(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2020-518210
(P2020-518210A)
(43)公表日　令和2年6月18日(2020.6.18)

(51)Int.Cl.		F I		テーマコード（参考）
H04N 19/513	***(2014.01)***	H04N	19/513	5C159
H04N 21/235	***(2011.01)***	H04N	21/235	5C164
H04N 21/438	***(2011.01)***	H04N	21/438	

審査請求 有　予備審査請求 未請求　（全 46 頁）

(21)出願番号　特願2020-507502 (P2020-507502)
(86)(22)出願日　平成30年4月20日 (2018.4.20)
(85)翻訳文提出日　令和1年12月11日 (2019.12.11)
(86)国際出願番号　PCT/US2018/028620
(87)国際公開番号　WO2018/195461
(87)国際公開日　平成30年10月25日 (2018.10.25)
(31)優先権主張番号　62/488,526
(32)優先日　平成29年4月21日 (2017.4.21)
(33)優先権主張国・地域又は機関
米国 (US)
(31)優先権主張番号　62/634,464
(32)優先日　平成30年2月23日 (2018.2.23)
(33)優先権主張国・地域又は機関
米国 (US)

(71)出願人　519376166
ゼニマックス　メディア　インク.
ＺＥＮＩＭＡＸ　ＭＥＤＩＡ　ＩＮＣ.
アメリカ合衆国　２０８５０　メリーランド州　ロックビル，ピカード　ドライブ　１３７０，スイート１２０
(74)代理人　100107364
弁理士　斉藤　達也
(72)発明者　コピエッツ，ミヒャエル
ドイツ連邦共和国　６０３２７　フランクフルト　アム　マイン　シュパイヒャーシュトラーセ　３８

Fターム(参考)　5C159 KK19 MA05 MA21 MC11 ME01 NN21 NN27 NN28 NN31 SS06 TA61 TB07 TC41 TC47

最終頁に続く

(54)【発明の名称】動きベクトル予測のためのプレイヤ入力の動き補償のためのシステムおよび方法

(57)【要約】
動き推定および補償によって待機時間を減少させるシステムおよび方法を提供する。本発明によると、動きベクトルをマッチング、タギング、および合算するためにリモートサーバからのルックアップテーブルを使用するクライアントデバイスが統合されている。リモートサーバがエンコードされたビデオフレームをクライアントに送信するとき、クライアントはこれをデコードし、動きを推定するのに合算された動きベクトルを適用する。また、この技術は、予め決定された基準に基づいて動きベクトルを生成し、これとインバリデータを、これをキャッシングするクライアントに送信する。サーバは、クライアントに入力を受信するように指示し、これをキャッシングされた動きベクトルまたはインバリデータとマッチングさせるのに使用する。また、この技術は、繰り返される動きベクトルをキャッシングし、以前に生成された動きベクトルライブラリをクライアントに送信する。サーバは、クライアントに動き推定を計算するように指示し、クライアントに記録された動きベクトルライブラリをアップデートするように指示する。したがって、ク

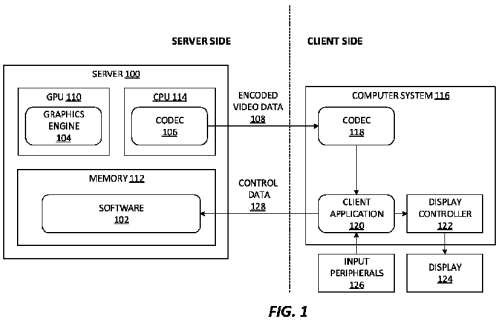


FIG. 1

【特許請求の範囲】
【請求項１】
段階を含むコンピュータで実現される動き推定方法であって、
段階は、
１つ以上のユーザ入力と１つ以上の関連する動きベクトル（ｍｏｔｉｏｎ　ｖｅｃｔｏｒ）で構成されたルックアップテーブルを送信する段階（前記ルックアップテーブルは、前記ルックアップテーブルをキャッシング（ｃａｃｈｉｎｇ）するための命令語（ｉｎｓｔｒｕｃｔｉｏｎ）とともに送信される）、
プレイヤ入力の受信時、マッチングされる動きベクトルに対して前記ルックアップテーブルに質疑（ｑｕｅｒｙ）するための命令語を送信する段階、
固有のタグ（ｕｎｉｑｕｅ　ｔａｇ）を前記ルックアップテーブルからの前記マッチングされる動きベクトルと関連させるための命令語を送信し、タギングされた動きベクトルをキュー（ｑｕｅｕｅ）に追加する段階（前記タギングされた動きベクトルは、合算される）、
フレームを固有の識別子タグ（ｉｄｅｎｔｉｆｉｅｒ　ｔａｇ）とともに送信する段階（前記タグは、前記フレームが前記プレイヤ入力と関連する動きを含む時系列地点（ｃｈｒｏｎｏｌｏｇｉｃａｌ　ｐｏｉｎｔ）を示す）、
サーバから受信されたタギングされたフレームと関連するタグを有する前記キューから動きベクトルを除去するための命令語を送信する段階
を含み、
サーバがクライアントにエンコードされたビデオフレームを送信するとき、前記クライアントは、
前記ビデオフレームをデコードし、ビデオ出力以前の動きを推定するために前記合算された動きベクトルを前記デコードされたフレームに適用するように指示を受ける、コンピュータで実現される方法。
【請求項２】
前記エンコードされたビデオフレームは、
残差ハンドリング（ｒｅｓｉｄｕａｌ　ｈａｎｄｌｉｎｇ）されずに、デコードされる、請求項１に記載のコンピュータで実現する方法。
【請求項３】
前記クライアントに、１つ以上の滑らかな関数（ｓｍｏｏｔｈｉｎｇ　ｆｕｎｃｔｉｏｎ）を前記キュー内の前記合算されたタギングされた動きベクトルに適用するように指示する段階
をさらに含む、請求項１に記載のコンピュータで実現される方法。
【請求項４】
前記動きベクトルと関連する固有のタグは、
実質的に時系列である、請求項１に記載のコンピュータで実現される方法。
【請求項５】
前記合算されたタギングされた動きベクトルは、
対応するマクロブロック（ｍａｃｒｏｂｌｏｃｋ）と関連する、請求項１に記載のコンピュータで実現される方法。
【請求項６】
前記キャッシングされたルックアップテーブルは、
プレイヤ入力に基づいて修正されるように構成される、請求項１に記載のコンピュータで実現される方法。
【請求項７】
前記合算されたタギングされた動きベクトルは、
一度適用されるように構成される、請求項１に記載のコンピュータで実現される方法。
【請求項８】
前記合算されたタギングされた動きベクトルは、

任意複雑度（ｃｏｍｐｌｅｘｉｔｙ）の動きを説明する、請求項１に記載のコンピュータで実現される方法。
【請求項９】
前記合算されたタギングされた動きベクトルは、
前記プレイヤ入力に対する未来（ｆｕｔｕｒｅ）フィードバック（ｆｅｅｄｂａｃｋ）を推定する、請求項１に記載のコンピュータで実現される方法。
【請求項１０】
前記合算されたタギングされた動きベクトルは、
Ｈ．２６４コーディング標準にしたがう、請求項１に記載のコンピュータで実現される方法。
【請求項１１】
動き推定システムであって、
ネットワークを介し、サーバは、
プレイヤ入力の受信時、マッチングされる動きベクトルに対してルックアップテーブルに質疑するために、クライアントに命令語を送信し、
固有のタグを前記ルックアップテーブルからの前記マッチングされる動きベクトルと関連させるために、前記クライアントに命令語を送信し、タギングされた動きベクトルをキューに追加し（前記タギングされた動きベクトルは合算される）、
フレームを固有の識別子タグとともに前記クライアントに送信し（前記タグは、前記フレームが前記プレイヤ入力と関連する動きを含む時系列地点を示す）、
前記サーバから受信されたタギングされたフレームと関連するタグを有する前記キューから動きベクトルを除去するために、前記クライアントに命令語を送信し、
前記サーバが前記クライアントにエンコードされたビデオフレームを送信するとき、前記クライアントは、
前記ビデオフレームをデコードし、動きを推定するために前記合算された動きベクトルを前記デコードされたフレームに適用するように指示を受ける、システム。
【請求項１２】
前記エンコードされたビデオフレームは、
残差ハンドリングされずに、デコードされる、請求項１１に記載のシステム。
【請求項１３】
前記サーバは、
１つ以上の滑らかな関数（ｓｍｏｏｔｈｉｎｇ　ｆｕｎｃｔｉｏｎ）を前記キュー内の前記合算されたタギングされた動きベクトルに適用するように、前記クライアントにさらに指示する、請求項１１に記載のシステム。
【請求項１４】
前記動きベクトルと関連するタグは、
実質的に時系列である、請求項１１に記載のシステム。
【請求項１５】
前記合算されたタギングされた動きベクトルは、
対応するマクロブロックと関連する、請求項１１に記載のシステム。
【請求項１６】
前記キャッシングされたルックアップテーブルは、
プレイヤ入力に基づいて修正されるように構成される、請求項１１に記載のシステム。
【請求項１７】
前記合算されたタギングされた動きベクトルは、
一度適用されるように構成される、請求項１１に記載のシステム。
【請求項１８】
前記合算されたタギングされた動きベクトルは、
任意複雑度（ｃｏｍｐｌｅｘｉｔｙ）の動きを説明する、請求項１１に記載のシステム。

【請求項１９】
前記合算されたタギングされた動きベクトルは、
前記プレイヤ入力に対する未来フィードバックを推定する、請求項１１に記載のシステム。
【請求項２０】
前記合算されたタギングされた動きベクトルは、
Ｈ．２６４コーディング標準にしたがう、請求項１１に記載のシステム。
【請求項２１】
段階を含む動きベクトルをキャッシングするためのコンピュータで実現される方法であって、
段階は、
サーバで１つ以上の動きベクトルを生成する段階（前記動きベクトルは、予め決定された基準に基づいて生成される）、
クライアントに前記生成された動きベクトルと１つ以上のインバリデータ（ｉｎｖａｌｉｄａｔｏｒ）を送信する段階（前記生成された動きベクトルとインバリデータは、前記クライアントでキャッシングされる）、
ユーザから入力を受信するために、前記クライアントに命令語を送信する段階（前記クライアントは、前記入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるように構成される）、および
前記マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース（ｇｒａｐｈｉｃ　ｉｎｔｅｒｆａｃｅ）のエフェクト動き（ｅｆｆｅｃｔ　ｍｏｔｉｏｎ）補償に適用するために、前記クライアントに命令語を送信する段階
を含む、方法。
【請求項２２】
前記動きベクトルは、
ルックアップテーブルにキャッシングされる、請求項２１に記載の方法。
【請求項２３】
前記１つ以上のインバリデータは、
１つ以上のキャッシングされた動きベクトルと関連する、請求項２１に記載の方法。
【請求項２４】
前記１つ以上のインバリデータは、
前記１つ以上のキャッシングされた動きベクトルをディセーブル（ｄｉｓａｂｌｅ）させるように構成される、請求項２３に記載の方法。
【請求項２５】
入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するように、前記クライアントに指示する段階
をさらに含む、請求項２１に記載の方法。
【請求項２６】
前記動きベクトルは、
固有であり予測可能である、請求項２１に記載の方法。
【請求項２７】
前記動きベクトルは、
予め（ａｈｅａｄｏｆｔｉｍｅ）またはランタイム（ｒｕｎｔｉｍｅ）中に生成される、請求項２１に記載の方法。
【請求項２８】
前記動きベクトルは、
前記サーバに記録され、
要求時にキャッシングされるように前記クライアントに送信される、請求項２１に記載の方法。

【請求項２９】
前記動きベクトルは、
ゲームで生成される（ｇａｍｅ－ｇｅｎｅｒａｔｅｄ）、請求項２１に記載の方法。
【請求項３０】
前記ゲームで生成された動きベクトルは、
ピクセルあたりの（ｐｅｒ－ｐｉｘｅｌ）動きベクトルで生成され、
マクロブロックあたりの（ｐｅｒ－ｍａｃｏｒｂｌｏｃｋ）動きベクトルに変換される、請求項２９に記載の方法。
【請求項３１】
動きベクトルをキャッシングするためのシステムであって、
ネットワークを介し、サーバは、
１つ以上の動きベクトルを生成し（前記動きベクトルは、予め決定された基準に基づいて生成される）、
クライアントに前記生成された動きベクトルと１つ以上のインバリデータを送信し（前記生成された動きベクトルとインバリデータは、前記クライアントでキャッシングされる）、
ユーザから入力を受信するために、前記クライアントに命令語を送信し（前記クライアントは、前記入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるように構成される）、
前記マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース内のエフェクト動き補償に適用するために、前記クライアントに命令語を送信する、システム。
【請求項３２】
前記動きベクトルは、
ルックアップテーブルにキャッシングされる、請求項３１に記載のシステム。
【請求項３３】
前記１つ以上のインバリデータは、
１つ以上のキャッシングされた動きベクトルと関連する、請求項３１に記載のシステム。
【請求項３４】
前記１つ以上のインバリデータは、
前記１つ以上のキャッシングされた動きベクトルをディセーブルさせるように構成される、請求項３３に記載のシステム。
【請求項３５】
入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するように、前記クライアントに指示する段階
をさらに含む、請求項３１に記載のシステム。
【請求項３６】
前記動きベクトルは、
固有であり予測可能である、請求項３１に記載のシステム。
【請求項３７】
前記動きベクトルは、
予めまたはランタイム中に生成される、請求項３１に記載のシステム。
【請求項３８】
前記動きベクトルは、
前記サーバに記録され、
要求時にキャッシングされるように前記クライアントに送信される、請求項３１に記載のシステム。
【請求項３９】

前記動きベクトルは、
ゲームで生成される、請求項３１に記載のシステム。
【請求項４０】
前記ゲームで生成された動きベクトルは、
ピクセルあたりの動きベクトルで生成され、
マクロブロックあたりの動きベクトルに変換される、請求項３９に記載のシステム。
【請求項４１】
動きベクトルをキャッシングするためのコンピュータで実現される方法であって、
サーバからクライアントに予め生成された動きベクトルライブラリ（ｍｏｔｉｏｎ　ｖｅｃｔｏｒ　ｌｉｂｒａｒｙ）を送信する段階（前記動きベクトルライブラリは、前記クライアントに記録されるように構成される）、
ユーザからの入力データをモニタリングするために、前記クライアントに命令語を送信する段階、
前記入力データから動き推定を計算するために、前記クライアントに命令語を送信する段階、および
前記入力データに基づいて前記記録された動きベクトルライブラリをアップデートするために、前記クライアントに命令語を送信する段階
を含み、
前記クライアントは、
前記サーバから実際の動きベクトルデータを受信する前にグラフィックインタフェースの動きを開始させるために、前記記録された動きベクトルライブラリを適用するように構成される、コンピュータで実現される方法。
【請求項４２】
前記記録された動きベクトルライブラリの適用をディセーブルさせるために、前記サーバから前記クライアントにコンテキストアップデート（ｃｏｎｔｅｘｔ　ｕｐｄａｔｅ）を送信する段階
をさらに含む、請求項４１に記載のコンピュータで実現される方法。
【請求項４３】
１つ以上のスケーリングファクタ（ｓｃａｌｉｎｇ　ｆａｃｔｏｒ）を前記動きベクトルライブラリに適用するための命令語を送信する段階
をさらに含む、請求項４１に記載のコンピュータで実現される方法。
【請求項４４】
前記スケーリングファクタは、
一般数学式：
【数１】

$$\text{再生速度スケール} = \frac{\text{キャッシングされたアニメーション時間}}{\text{キャッシングされたアニメーション時間} + \text{ディレイ}}$$

に基づいて計算される、請求項３に記載のコンピュータで実現される方法。
【請求項４５】
前記生成された動きベクトルライブラリは、
複数の動きベクトルで構成される、請求項４１に記載のコンピュータで実現される方法。
【請求項４６】
前記動きベクトルは、
ゲームで生成される、請求項４５に記載のコンピュータで実現される方法。
【請求項４７】
前記生成された動きベクトルライブラリは、
前記クライアントに永久的に記録されるように構成される、請求項４１に記載のコンピ

ュータで実現される方法。
【請求項４８】
前記動きベクトルライブラリは、
ビルドプロセス（ｂｕｉｌｄｐｒｏｃｅｓｓ）中に生成される、請求項４１に記載のコンピュータに実現された方法。
【請求項４９】
前記生成された動きベクトルライブラリは、
前記ユーザからの前記入力データと関連する、請求項４１に記載のコンピュータに実現された方法。
【請求項５０】
前記命令語は、
相関タグ（ｃｏｒｒｅｌａｔｉｏｎ　ｔａｇ）で構成され、
前記相関タグは、
前記ユーザからの前記入力データと関連する、請求項４１に記載のコンピュータに実現された方法。
【請求項５１】
動きベクトルをキャッシングするためのシステムであって、
ネットワークを介し、サーバは、
クライアントに予め生成された動きベクトルライブラリを送信し（前記動きベクトルライブラリは、前記クライアントに記録されるように構成される）、
ユーザからの入力データをモニタリングするために、前記クライアントに命令語を送信し、
前記入力データから動き推定を計算するために、前記クライアントに命令語を送信し、
前記入力データに基づいて前記記録された動きベクトルライブラリをアップデートするために、前記クライアントに命令語を送信し、
前記クライアントは、
前記サーバから実際の動きベクトルデータを受信する前にグラフィックインタフェースの動きを開始させるために、前記記録された動きベクトルライブラリを適用するように構成される、システム。
【請求項５２】
前記記録された動きベクトルライブラリの適用をディセーブルさせるために、前記サーバから前記クライアントにコンテキストアップデートを送信する段階
をさらに含む、請求項５１に記載のシステム。
【請求項５３】
前記サーバは、
１つ以上のスケーリングファクタを前記動きベクトルライブラリに適用するための命令語をさらに送信する、請求項５１に記載のシステム。
【請求項５４】
前記スケーリングファクタは、
一般数学式：
【数２】

$$\text{再生速度スケール} = \frac{\text{キャッシングされたアニメーション時間}}{\text{キャッシングされたアニメーション時間} + \text{ディレイ}}$$

に基づいて計算される、請求項５３に記載のシステム。
【請求項５５】
前記生成された動きベクトルライブラリは、
複数の動きベクトルで構成される、請求項５１に記載のシステム。
【請求項５６】

前記動きベクトルは、
ゲームで生成される、請求項５５に記載のシステム。
【請求項５７】
前記生成された動きベクトルライブラリは、
前記クライアントに永久的に記録されるように構成される、請求項５１に記載のシステム。
【請求項５８】
前記動きベクトルライブラリは、
ビルドプロセス中に生成される、請求項５１に記載のシステム。
【請求項５９】
前記生成された動きベクトルライブラリは、
前記ユーザからの前記入力データと関連する、請求項５１に記載のシステム。
【請求項６０】
前記命令語は、
相関タグで構成され、
前記相関タグは、
前記ユーザからの前記入力データと関連する、請求項５１に記載のシステム。
【発明の詳細な説明】
【技術分野】
【０００１】
本出願は、２０１７年４月２１日に出願された米国仮出願第６２／４８８，５２６号、２０１８年２月２３日に出願された米国仮出願第６２／６３４，４６４号、２０１８年３月９日に出願された米国仮出願第６２／６４０，９４５号、および２０１８年３月１６日に出願された米国仮出願第６２／６４４，１６４号の利益を主張する。
【背景技術】
【０００２】
サーバ側のゲームがクライアント側のプレイヤによって制御されるリモートゲーミングアプリケーションは、既存のまたはカスタマイズされたエンコーダを使用し、リアルタイムで３Ｄグラフィックスエンジン（ｇｒａｐｈｉｃｓ　ｅｎｇｉｎｅ）からのビデオ出力をエンコードすることを試みていた。しかし、ビデオゲームの会話形式の特性、特に、ビデオ出力とプレイヤ入力との間のプレイヤフィードバックループ（ｐｌａｙｅｒ　ｆｅｅｄｂａｃｋ　ｌｏｏｐ）は、ゲームビデオストリーミングの待機時間（ｌａｔｅｎｃｙ）が一般的なビデオストリーミングよりも極めて長い。既存のビデオコーディング方法は、エンコード時間を減らすために計算能力（ｃｏｍｐｕｔａｔｉｏｎａｌ　ｐｏｗｅｒ）を交換することができるが、その他の方法はない。エンコードプロセスをビデオレンダリングプロセスに統合するための新たな方法は、エンコード時間を著しく減らすことができる上に、計算能力を低め、エンコードされたビデオの品質を改善し、既存のハードウェアデバイスの相互運用性を維持するためにオリジナルビットストリームデータ形式（ｏｒｉｇｉｎａｌ　ｂｉｔｓｔｒｅａｍ　ｄａｔａ　ｆｏｒｍａｔ）を維持するようになる。
【０００３】
通常のビデオ再生とは異なり、ビデオゲームは、ビデオフィードループ（ｌｏｏｐ）に固有の（ｕｎｉｑｕｅ）プレイヤ入力が生じる。プレイヤは、入力からビデオ出力までの待機時間（ｌａｔｅｎｃｙ）に極めて敏感である。このようなプレイヤ入力フィードバックループにおける長い待機時間は、サーバホストビデオゲームインスタンス（ｉｎｓｔａｎｃｅ）が遠隔のプレイヤによって制御されるビデオゲームストリーミングアプリケーションに対して大きな障害物となる。入力からフィードバックまでの時間を減らすことができるプロセスがあれば、ユーザ経験の直接的な改善に繋がるはずである。
【０００４】
ゲームストリーミング環境において、クライアントハードウェアは多様な水準の計算性能を備えるが、専用Ｈ．２６４ハードウェアデコーダがモバイルおよび他の低電力デバイ

スにおいて最も普遍的である。ハードウェアデコーダは、動き補償（ｍｏｔｉｏｎ　ｃｏｍｐｅｎｓａｔｉｏｎ）のようなＨ．２６４コーディング標準によって通常的に行われる計算の小さな選択を実行するのに強い。専用コーディングハードウェアの強みは、クライアントの全体計算能力に関係なく、ゲームストリーミング環境において優れたプレイヤ経験を提供できるというところにある。

【０００５】

ローカル（ｌｏｃａｌ）、すなわち、ストリーミングされない（ｎｏｎ－ｓｔｒｅａｍｅｄ）レンダリングアプリケーションにおいて、ゲームエンジンは、プレイヤ入力とビデオフィードバックとの間に複数のフレームからなる待機時間を追加する。ゲームストリーミングアプリケーションではプレイヤ入力がネットワークを介してリモートサーバに伝達されなければならず、プレイヤがフィードバックを受信する前に、ビデオ出力がエンコードされて送信されてデコードされなければならないため、追加の待機時間がプレイヤ入力フィードバックサイクルに追加されざるを得ない。いくつかのプレイヤ入力に対し、クライアントは、即時に動き補償を実行してビデオフィードバックに対する結果を推定し、ネットワークの待機時間を減らすことができる。

【０００６】

ビデオゲームがネットワークのクライアントによって遠隔で制御されながらサーバで実行される状況において、入力フィードバック待機時間の減少のために一部のイメージ正確度を犠牲にするために、プレイヤ入力の動き補償は、最も基本的にピクセルのグループをシフト（ｓｈｉｆｔ）させる技術である。このような技術は、１人称ゲームにおける、プレイヤビュー（ｖｉｅｗ）の回転のような一定の動きベクトルをもたらす入力に対するプレイヤフィードバック待機時間を減らすのに強い。

【０００７】

ビデオゲームにおいて、プレイヤコンテキストは、以前のプレイヤアクション、入力、および決定の結果である現在のゲーム状態によって定義される。ゲームストリーミングシステムにおいて、クライアントは、プレイヤコンテキストを認知していない。すなわち、クライアントは、ビデオ出力を受信するだけで、特定のプレイヤ入力の結果を決定するゲーム状態情報は受信しないのである。ゲーム状態情報を基盤に、固有でありながらも予測可能な動き成果（ｏｕｔｃｏｍｅ）を引き起こす多様な入力がある。このような入力は、プレイヤフィードバック待機時間の減少という利点はあるが、クライアントはプレイヤコンテキスト情報を有さないため、従来のプレイヤ入力の動き補償のためにクライアントにプレキャッシングされることができない。また、プレイヤコンテキストの順列空間は、キャッシングされた、繰り返される動きベクトルのような方法によって動きベクトルを予め生成するため、極めて完全であると言える。このようなシステムおよび方法は、米国仮出願第６２／４８８，５２６号、第６２／６３４，４６４号、および第６２／６４０，９４５号に記載されており、このすべては本明細書に統合される。ゲームサーバは、予測された動きベクトルを生成し、プレイヤコンテキストの変更によってクライアント入力の動き補償キャッシュ（ｃａｃｈｅ）をアップデートすることにより、補償が成り立つ。これは、クライアントが制限されたコンテキスト従属入力のセットに対してプレイヤ入力の動き補償手法を使用することを許容し、入力フィードバック待機時間の減少をもたらす。

【０００８】

米国特許第９，６６１，３５１号（‘３５１特許）は、サーバからクライアントデバイスへの送信の間、フレームをスキップ（ｓｋｉｐ）するためのシステムおよび方法を開示している。ここで、スキップされたフレームを検出することに応答し、クライアントデバイスは、圧縮されたビデオストリームでスキップされたフレームの代わりとなる予測されたフレームを生成し、予測されたフレームは、クライアントデバイスによってデコードされた１つ以上の以前フレームからデルタ（ｄｅｌｔａ）情報を拡張させることを基盤に生成される。このような技術に対し、１つ以上の再構成されるか予測されたフレームのクライアント側のフレーム予測は、１つ以上の先行する（ｐｒｅｃｅｄｉｎｇ）プレイのデータ（例えば、動きベクトル、残差（ｒｅｓｉｄｕａｌ）など）を基盤としてスキップされ

たフレームの次に使用される。また、技術は、ビート割り当ておよび／または補助機能（ｓｕｂ　ｆｅａｔｕｒｅ）エンコードを優先視する。エンコードされたネットワーク抽象化層ユニット（Ｅｎｃｏｄｅｄ　Ｎｅｔｗｏｒｋ　Ａｂｓｔｒａｃｔｉｏｎ　Ｌａｙｅｒ　Ｕｎｉｔ：ＮＡＬＵ）は、（１）動きベクトル、および（２）残差に分割される。実際にフレームをスキップする代りに、装置は、優先順位にしたがって最小限のエンコードデータだけを送信することができる。例えば、動きの優先順位が指定されると、動きベクトルだけが送信されるようになる。少なくとも‘３５１特許には、サーバから送信されたルックアップテーブルを、ユーザ入力を動きベクトルとタグ（ｔａｇ）にマッチングさせることに使用し、この動きベクトルを合算するクライアントデバイスは開示されていなため、本発明は‘３５１特許の技術よりも優れていると言える。また、‘３５１特許には、デコードされたフレームにおける動きを推定するために、このフレームに対するこの合算された動きベクトルを適用することは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターン（ｒｅｔｕｒｎ）するのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【０００９】

米国特許第８，６７８，９２９号（‘９２９特許）は、ネットワークの会話型ゲームシステムの動作に関する。開示される方法は、２つのブレンディング（ｂｌｅｎｄｉｎｇ）方式によって共有されたグローバルオブジェクト（ｇｌｏｂａｌ　ｏｂｊｅｃｔ）に対してコース訂正された位置値を決定することにより、ネットワーク遅延を減少させることにフォーカスを合わせている。この特許に記載された「ブレンディング」は、ローカルコンソール（ｌｏｃａｌ　ｃｏｎｓｏｌｅ）からローカルプレイヤのポーズ（ｐｏｓｅ）計算を送信するネットワークシミュレーション（ｎｅｔｗｏｒｋ　ｓｉｍｕｌａｔｉｏｎ）段階を含む。コンソールは、ローカルおよびネットワークシミュレーションをブレンディングする。また、方法は、ローカルコンソールで共有されたグローバルオブジェクトに対する位置値を決定するのにローカルプレイヤのブレンディングされたポーズを使用することにより、共有されたグローバルオブジェクトをブレンディングする。少なくとも‘９２９特許には、サーバから送信されたルックアップテーブルを、ユーザ入力を動きベクトルにマッチングさせることに使用し、この動きベクトルをタギングして合算するクライアントデバイスは開示されていなため、本発明は‘９２９特許の技術よりも優れていると言える。また、‘９２９特許には、デコードされたフレームにおける動きを推定するために、このフレームに対するこの合算された動きベクトルを適用することは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１０】

米国特許第８，０６９，２５８号（‘２５８特許）は、マルチプレイヤシミュレーションのネットワーク支援を明白に減少させるためにローカルフレーム処理を使用することに関する。ここに記載された方法は、ローカルユーザからの入力をインタセプト（ｉｎｔｅｒｃｅｐｔ）する段階、以前のゲームフレームからネットワークシミュレーションのリモートオブジェクトの状態データを決定する段階、およびネットワークシミュレーションの一部である複数のゲームシステムから非決定的（ｎｏｎｄｅｔｅｒｍｉｎｉｓｔｉｃ）オブジェクトの相互作用（ｉｎｔｅｒａｃｔｉｏｎ）を決定する段階を含む。この相互作用データは、状態データおよびローカル入力とともに、ビデオフレームのローカルシミュレーションを生成するのに使用される。このような方式により、ローカルおよびネットワークシミュレーションは、単一フレームに対して非同期的に実行されるようになり、各フレームは、ゲーム内の単一時間フェーズ（ｐｈａｓｅ）に対応する。これは、ローカルシミュレーションがネットワークゲームプレイ（ｇａｍｅｐｌａｙ）中にリアルタイムでアップデートされ、ネットワークと（実質的に）同期化されるように許容する。少なくとも‘２５８特許には、サーバから送信されたルックアップテーブルを、ユーザ入力を動きベク

トルにマッチングさせるのに使用し、この動きベクトルをタギングして合算するクライアントデバイスは開示されていないため、本発明は‘２５８特許の技術よりも優れていると言える。また、‘２５８特許には、デコードされたフレームにおける動きを推定するために、このフレームに対するこの合算された動きベクトルを適用することは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１１】

米国特許第９，６６５，３３４Ｂ２号（‘３３４特許）は、ディスプレイ上の結合するグラフィックスをレンダリングするために複数のプロセスと合成器（ｃｏｍｐｏｓｉｔｏｒ）を適用するプロトコルをレンダリングするためのシステムおよび方法を開示している。この技術は次のように動作する。サーバが同時に複数のクライアントデバイスにゲームスクリーンを提供するとき、サーバにおけるレンダリングプロセッシングの計算負荷（ｌｏａｄ）は、例えば、迅速な応答性を求めるゲームコンテンツにおいて過重となる。すなわち、サーバがスクリーンを提供する複数クライアントデバイスは、サーバのレンダリング性能および求められる応答性によって制限される。これとは対照的に、各クライアントデバイスは、サーバとクライアントデバイスとの間にレンダリングプロセスを共有させるために、一般的なレンダリング性能によって実行されるプロセッシングを実行するように制御されるとき、スクリーンはより多くのクライアントデバイスに提供されることができる。また、一般的に、テクスチャマッピング（ｔｅｘｔｕｒｅ　ｍａｐｐｉｎｇ）を適用しなくても、レンダリングされるゲームスクリーンは高い圧縮効率を有し、インターネットのようなネットワークを介してより小さな帯域幅で送信されることができる。少なくとも‘３３４特許には、予め決定された基準に基づいてサーバで動きベクトルを生成し、生成された動きベクトルと１つ以上のインバリデータ（ｉｎｖａｌｉｄａｔｏｒ）をクライアントに送信し、クライアントがこの動きベクトルとインバリデータをキャッシングすることは開示されていないため、本発明は‘３３４特許に記載された技術よりも優れていると言える。また、‘３３４特許には、サーバがユーザから入力を受信するようにクライアントに指示し、その入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるのに使用し、このベクトルまたはインバリデータが動き補償に使用されることは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるため優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１２】

米国特許第９，７３６，４５４号（‘４５４特許）は、テクスチャブロック（ｔｅｘｔｕｒｅ　ｂｌｏｃｋ）とともに配置された深さブロック（ｄｅｐｔｈ　ｂｌｏｃｋ）の利用可能性を検査し、ともに配置された深さブロックの利用可能性に基づいてテクスチャブロックに対する予測方法を決定し、ともに配置された深さブロックの利用可能性に基づいてテクスチャブロックに対する第１予測ブロック（ｆｉｒｓｔ　ｐｒｅｄｉｃｔｉｏｎ　ｂｌｏｃｋ）を導き出すことを含む、エンコードのためのシステムおよび方法を開示している。少なくとも‘４５４特許には、予め決定された基準に基づいてサーバで動きベクトルを生成し、生成された動きベクトルと１つ以上のインバリデータをクライアントに送信し、クライアントがこの動きベクトルとインバリデータをキャッシングすることは開示されていないため、本発明は‘４５４特許に記載された技術よりも優れていると言える。また、‘４５４特許には、サーバがユーザから入力を受信するようにクライアントに指示し、その入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるのに使用し、このベクトルまたはインバリデータが動き補償に使用されることは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１３】

米国特許第９，７０５，５２６号（‘５２６特許）は、メディア（ｍｅｄｉａ）およびイメージアプリケーションにおけるエントロピ（ｅｎｔｒｏｐｙ）エンコードのためのシステムおよび方法を開示している。その技術は、圧縮が指示されたイメージおよび／またはビデオデータのソースを受信することによって始まるシステムを開示している。また、無損失の（ｌｏｓｓｌｅｓｓ）圧縮方式が適用される。また、予測器（ｐｒｅｄｉｃｔｏｒ）／デルタ計算ユニット（ｄｅｌｔａ　ｃｏｍｐｕｔａｔｉｏｎ　ｕｎｉｔ）が入力を採択し、隣接する入力エレメント（ｅｌｅｍｅｎｔ）間のデルタ計算を使用して入力データの余分（ｒｅｄｕｎｄａｎｃｙ）を減少しようとする。さらに、この値は、圧縮されたイメージおよび／またはビデオデータを生成するために、エントロピエンコーダにおいて予め定義された統計モデリングによってエンコードされる。上述と同じように、少なくとも‘５２６特許には、予め決定された基準に基づいてサーバで動きベクトルを生成し、生成された動きベクトルと１つ以上のインバリデータをクライアントに送信し、クライアントがこの動きベクトルとインバリデータをキャッシングすることは開示されていないため、本発明は‘５２６特許に記載された技術よりも優れていると言える。また、‘５２６特許には、サーバがユーザから入力を受信するようにクライアントに指示し、その入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせることに使用され、このベクトルまたはインバリデータが動き補償に使用されることは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１４】

米国特許第８，８７３，６３６Ｂ２号（‘６２６特許）は、ゲームのローカルインスタンスを実行するユーザＰＣに、コーディングされたイメージデータを提供する動画（ｍｏｖｉｎｇ－ｉｍａｇｅ）配布（ｄｉｓｔｒｉｂｕｔｉｏｎ）サーバ（例えば、オンラインゲームを実行するサーバ）に関する。このプロセスを具体的に実行するために、ユーザＰＣのＣＰＵは、先行するフレームスクリーンで選択されたブロックと関連する動きベクトルをデコードするために参照される領域を指定する。これは、選択されたブロックと関連する動きベクトル（提供されるプレプロセッシング（ｐｒｅ－ｐｒｏｃｅｓｓｉｎｇ）情報に含まれたベクトル）を参照して実行され、領域のイメージを参照イメージで抽出する。他の参考文献と同じように、少なくとも‘６３６特許には、予め決定された基準に基づいてサーバで動きベクトルを生成し、生成された動きベクトルと１つ以上のインバリデータをクライアントに送信し、クライアントがこの動きベクトルとインバリデータをキャッシングすることは開示されていないため、本発明は‘６３６特許に記載された技術よりも優れていると言える。また、‘６３６特許には、サーバがユーザから入力を受信するようにクライアントに指示し、その入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせることに使用し、このベクトルまたはインバリデータが動き補償に使用されることは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１５】

国際公開特許第ＷＯ２００９１３８８７８　Ａ２号（‘８７８公報）は、多様なレンダリングされたオブジェクトの詳細レベルおよびポストフィルタリング（ｐｏｓｔ－ｆｉｌｔｅｒｉｎｇ）を制御しながら、中央集中式ストリーミングアプリケーションサーバで複数の会話型アプリケーションをプロセッシングおよびストリーミングすることに関する。システムにおいて、中央集中式ストリーミングアプリケーションサーバは、このビデオプレプロセッサにおいて圧縮されたストリームをデコードし、コンテンツをディスプレイするクライアントデバイスへのオーディオビジュアル（ａｕｄｉｏ－ｖｉｓｕａｌ）コンテンツの圧縮されたストリームをエンコードする前に、フレームシーケンスに空間および時間フィルタリングを実行する。中央集中式の会話型アプリケーションサーバのＧＰＵコマンド（ｃｏｍｍａｎｄ）プロセッサは、ビデオエンコーダのターゲットエンコードされた

フレーム内の各マクロブロック（ｍａｃｒｏｂｌｏｃｋ）に対して動き補償推定を計算したりするモジュールを含む。これにもかかわらず、少なくとも‘８７８公報には、予め決定された基準に基づいてサーバで動きベクトルを生成し、生成された動きベクトルと１つ以上のインバリデータをクライアントに送信し、クライアントがこの動きベクトルとインバリデータをキャッシングすることは開示されていないため、本発明は‘８７８公報に記載された技術よりも優れていると言える。また、‘８７８公報には、サーバがユーザから入力を受信するようにクライアントに指示し、その入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせることに使用し、このベクトルまたはインバリデータが動き補償に使用されることは開示されていない。さらに、本発明は、入力フィードバック待機時間を減少させるのに優れており、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００１６】

米国特許第９，３５８，４６６Ｂ２号（‘４６６特許）は、キャッシングされたデータを再使用することによってビデオゲーム性能を向上させることに関する。開示されるシステムは、使用されたデジタル資産を識別し、識別されたデジタル資産がキャッシュ（ｃａｃｈｅ）内に存在するか否かを判断することにより、相異して生成されたビデオゲームミッション（ｍｉｓｓｉｏｎ）のキャッシュ性能に少なくとも部分的にスコア（ｓｃｏｒｅ）を付与する。キャッシュスコアは、既にキャッシュに存在するミッションに対するデジタル資産の割合に該当するキャッシュ再使用率に基づいて計算される。キャッシュスコアを生成する他の技術は、既にキャッシュに存在するミッションに対する結合されたデジタル資産の全体サイズ、および／またはまだキャッシュに存在しないミッションに対する結合されたデジタル資産の全体サイズのようなファクタ（ｆａｃｔｏｒ）を説明する。このような方式によってデータをキャッシングすることにより、このデータと異なる、キャッシングされていないデータはより効率的になる。少なくとも‘４６６特許には、繰り返される動きベクトルをキャッシングしたり、入力データから動き推定を計算したり、入力データに基づいて記録された動きベクトルライブラリ（ｍｏｔｉｏｎ ｖｅｃｔｏｒ ｌｉｂｒａｒｙ）をアップデートしたりするため、これにより、クライアントがサーバから実際の動きベクトルデータを受信する前に動きを開始させないために、記録された動きベクトルライブラリを使用することができることは開示されていないため、本発明は‘４６６特許に記載された技術よりも優れていると言える。

【００１７】

米国特許第６，９０３，６６２　Ｂ２号（‘６６２特許）は、構成可能なコンピュータ入力デバイス、特に、迅速な応答時間を維持するために入力データをキャッシングする入力デバイスに関する。システムは、特定のキーが以前に識別されたかを確認するために、内部メモリ（またはキャッシュ）に対してキープッシュを確認することにより動作する。この後、この時点よりも前にシステムがキーと通信していなければ、システムはこのキーメモリから入力に対応するデータを検索する。この後、システムは、その内部メモリ（キャッシュ）をキーアイデンティティ（ｉｄｅｎｔｉｔｙ）とこれに対応するデータとしてアップデートする。この後、システムは、キーからの入力データをホストコンピュータに送信する。しかし、次に、システムがプッシュされた状態の同一のキーを見つければ、キーメモリから同一のスキャンコードを再検索する代りに、自体のメモリ（キャッシュ）を照会する（ｑｕｅｒｙ）。少なくとも‘６６２特許には、繰り返される動きベクトルをキャッシングしたり、入力データから動き推定を計算したり、入力データに基づいて記録された動きベクトルライブラリをアップデートしたりし、これにより、クライアントがサーバから実際の動きベクトルデータを受信する前に動きを開始させるために、記録された動きベクトルライブラリを使用することができることは開示されていないため、本発明は‘６２２特許に記録された技術よりも優れていると言える。

【００１８】

日本特許第ＪＰ６１２９８６５Ｂ２号（‘８６５特許）は、ゲームコンテンツデータが

リアルタイムゲームプレイ中の必要時に利用可能なように、経路（ｐａｔｈ）セグメント（ｓｅｇｍｅｎｔ）のサブセット（ｓｕｂｓｅｔ）に対するレンダリングされたゲームコンテンツデータをローカルキャッシュにキャッシングするための、ゲームクライアントに送信するためのシステムおよび方法を開示している。少なくとも‘８６５特許は、繰り返される動きベクトルをキャッシングしたり、入力データから動き推定を計算したり、入力データに基づいて記録された動きベクトルライブラリをアップデートしたりし、これにより、クライアントがサーバから実際の動きベクトルデータを受信する前に動きを開始させるために、記録された動きベクトルライブラリを使用することができることは開示されていないため、本発明は‘８６５特許に記載された技術よりも優れていると言える。

【００１９】

米国特許第９，７６２，９１９号（‘９１９特許）は、ブロックプロセッシングパイプライン（ｂｌｏｃｋ　ｐｒｏｃｅｓｓｉｎｇ　ｐｉｐｅｌｉｎｅ）において参照データをキャッシングするためのシステムおよび方法を開示している。‘９１９特許技術は、例えば、ＳＲＡＭ（ｓｔａｔｉｃ　ｒａｎｄｏｍ　ａｃｃｅｓｓ　ｍｅｍｏｒｙ）のようなローカル（パイプライン）メモリで実現されることのできるキャッシュメモリを開示しており、パイプラインの初期ステージ（ｓｔａｇｅ）でマクロブロックに対して決定される動きベクトルに対応するクロマ参照データ（ｃｈｒｏｍａ　ｒｅｆｅｒｅｎｃｅ　ｄａｔａ）の部分（例えば、６４バイトのメモリブロック）がメモリからプレフェッチ（ｐｒｅｆｅｔｃｈｅｄ）される。クロマキャッシュロジック（ｃｈｒｏｍａ　ｃｈｃｈｅ　ｌｏｇｉｃ）は、キャッシュを維持することができ、パイプラインの複数のステージに拡張される。パイプラインを通過する与えられたマクロブロックの動きベクトルに対するフェッチ（ｆｅｔｃｈ）は、クロマ動き補償が必要となる前に、メモリからキャッシュに繰り返されるメモリブロックを読み取るため、時間（すなわち、複数のパイプラインサイクル（ｃｙｃｌｅ））を提供するクロマ動き補償ステージの以前の１つ以上のステージでクロマキャッシュロジックによって始めることができる。しかし、‘９１９特許も、本発明に比べると不足である。少なくとも‘９１９特許には、繰り返される動きベクトルをキャッシングしたり、入力データから動き推定を計算したり、入力データに基づいて記録された動きベクトルライブラリをアップデートしたりし、これにより、クライアントがサーバから実際の動きベクトルデータを受信する前に動きを開始させるために、記録された動きベクトルライブラリを使用することができることは開示されていないため、本発明は‘９１９特許に記載された技術よりも優れていると言える。

【００２０】

このような技術における最新技術に関する説明によって明らかであるが、リアルタイムゲーム環境のエンコードと関連する現在のコンピュータ技術に対する改善が、本技術分野において求められている。

【発明の概要】

【発明が解決しようとする課題】

【００２１】

したがって、本発明の目的は、クライアントデバイスがユーザ入力を動きベクトルにマッチングさせるためにサーバから送信されたルックアップテーブルを使用し、この動きベクトルをタギングして合算する、動き補償手法によって待機時間を減少させるためのシステムおよび方法を開示することにある。リモートサーバがクライアントにエンコードされたビデオフレームを送信するとき、クライアントは、このビデオフレームをデコードし、このフレーム内の動きを推定するために合算された動きベクトルをデコードされたフレームに適用する。

【００２２】

本発明の他の目的は、エンコードされたフレームが残差ハンドリング（ｒｅｓｉｄｕａｌ　ｈａｎｄｌｉｎｇ）されることなくデコードされるシステムおよび方法を開示することにある。

【００２３】

本発明のまた他の目的は、クライアントが１つ以上の滑らかな関数（ｓｍｏｏｔｈｉｎｇ　ｆｕｎｃｔｉｏｎ）をキュー内の合算されたタギングされた動きベクトルに適用する、動き補償手法によって待機時間を減少させるためのシステムおよび方法を開示することにある。

【００２４】

本発明のさらに他の目的は、クライアントデバイスにおいて、動きベクトルと関連するタグは、実質的に時系列（ｃｈｒｏｎｏｌｏｇｉｃａｌ）である、動き補償手法によって待機時間を減少させるためのシステムおよび方法を開示することにある。

【００２５】

本発明の目的は、予め決定された基準に基づいてサーバで動きベクトルを生成し、クライアントに生成された動きベクトルと１つ以上のインバリデータ（ｉｎｖａｌｉｄａｔｏｒ）を送信することにより、クライアントがこの動きベクトルとインバリデータをキャッシングする、入力フィードバック待機時間を減少させるためのシステムおよび方法を提供することにある。サーバは、ユーザから入力を受信し、この入力を動きベクトルまたはインバリデータとマッチングさせることに使用するようにクライアントに指示する。また、この比較に基づき、クライアントは、マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース（ｇｒａｐｈｉｃ　ｉｎｔｅｒｆａｃｅ）のエフェクト動き（ｅｆｆｅｃｔ　ｍｏｔｉｏｎ）補償に適用する。

【００２６】

本発明の他の目的は、動きベクトルをルックアップテーブルにキャッシングすることにより、入力フィードバック待機時間を減少させるためのシステムおよび方法を提供することにある。

【００２７】

本発明のまた他の目的は、インバリデータを１つ以上のキャッシングされた動きベクトルと関連させることにより、入力フィードバック待機時間を減少させるためのシステムおよび方法を提供することにある。

【００２８】

本発明のさらに他の目的は、入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するようにクライアントに指示することにより、入力フィードバック待機時間を減少させるためのシステムおよび方法を提供することにある。

【００２９】

本発明の目的は、クライアントに予め生成された動きベクトルライブラリ（ｍｏｔｉｏｎ　ｖｅｃｔｏｒ　ｌｉｂｒａｒｙ）を送信するサーバにおいて、繰り返される（ｒｅｐｅｔｉｔｉｖｅ）動きベクトルをキャッシングすることにより、待機時間を減少させるためのシステムおよび方法を開示することにある。クライアントは、動きベクトルライブラリを記録し、ユーザ入力データをモニタリングする。サーバは、入力データから動き推定を計算するようにクライアントに指示し、入力データに基づいて記録された動きベクトルライブラリをアップデートするようにクライアントに指示する。これにより、クライアントは、サーバから実際の動きベクトルデータを受信する前にグラフィックインタフェースの動きを開始させるために、記録された動きベクトルライブラリを適用する。

【００３０】

本発明の他の目的は、サーバが記録された動きベクトルライブラリの適用をディスエーブルさせるためにコンテキストアップデート（ｃｏｎｔｅｘｔ　ｕｐｄａｔｅ）を送信する、繰り返される動きベクトルをキャッシングすることにより、待機時間を減少させるためのシステムおよび方法を開示することにある。

【００３１】

本発明のさらに他の目的は、１つ以上のスケーリングファクタ（ｓｃａｌｉｎｇ　ｆａｃｔｏｒ）が動きベクトルライブラリに適用される、繰り返される動きベクトルをキャッシングすることにより、待機時間を減少させるためのシステムおよび方法を開示すること

にある。
【００３２】
添付の図面は、後述する詳細な説明を参照することでより適切に理解することができるため、本発明の完全な理解およびこれによる多くの利点を容易に得ることができるであろう。
【図面の簡単な説明】
【００３３】
【図１】ビデオゲームがリモートクライアントの入力によって制御されながらサーバで実行される遠隔ゲームシステムを例示的に示したブロック図である。
【図２】適切なプレイヤ入力に対する動き補償を適用することにより、ゲームストリーミングアプリケーションにおける入力フィードバック待機時間の減少を例示的に説明示したフローチャートである。
【図３】プレイヤ入力の動き補償を利用するビデオゲームストリーミング環境のランタイム中の例示的な瞬間を例示的に示したブロック図である。
【図４】図３のプレイヤ入力の動き補償中のマクロブロックを例示的に示した図である。
【図５】クライアントにおいてプレイヤ入力の動き補償中の動きベクトルを適用するための代案的な方法を例示的に示した図である。
【図６】図６ａ、図６ｂ、および図６ｃは、プレイヤ入力の動き補償および図５に示された代案的な方法に対するブレンディング（ｂｌｅｎｄｉｎｇ）を経る例示的なマクロブロックを示した図である。
【図７】予測された動きベクトルのランタイム生成を示した図である。
【図８】プレイヤ入力の動き補償を目的とする予測された動きベクトルの送信およびクライアント側の記録を例示的に示したフローチャートである。
【図９】予測された動きベクトルを無効にする（ｉｎｖａｌｉｄａｔｉｎｇ）例示的なプロセスを示した図である。
【図１０】動きベクトルライブラリを生成するための例示的な方法およびキャッシング目的のための例示的な繰り返される動きベクトルライブラリを示した図である。
【図１１】プレイヤ入力の動き補償のために動きベクトルライブラリをキャッシング、適用、およびアップデートするためのプロセスを例示的に示したフローチャートである。
【図１２】動きベクトルマッピングがアップデートされるか否かを例示的に示した図である。
【図１３】キャッシングされた動きベクトルの場合に、プレイヤ入力の動き補償中にマルチフレーム動きベクトルを適用するための例示的な修正を示した図である。
【発明を実施するための形態】
【００３４】
図面に示す発明の好ましい実施形態を説明するにあたり、明確化のために、特定の用語に依存することがある。しかし、本発明は、このように選択された特定の用語に限定されることを意図しておらず、それぞれの特定の用語は、類似の目的を達成するために類似の方式で作動するすべての技術的等価物を含むものと理解されなければならない。本発明の好ましい実施形態は、例示の目的として記載されているが、本発明は、図面に具体的に示されていない他の形態によっても実現可能であることが理解されなければならない。
特定の類型のプレイヤ入力は、プレイヤ入力の動き補償のためのより適した候補を生成する。２つのファクタ（ｆａｃｔｏｒ）が与えられた入力の適合性（ｓｕｉｔａｂｉｌｉｔｙ）に寄与する。入力フィードバック待機時間に対するプレイヤ敏感度（ｓｅｎｓｉｔｉｖｉｔｙ）および相当なアーティファクト（ａｒｔｉｆａｃｔ）を導入せずにプレイヤ入力の動き補償を実現するには困難がある。各入力は、適合性に対して評価される必要がある。例えば、１人称シューティングゲーム（ｓｈｏｏｔｅｒ　ｇａｍｅ）において、プレイヤは、マウス基盤（ｍｏｕｓｅ－ｂａｓｅｄ）のビュー回転に極めて敏感であり、プレイヤ入力からビデオ出力まで約１６ｍｓという１秒にもならないディレイ（ｄｅｌａｙ）まで認識されるであろう。しかし、同じような状況において、ゲームパッド（ｇａｍｅ

ｐａｄ）コントローラ基盤のビュー回転は、一般的にこれよりは遅く、プレイヤは、入力フィードバック待機時間にそれほど敏感でない。ビュー回転は、回転の反対方向にシーン（ｓｃｅｎｅ）をシフトさせることによって同じようにすることができるが、回転方向にイメージのエッジ（ｅｄｇｅ）に沿って好ましくないアーティファクトが存在することがある。スクリーン上の照準（ａｉｍ）を調節することのような小さなビュー回転に対し、プレイヤは、エッジアーティファクトを認識できないこともある。他の例において、ドライビングゲーム（ｄｒｉｖｉｎｇ　ｇａｍｅ）において、加速する車は、プレイヤの敏感性および／または待機時間に対する慣性（ｉｎｅｒｔｉａ）の不足によりプレイヤ入力の動き補償に対して低い優先順位となるが、ステアリング（ｓｔｅｅｒｉｎｇ）およびブレーキング（ｂｒａｋｉｎｇ）入力は、プレイヤが入力フィードバック待機時間を認識するはずであるため、高い優先順位となる。

【００３５】

プレイヤ入力の受信から動き出力がディスプレイされるまでの時間がプレイヤフィードバック待機時間となる。動き補償を使用することによって推定された動きは、プレイヤ入力を処理するようにサーバを待ちながら、ほぼ即時にフィードバックを提供することができる。このような方式により、プレイヤフィードバック待機時間は、ゲームストリーミングアプリケーションにおいて大きく減少される。ゲームストリーミングアプリケーションにおいてプレイヤ入力の動き補償を実現することにより、動き推定は、次の利用可能なフレームで提供されるようになる。これとは対照的に、入力がサーバに移動し、フレームを生成し、リターンされるには、複数のフレームを経るようになる。また、プレイヤ入力の動き補償は、ゲームエンジンおよびレンダラ（ｒｅｎｄｅｒｅｒ）が複数のフレームのプレイヤフィードバック待機時間を有する既存のストリーミングされない（ｎｏｎ－ｓｔｒｅａｍｅｄ）ゲームアプリケーションにおいて利点を提供することができる。

【００３６】

クライアントは、スクリーン周辺のオブジェクトの移動を追跡するための適切なコンテキストを有さないであろう。プレイヤ入力の動き補償は、特定のマクロブロックまたはビデオオブジェクトの位置をクライアントが認知しない場合には適さない。例えば、二次元プラットフォームゲーム（ｐｌａｔｆｏｒｍｅｒ　ｇａｍｅ）において、キャラクタ（ｃｈａｒａｃｔｅｒ）は、左側から右側にスクリーン周辺を移動する。クライアントは、プレイヤがジャンプを入力（ｊｕｍｐ　ｉｎｐｕｔ）したとき、キャラクタがどこに位置するか認知することができない。したがって、このような場合に入力フィードバック待機時間を減少させるために、プレイヤ入力の動き補償だけが使用されることはない。

【００３７】

一般的に、プレイヤ入力の動き補償のための動きベクトルは、予め生成されなければならない。プレイヤのカメラ回転のような動きに対し、動きベクトルは、ゲームが入力にどのように加重値を付与するかに基づいて計算されてよい。一例として、動きベクトルは、入力値に感度重み（ｓｅｎｓｉｔｉｖｉｔｙ　ｗｅｉｇｈｔ）を掛けたものであってよい。アニメーションの移動のような直接に計算することのできない動きに対し、動きが直接に測定されて記録されるように、アニメーションは、デベロップメント（ｄｅｖｅｌｏｐｍｅｎｔ）中にトリガーされてよい。動きベクトルを測定することは、Ｈ．２６４エンコード中に実行される同一の動き推定技術によって達成されてよい。

【００３８】

図１は、ビデオゲームがリモートクライアントによって制御される例示的なシステムを示した図である。このシステムにおいて、サーバ１００は、ビデオゲームソフトウェア１２０およびビデオ出力をレンダリングするグラフィックスエンジン（ｇｒａｐｈｉｃｓ　ｅｎｇｉｎｅ）１０４をホスティングする。ビデオは、コーデック（コーディングエンジンまたはエンコーダとも呼ばれる）１０６でエンコードされ、エンコードされたビデオデータ１０８はリモートクライアントに送信される。サーバアキテクチャ（ｓｅｒｖｅｒ　ａｒｃｈｉｔｅｃｔｕｒｅ）１００は、グラフィックスエンジン１０４とコーデック１０６の両方の機能を支援することのできるハードウェアまたはソフトウェアのいずれかの組

み合わせであってよい。図に示す例において、グラフィックスエンジン１０４は、例えば、複数のコンピュータ読み取り可能なメモリ１１２にロードされたビデオゲームソフトウェア１２０によって制御されるＧＰＵ１１０で実現されてよく、この反面、コーデック１０６は、ビデオエンコードソフトウェアを実行するＣＰＵ１１４で実現されてよい。

【００３９】

リモートクライアントは、送信されたエンコードされたビデオデータ１０８をデコードするためのコーデック１１８と、プレイヤ入力の動き補償を適用するためのクライアントアプリケーション１２０を実行することのできるコンピュータシステム１１６で構成される。また、クライアントコンピュータシステム１１６は、ディスプレイハードウェア１２４を実行させるためのディスプレイコントローラ１２２を含む。クライアント側の入力周辺機器（ｐｅｒｉｐｈｅｒａｌ）１２６からの入力は、クライアントアプリケーション１２０により、サーバ１００から実行されるゲームソフトウェア１０２に再送信されるコントロールデータ１２８として変換される。また、周辺機器１２６からの入力は、図２でより詳細に示すように、適用するプレイヤ入力の動き補償があれば、どのようなものであるかを決定するのに使用されるであろう。

【００４０】

図２は、単一（ｓｉｎｇｌｅ）入力に対するプレイヤ入力の動き補償を実行するのに求められる段階を説明するためのフローチャートである。クライアントが初期化するとき、段階２００において、サーバは、入力と、これと関連する動きベクトルを含むルックアップテーブルを送信する。この後、段階２０２において、ルックアップテーブルは、クライアントによってキャッシングされる。このような実施形態において、クライアントは、複数のストリーミングゲームの要求を満たすように一般化される。特定の実施形態において、特定のクライアントは、ゲームに対するルックアップテーブルを既に有していることがあるため、段階２００および２０２はスキップされてよい。代案的な実現において、ゲームの特定のクライアントは、サーバからキャッシングされることなく、動きベクトルルックアップテーブルを永久的に記録することも可能である。

【００４１】

段階２０４において、クライアントがマウス（ｍｏｕｓｅ）またはゲームパッド（ｇａｍｅｐａｄ）コントローラのような入力デバイスからプレイヤ入力を受信すると、段階２０６において、クライアントアプリケーションは、キャッシングされた動きベクトルルックアップテーブルからマッチングされる入力を確認する。マッチングされるプレイヤ入力がなければ、クライアントは、追加のアクション（ａｃｔｉｏｎ）は取らずに、追加の修正なくサーバに入力を送信する。キャッシュ（ｃａｃｈｅ）にマッチングされるプレイヤ入力があれば、クライアントは、プレイヤ入力の動き補償を適用する。選択的に、キャッシュは、プレイヤ入力に基づいてルックアップテーブルでエントリ（ｅｎｔｒｙ）を変更してよい。例えば、プレイヤが一時停止ボタン（ｐａｕｓｅ　ｂｕｔｔｏｎ）を押すと、プレイヤが一時停止スクリーンを終えるまで、すべてのプレイヤ移動入力がディセーブル（ｄｉｓａｂｌｅ）されなければならない。一実施形態において、クライアント制御ルックアップテーブルは、好ましくはクライアントにより、プレイヤが一時停止入力を選択するたびに転換される、すなわち、一時停止メニューにおける使用のための１つと、一時停止メニュー以外における使用のための１つである、２つのセットの入力を有するようになる。代案的な実施形態において、サーバは、クライアントのキャッシングされたルックアップテーブルのコンテンツを転換してよい。

【００４２】

クライアントが動き補償のためのプレイヤ入力を受信すると、段階２０８において、クライアントは、プレイヤ入力と、これと関連する動きベクトルにタグ（ｔａｇ）を追加する。段階２１０において、タギングされた入力は、サーバに送信される。タグは、プレイヤ入力を未来（ｆｕｔｕｒｅ）フィードバックと関連させることのできる識別子（ｉｄｅｎｔｉｆｉｅｒ）である。例えば、タグは、クライアントがプレイヤ入力の動き補償を実行するのに使用される入力を受信するたびに増加する定数（ｉｎｔｅｇｅｒ）であってよ

い。タグは、プレイヤ入力と同一のネットワークパケット（ｎｅｔｗｏｒｋ　ｐａｃｋｅｔ）内にメタデータ（ｍｅｔａｄａｔａ）として追加されるか、入力情報と同期化されたタグ情報を維持させる類似のメッセージングパターン内で送信されてよい。段階２１３において、クライアントは、タギングされたタグの受信を確認する。プレイヤ入力がタギングされて送信される間、段階２１２において、クライアントは、キャッシングされたルックアップテーブルに含まれた動きベクトルを適用する。このような動きベクトルは、相関タグ（ｃｏｒｒｅｌａｔｉｏｎ　ｔａｇ）がサーバからリターンされるまで、各受信フレーム（ｉｎｃｏｍｉｎｇ　ｆｒａｍｅ）に対して適用される。このような動きベクトルを適用するための方法の詳細な説明は、図３に示すとおりである。

【００４３】

サーバがタギングされたプレイヤ入力を受信すると、段階２１４において、タギングされたプレイヤ入力は、出力フレームを生成するゲームに伝達される。この後、段階２１６において、ビデオイメージがエンコードされる。エンコードされたフレームがクライアントに再送信される前に、段階２１８において、プレイヤ入力タグがエンコードされたフレームに付与される（ａｔｔａｃｈｅｄ）。これは、プレイヤ入力とともに既に送信されたものと同一のタグであり、出力フレームがプレイヤ入力からの実際のビデオフィードバックを含むことを意味する。エンコードされたフレームにタグを付与することは、タグをメタデータとしてエンコードされたフレームと同一のネットワークパケットに追加することによって達成されてよい。段階２２０において、タギングされたエンコードされたフレームは、クライアントに再送信される。クライアントがエンコードされたフレームをタグとともに受信すると、クライアントは、そのタグを以前のプレイヤ入力の動き補償に関連させてよい。この後、段階２２２において、クライアントは、以前の動き補償を適用することを中断する。

【００４４】

図３は、プレイヤ入力の動き補償を利用するビデオゲームストリーミング環境のランタイム中の例示的な瞬間を示した図である。段階３００において、クライアントがプレイヤ入力を受信すると、段階３０２において、プレイヤ入力は、動きベクトルルックアップテーブルと比較される。マッチングするプレイヤ入力があれば、関連する動きベクトルがプレイヤ入力の動き補償のために使用される。段階３０４の固有の入力タグとして、段階３０６において、動きベクトルがタギングされる。このような例において、選択されたタグは、定数「１００３」である。段階３０８において、タギングされた動きベクトルは、プレイヤ入力の動き補償のために現在使用されている他のタギングされた動きベクトルを含むキュー（ｑｕｅｕｅ）に追加される。

【００４５】

段階３２０において、次のフレームは、サーバからのビットストリーム（ｂｉｔｓｔｒｅａｍ）に到達する。段階３１８において、このフレームは、固有の識別子、このような場合は定数「１００１」でタギングされるが、これは、フレームがタグ「１００１」に対応する入力を含んですべての以前プレイヤ入力の結果動きを含むことを示す。段階３２２において、タグ「１００１」は、このようにタギングされた入力に対して動き補償を適用することを中断できることをクライアントに指示する。この後、段階３０８において、タグ「１００１」を有する動きベクトルは、以前のパケットが損失された場合に、キューに残っている以前のタグを有する動きベクトルとともに、タギングされた動きベクトルキューから除去される。

【００４６】

段階３２４において、コーディングされたビデオ３１６は、デコードされる。一方、段階３１０において、段階３０８において、動きベクトルキュー内の残り動きのベクトルが合算される。動きベクトルは、一般的に、イメージの各マクロブロックに対するベクトルを有するベクトルフィールド（ｖｅｃｔｏｒ　ｆｉｅｌｄ）である。動きベクトルを合算するために、その結果が各マクロブロックに対するベクトルを有するベクトルフィールドになるように、ベクトルはエレメント単位（ｅｌｅｍｅｎｔ－ｗｉｓｅ）で合算される。

２つのベクトルフィールドの合計は、フィールドの各ポイント（ｐｏｉｎｔ）に対するベクトルの合計であり、２つのセットの動きベクトルの合計は、イメージの各マクロブロックに対するベクトルの合計である。２つのベクトルの合計は、この成分（ｃｏｍｐｏｎｅｎｔ）の成分単位の合計によって定義され、これは｛u_1、u_2｝＋｛v_1、v_2｝＝｛u_1＋v_1、u_2＋v_2｝のように表現されてよい。このような例において、タグ「１００２」および「１００３」を有する２つのセットの動きベクトルは、キューに含まれる。このような２つのセットの動きベクトルは合算される。タグは実質的に時系列（ｃｈｒｏｎｏｌｏｇｉｃａｌ）であり、これは、以前のタギングされたプレイヤ入力の順序をユーザに認知させる。これは、受信フレームのリターンタグを含むことにより、ユーザがタギングされた動きベクトルを捨てることを許容する。また、上述されたタギングは、さらに複雑な方法よりも計算上において安価である。クランピングアーティファクト（ｃｌａｍｐｉｎｇ　ａｒｔｉｆａｃｔ）を防いだり、類似の導入されるアーティファクトを緩和させたりするために、選択的な滑らかな関数（ｓｍｏｏｔｈｉｎｇ　ｆｕｎｃｔｉｏｎ）がこの時点に適用されてよい。

【００４７】

クランピングアーティファクトは、マクロブロックがオフスクリーン（ｏｆｆｓｃｒｅｅｎ）に移動し、スクリーンの端に垂直のピクセルカラーの滲みとして示されるときに、導入される。滑らかな関数の一例として、外向（ｏｕｔｗａｒｄ－ｐｏｉｎｔｉｎｇ）動きベクトルがイメージのエッジに近くなるほど、この強度（ｓｔｒｅｎｇｔｈ）を減少させるであろう。外向成分、すなわち、イメージの上側端および下側端に向かう外向ベクトルのｙ成分およびイメージの左側端および右側端に向かう外向ベクトルのｘ－成分だけを弱化させる必要がある。境界（ｂｏｒｄｅｒ）に向かうベクトルに対し、境界からの距離が０に近くなるとき、ベクトル成分が０に近くなるように、ベクトル成分はエッジからの距離の二乗に掛けられてよい。このような例において、イメージの右側端に向かう外向ベクトルは｛ｘ、ｙ｝から｛ｘ×ｄ×ｄ、ｙ｝に変換されるはずであり、ここで、ｄは、エッジからの距離である。これは、イメージの境界周辺に少しのイメージ歪曲が発生する代りに、クランピングアーティファクトを緩和させるであろう。歪曲は、クランピングアーティファクトよりもプレイヤに認識され難い。

【００４８】

段階３２４において、デコードプロセスがエンコードされたビデオフレームで完了した後、段階３１２において、合算された動きベクトルが動き補償に使用される。段階３１４において、結果ビデオが出力される。この出力はプレイヤ入力の動き補償データを含み、クライアントにディスプレイされるであろう。この出力されたフレームは、相関タグ「１００３」の入力に対する動き補償を含む１番目のフレームである。また、この出力されたフレームは、クライアントがサーバから実際の動きベクトルがリターンされるのを待つ相関タグ「１００２」の以前の入力に対する動き推定を含む。また、この出力されたフレームは、クライアントが既に動きを推定した相関タグ「１００１」の以前の入力に対する実際の動きベクトルを含む１番目のフレームである。結果的に、３種類の動き推定状態が、この方法のこの時点に存在するようになる（新たな入力に対する新たな推定状態、実際の結果を待ち続ける推定状態、および実際の結果がクライアントに到達することによって中断された他の推定状態）。

【００４９】

図４は、図３のプレイヤ入力の動き補償中の例示的なマクロブロックを示した図である。プレイヤ入力の動き補償段階は、現在フレームのマクロブロックが以前フレームのどこから移動されたものであるかを決定するのにデコードされた動きベクトルが使用されるＨ．２６４デコード中に実行されたプロセスと、その手順が類似する。プレイヤ入力の動き補償に対し、適用された動きベクトルは、プレイヤに出力ビデオをディスプレイする前に現在フレームのマクロブロックを移動させることにより、プレイヤ入力に対する未来フィードバックを推定するであろう。図４ａは、図３の段階３０８で示した例示的なタギングされた動きベクトルからの、段階４００でキュー内の２つのセットのタギングされた動き

ベクトルを示している。このような２つのセットは、図４ｂの段階４０２において一セットの動きベクトルを生成するために、イメージの各マクロブロックに対するベクトルの合計を取ることによって合算される。このセット内において、動きベクトルそれぞれは、現在フレームの各マクロブロックに対する推定された移動を示す。段階４０２で合算された動きベクトルの各ベクトルに対し、図４ｃの例示的なイメージの対応するマクロブロックはシフト（ｓｈｉｆｔ）されるであろう。一例として、マクロブロックが段階４０４で対応するプレイヤ入力の動きベクトルによってシフトされるが、これは図４ｃに示されている。例示的なイメージの各マクロブロックは、このような方式によってシフトされるであろう。図４ｃの例示的なイメージは、５０個のマクロブロックだけを含むが、高画質イメージは数千個のマクロブロックを含むであろう。図４に示した例示的な動きベクトルは、均一な剛体運動（ｒｉｇｉｄ　ｍｏｔｉｏｎ）を示すが、説明されたプレイヤ入力動き補償手法は、回転、振動、または該当の技術分野において周知のスクリーン空間における他の複雑な動きを説明するために、任意の複雑度（ｃｏｍｐｌｅｘｉｔｙ）の動きベクトルとともに使用されてよい。

【００５０】

図５は、クライアントでプレイヤ入力の動き補償中に動きベクトルを適用するための代案的な方法を示した図である。この方法の特定の実施形態において、ビデオのエンコードおよびデコード間の残差（ｒｅｓｉｄｕａｌ）をハンドリングする必要はない。マッチングするプレイヤ入力がルックアップテーブルから探索された後、段階５００において関連する動きベクトルがタギングされ、図２の段階２０８および２１２に示すように、プレイヤ入力の動き補償のために使用される。次のフレームにおけるデコードプロセスの間、プレイヤ入力の動きベクトルはデコードされた動きベクトルに追加され、Ｈ．２６４コーディング標準によって定義された、段階５０２の動き補償段階に追加される。段階５０４において、Ｈ．２６４コーディング標準によって定義されたようにデブロッキングフィルタ（ｄｅｂｌｏｃｋｉｎｇ　ｆｉｌｔｅｒ）が適用されるが、プレイヤ入力の動き補償によって修正されなかったブロックだけに適用されなければならない。段階５０６に示した結果出力は、プレイヤ入力の動き補償を含み、クライアントにディスプレイされるであろう。次に、段階５０６と段階５１８において、出力はバッファリングされ、次のフレームで動き補償に使用するための以前イメージとなる。図３の実施形態とは異なり、この実施形態において、タギングされたプレイヤ入力動きベクトルは１度だけ適用される。これは、段階５１８に示された以前イメージは、以前のタギングされた動きベクトルの合計を含むため、タギングされたプレイヤ入力の動きベクトルがキュー内のすべてのタギングされた動きベクトルと合算される必要がないことを意味する。入力の受信からビデオ出力のディスプレイまでの時間が入力フィードバック待機時間となり、これは、サーバが出力ビデオをリターンするのを待つ代りに、プレイヤ入力の動き補償を使用することによって大きく減少される。

【００５１】

段階５００と同時に、段階２０８および２１０に示すように、クライアントは、対応するタギングされたプレイヤ入力をサーバに送信する。図２の段階２２０に示すように、クライアントは、段階５０８に示すようなコーディングされたビデオを、同一のタグ５１０とともに、段階５１２のサーバからのビットストリームで受信する。ビデオと関連するタグ５１０は、プレイヤ入力の動き補償によって既に推定された実際の動きが、段階５０８に示すようなエンコードされたビデオで表現されるということを意味する。エンコードされたビデオは、Ｈ．２６４コーディング標準によって定義されたように、段階５１４のエントロピデコード（ｅｎｔｒｏｐｙ　ｄｅｃｏｄｉｎｇ）を通過する。段階５１６の逆量子化（ｉｎｖｅｒｓｅ　ｑｕａｎｔｉｚａｔｉｏｎ）および逆変換（ｉｎｖｅｒｓｅｔｒａｎｓｆｏｒｍａｔｉｏｎ）を通過する。

【００５２】

上述した段階５０８、５１０、５１２、５１４、および５１６以前に実行された以前プレイヤ入力の動き補償段階は、エンコードされたフレームからの実際の動きベクトルがシ

フトされようとするマクロブロックを既にシフトさせた。したがって、実際の動きベクトルを直接に適用することは、誤ったマクロブロックをシフトさせることに繋がることになる。ビットストリームは、２つの関連するデータ（エンコードされたビデオフレームおよび相関タグ）ピース（ｐｉｅｃｅ）を含むであろう。エンコードされたフレームは、段階５１４のエントロピデコードに送信される反面、タグは、段階５２０のブレンディングプロセスに送信される。補償のために、段階５２０のブレンディングプロセスは、マッチングされた相関タグを有し、追加される実際の動きベクトルと以前に使用されたプレイヤ入力の動きベクトル５０との間の差を計算することにより、訂正（ｃｏｒｒｅｃｔｉｏｎ）動きベクトルを供給する。その差は、実際の動きベクトルに相関タグに対する逆動きベクトルを追加することによって計算されてよいが、これは図６を参照しながらさらに詳しく説明する。訂正ベクトルは、段階５０２の動き補償のためにデコードされた動きベクトルに代わって使用される。デコードパイプライン（ｄｅｃｏｄｉｎｇ　ｐｉｐｅｌｉｎｅ）の残りは、段階５０４のデブロッキングフィルタ、段階５０６の次の出力ビデオフレーム、および段階５１８の出力バッファリングを通常とおりに続ける。このような段階は、各フレームで継続して繰り返される。一般的に、段階５２０のブレンディングプロセスは、逆量子化（段階５１４）および逆変換（段階５１６）後に発生するが、これは、実際の動きベクトルがこの時点まで利用可能でないためである。実際の動きベクトルが利用可能になれば、ブレンディングプロセスはこれを使用し、実際のものと推定された動きベクトルとの差を計算するであろう。

【００５３】

図６は、プレイヤ入力の動き補償およびブレンディングを経る例示的なマクロブロックを示した図である。０ｍｓ時間に、プレイヤは、左側にカメラビューを回転させるために入力をする。ルックアップテーブルから「ＰＩＭＣ」６００で示される関連するプレイヤ入力の動き補償動きベクトルが、次のフレームのすべてのマクロブロックに適用される。図６ａは、右側にシフトされる例示的なマクロブロックを示している。プレイヤ入力の動き補償の結果は、次のデコードされたフレームに適用され、１秒あたり６０フレームで実行されるビデオに対して１６ｍｓ（一フレームの長さ）の最大入力フィードバック待機時間を発生させる。他の実施形態において、最大入力フィードバック待機時間は、１秒あたり３０フレームで実行されるビデオに対して３３ｍｓとなることがある。したがって、本発明は、多様なフレームレート（ｆｒａｍｅｒａｔｅ）に動作され、最大入力フィードバック待機時間を一フレームの長さに制限するものと考えられる。タギングされたビデオがサーバからリターンされるとき、これは、図６ｂに示すように、サーバによってエンコードされた実際の動きベクトル６０２を含む。このような例に対し、コーディングされたビデオは１００ｍｓ時間にリターンされるが、これは、ネットワーク待機時間によって大きく異なるであろう。実際のベクトル６０２は、プレイヤ入力の動き補償６００で既にシフトされたマクロブロックを参照するため、実際のベクトル６０２は、既存のフレームに直接適用されることができない。この代わりとなる、訂正ベクトル６０４が、実際のベクトル６０２とプレイヤ入力の動きベクトル６００との差を求めることによって計算される必要がある。この差は、プレイヤ入力の逆動きベクトルを実際の動きベクトルに追加することによって計算されてよい。このようなベクトルの差を求めることは、図５の段階５２４のブレンディングによって参照される。訂正ベクトル６０４が小さいほど、プレイヤ入力の動き補償方法が実際の動きベクトルを推定するのにより成功的であった。例示的なマクロブロックに対する図６ｃに示された結果動き６０８は、実際の動きベクトル６０２と同一である。このような例は、プレイヤ入力の動き補償がプレイヤ入力に対するビデオフィードバックをどのように推定し、１６ｍｓと１００ｍｓとの間の時間に対する結果をどのように示すかを示している反面、修正されていないシステムは、プレイヤ入力が受信された後、１１６ｍｓまでプレイヤフィードバックをディスプレイしない。

【００５４】

デベロップメント（ｄｅｖｅｌｏｐｍｅｎｔ）の間、ゲーム開発者（ｄｅｖｅｌｏｐｅｒ）は、どのような動きとアニメーションがランタイム間に予測された動きベクトルを送

信するかを決定する必要がある。固有でありながらも予測可能な動きベクトルは、予測された動きベクトルに対する最適な候補である。範囲型の例では、ジョイント角度（ｊｏｉｎｔ　ａｎｇｌｅ）を計算するのに運動学方程式（ｋｉｎｅｍａｔｉｃｓ　ｅｑｕａｔｉｏｎ）を使用するアニメーション、時間歪曲されたアニメーション、またはその他に増加するか圧縮されたアニメーションのような、エンジンによって適応的に変更されるアニメーションを含むであろう。例えば、プレイヤが提議されたレッジ（ｌｅｄｇｅ）内に入ってジャンプするとき、レッジグラブ（ｌｅｄｇｅ　ｇｒａｂ）アニメーションが再生される。レッジグラブアニメーションは、プレイヤの手はレッジの上にあるが、依然としてプレイヤの体に付いているように、増える。アニメーションは、定義された数のフレームで再生され、レッジの上にプレイヤを置く。このような例の開始点は、許容可能な位置と方向の範囲内で可変的である。正確なアニメーションは予め知らされないが、注文型（ｏｎ－ｄｅｍａｎｄ）ゲームエンジンによってプログラミング方式で生成されるため、このようなレッジグラブアニメーションは、予測された動きベクトルを生成するのに適した候補である。特に、クライアントは、ゲームストリーミング環境におけるプレイヤ位置に対する状況情報を有さないため、クライアントは、このアニメーションに対する動きベクトルを認知することができない。

【００５５】

予測された動きベクトルは、特定のカメラ位置、小さな時間ウィンドウ、または複数の他の特定プレイヤコンテキストのような制限された状況または時間範囲に対してのみ有用である。各セットの予測された動きベクトルに対し、対応するインバリデータ（ｉｎｖａｌｉｄａｔｏｒ）が生成される必要がある。インバリデータは、クライアントにおいて、予測された動きベクトルが有効となった後に適用されることを防ぐために使用されてよい。特定の実施形態において、インバリデータは、ゲームコンテキストを変更するプレイヤ入力のセットであってよく、したがって、予測された動きベクトルを再生することは、これ以上可能にならない。他の実施形態において、インバリデータは、予測された動きベクトルが有効となる時間ウィンドウであってよい。また他の実施形態において、インバリデータは、無効化（ｉｎｖａｌｉｄａｔｉｎｇ）入力および時間ウィンドウの組み合わせであってよい。例えば、レッジグラブアニメーションに対して生成された予測された動きベクトルは、制限されたプレイヤ位置と方向内でのみ有効であり、これにより、インバリデータは、並進または回転移動入力を必須で含むようになるであろう。インバリデータは、予測された動きベクトル特徴のデベロップメント中に設計されて実現される必要がある。さらに、予測された動きベクトルは、動き補償のためにキャッシングされた、繰り返される動きベクトルを使用することを説明する図１０～図１３を参照しながら後述するように、サーバから送信されたイベント（ｅｖｅｎｔ）やメッセージによってディセーブルされるかアップデートされてよい。

【００５６】

予測された動きベクトルは、予め生成されてもよいし、ランタイム中に必要に応じて生成されてもよい。例えば、制限された数の順列を有するアニメーションに対し、予測された動きベクトルは、各順列をトリガーし、動きベクトルを記録することによってオフライン（ｏｆｆｌｉｎｅ）で生成されてよい。特定の実施形態において、一般化されたクライアントに対し、動きベクトルは、サーバ側に記録された後、注文型でキャッシングされるようにクライアントに送信されるであろう。動きベクトルがクライアントに送信されるとき、これは、ルックアップテーブルにキャッシングされる。事前生成された予測された動きベクトルは、サーバに記録され、ゲームのランタイム中にクライアントのルックアップテーブル内にキャッシングされるように、サーバが事前生成された動きベクトルを送信することのできるゲーム読み取り可能なファイル形式で利用可能となる。逆運動学によって計算されたアニメーションのように、ランタイム中に生成されたアニメーションは、個別の数の可能なアニメーション順列がないことがあるため、事前生成されないこともある。逆運動学は、アニメーションを一セットの境界条件内に合わせのために、リアルタイムレンダリングに一般的に使用される方法である。例えば、テレビゲーム内のプレイヤキャラ

クタは、近くのレッジを、グラブ（ｇｒａｂ）を要求し、境界条件は、プレイヤの手がレッジに触れる位置によって定義されるはずであり、レッジグラブアニメーションは、逆運動学によって変更されるであろう。このように適応的に変更されるアニメーションに対し、ゲームは、ランタイム中の可能なアニメーションをオフスクリーン動きベクトルイメージに推論的にレンダリングし、必要に応じて予測された動きベクトルを記録してよい。例えば、プレイヤが掴むことのできるレッジの近くにいれば、ゲームは、プレイヤが直ぐにレッジを掴むものと予想することができ、ゲームは、予測された動きベクトルを生成するようにレッジグラブアニメーションを推論的にレンダリングしてよい。ランタイム中に予測された動きベクトルが生成される必要のある適応的に変更されたアニメーションは、開発者（ｄｅｖｅｌｏｐｅｒ）によって予め識別される必要がある。

【００５７】

プレイヤ位置追跡、スクリプティング（ｓｃｒｉｐｔｉｎｇ）システム、トリガーボリューム、または経路探索（ｐａｔｈｆｉｎｄｉｎｇ）システムのような、プレイヤコンテキストを説明する従来のゲームシステムは、ゲームがアニメーションを推論的にレンダリングする必要があるときに、信号を送信するイベントを生成するのに使用されてよい。例えば、ゲームは、掴むことのできるレッジに対するプレイヤの接近度（ｐｒｏｘｉｍｉｔｙ）を追跡し、レッジグラブアニメーションを推論的にレンダリングするようにゲームに信号を送信し、予測された動きベクトルを記録してよい。武器をピックアップしたり、レバーを上げたり、ボタンを押したりするような特定のアニメーションは、相互作用およびプレイヤ方向に対し、プレイヤ接近度に基づいて増やしたり調節されたりしてよい。このようなアニメーションは、事前生成を可能にするには極めて多くの順列を有するが、図７に示すように、ランタイム中に生成されてもよい。常に同じ方式によって発生する動きは、オフラインで生成されて記録されることもある。これらは、一般的に、トリガーされるたびに同一の割合で同じスクリーン空間に発生する動きである。このようなアニメーションに対する動きベクトルは、アニメーションのすべての可能な順列をトリガーし、ゲームで生成された動きベクトルを記録するものであっても、Ｈ．２６４コーデックで使用されるもののような従来の動き推定技術によって動きベクトルを生成することにより、オフラインで記録されるものであってもよい。好ましい実施形態において、図１～図６を参照しながら説明したようなゲームで生成された動きベクトルは、高品質の動き推定を保障するために使用される。このようなプロセスは、デベロップメント中にいつでも発生されるが、ビルドプロセス（ｂｕｉｌｄ　ｐｒｏｃｅｓｓ）またはミップマップ（ｍｉｐ－ｍａｐ）とディテール水準（ｌｅｖｅｌ　ｏｆ　ｄｅｔａｉｌｓ：ＬＯＤｓ）を事前生成することのような他の従来のアセットコンディショニング（ａｓｓｅｔ－ｃｏｎｄｉｔｉｏｎｉｎｇ）プロセスの１つの段階として、このプロセスを追加することが好ましい。アセットコンディショニングは、ゲームアセット（ｇａｍｅ　ａｓｓｅｔ）を人間で読み取り可能なソース形式から機械で読み取り可能形式にコンパイル（ｃｏｍｐｉｌｅ）するプロセスを含んでよい。例えば、ミップマップは、アーティスト生成されたテクスチャファイルを複数の解像度を含むゲームサポート形式に変換することにより、予め生成されてよい。これと同じように、ディテール水準は、アーティスト生成されたモデルファイルを複数のディテール水準を含むゲームサポート形式に変換することにより、予め生成されてよい。動きベクトル生成は、アーティスト生成されたアニメーション形式をゲームサポート形式に変換する従来のアセットコンディショニングプロセスに追加されてよい。

【００５８】

図７は、オフラインまたはランタイムシナリオで動きベクトルを生成するための例示的な方法を示した図である。アニメーションは、段階７００のオフスクリーン表面／イメージとしてレンダリングされてよく、動きベクトルは、即時的な使用のために記録されてよい。移動するスクリーンの部分だけがレンダリングされる必要があり、シーン（ｓｃｅｎｅ）の他のオブジェクトは無視されてよい。段階７０２に示される点線のオブジェクトと、段階７０４に示される実線のオブジェクトは、以前フレームと現在フレームそれぞれにおけるアニメーションオブジェクトの位置を示す。以前フレームから現在フレームへの移

動は、段階７０６において、段階７０８に示される動きベクトルの形式でキャプチャされるであろう。動きベクトルは、ゲームで生成された動きベクトルからキャプチャされてもよいし、Ｈ．２６４コーデックで使用されるもののように従来の動き推定技術によってキャプチャされてもよい。好ましい実施形態において、図１～図６を参照しながら説明したように、ゲームで生成された動きベクトルは、高品質の動きベクトルを保障するのに使用される。動きベクトルが必要なすべてのフレームに対して生成されるまで、段階７００～７０８に示されたプロセスを繰り返すことにより、与えられたアニメーションに対して多くのフレームの予測された動きベクトルが迅速に計算されるようになる。アニメーションのすべてのフレームが生成される必要はなく、ビデオストリームに追いつくまではクライアントで再生するのに十分に生成される必要がある。生成されたフレームの最小数は、プレイヤ入力を送信するものと、クライアントで結果ビデオストリームを受信するもののディレイによって異なるはずであり、アニメーションの生成された部分の長さは、少なくともディレイの分だけ長くなければならない。予測された動きベクトルフレームは、動き推定にキャッシングされた、繰り返される動きベクトルの使用を説明する図１０～図１３を参照しながら後述するように、再生される間、レートスケーリング（ｒａｔｅ－ｓｃａｌｅｄ）されてよい。予測された動きベクトルフレームの再生がレートスケーリングされれば、ディレイと同一のアニメーションの一部に対する予測された動きベクトルの生成は、５０％の再生レート－スケーリングをもたらす。アニメーションのさらに長い部分に対する動きベクトルの生成は、消極的にレートスケーリングされた再生をもたらす。

【００５９】

ビデオエンコードのために使用されたマクロブロックサイズは、動きベクトルを記録するときに考慮されなければならず、マクロブロックごとに動きベクトルがなければならない。好ましい実施形態において、ゲームで生成された動きベクトルは、ピクセルあたりの（ｐｅｒ－ｐｉｘｅｌ）動きベクトルで生成され、ピクセルあたりの動きベクトルの各マクロブロックグループに対して算術平均（ａｒｉｔｈｍｅｔｉｃ　ｍｅａｎ）を求めることにより、マクロブロックあたりの（ｐｅｒ－ｍａｃｒｏｂｌｏｃｋ）動きベクトルに変換される。

【００６０】

図８は、プレイヤ入力の動き補償を目的として予測された動きベクトルの送信およびクライアント側の記録を示している。ビデオゲームソフトウェアデベロップメントの間、イベントなどは、近づいてくる（ｕｐｃｏｍｉｎｇ）入力中心の（ｉｎｐｕｔ－ｄｒｉｖｅｎ）コンテキストに適する（ｃｏｎｔｅｘｔ－ｓｅｎｓｉｔｉｖｅ）信号を送信するように構成される必要がある。例えば、プレイヤがレッジグラブアニメーションを実行するときに予測された動きベクトルが利用可能となるように、ゲーム開発者は、予測された動きベクトルを送信しようとするであろう。開発者は、プレイヤが掴むことのできるレッジを、向い合って掴むことのできるレッジの範囲内にいるたびに、トリガーされるイベントを実現する。このような例において、「ゲームランタイム中のイベント受信」段階８００において、プレイヤがゲームをしている間、掴むことのできるレッジに近づくとき、例示的なイベントが受信される。イベント類型は、適応的に変更されたアニメーションの場合のように、予測された動きベクトルが生成される必要があるか否か、または変更はされないが頻繁に再生されるかプレイヤコンテキストに依存するアニメーションの場合のように、予測された動きベクトルがオフラインで事前生成されたか否かを説明するであろう。このような例において、「予測された動きベクトルの生成」段階８０２において、レッジグラブアニメーションは、プレイヤのレッジからの距離に基づいて増加するが、これは、動きベクトルがランタイムで生成される必要があることを意味する。他の場合、「事前生成された動きベクトルを読み取る」段階８０４において、動きベクトルは、オフラインで生成され、保存場所から読み取られてよい。

【００６１】

事前生成された動きベクトルの一例として、大きな武器倉庫の武器に転換が生じる場合が挙げられる。可能な武器転換順列の数は大多数である場合があり、この結果、動きベク

トルの全体セットのキャッシングを非現実的にさせる。一般的に、動きベクトルが制限されたキャッシュ空間を過度に占めていて頻繁に使用されなくなれば、動きベクトルはプレキャッシングのための主要候補にはならない。「予測された動きベクトルとインバリデータの送信」段階８０６において、予測された動きベクトルは、クライアントに送信される。「予測された動きベクトルとインバリデータのキャッシング」段階８０８において、予測された動きベクトルは、動きベクトルルックアップテーブルに追加される。一実施形態において、無効化（ｉｎｖａｌｉｄａｔｉｏｎ）システムは、ルックアップテーブルに動きベクトルの適用をトリガーするが、この代わりに、動きベクトルをディスエーブルさせるシステムと同じように機能する。動きベクトルとインバリデータのセットが「予測された動きベクトルおよびインバリデータのキャッシング」段階８０８において受信されるとき、インバリデータは、無効化システムによって登録される必要がある。

【００６２】

図１～図６を参照しながら説明したようなプレイヤ入力の動き補償方法は、すべてのプレイヤ入力を動きベクトルルックアップテーブル内のエントリーと比較する。上述した例において、「受信されたプレイヤ入力のマッチング」段階８１０において、プレイヤが、レッジグラブアニメーションを始めるために求められる入力を行うとき、入力は、ルックアップテーブルに予めキャッシングされた入力とマッチングされるであろう。「プレイヤ入力の動き補償の適用」段階８１２において、キャッシングされた予測された動きベクトルがまだ無効化されていなければ、予測された動きベクトルが適用されるであろう。「無効化」段階８１４において、プレイヤ入力が予測された動きベクトルを無効にしたり、予測された動きベクトルが予め決定された時間後に満了になったり、または予測された動きベクトルが１度適用された後に満了になったりすれば、予測された動きベクトルはルックアップテーブルから除去され、適用されない。

【００６３】

図９は、予測された動きベクトルセットを無効にすることのできる信号の例示的な方法を説明するための図である。無効化は、好ましくは、クライアントにキャッシングされたルックアップテーブルから一セットの予測された動きベクトルが除去される、ルックアップテーブルに対するアップデート類型である。サーバから受信されたアップデートイベントなどに応答することに加え、アップデートメカニズムは、プレイヤ入力をモニタリングし、無効化カウントダウンタイマー（ｃｏｕｎｔｄｏｗｎ　ｔｉｍｅｒ）を含んでよい。一セットの予測された動きベクトルがルックアップテーブルにキャッシングされるとき、このインバリデータがアップデート機能／メカニズムに登録されてよい。インバリデータは、ルックアップテーブルに対する無効化アップデートをトリガーするデータ信号である。ルックアップテーブルの例として、段階９００に示された「ルックアップテーブル」は、以下のような３セットの予測された動きベクトルを含む（「予測されたドア（ｄｏｏｒ）アニメーション」段階９０２に示された予測されたドアアニメーション、「予測されたレッジグラブ」段階９０４に示された予測されたレッジグラブアニメーション、および「予測されたキルショット（ｋｉｌｌ　ｓｈｏｔ）アニメーション」段階９０６に示された予測されたキルショットアニメーション）。予測された動きベクトルは、１度適用された後に無効化されてよい。一例として、「予測されたドア（ｄｏｏｒ）アニメーション」段階９０２に示された予測されたドアアニメーションは、「オープンドア入力」段階９０８においてプレイヤが入力を行ったとき、近くのドアを開くように適用される。これと同時に、「オープンドア入力」段階９０８におけるオープンドア入力の登録は解除され、「予測されたドアアニメーション」段階９０２に示された予測されたドアアニメーションは、「使用後の無効化」段階９１０においてルックアップテーブルから除去されてよい。これと同じように、他の入力も、これが適用される前に予測された動きベクトルを無効にしてよい。一例として、「予測されたレッジグラブ」段階９０４に示されたレッジグラブアニメーションに対する一セットの予測された動きベクトルは、レッジから制限された距離内のみで有効となる。

【００６４】

プレイヤが（「ジャンプ入力」段階９１４に示されたジャンプ入力を使用して）レッジをグラブするためにジャンプする前に、プレイヤが「移動入力」段階９１２に示された移動入力を使用してレッジから遠ざかれば、「予測されたレッジグラブ」段階９０４に示された予測されたレッジグラブの動きベクトルが「入力に対する無効化」段階９１６において無効となるであろう。無効にする入力の他の例として、プレイヤがグラップリングフック（ｇｒａｐｐｌｉｎｇ　ｈｏｏｋ）または予測された動きベクトルを無効にする他の移動基盤の武器を持っている場合、およびプレイヤが特殊能力を活性化させるボタンを押す場合などが挙げられてよい。無効化する入力はコンテキストによって異なるため、他のものは特定の実施形態に基づいて明らかになるであろう。また、予測された動きベクトルは、時間の経過とともに満了となってよい。一例として、キルショットのチャンスが３秒のウィンドウ中にプレイヤに提供されてよい。プレイヤが３秒のウィンドウ中に「メイレイ（ｍｅｌｅｅ）入力」段階９１８に示されたメイレイ入力を押さなかった場合、「予測されたキルショットアニメーション」段階９０６に示された予測されたキルショットアニメーションに対する動きベクトルは、「満了タイマーに対する無効化」段階９２０において無効となるであろう。予測された動きベクトルは複数のインバリデータを有してよく、その反対も可能である。例えば、予測されたレッジグラブは、移動入力が受信されるか、レッジグラブの動きベクトルが使用された後という、２つのうちの早い方により、無効化されてよい。

【００６５】

図１０は、動きベクトルライブラリを生成するための例示的な方法およびキャッシング目的のための例示的な繰り返される動きベクトルライブラリを示した図である。選択された動きは、実質的に反復性が高いため、これはトリガーされるたびに同一の方式によって再生されるであろう。これは、動きベクトルを予め生成し、これをライブラリとして組職化できるようにする。動きベクトルライブラリ生成は、デベロップメント中にいつでも発生するが、このプロセスをビルドプロセスまたは複数の他のアセットコンディショニングフェーズ（ａｓｓｅｔ　ｃｏｎｄｉｔｉｏｎｉｎｇ　ｐｈａｓｅ）の一段階として追加することが好ましいこともある。このような例において、動きベクトルライブラリは、１人称シュータにおいて利用可能なそれぞれの武器に対して生成されるであろう。「ライブラリ生成」段階１０００において、第１武器に対するライブラリ生成が始まるとき、「アニメーショントリガー」段階１００２において第１武器アニメーションがトリガーされ、「動きベクトル記録」段階１００４において動きベクトルが記録される。動きベクトルは、ゲームで生成されてもよいし、Ｈ．２６４コーデックで使用されるもののような従来の動き推定技術によって生成されてもよい。好ましい実施形態において、本明細書にその全体が統合された米国仮出願第６２／４８８，２５６号および第６２／６３４，４６４に記載されるようなゲームで生成された動きベクトルは、高品質の動き推定を保障するために使用される。記録された動きベクトルが正確でないか正確に量子化されなければ、これがプレイヤ入力の動き補償中に使用されるときに、アーティファクトを取り入れるであろう。段階１００２および１００４は、動きベクトルがライブラリでそれぞれの高い反復性を有するアニメーションに対して記録されるまで繰り返される。ライブラリ生成は、すべてのライブラリが生成されるまで、段階１０００で再び始まる。

【００６６】

「動きベクトルライブラリ」段階１００６に示された例は、１人称シューターゲームにおけるプラズマライフル武器（ｐｌａｓｍａ　ｒｉｆｌｅ　ｗｅａｐｏｎ）に対する、繰り返される動きベクトルライブラリの極めて単純化されたバージョンである。この例は、２つの簡単なアニメーションを含むように単純化される（段階１００８に示されたプレイヤがプラズマライフルを発射するときに再生されるフィードバックアニメーションのための２フレームアニメーション、および段階１０１０に示されたプレイヤが前方に歩いていくときに発生するライフルのボビングモーション（ｂｏｂｂｉｎｇ　ｍｏｔｉｏｎ）のための４フレームアニメーション）。一般的な実世界の環境（ｒｅａｌ－ｗｏｒｌｄ　ｅｎｖｉｒｏｎｍｅｎｔ）において、武器は、より多くのアニメーションを有し、そのアニメ

ーションは４つのフレームよりも相当に長い。
【００６７】
図１１は、プレイヤ入力の動き補償のために動きベクトルライブラリをキャッシング、適用、およびアップデートするためのプロセスを示した図である。「始まるときの繰り返される動きベクトルライブラリの送信」段階１１００において、ゲームが初期化されるとき、サーバは、クライアントに事前生成された動きベクトルライブラリを送信する。「繰り返される動きベクトルのキャッシング」段階１１０２において、クライアントは、メモリに、一般的にルックアップテーブルの形態で、動きベクトルライブラリを記録する。このような実施形態おいて、クライアントは、複数のストリーミングゲームの要求を提供するように一般化される。代案的な実施形態において、ゲームの特定クライアントは、サーバからの繰り返される動きベクトルをキャッシングする必要なく、動きベクトルライブラリを永久的に記録してもよい。
【００６８】
この時点において、クライアントは、プレイヤ入力をモニタリングし、入力とプレイヤ入力の動き補償ルックアップテーブル内のエントリーとの比較を開始してよい。「受信されたプレイヤ入力のマッチング」段階１１０４において、入力がルックアップテーブル内のマッチングされたエントリーを有するとき、「プレイヤ入力の動き補償の適用」段階１１０６において、プレイヤ入力と関連する動きベクトルは、図１～図１０を参照しながら説明したような動き推定の提供に使用される。「キャッシングされた、繰り返される動きベクトルマッピングのアップデート」段階１１１２において、特定の入力を受信することが、ルックアップテーブルが変更されるように要求する特定の場合となることがある。ルックアップテーブルを変更する入力の例としては、キャラクタ移動、武器発射アニメーション、または他の再生関連モーションのようなプレイヤ入力をディスエーブルさせることのできる一時停止ボタンがある。動きベクトルを適用して選択的にルックアップテーブルをアップデートした後、クライアントは、プレイヤ入力をモニタリングし続けるであろう。「キャッシングされた、繰り返される動きベクトルマッピングのアップデート」段階１１１２において、ルックアップテーブルがアップデートされた後に挿入されるプレイヤ入力は、ルックアップテーブル内のアップデートされたエントリーと比較されるであろう。
【００６９】
ゲームのランタイム中にいつでも、プレイヤに対するコンテキストの変更は、ルックアップテーブルが変更されるように要求してよい。例えば、プレイヤは、以前に保有していた武器が新たな武器に転換されるように、キャッシングされた動きライブラリを要求する武器を転換してよい。一例としては、特定のゲームイベントをモニタリングするサーバ側のゲームを示してよい。このようなイベントは、ゲームデベロップメント中に、該当の技術分野で周知の方法によって構成可能であり、ゲームの従来のメッセージングまたはイベントシステムによって送信されてよい。「ランタイム中にイベントを受信」段階１１０８において、実行されるゲームインスタンスがこのようなイベントのうちの１つを受信するとき、「コンテキストアップデート送信」段階１１１０において、メッセージが生成されてクライアントに送信されてよい。コンテキストアップデートは、プレイヤ入力とルックアップテーブルに含まれた動きベクトルとの関係を変更するゲームイベントに対して送信されてよい。コンテキストアップデートは、プレイヤ入力が一セットの動きベクトルをトリガーするものをイネーブル（ｅｎａｂｌｅ）またはディセーブル（ｄｉｓａｂｌｅ）させるか、与えられた入力と関連する動きベクトルを変更するか、そうではなければ、プレイヤ入力されたプレイヤ入力の動き補償のための動きベクトルとの関連を追加または除去してよい。「キャッシングされた、繰り返される動きベクトルマッピングのアップデート」段階１１１２において、メッセージがクライアントに到達するとき、ルックアップテーブルが変更されるコンテキストによって変更される。
【００７０】
図１２は、動きベクトルマッピングがどのようにアップデートされるかを示した図である。クライアントのキャッシュ１２０２に記録されたルックアップテーブル１２００は、

クライアントにより、与えられたプレイヤ入力と関連する動きベクトルの探索に使用される。状況による変更がプレイヤ入力の行動を変更する、ゲームのランタイム時間がある。例えば、プレイヤが前方に移動したが、壁（ｗａｌｌ）のような移動ブロッカー（ｍｏｖｅｍｅｎｔ　ｂｌｏｃｋｅｒ）にぶつかれば、前方移動に対して予測された動きベクトルの適用を中断する必要が生じる。これが発生するとき、サーバは、図１１の段階１１１０に示すように、クライアントにコンテキストアップデートを送信する。このような移動ブロッキングイベント１２０４が受信されるが、これは、クライアントに前方移動入力１２０６と対応する動きベクトル１２０８との連結をディセーブルさせるように信号を送信する。前方移動の動きベクトルは、サーバからの他のイベントが前方移動入力と前方移動の動きベクトルとの連結を再びイネーブルさせるまでプレイヤが前方移動入力を使用するとき、適用を中断する。このような例に対し、ゲームは、プレイヤの移動がアンブロックされる（ｕｎｂｌｏｃｋｅｄ）ときにイベントを生成し、前方入力とルックアップテーブル内の前方移動の動きベクトルとの連結を再設定するように、クライアントにコンテキストアップデートを送信する。

【００７１】

他の例において、プレイヤが、プラズマライフルをショットガン（ｓｈｏｔｇｕｎ）に切り換えることが考えられる。プラズマライフルの動きベクトルライブラリ１２１０は、特定の発射入力１２１１に対応する発射動きベクトル１２０９とともにキャッシュに記録される。また、キャッシュは、ショットガン１２１２とピストル（ｐｉｓｔｏｌ）１２１４を含む他の武器に対する動きベクトルライブラリを記録する。クライアントがサーバから武器転換イベント１２１６を受信すると、プラズマライフルの動きベクトルライブラリ１２１０がショットガンの動きベクトルライブラリ１２１２に対してルックアップテーブル１２００から転換される。プレイヤ入力の動き補償が武器転換中に不適切に発生することを防ぐために、２つのイベントが、武器転換アニメーションが再生される間にプラズマライフルの動きベクトルライブラリ１２１０を先にディセーブルさせるように同時に使用された後、武器転換アニメーションが完了した後、２つの動きベクトルライブラリを転換してよい。

【００７２】

さらに長いマルチフレーム動きベクトルに対し、最後のフレームの実際のビデオがサーバから受信されると、最後のフレームの動きベクトルが適用されるように、このアプリケーションを増加させてよい。これにより、サーバがクライアントにキャッシングされた動きベクトルが足りないとき、推定された動きに追いつくようにできるであろう。動きベクトルに対するスケーリングファクタは、再生速度スケール（ｐｌａｙｂａｃｋ　Ｓｐｅｅｄ　Ｓｃａｌｅ）によって定義され、例示的に下記のように計算される。

【００７３】

【数１】

$$\text{再生速度スケール} = \frac{\text{キャッシングされたアニメーション時間}}{\text{キャッシングされたアニメーション時間} + \text{ディレイ}}$$

【００７４】

ここで、ディレイは、初期プレイヤ入力イベントからクライアントで実際のビデオを受信するまでの時間によって定義される。このディレイは、ネットワークを介して入力を送信するのにかかる時間、ゲームロジック、レンダリングロジック、ＧＰＵレンダリング時間、およびエンコード時間を含んでサーバで処理する時間、さらにプレイヤにビデオをリターンするためのネットワーク時間を含む。ディレイは、ゲームストリーミング環境で持続的に測定されなければならない。プレイヤ入力の動き補償の好ましい実施形態は、プレイヤ入力と実際のビデオとを関連させるために、米国仮出願第６２／４８８，２５６号および第６２／６３４，４６４号に記載されるように、相関タグを使用する。入力されたプ

レイヤ入力がサーバに送信される前に、相関タグが固有の識別子として付与される。ビデオフレームが相関タグとともにサーバからリターンされるとき、クライアントは、固有の識別子を以前の入力とマッチングさせる。これは、関連する入力に対する動き推定を中断したり、ブレンディング技術による以前の動き推定を取り消したりするように、クライアントに信号を送る。アニメーションのキャッシングされた部分の長さやキャッシングされたアニメーション時間は、キャッシングされたアニメーションのフレームの数に各フレームの長さを掛けることによって計算されてよい。

【００７５】

図１３において、１０個のフレームを含むアニメーションの例として、１００ｍｓのディレイとともに１秒あたり６０フレームを実行するゲームにおいてプレイヤ入力の動き補償のために使用される。プレイヤ入力がプレイヤ入力の動き補償をトリガーするとき、再生速度スケールが下記のようにキャッシングされた動きベクトルに対して計算される。

【００７６】

【数２】

$$\text{再生速度スケール} = \frac{\text{キャッシングされたアニメーション時間}}{\text{キャッシングされたアニメーション時間} + \text{ディレイ}}$$

【００７７】

【数３】

$$\text{再生速度スケール} = \frac{10\ \text{フレーム} * \frac{1000\ \text{ms}}{60\ \text{フレーム}}}{\left(10\ \text{フレーム} * \frac{1000\ \text{ms}}{60\ \text{フレーム}}\right) + 100\ \text{ms}}$$

【００７８】

【数４】

$$\text{再生速度スケール} = \frac{166.66\ \text{ms}}{166.66\ \text{ms} + 100\ \text{ms}} = 0.625$$

【００７９】

プレイヤ入力は０ｍｓ時間に受信された。動きベクトルの再生の割合は、計算された再生速度スケールによってスケーリングされる（１３００）。１番目のフレームの動きベクトルは、サーバからのビデオストリーム内の利用可能な次のフレームに適用される。スケーリングされた動きベクトルフレームは、アニメーションスムーズ（ａｎｉｍａｔｉｏｎ　ｓｍｏｏｔｈ）を維持するために補間されるであろう。動きベクトルフレームは、複数のフレームにわたってスケーリングされるため、補間は与えられたフレームに適用する動きベクトルが「どれほど多いか」を計算するのに使用可能な方法である。一例として、計算された再生速度スケールに基づいて線形補間を使用してよい。この例に対し、計算された再生速度スケールは０．６２５であり、これは一セットの動きベクトルを１．６個のディスプレイフレームの上に増やすであろう。補間は、どれほど多くの動きベクトルを与えられたフレームに適用するかを計算するための方法である。すなわち、補間は、このセットの動きベクトルが複数のディスプレイフレームの上に増えるとき、どれほど遠くにマクロブロックを動きベクトルの下に移動させるかを計算する。第１スケーリングされた動きベクトルの一部だけが０．６２５の再生速度スケールと同一な、１７ｍｓで第１ディスプレイフレームに適用されなければならない。３３ｍｓにおける第２ディスプレイフレームに、第１スケーリングされた動きベクトルの残りが適用され、１－０．６２５＝０．３７５で計算された後、第２スケーリングされたベクトルの１番目の部分が適用され、再生速度スケールから第１スケーリングされた動きベクトルの残り部分を引いた値、または０．

６２５－０．３７５＝０．２５で計算される。５０ｍｓの第３ディスプレイフレームにおいて、第２セットのスケーリングされた動きベクトルが継続して適用され、マクロブロックが動きベクトルの下に６２．５％で移動する。６７ｍｓの第４ディスプレイフレームにおいて、第２スケーリングされた動きベクトルの残りが適用され、１－０．２５－０．６２５＝０．１２５で計算され、第３スケーリングされた動きベクトルの１番目の部分が適用され、再生速度スケールから第２スケーリングされた動きベクトルの残り部分を引いた値、または０．６２５－０．１２５＝０．５で計算される。スケーリングされた動きベクトルが適用されることにより、線形補間が続く。

【００８０】

推定された動きの各フレームを未来の実際のビデオと関連させるために、マルチフレーム動きベクトルは、キャッシングされたアニメーションの各フレームに対する相関タグを送信してよい。

【００８１】

ディレイは、クライアントとサーバとの間のネットワーク経路およびアキテクチャに大きく左右されるであろう。これは、１００ｍｓディレイを使用するが、数十から数百ミリ秒（ｍｉｌｌｉｓｅｃｏｎｄｓ）の間で変わる。より短いディレイがより優れたプレイヤ経験を提供するが、プレイヤ入力の動き補償手法は、特定の場合に、高いディレイ時間の影響を偽装（ｄｉｓｇｕｉｓｅ）するのに繋がる。ディレイ１３０４後に、実際のビデオ１３０２が受信される。エッジ位置の（ｅｄｇｅ－ｌｏｃａｔｅｄ）サーバまたは消費者と物理的に近いサーバに対し、ディレイ時間は約３０ｍｓと低い。より一般的なサーバ位置に対し、１００ｍｓはさらに可能性が高い。実際のビデオはスケーリングされないため、オリジナルアニメーションの長さ１３０６を維持する。実際のビデオは、プレイヤ入力の動き補償手法によって適用される。

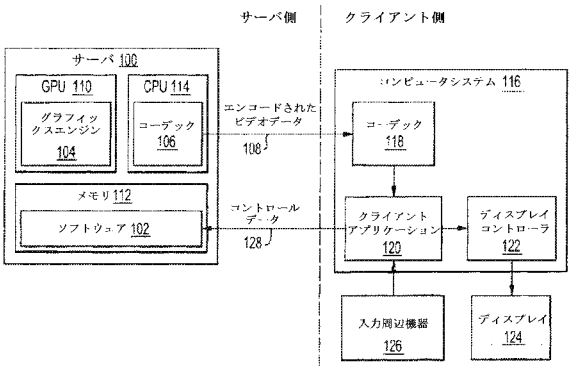
【００８２】

クライアントが以前の動き補償を完璧にブレンディングすることができなければ、Ｈ．２６４コーディング標準がいずれかの時間的に伝播されるエラーを訂正することのできるリンダンダントスライス特徴（ｒｅｄｕｎｄａｎｔ　ｓｌｉｃｅ　ｆｅａｔｕｒｅ）を提供する。Ｈ．２６４プロファイル設定を基盤に、各スライスは、イントラスライス（ｉｎｔｒａ　ｓｌｉｃｅ）（Ｉ－スライス）にエンコードされ、特定の周波数で回転スケジュール（ｒｏｔａｔｉｎｇ　ｓｃｈｅｄｕｌｅ）に送信されるであろう。イントラスライスが動きベクトルを含まないため、ブレンド動きベクトルは、実際の動きベクトルがｐ－スライスに到達するときだけに適用されなければならない。これは、タギングされたフレームがサーバからリターンされる前に、ブレンド動きベクトルがＩ－スライスに示したマクロブロックに適用されることを防ぐことができる。

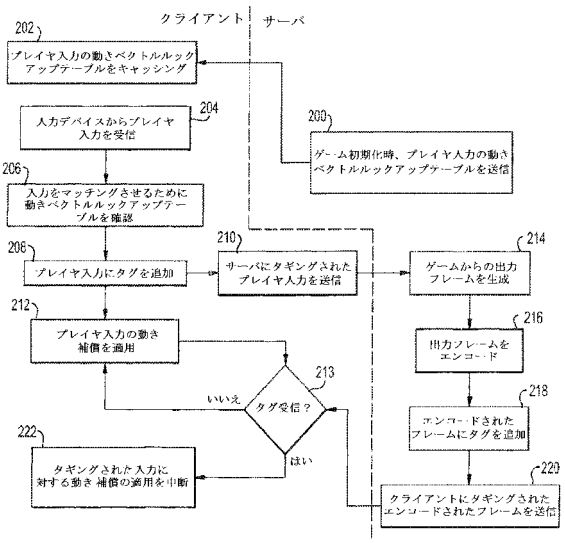
【００８３】

上述した説明および図面は、本発明の原理を例示するためのものに過ぎない。本発明は、好ましい実施形態によって制限されるように意図されてはならず、該当の技術分野において通常の知識を有する者にとって明白である、多様な方式で実現されるであろう。本発明の多数の応用は、該当の技術分野において通常の知識を有する者によって容易に実現されるであろう。したがって、開示された特定の例示または図に示されて説明された正確な構成と動作によって本発明を制限することは好ましくない。むしろ、すべての適切な修正および均等物が、本発明の範囲内に属するものと理解されなければならない。

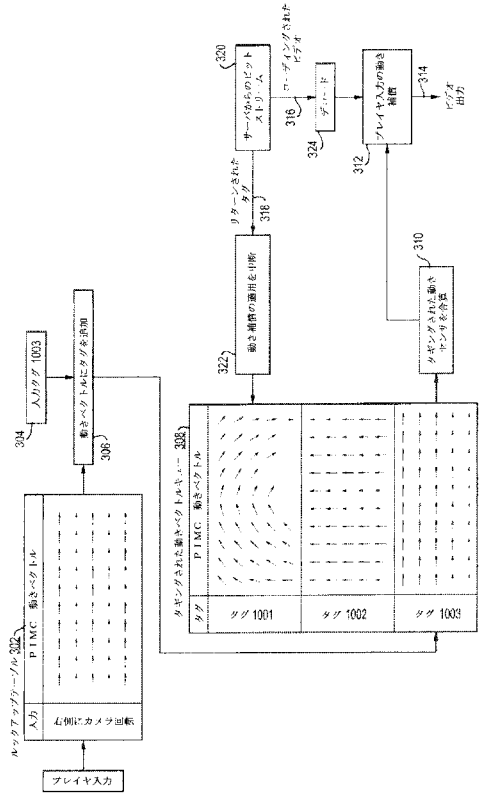
【図１】



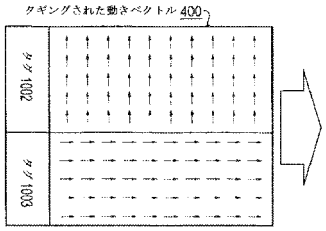
【図２】



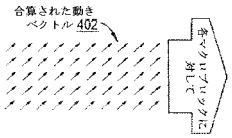
【図３】



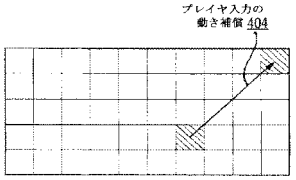
【図４ａ】



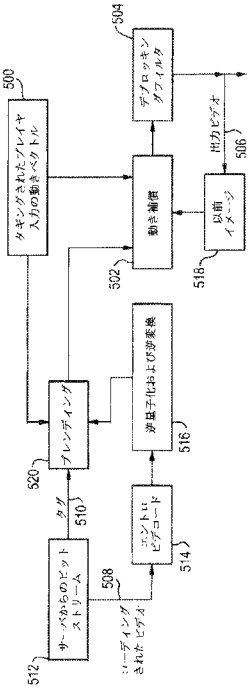
【図４ｂ】



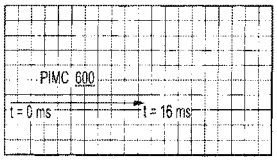
【図４ｃ】



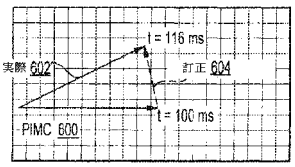
【図５】



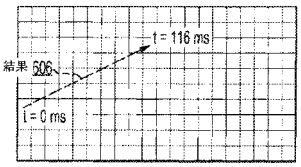
【図６ａ】



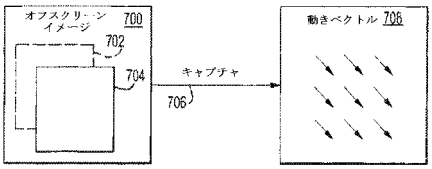
【図６ｂ】



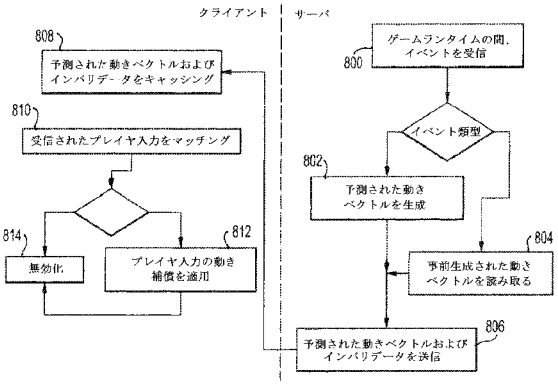
【図６ｃ】



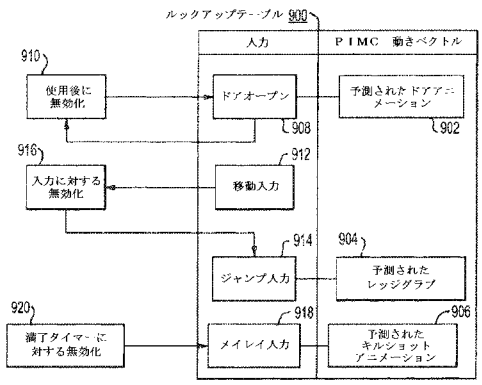
【図７】



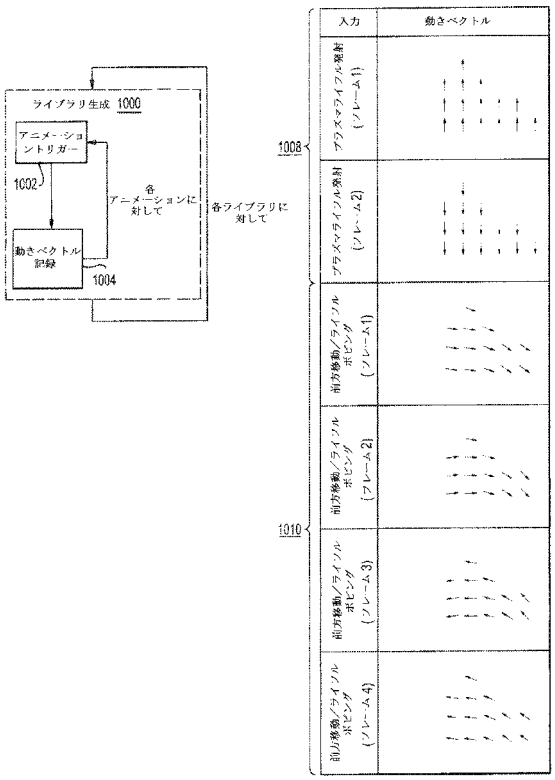
【図８】



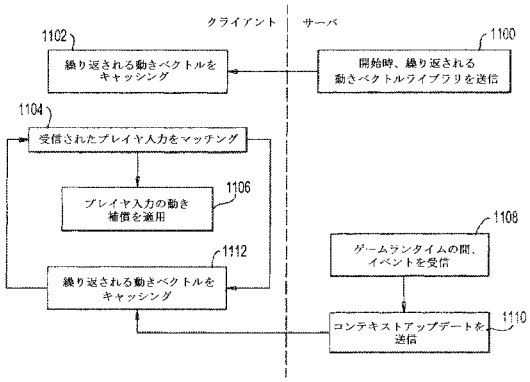
【図９】



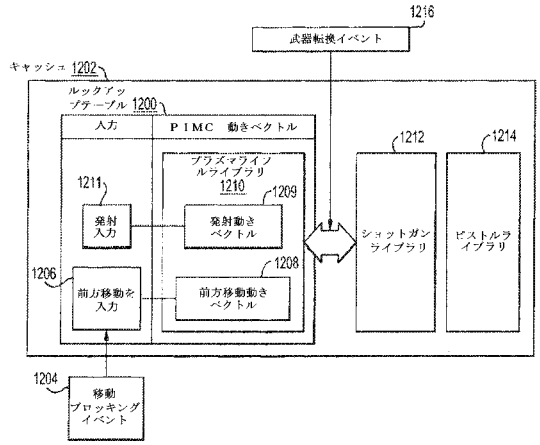
【図１０】



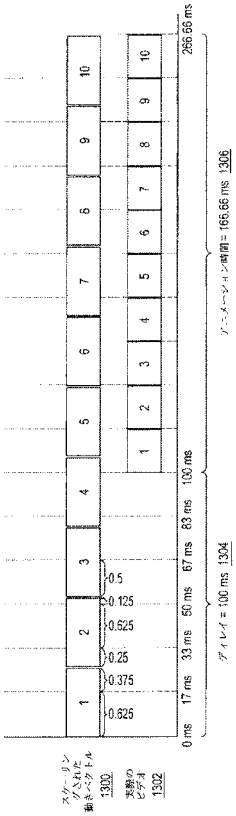
【図１１】



【図１２】



【図１３】



【手続補正書】
【提出日】令和2年4月23日(2020.4.23)
【手続補正１】
【補正対象書類名】特許請求の範囲
【補正対象項目名】全文
【補正方法】変更
【補正の内容】
【特許請求の範囲】
【請求項１】
動きベクトル（ｍｏｔｉｏｎ　ｖｅｃｔｏｒ）をキャッシングするためのコンピュータで実現される方法であって、
サーバで１つ以上の動きベクトルを生成する段階（前記動きベクトルは、予め決定された基準に基づいて生成される）、
クライアントに前記生成された動きベクトルと１つ以上のインバリデータ（ｉｎｖａｌｉｄａｔｏｒ）を送信する段階（前記生成された動きベクトルとインバリデータは、前記クライアントでキャッシングされる）、
ユーザから入力を受信するために、前記クライアントに命令語を送信する段階（前記クライアントは、前記入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるように構成される）、および
前記マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース（ｇｒａｐｈｉｃ　ｉｎｔｅｒｆａｃｅ）のエフェクト動き（ｅｆｆｅｃｔ　ｍｏｔｉｏｎ）補償に適用するために、前記クライアントに命令語を送信する段階
を含む、方法。
【請求項２】
前記動きベクトルは、

ルックアップテーブルにキャッシングされる、請求項１に記載の方法。
【請求項３】
前記１つ以上のインバリデータは、
１つ以上のキャッシングされた動きベクトルと関連する、請求項１に記載の方法。
【請求項４】
前記１つ以上のインバリデータは、
前記１つ以上のキャッシングされた動きベクトルをディセーブル（ｄｉｓａｂｌｅ）させるように構成される、請求項３に記載の方法。
【請求項５】
入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するように、前記クライアントに指示する段階
をさらに含む、請求項１に記載の方法。
【請求項６】
前記動きベクトルは、
前記サーバに記録され、
要求時にキャッシングされるように前記クライアントに送信される、請求項１に記載の方法。
【請求項７】
前記動きベクトルは、
ゲームで生成される（ｇａｍｅ－ｇｅｎｅｒａｔｅｄ）、請求項１に記載の方法。
【請求項８】
前記ゲームで生成された動きベクトルは、
ピクセルあたりの（ｐｅｒ－ｐｉｘｅｌ）動きベクトルで生成され、
マクロブロックあたりの（ｐｅｒ－ｍａｃｏｒｂｌｏｃｋ）動きベクトルに変換される、請求項７に記載の方法。
【請求項９】
動きベクトルをキャッシングするためのシステムであって、
ネットワークを介し、サーバは、
１つ以上の動きベクトルを生成し（前記動きベクトルは、予め決定された基準に基づいて生成される）、
クライアントに前記生成された動きベクトルと１つ以上のインバリデータを送信し（前記生成された動きベクトルとインバリデータは、前記クライアントでキャッシングされる）、
ユーザから入力を受信するために、前記クライアントに命令語を送信し（前記クライアントは、前記入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるように構成される）、
前記マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース内のエフェクト動き補償に適用するために、前記クライアントに命令語を送信する、システム。
【請求項１０】
前記動きベクトルは、
ルックアップテーブルにキャッシングされる、請求項９に記載のシステム。
【請求項１１】
前記１つ以上のインバリデータは、
１つ以上のキャッシングされた動きベクトルと関連する、請求項９に記載のシステム。
【請求項１２】
前記１つ以上のインバリデータは、
前記１つ以上のキャッシングされた動きベクトルをディセーブルさせるように構成される、請求項１１に記載のシステム。

【請求項１３】
入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するように、前記クライアントに指示する段階
をさらに含む、請求項９に記載のシステム。
【請求項１４】
前記動きベクトルは、
ゲームで生成される、請求項９に記載のシステム。
【請求項１５】
前記インバリデータは、
プレイヤ入力と時間ウィンドウ（ｔｉｍｅ　ｗｉｎｄｏｗ）のうちの１つまたはその組み合わせである、請求項９に記載のシステム。
【手続補正３】
【補正対象書類名】明細書
【補正対象項目名】００８３
【補正方法】変更
【補正の内容】
【００８３】
上述した説明および図面は、本発明の原理を例示するためのものに過ぎない。本発明は、好ましい実施形態によって制限されるように意図されてはならず、該当の技術分野において通常の知識を有する者にとって明白である、多様な方式で実現されるであろう。本発明の多数の応用は、該当の技術分野において通常の知識を有する者によって容易に実現されるであろう。したがって、開示された特定の例示または図に示されて説明された正確な構成と動作によって本発明を制限することは好ましくない。むしろ、すべての適切な修正および均等物が、本発明の範囲内に属するものと理解されなければならない。
〔付記１〕
段階を含むコンピュータで実現される動き推定方法であって、
段階は、
１つ以上のユーザ入力と１つ以上の関連する動きベクトル（ｍｏｔｉｏｎ　ｖｅｃｔｏｒ）で構成されたルックアップテーブルを送信する段階（前記ルックアップテーブルは、前記ルックアップテーブルをキャッシング（ｃａｃｈｉｎｇ）するための命令語（ｉｎｓｔｒｕｃｔｉｏｎ）とともに送信される）、
プレイヤ入力の受信時、マッチングされる動きベクトルに対して前記ルックアップテーブルに質疑（ｑｕｅｒｙ）するための命令語を送信する段階、
固有のタグ（ｕｎｉｑｕｅ　ｔａｇ）を前記ルックアップテーブルからの前記マッチングされる動きベクトルと関連させるための命令語を送信し、タギングされた動きベクトルをキュー（ｑｕｅｕｅ）に追加する段階（前記タギングされた動きベクトルは、合算される）、
フレームを固有の識別子タグ（ｉｄｅｎｔｉｆｉｅｒ　ｔａｇ）とともに送信する段階（前記タグは、前記フレームが前記プレイヤ入力と関連する動きを含む時系列地点（ｃｈｒｏｎｏｌｏｇｉｃａｌ　ｐｏｉｎｔ）を示す）、
サーバから受信されたタギングされたフレームと関連するタグを有する前記キューから動きベクトルを除去するための命令語を送信する段階
を含み、
サーバがクライアントにエンコードされたビデオフレームを送信するとき、前記クライアントは、
前記ビデオフレームをデコードし、ビデオ出力以前の動きを推定するために前記合算された動きベクトルを前記デコードされたフレームに適用するように指示を受ける、コンピュータで実現される方法。
〔付記２〕

前記エンコードされたビデオフレームは、
残差ハンドリング（ｒｅｓｉｄｕａｌ　ｈａｎｄｌｉｎｇ）されずに、デコードされる、請求項１に記載のコンピュータで実現する方法。
〔付記３〕
前記クライアントに、１つ以上の滑らかな関数（ｓｍｏｏｔｈｉｎｇ　ｆｕｎｃｔｉｏｎ）を前記キュー内の前記合算されたタギングされた動きベクトルに適用するように指示する段階
をさらに含む、付記１に記載のコンピュータで実現される方法。
〔付記４〕
前記動きベクトルと関連する固有のタグは、
実質的に時系列である、付記１に記載のコンピュータで実現される方法。
〔付記５〕
前記合算されたタギングされた動きベクトルは、
対応するマクロブロック（ｍａｃｒｏｂｌｏｃｋ）と関連する、付記１に記載のコンピュータで実現される方法。
〔付記６〕
前記キャッシングされたルックアップテーブルは、
プレイヤ入力に基づいて修正されるように構成される、付記１に記載のコンピュータで実現される方法。
〔付記７〕
前記合算されたタギングされた動きベクトルは、
一度適用されるように構成される、付記１に記載のコンピュータで実現される方法。
〔付記８〕
前記合算されたタギングされた動きベクトルは、
任意複雑度（ｃｏｍｐｌｅｘｉｔｙ）の動きを説明する、付記１に記載のコンピュータで実現される方法。
〔付記９〕
前記合算されたタギングされた動きベクトルは、
前記プレイヤ入力に対する未来（ｆｕｔｕｒｅ）フィードバック（ｆｅｅｄｂａｃｋ）を推定する、付記１に記載のコンピュータで実現される方法。
〔付記１０〕
前記合算されたタギングされた動きベクトルは、
Ｈ．２６４コーディング標準にしたがう、付記１に記載のコンピュータで実現される方法。
〔付記１１〕
動き推定システムであって、
ネットワークを介し、サーバは、
プレイヤ入力の受信時、マッチングされる動きベクトルに対してルックアップテーブルに質疑するために、クライアントに命令語を送信し、
固有のタグを前記ルックアップテーブルからの前記マッチングされる動きベクトルと関連させるために、前記クライアントに命令語を送信し、タギングされた動きベクトルをキューに追加し（前記タギングされた動きベクトルは合算される）、
フレームを固有の識別子タグとともに前記クライアントに送信し（前記タグは、前記フレームが前記プレイヤ入力と関連する動きを含む時系列地点を示す）、
前記サーバから受信されたタギングされたフレームと関連するタグを有する前記キューから動きベクトルを除去するために、前記クライアントに命令語を送信し、
前記サーバが前記クライアントにエンコードされたビデオフレームを送信するとき、前記クライアントは、
前記ビデオフレームをデコードし、動きを推定するために前記合算された動きベクトルを前記デコードされたフレームに適用するように指示を受ける、システム。

〔付記１２〕
前記エンコードされたビデオフレームは、
残差ハンドリングされずに、デコードされる、付記１１に記載のシステム。
〔付記１３〕
前記サーバは、
１つ以上の滑らかな関数（ｓｍｏｏｔｈｉｎｇ　ｆｕｎｃｔｉｏｎ）を前記キュー内の前記合算されたタギングされた動きベクトルに適用するように、前記クライアントにさらに指示する、付記１１に記載のシステム。
〔付記１４〕
前記動きベクトルと関連するタグは、
実質的に時系列である、付記１１に記載のシステム。
〔付記１５〕
前記合算されたタギングされた動きベクトルは、
対応するマクロブロックと関連する、付記１１に記載のシステム。
〔付記１６〕
前記キャッシングされたルックアップテーブルは、
プレイヤ入力に基づいて修正されるように構成される、付記１１に記載のシステム。
〔付記１７〕
前記合算されたタギングされた動きベクトルは、
一度適用されるように構成される、付記１１に記載のシステム。
〔付記１８〕
前記合算されたタギングされた動きベクトルは、
任意複雑度（ｃｏｍｐｌｅｘｉｔｙ）の動きを説明する、付記１１に記載のシステム。
〔付記１９〕
前記合算されたタギングされた動きベクトルは、
前記プレイヤ入力に対する未来フィードバックを推定する、付記１１に記載のシステム。
〔付記２０〕
前記合算されたタギングされた動きベクトルは、
Ｈ．２６４コーディング標準にしたがう、付記１１に記載のシステム。
〔付記２１〕
段階を含む動きベクトルをキャッシングするためのコンピュータで実現される方法であって、
段階は、
サーバで１つ以上の動きベクトルを生成する段階（前記動きベクトルは、予め決定された基準に基づいて生成される）、
クライアントに前記生成された動きベクトルと１つ以上のインバリデータ（ｉｎｖａｌｉｄａｔｏｒ）を送信する段階（前記生成された動きベクトルとインバリデータは、前記クライアントでキャッシングされる）、
ユーザから入力を受信するために、前記クライアントに命令語を送信する段階（前記クライアントは、前記入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるように構成される）、および
前記マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース（ｇｒａｐｈｉｃ　ｉｎｔｅｒｆａｃｅ）のエフェクト動き（ｅｆｆｅｃｔ　ｍｏｔｉｏｎ）補償に適用するために、前記クライアントに命令語を送信する段階
を含む、方法。
〔付記２２〕
前記動きベクトルは、
ルックアップテーブルにキャッシングされる、付記２１に記載の方法。
〔付記２３〕

前記１つ以上のインバリデータは、
１つ以上のキャッシングされた動きベクトルと関連する、付記２１に記載の方法。
〔付記２４〕
前記１つ以上のインバリデータは、
前記１つ以上のキャッシングされた動きベクトルをディセーブル（ｄｉｓａｂｌｅ）させるように構成される、付記２３に記載の方法。
〔付記２５〕
入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するように、前記クライアントに指示する段階
をさらに含む、付記２１に記載の方法。
〔付記２６〕
前記動きベクトルは、
固有であり予測可能である、付記２１に記載の方法。
〔付記２７〕
前記動きベクトルは、
予め（ａｈｅａｄｏｆｔｉｍｅ）またはランタイム（ｒｕｎｔｉｍｅ）中に生成される、付記２１に記載の方法。
〔付記２８〕
前記動きベクトルは、
前記サーバに記録され、
要求時にキャッシングされるように前記クライアントに送信される、付記２１に記載の方法。
〔付記２９〕
前記動きベクトルは、
ゲームで生成される（ｇａｍｅ－ｇｅｎｅｒａｔｅｄ）、付記２１に記載の方法。
〔付記３０〕
前記ゲームで生成された動きベクトルは、
ピクセルあたりの（ｐｅｒ－ｐｉｘｅｌ）動きベクトルで生成され、
マクロブロックあたりの（ｐｅｒ－ｍａｃｏｒｂｌｏｃｋ）動きベクトルに変換される、付記２９に記載の方法。
〔付記３１〕
動きベクトルをキャッシングするためのシステムであって、
ネットワークを介し、サーバは、
１つ以上の動きベクトルを生成し（前記動きベクトルは、予め決定された基準に基づいて生成される）、
クライアントに前記生成された動きベクトルと１つ以上のインバリデータを送信し（前記生成された動きベクトルとインバリデータは、前記クライアントでキャッシングされる）、
ユーザから入力を受信するために、前記クライアントに命令語を送信し（前記クライアントは、前記入力をキャッシングされた動きベクトルまたはインバリデータとマッチングさせるように構成される）、
前記マッチングされた動きベクトルまたはインバリデータをグラフィックインタフェース内のエフェクト動き補償に適用するために、前記クライアントに命令語を送信する、システム。
〔付記３２〕
前記動きベクトルは、
ルックアップテーブルにキャッシングされる、付記３１に記載のシステム。
〔付記３３〕
前記１つ以上のインバリデータは、

１つ以上のキャッシングされた動きベクトルと関連する、付記３１に記載のシステム。
〔付記３４〕
前記１つ以上のインバリデータは、
前記１つ以上のキャッシングされた動きベクトルをディセーブルさせるように構成される、付記３３に記載のシステム。
〔付記３５〕
入力が１つ以上の動きベクトルと関連するキャッシングされたインバリデータにマッチングされると、１つ以上のキャッシングされた動きベクトルを削除するように、前記クライアントに指示する段階
をさらに含む、付記３１に記載のシステム。
〔付記３６〕
前記動きベクトルは、
固有であり予測可能である、付記３１に記載のシステム。
〔付記３７〕
前記動きベクトルは、
予めまたはランタイム中に生成される、付記３１に記載のシステム。
〔付記３８〕
前記動きベクトルは、
前記サーバに記録され、
要求時にキャッシングされるように前記クライアントに送信される、付記３１に記載のシステム。
〔付記３９〕
前記動きベクトルは、
ゲームで生成される、付記３１に記載のシステム。
〔付記４０〕
前記ゲームで生成された動きベクトルは、
ピクセルあたりの動きベクトルで生成され、
マクロブロックあたりの動きベクトルに変換される、付記３９に記載のシステム。
〔付記４１〕
動きベクトルをキャッシングするためのコンピュータで実現される方法であって、
サーバからクライアントに予め生成された動きベクトルライブラリ（ｍｏｔｉｏｎ　ｖｅｃｔｏｒ　ｌｉｂｒａｒｙ）を送信する段階（前記動きベクトルライブラリは、前記クライアントに記録されるように構成される）、
ユーザからの入力データをモニタリングするために、前記クライアントに命令語を送信する段階、
前記入力データから動き推定を計算するために、前記クライアントに命令語を送信する段階、および
前記入力データに基づいて前記記録された動きベクトルライブラリをアップデートするために、前記クライアントに命令語を送信する段階
を含み、
前記クライアントは、
前記サーバから実際の動きベクトルデータを受信する前にグラフィックインタフェースの動きを開始させるために、前記記録された動きベクトルライブラリを適用するように構成される、コンピュータで実現される方法。
〔付記４２〕
前記記録された動きベクトルライブラリの適用をディセーブルさせるために、前記サーバから前記クライアントにコンテキストアップデート（ｃｏｎｔｅｘｔ　ｕｐｄａｔｅ）を送信する段階
をさらに含む、付記４１に記載のコンピュータで実現される方法。
〔付記４３〕

１つ以上のスケーリングファクタ（ｓｃａｌｉｎｇ　ｆａｃｔｏｒ）を前記動きベクトルライブラリに適用するための命令語を送信する段階
をさらに含む、付記４１に記載のコンピュータで実現される方法。
〔付記４４〕
前記スケーリングファクタは、
一般数学式：

$$\text{再生速度スケール} = \frac{\text{キャッシングされたアニメーション時間}}{\text{キャッシングされたアニメーション時間} + \text{ディレイ}}$$

に基づいて計算される、付記３に記載のコンピュータで実現される方法。
〔付記４５〕
前記生成された動きベクトルライブラリは、
複数の動きベクトルで構成される、付記４１に記載のコンピュータで実現される方法。
〔付記４６〕
前記動きベクトルは、
ゲームで生成される、付記４５に記載のコンピュータで実現される方法。
〔付記４７〕
前記生成された動きベクトルライブラリは、
前記クライアントに永久的に記録されるように構成される、付記４１に記載のコンピュータで実現される方法。
〔付記４８〕
前記動きベクトルライブラリは、
ビルドプロセス（ｂｕｉｌｄｐｒｏｃｅｓｓ）中に生成される、付記４１に記載のコンピュータに実現された方法。
〔付記４９〕
前記生成された動きベクトルライブラリは、
前記ユーザからの前記入力データと関連する、付記４１に記載のコンピュータに実現された方法。
〔付記５０〕
前記命令語は、
相関タグ（ｃｏｒｒｅｌａｔｉｏｎ　ｔａｇ）で構成され、
前記相関タグは、
前記ユーザからの前記入力データと関連する、付記４１に記載のコンピュータに実現された方法。
〔付記５１〕
動きベクトルをキャッシングするためのシステムであって、
ネットワークを介し、サーバは、
クライアントに予め生成された動きベクトルライブラリを送信し（前記動きベクトルライブラリは、前記クライアントに記録されるように構成される）、
ユーザからの入力データをモニタリングするために、前記クライアントに命令語を送信し、
前記入力データから動き推定を計算するために、前記クライアントに命令語を送信し、
前記入力データに基づいて前記記録された動きベクトルライブラリをアップデートするために、前記クライアントに命令語を送信し、
前記クライアントは、
前記サーバから実際の動きベクトルデータを受信する前にグラフィックインタフェースの動きを開始させるために、前記記録された動きベクトルライブラリを適用するように構成される、システム。
〔付記５２〕

前記記録された動きベクトルライブラリの適用をディセーブルさせるために、前記サーバから前記クライアントにコンテキストアップデートを送信する段階
をさらに含む、付記５１に記載のシステム。
〔付記５３〕
前記サーバは、
１つ以上のスケーリングファクタを前記動きベクトルライブラリに適用するための命令語をさらに送信する、付記５１に記載のシステム。
〔付記５４〕
前記スケーリングファクタは、
一般数学式：

$$\text{再生速度スケール} = \frac{\text{キャッシングされたアニメーション時間}}{\text{キャッシングされたアニメーション時間} + \text{ディレイ}}$$

に基づいて計算される、付記５３に記載のシステム。
〔付記５５〕
前記生成された動きベクトルライブラリは、
複数の動きベクトルで構成される、付記５１に記載のシステム。
〔付記５６〕
前記動きベクトルは、
ゲームで生成される、付記５５に記載のシステム。
〔付記５７〕
前記生成された動きベクトルライブラリは、
前記クライアントに永久的に記録されるように構成される、付記５１に記載のシステム。
〔付記５８〕
前記動きベクトルライブラリは、
ビルドプロセス中に生成される、付記５１に記載のシステム。
〔付記５９〕
前記生成された動きベクトルライブラリは、
前記ユーザからの前記入力データと関連する、付記５１に記載のシステム。
〔付記６０〕
前記命令語は、
相関タグで構成され、
前記相関タグは、
前記ユーザからの前記入力データと関連する、付記５１に記載のシステム。

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US18/28620

A. CLASSIFICATION OF SUBJECT MATTER
IPC - G06T 7/20, 5/00, 7/00 (2018.01)
CPC - G06T 7/20, 5/00, 7/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
See Search History document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
See Search History document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2016/0227218 A1 (ECOLE DE TECHNOLOGIE SUPERIEURE) 04 August 2016, entire document	1-20
A	US 8,069,258 B1 (HOWELL, J) 29 November 2011, entire document	1-20
A	US 8,854,376 B1 (BHAT, K et al) 07 October 2014, enitre document	1-20

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 June 2018 (22.06.2018)	24 AUG 2018
Name and mailing address of the ISA/ Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer Shane Thomas PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US18/28620

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:
*** Please see extra sheet ***

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:
Group I: Claims 1-20

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/US18/28620

-***-Continued from Box No. III Observations where unity of invention is lacking -***-

This application contains the following inventions or groups of inventions which are not so linked as to form a single general inventive concept under PCT Rule 13.1. In order for all inventions to be examined, the appropriate additional examination fee must be paid.

Group I: Claims 1-20 are directed towards a method and system for motion estimation.
Group II: Claims 21-40 are directed towards a method and system for caching motion vectors using invalidators.
Group III: Claims 41-60 are directed towards a method and system for caching motion vectors using a motion vector library.

The inventions listed as Groups I-II do not relate to a single general inventive concept under PCT Rule 13.1 because, under PCT Rule 13.2, they lack the same or corresponding special technical features for the following reasons:

The special technical features of Group I include at least a computer-implemented method of motion estimation comprising the steps of: a system for motion estimation, wherein, over a network, a server: transmitting a lookup table comprised of one or more user inputs and one or more associated motion vectors, wherein the lookup table is transmitted with an instruction to cache the lookup table; transmitting an instruction to query the lookup table for matching motion vectors upon the receipt of a player input; transmitting an instruction to associate a unique tag with the matching motion vectors from the lookup table and adding the tagged motion vectors to a queue, wherein the tagged motion vectors are summed; transmitting a frame with a unique identifier tag, wherein the tag indicates the chronological point to which the frame contains motion associated with the player input; transmitting an instruction to remove motion vectors from the queue that have a tag associated with the tagged frame received from the server; and wherein when a server transmits encoded video frames to a client, the client is instructed to decode the video frames and apply the summed motion vectors to the decoded frames to estimate motion prior to video output., which are not present in Groups II-III.

The special technical features of Group II include at least generating one or more motion vectors at a server, wherein the motion vectors are generated based on predetermined criteria; transmitting the generated motion vectors and one or more invalidators to a client, wherein the generated motion vectors and invalidators are cached at the client; transmitting an instruction to the client to receive input from a user, wherein the client is configured to match the input to cached motion vectors or invalidators; and transmitting an instruction to the client to apply the matched motion vectors or invalidators to effect motion compensation in a graphic interface, which are not present in Groups I & III.

The special technical features of Group III include at least transmitting a previously generated motion vector library from a server to a client, wherein the motion vector library is configured to be stored at the client; transmitting an instruction to the client to monitor for input data from a user; transmitting an instruction to the client to calculate a motion estimate from the input data; and transmitting an instruction to the client to update the stored motion vector library based on the input data, wherein the client is configured to apply the stored motion vector library to initiate motion in a graphic interface prior to receiving actual motion vector data from the server, which are not present in Groups I-II.

The common technical features shared by Groups I-III are a computer-implemented method comprising; a system, wherein, over a network, a server; motion vectors; user input; transmitting an instruction to the client; and a graphic interface.

However, these common features are previously disclosed by WO 2009/042433 A2 to MICROSOFT CORPORATION (hereinafter "Microsoft"). Microsoft discloses a computer-implemented method comprising (using a thin client, paragraph [0043]); a system, wherein, over a network, a server (a server provides graphical interface data to a client, paragraph [0043]); motion vectors (motion vectors, paragraph [0054]); user input (user input, paragraph [0053]); transmitting an instruction to the client (commands are sent to a thin client from the server, paragraph [0043]); and a graphic interface (user interfaces with graphical data, paragraph [0043]).

Since the common technical features are previously disclosed by the Microsoft reference, these common features are not special and so Groups I-III lack unity.

フロントページの続き

(31)優先権主張番号 62/640,945
(32)優先日 平成30年3月9日(2018.3.9)
(33)優先権主張国・地域又は機関
米国(US)
(31)優先権主張番号 62/644,164
(32)優先日 平成30年3月16日(2018.3.16)
(33)優先権主張国・地域又は機関
米国(US)

(81)指定国・地域 AP(BW,GH,GM,KE,LR,LS,MW,MZ,NA,RW,SD,SL,ST,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,RU,TJ,TM),EP(AL,AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HR,HU,IE,IS,IT,LT,LU,LV,MC,MK,MT,NL,NO,PL,PT,RO,RS,SE,SI,SK,SM,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,KM,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AO,AT,AU,AZ,BA,BB,BG,BH,BN,BR,BW,BY,BZ,CA,CH,CL,CN,CO,CR,CU,CZ,DE,DJ,DK,DM,DO,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,GT,HN,HR,HU,ID,IL,IN,IR,IS,JO,JP,KE,KG,KH,KN,KP,KR,KW,KZ,LA,LC,LK,LR,LS,LU,LY,MA,MD,ME,MG,MK,MN,MW,MX,MY,MZ,NA,NG,NI,NO,NZ,OM,PA,PE,PG,PH,PL,PT,QA,RO,RS,RU,RW,SA,SC,SD,SE,SG,SK,SL,SM,ST,SV,SY,TH,TJ,TM,TN,TR,TT

Fターム(参考) 5C164 FA22 MB13S SB08P SB41S UB10S UB23P UB41S YA11

【要約の続き】
ライアントは、実際の動きベクトルデータを受信する前に動きを開始するように、記録されたライブラリを適用する。