280
パケットA
描画0、
描画1
描画2
インクライン
BLT C→A
パケットB
描画0
描画1
パケットBLT
A → D
282
284
FBO B-
描画2
パケットB'
描画2
286
288
フラッシュ
IB1
260
262
274
276
268
270
278

【図６】


FBO A-
描画0
FBO A-
描画1
FBO B-
描画0
FBO A-
描画2
(テクスチャ
としてBを
使用する)
パケットA
描画0、
描画1
300
302
パケットB
描画0
304
306
ここでフラッシュする、
またはレンダリングを
オフにする
パケットA'
描画2
308
310

【図７】


320
コマンドを
初期の順序で受信する
322
コマンドの
実行順序を決定する
324
決定された実行順序を使用して
コマンドを実行する

フロントページの続き

(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
(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　モーリス・フランクリン・リブル
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　ヒテンドラ・モハン・ガンガニ
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　村松　貴士

(56)参考文献　国際公開第２００６／１２３５４７（WO，Ａ1）
米国特許出願公開第２０１４／０１８４６２３（ＵＳ，Ａ1）
米国特許出願公開第２０１４／０１１８３６２（ＵＳ，Ａ1）
米国特許出願公開第２０１２／００１７０６９（ＵＳ，Ａ1）
米国特許出願公開第２０１３／０１２７８９１（ＵＳ，Ａ1）

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