(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 660 434**

(51) Int. CI.:

G06T 15/00 (2011.01)
G06F 15/78 (2006.01)
G06F 9/46 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **19.04.2005 E 05103115 (1)**

(97) Fecha y número de publicación de la concesión europea: **03.01.2018 EP 1594091**

(54) Título: **Sistema y procedimiento para proporcionar una canalización de gráficos mejorada**

(30) Prioridad:

03.05.2004 US 567490 P
03.09.2004 US 934249

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
22.03.2018

(73) Titular/es:

MICROSOFT TECHNOLOGY LICENSING, LLC (100.0%)
One Microsoft Way
Redmond, WA 98052, US

(72) Inventor/es:

PATEL, AMAR;
BOYD, CHARLES N.;
BLYTHE, DAVID R.;
NOYLE, JEFF M. J.;
TOELLE, MICHAEL A. y
WRIGHT, STEPHEN H.

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

DESCRIPCIÓN

Sistema y procedimiento para proporcionar una canalización de gráficos mejorada

Campo de la invención

La presente invención está dirigida a una unidad de procesamiento de gráficos (GPU) que permite una canalización de gráficos mejorada usando software desarrollado para hardware para añadir una funcionalidad nueva y mejorada en la canalización de gráficos.

Antecedentes

Los sistemas informáticos se usan comúnmente para mostrar objetos gráficos en una pantalla de visualización. El propósito de los gráficos informáticos tridimensionales (3-D) es generalmente crear imágenes bidimensionales (2-D) en una pantalla de ordenador que representen de manera realista un objeto u objetos en tres dimensiones. En el mundo real, los objetos ocupan tres dimensiones, teniendo una altura real, una anchura real y una profundidad real. Una fotografía es un ejemplo de una representación bidimensional de un espacio tridimensional. Los gráficos informáticos en 3-D son generalmente como una fotografía, ya que representan un mundo en 3-D en el espacio en 2-D de una pantalla de ordenador, excepto que la imagen subyacente generalmente está modelada con geometría tridimensional y texturas superficiales.

Las imágenes creadas con gráficos informáticos en 3-D se utilizan en una amplia gama de aplicaciones, desde vídeojuegos de entretenimiento hasta simuladores de vuelo de aeronaves, para retratar de forma realista la vista individual de una escena en un punto dado en el tiempo. Ejemplos bien conocidos de gráficos informáticos en 3-D incluyen efectos especiales en películas de Hollywood como Terminator II, Jurassic Park, Toy Story y similares.

Una industria que ha visto particularmente una enorme cantidad de crecimiento en los últimos años es la industria de los juegos de ordenador y la actual generación de juegos de ordenador se aplican técnicas de gráficos en 3-D de una manera cada vez mayor. Al mismo tiempo, la velocidad de juego se realiza cada vez más rápido. Esta combinación ha alimentado una necesidad genuina de una representación rápida y flexible de gráficos en 3-D en sistemas relativamente baratos.

Para crear una representación gráfica informática en 3-D, típicamente, los objetos a ser representados se representan como modelos matemáticos dentro del ordenador. Por ejemplo, los modelos en 3-D pueden estar formados por puntos geométricos dentro de un sistema de coordenadas que consiste en un eje x, y y z, por ejemplo, que corresponden a la anchura, altura y profundidad, respectivamente. Los objetos están definidos por una serie de puntos, llamados vértices. La ubicación de un punto o vértice se define por sus coordenadas x, y y z (u otro sistema de coordenadas). En terminología de gráficos, un vértice es un punto, dos vértices definen una línea, o un segmento de línea, y tres vértices definen un triángulo donde los tres son "primitivos". Cuando tres o más de estos puntos están conectados, se forma un polígono, siendo el triángulo el polígono más simple.

La representación y visualización de gráficos tridimensionales (3-D) en una pantalla implica típicamente muchos cálculos y computaciones. En un sistema de gráficos simple, dichos cálculos se producen según algún nivel de procesamiento cooperativo o compartido mediante la unidad de procesamiento central (CPU) y la unidad de procesamiento de gráficos (GPU). En un escenario ejemplar, después de procesar las instrucciones y realizar algunos cálculos iniciales en la CPU, un conjunto de puntos de coordenadas o vértices que definen el objeto a representar se almacenan en la memoria de vídeo para su posterior procesamiento por la GPU en la canalización de gráficos. Un teselador puede dividir los datos gráficos en polígonos simples de acuerdo con algoritmos predeterminados diseñados para cubrir eficientemente la superficie del objeto que se está representando, esto se conoce como teselación. Actualmente, en la mayoría de los canales gráficos, los datos pueden ser operados por uno o más sombreadores de procedimiento, dependiendo de las instrucciones que se suministrar a la GPU.

Los sombreadores de procedimiento son subunidades de procesamiento especializadas de la GPU para realizar operaciones especializadas en datos de gráficos. Un ejemplo de un sombreador de procedimiento es un sombreador de vértices, que generalmente opera en vértices. Por ejemplo, un sombreador de vértices puede aplicar cálculos de posiciones, colores y coordenadas de texturización a vértices individuales. Los sombreadores de vértices realizan cálculos de funciones fijas o programables en secuencias de vértices especificadas en la memoria de vídeo de la canalización de gráficos. Los sombreadores de vértices no pueden generar geometría adicional (vértices), sino que operan en los vértices que se especifican para la transformación algorítmica mediante el programa descargado al sombreador de vértices desde el servidor.

Generalmente montado después del sombreador de vértices, otro ejemplo de un sombreador de procedimiento es un sombreador de píxeles. Por ejemplo, las salidas del sombreador de vértices se pueden pasar a través de una unidad no programable, denominada motor de configuración, que es una unidad de sombreador no procedimental que define la gama de píxeles en los que opera el sombreador de píxeles. La gama de píxeles puede ser operada por un sombreador de píxeles, que a su vez opera en cada píxel individual. Desde el sombreador de píxeles, los valores de salida se envían a la memoria intermedia de tramas donde esperan para visualizarse en la pantalla, o esperan a ser recuperados por el sistema de servidor. Aunque la geometría en 3-D generalmente se combina con

triángulos (conjuntos de 3 vértices), actualmente, no hay sombreadores de procedimiento que operen directamente sobre triángulos como tales (conjuntos de tres vértices relacionados), o segmentos de líneas, o líneas (conjuntos de dos vértices relacionados).

Cuando es programable, se observa que el término "sombreador" está sujeto a uso engañoso porque un "sombreador" puede referirse a la subunidad de hardware de la GPU que realiza el sombreado y un "sombreador" puede referirse al conjunto de instrucciones o piezas descargadas a la GPU que se cargan posteriormente en la memoria, por ejemplo, almacenamiento de registro, utilizado por el sombreador (hardware) para realizar el sombreado. El término también puede referirse a ambos trabajando juntos. Por lo tanto, aunque a veces la palabra se usa en este documento genéricamente para referirse a cualquiera de los dos, y, en consecuencia, debe tomarse que abarca ambos significados, donde el término "subunidad" también se usa en relación con el término sombreador, los términos deben interpretarse para referirse a la subunidad de la GPU que realiza el procesamiento asociado con el sombreado.

También se hace notar que el término "memoria intermedia de tramas" en las arquitecturas de gráficos de hoy en día no se refiere en general a una porción predividida, preasignada y predefinida de memoria de vídeo reservada exclusivamente para la salida del motor de configuración o de píxeles unidad de procesamiento de gráficos, sino que la memoria intermedia de cuadros generalmente se refiere a cualquier memoria (generalmente memoria de vídeo incluida para la interoperación con la GPU) utilizada en conexión con procesos de rasterización y/o conversión de salida de digital a analógico (DAC). A este respecto, si bien el término rasterización a veces se usa de manera más general, el procesamiento realizado en relación con el procesamiento de píxeles o el procesamiento del motor de configuración en la canalización de gráficos generalmente se conoce como rasterización. El escaneo o DAC de salida, por otro lado, es el proceso de transmisión de señales a un monitor o LCD basado en el contenido de la memoria intermedia de tramas.

Hoy en día, los valores en la memoria intermedia de tramas se pueden usar para otros fines que, para su visualización en una pantalla, pero tal uso es limitado. Por ejemplo, se puede desear devolver los valores a una aplicación en el servidor después de que hayan sido operados por un sombreador de píxeles o un sombreador de vértices para guardar, imprimir, realizar transformaciones adicionales, etc. En estos casos, los valores "técnicamente" pueden ser leídos; sin embargo, recuperar los valores en la memoria de vídeo, aparte de las porciones limitadas dedicadas a almacenar imágenes para mostrar, es una tarea extremadamente ardua que requiere cooperación con la memoria del servidor, la CPU y conocimiento avanzado sobre cómo la memoria de vídeo está dispuesta para su uso en conexión con el hardware de gráficos particular. Esto es especialmente cierto si se debe leer cualquier parte de la memoria de vídeo que no sea la memoria intermedia de tramas. Aunque no es imposible, en la práctica, está lejos de ser sencillo leer el almacenamiento intermedio de la memoria de vídeo utilizada para la salida mediante el teselador, sombreadores de vértices y sombreadores de píxeles. En algunos casos, las salidas de teseladores, sombreadores de vértices y sombreadores de píxeles pueden no llegar nunca a la memoria de vídeo, sino que pueden limitarse a una trayectoria de datos especializada para usar entre etapas de procesamiento de gráficos de la GPU (por ejemplo, la salida de un sombreador de vértices se transmite generalmente a un sombreador de píxeles a través de una trayectoria predefinida sin alcanzar la memoria de vídeo). Las operaciones de la GPU se producen tan rápido y de acuerdo con la arquitectura especializada que la lectura del almacenamiento intermedio de la memoria de vídeo en los canales de gráficos actuales no es factible hasta que la salida llegue a la memoria intermedia de tramas para la rasterización u otra operación.

Aunque la memoria intermedia de tramas, que recibe la salida del sombreador de píxeles en un sistema convencional, se puede acceder de una manera conocida por la CPU del servidor para guardar (o de otro modo operar en) los datos de gráficos, tales como imágenes, procesados por la GPU, aún se debe acceder a los datos y recuperarlos en la CPU del servidor para tales operaciones adicionales. La figura 1B ilustra lo que se debe hacer con la salida de datos desde una memoria intermedia de tramas para lograr una operación adicional sobre los datos mediante la canalización de gráficos. Después de la operación de un ciclo de procesamiento de canal representado por las operaciones de la GPU y después de recuperar los valores de la memoria intermedia de tramas a la CPU del servidor, la CPU coopera para enviar los valores nuevamente a través de la canalización de gráficos para cualquier procesamiento posterior. Por lo tanto, lo que se desea en el estado de la técnica son dos cosas principales: (1) la capacidad de reutilizar, es decir, volver a procesar, los datos que se envían desde subcomponentes, como un sombreador de vértices o un sombreador de píxeles, dentro de la canalización de gráficos antes de llegar a la memoria intermedia de tramas y (2) la capacidad de hacerlo de manera recursiva sin implicar los recursos del servidor y la memoria.

Como se ilustra en la figura 1A, los sistemas informáticos están divididos entre la CPU del servidor y el hardware de gráficos. La CPU facilita la realización de llamadas a API de gráficos mediante aplicaciones y servicios que solicitan su uso. Convencionalmente, la aplicación y los controladores se encuentran en el lado de la CPU y la información de esas fuentes se envía para mostrarse en un monitor. En primer lugar, la información se envía desde la CPU a la GPU, según el paquete mediante la CPU según las API. Luego, la información desde la aplicación espera en la memoria hasta que se accede mediante el sombreador de vértices. Después de que el sombreador de vértices concluye sus operaciones, la información, como salida desde el sombreador de vértices, se envía a través de una trayectoria de datos especial al sombreador de píxeles (colocar la salida en la memoria de vídeo es generalmente demasiado lento) hasta que se accede mediante el sombreador de píxeles, y el sombreador de vértices permanece

inactivo o continúa procesando más datos. Después de que el sombreador de píxeles haya realizado sus operaciones, la información se coloca en una memoria intermedia de tramas para escanearse en una pantalla o enviar al servidor para su posterior operación.

Se han desarrollado API especializados en gráficos 3-D que exponen la funcionalidad especializada de sombreadores de vértices y de píxeles de hoy en día. En este sentido, un desarrollador puede descargar instrucciones, o pequeños programas, a una unidad de sombreador de vértices que efectivamente programa el sombreador de vértices para realizar un comportamiento especializado. Por ejemplo, las API exponen la funcionalidad asociada con un mayor número de registros en los sombreadores de vértices, por ejemplo, la funcionalidad de sombreado de vértices especializada con respecto a los números de coma flotante en un nivel de registro. Además, es posible implementar un conjunto de instrucciones que hace que el sombreador de vértices extremadamente rápido devuelva solo la porción fraccionaria de los números de coma flotante. Se puede lograr una variedad de funcionalidades descargando estas instrucciones, suponiendo que no se excede el límite de recuento de instrucciones del sombreador de vértices y el almacenamiento de registro asociado.

En particular, la funcionalidad de una etapa de sombreador de vértices es para la transformación, la iluminación y la transformación de vez en cuando de la textura. Básicamente, la transformación es tomar datos de posición, los datos sobre dónde deben estar los vértices cuando se muestran y transformarlos en datos para el monitor, un espacio de pantalla bidimensional. Tradicionalmente, los procesos de transformación de vértices pasan datos de posición sin modificación o modifican los datos usando matrices. La etapa del sombreador de vértices generalmente se limita a realizar funciones de transformación, funciones de iluminación y algunas funciones de textura.

Dado que los juegos aumentan el nivel de detalle de gráficos, aumenta la densidad de polígono y la iluminación y otras técnicas de sombreado de vértices se vuelven más importantes, como una etapa de procesamiento de vértices. La iluminación estática, una forma popular de iluminación debido a su alta calidad, generalmente se calcula en el sombreador de vértices y se almacena como una textura. Debido a que coloca toda la información de iluminación en texturas, es difícil modificar la información durante el tiempo de ejecución haciendo posible la iluminación dinámica solo si las instrucciones por vértice se dan de antemano. Ocasionalmente, el sombreador de vértices aplica una transformación de matriz a coordenadas de vértice para coordenadas de textura. Esto se produce generalmente para el mapeo de reflexión esférica y cúbica y la animación de texturas.

Los tipos típicos de iluminación realizadas por un sombreador de vértice incluyen: posicional, direccional o proyector. Para agregar esta iluminación, los cálculos matemáticos, principalmente la manipulación de la matriz, cambian los vértices para reflejar un tipo de iluminación definido en una aplicación. Hay diferentes luces, generalmente imitando la realidad donde las fuentes de luz como la luz del sol y una luz de la calle tienen diferentes propiedades. Estas luces pueden ser posicionales, direccionales o proyectores que crean una multitud de combinaciones para que la etapa del sombreador de vértices las calcule.

Por lo tanto, el procesamiento de la geometría realizado usando un sombreador de vértices incluye algún tipo de transformación a aplicar a los datos, el entorno de iluminación para vértices, y el material para la transformación de la textura. La función fija y los sombreadores de vértices programables generalmente funcionan de esas maneras en esa etapa en el canal. Si una aplicación tiene más información para ser procesada en estas áreas, habrá un cuello de botella en el sombreador de vértices y otros componentes de la canalización de gráficos permanecerán inactivos.

Con respecto a los sombreadores de píxeles, la funcionalidad de sombreado de píxeles especializada se puede lograr mediante la descarga de instrucciones para el sombreador de píxeles. Por ejemplo, se expone la funcionalidad que proporciona un mecanismo de interpolación lineal en el sombreador de píxeles. Además, la funcionalidad de muchos modificadores de operación diferentes está expuesta a los desarrolladores en relación con los conjuntos de instrucciones adaptados a los sombreadores de píxeles. Por ejemplo, negar, reasignar, sesgar y otras funcionalidades son extremadamente útiles para muchas aplicaciones gráficas para las cuales es deseable un sombreado de píxeles eficiente, sin embargo, como se ejecutan como parte de una única instrucción, se expresan mejor como modificadores de esa instrucción. En resumen, la funcionalidad anterior es ventajosa para operaciones gráficas, y su incorporación funcional en conjuntos de instrucciones de sombreadores de píxeles y de vértices ya especializados agrega un enorme valor desde la perspectiva de la facilidad de desarrollo y el rendimiento mejorado. Por lo tanto, se puede lograr una variedad de funcionalidades descargando estas instrucciones, asumiendo que no se excede el límite de recuento de instrucciones y otras limitaciones de hardware del sombreador de píxeles.

Aunque el sombreador de píxeles no realiza algunas operaciones de la matriz (por ejemplo, las transformaciones de vectores para iluminación), puede ser útil pensar en la funcionalidad de un sombreador de píxeles que tiene capacidades computacionales matemáticas más o menos sencillas, en contraposición a los cálculos de matriz más complejos realizados por el sombreador de vértices para transformar vértices, en los que cada vértice puede estar representado por muchos valores que requieren procesamiento. La matemática involucrada con un sombreador de píxeles es probable que sea como cálculos de coma flotante y estos cálculos sombrean píxeles para crear reflectividad, brillo y relieve. Existen limitaciones en el número de cálculos que se pueden realizar a medida que se pasa información a través del sombreador de píxeles. Debido a estas limitaciones, parte de la información requiere operaciones de múltiples pases para crear múltiples texturas en cada píxel. Y cada pase abarca varios ciclos de reloj en el canal.

El sombreador de vértices calcula matrices sobre un vértice dado, mientras que el sombreado de píxeles puede calcular operaciones de punto más flotantes. Hay casos en que un programador puede querer ver las transformaciones matemáticas del vértice y no los valores como una visualización en pantalla. En este punto, un programador tendría que leer la memoria desde la memoria intermedia de tramas que, como se describió, es prohibitiva.

Los sombreadores de píxeles y los sombreadores de vértices son capaces de operar en píxeles y vértices. En la programación de gráficos, las primitivas son las formas básicas a partir de las cuales se crea cualquier gráfico tridimensional dado. Las primitivas usadas y definidas regularmente incluyen un vértice, una línea y un triángulo. En la actualidad, los sombreadores de píxeles y los sombreadores de vértices pueden operar con las instrucciones dadas en un vértice. Incluso si los sombreadores son programables, los sombreadores solo pueden operar en vértices o puntos, como primitivos.

Cuando estos sombreadores específicos, cualquiera de los sombreadores de píxeles o de vértices, operan en una canalización de gráficos, hay cuellos de botella regulares debido a las operaciones que se producen en cada etapa. Estos cuellos de botella pueden resolverse si un programador intenta limitar y equilibrar las instrucciones enviadas a cada sombreador para cada vértice. Sin embargo, al diseñar gráficos para una pantalla intrincada en 3-D, como un juego, equilibrar el número de instrucciones se convierte en una tarea abrumadora. La mayoría de los programadores no implementa ningún tipo de optimización hasta que los gráficos son notablemente más lentos en un programa determinado. Incluso en los mejores esquemas de optimización que un programador pueda usar, el hardware se encontrará inactivo y sin usar mientras espera el paso de la información, simplemente debido a la variación asociada con los diferentes tipos de tareas solicitadas de un subsistema de gráficos. Además, para optimizar cualquier programa de gráficos, un programador primero debe identificar la fuente del cuello de botella, que puede ser una tarea tediosa. Por lo tanto, sería deseable poder reconfigurar dinámicamente los núcleos de un subsistema de gráficos de modo que el procesamiento central de los datos se adapte automáticamente a la tarea que se solicita que se realice.

Por lo tanto, la representación de los datos de los gráficos en un sistema informático es un conjunto de procesos que consumen muchos recursos. El proceso de sombreado, es decir, el proceso de realizar algoritmos complejos sobre conjuntos de estructuras de datos gráficos especializados, usados para determinar valores para ciertas primitivas, como el color, etc. asociados con las estructuras de datos gráficos, ejemplifica este cálculo intensivo y proceso complejo. En general, el proceso de sombreado se ha normalizado en cierto grado. Al pasar el código fuente diseñado para trabajar con un sombreador en una aplicación, un sombreador se convierte en un objeto que la aplicación puede crear/utilizar para facilitar el dibujo eficiente de gráficos de vídeo complejos. Los sombreadores de vértices y los sombreadores de píxeles son ejemplos de dichos sombreadores.

Por lo tanto, la introducción de operaciones programables en una base por vértice y por píxel se ha vuelto en una difusión más amplia en hardware gráfico moderno. Esta capacidad de programación general permite el potencial de algoritmos creativos limitados a niveles de rendimiento aumentados. Sin embargo, además de los mencionados anteriormente, existen algunas limitaciones a lo que se puede lograr hoy. Normalmente, con los canales de representación actuales en los sombreadores de vértices y de píxeles, se ingresa una secuencia de datos de geometría al sombreador de vértices para realizar alguna operación de los vértices, como resultado de lo cual los datos de geometría se transforman en datos de píxeles, dando como resultado una secuencia de datos de píxeles. El sombreador de vértices puede recibir instrucciones que programen el sombreador de vértices para realizar una funcionalidad especializada, pero existen límites para el tamaño y la complejidad de las instrucciones del sombreador de vértices. De forma similar, un sombreador de píxeles puede realizar opcionalmente una o más transformaciones a los datos que emiten una secuencia de datos de píxeles. El sombreador de píxeles también puede recibir instrucciones que programen el sombreador de píxeles para realizar una funcionalidad especializada, pero existen límites para el tamaño y la complejidad de las instrucciones del sombreador de píxeles.

Hoy en día, los programadores utilizan regularmente sombreadores de vértices y sombreadores de píxeles. Sin embargo, el hardware programable actual tiene mecanismos programables algo limitados que no permiten que un programador especifique la reutilización de valores antes de llegar al final del canal antes de que los valores sean rasterizados. Los programadores pueden intentar equilibrar el uso de los diferentes componentes en la canalización de gráficos para evitar los cuellos de botella en uno o más de los sombreadores, sin embargo, el hardware de gráficos es fijo. Aunque algunos hardware especializados se construyen para tareas especializadas que tienen combinaciones especializadas y disposiciones de sombreadores de procedimientos para esas tareas especializadas, el hardware no se puede reorganizar para otras tareas. Por lo tanto, cuando se realizan tareas distintas de aquellas tareas especializadas, y la información debe modificarse específicamente mediante instrucciones para solo uno de los sombreadores, la modificación puede tener que esperar, tal vez de manera inaceptable, dependiendo de la aplicación. Además, aunque los sombreadores pueden operar los vértices, el programador no puede especificar mediante las API de gráficos operaciones que funcionen directamente en otras primitivas como tales. En otras palabras, con la excepción de los vértices, un programador no puede empaquetar primitivas arbitrariamente para el procesamiento en la canalización de gráficos como primitivas.

La teselación es un proceso que típicamente se produce al comienzo de una canalización de gráficos que implica cubrir una región geométrica limitada sin huecos o solapamientos mediante figuras de plano congruentes de un tipo

o unos pocos tipos. Aunque los teseladores existentes implementan algunos algoritmos básicos para crear una cuadrícula de vértices nuevos basados en un puñado de vértices de puntos de control, el proceso se basa en algoritmos prefijados y se limita al extremo frontal del canal y, por lo tanto, no es programable para crear geometría arbitraria adicional en el medio del canal después del procesamiento mediante un sombreador de procedimiento. Además, una vez que los vértices son generados por el teselador, no hay otro lugar en el canal donde se puedan generar los vértices. En otras palabras, los sombreadores de vértices actuales pueden recibir, procesar y enviar vértices, pero no pueden generar nuevos vértices.

Por consiguiente, sería deseable implementar sistemas y procedimientos que superen las deficiencias de la actual capacidad de programación en relación con arquitecturas de canales de gráficos presentes, API y hardware debido a las limitaciones en el recuento de instrucciones, limitaciones en forma de salida y falta de intercambio de datos en el canal.

Haaker T. et al: "A distributed frame buffer within a window-oriented high performance graphics system", Advances in Computer Graphics Hardware IV, Springer-Verlag, páginas 261 a 273, describe que dado que una memoria intermedia de tramas debe estar estrechamente acoplada a un procesador de rasterización, la arquitectura de la memoria intermedia de tramas tiene una gran influencia en el diseño del hardware y del software del sistema general de gráficos. Por otro lado, la generación de imágenes, la manipulación y la rasterización comprenden muchas tareas diferentes y cálculos complejos. Las etapas del canal de salida de generación de imágenes comprenden una sección de geometría, una sección de representación y una memoria intermedia de tramas. La sección de geometría comprende las transformaciones de modelado y visualización, la proyección en perspectiva y el recorte de las primitivas gráficas. La sección de representación consiste en la evaluación de un modelo de iluminación, del proceso de rasterización, sombreado y eliminación de superficies ocultas. Al final del proceso de renderizado, los píxeles se almacenan en la memoria intermedia de tramas y se muestran en el monitor. Se necesita una arquitectura de sistema donde muchos procesadores realicen todas estas tareas para lograr una rápida generación y manipulación de imágenes. Con una arquitectura paralela, el diseñador divide la carga computacional. En una arquitectura paralela para la sección geométrica, la tarea computacional se distribuye a un número configurable de procesadores de geometría MIMD. En un enfoque adicional, esto se aplica a la geometría y a la sección de representación.

El documento US 2002/033817 A1 proporciona una interfaz de software mejorada como una capa entre los desarrolladores de aplicaciones y la canalización de gráficos que convierte y procesa los datos de gráficos. Se proporciona un API que permite la capacidad de programación de un chip 3-D, en el que los elementos de programación o algorítmicos escritos por el desarrollador pueden descargarse al chip, programando así el chip para realizar estos algoritmos.

El documento US 6.483.505 B1 se refiere al procesamiento de píxeles de paso múltiple. Se recibe una secuencia de comandos que incluye una pluralidad de comandos de dibujo donde los comandos de dibujo de paso múltiple incluidos en la secuencia incluyen un número de conjuntos de información de estado y una o más primitivas de gráficos. Para una operación de procesamiento de píxeles de paso múltiple, la canalización de gráficos que realiza el procesamiento de píxeles se configura primero usando un primer conjunto de información de estado incluido en los conjuntos de información de estado para la operación de paso múltiple. Una vez que se ha configurado la interconexión de gráficos, al menos una porción del procesamiento que se realizará para el comando de dibujo se realiza utilizando la interconexión de gráficos configurada por este primer conjunto de información de estado. Los datos resultantes producidos a través de este procesamiento se almacenan como datos intermedios. La canalización de gráficos se reconfigura luego utilizando un conjunto posterior de información de estado correspondiente al comando de dibujo de paso múltiple. Una vez que la canalización de gráficos se ha reconfigurado, se realiza el segundo paso. El segundo paso incluye el procesamiento de la porción de los datos gráficos procesados durante el primer paso usando la canalización de gráficos configurada por el siguiente conjunto de información de estado.

Singh S. et al: "Virtual Hardware for Graphics Applications using FPGAs" IEEE Workshop en Napa Valley, 1994, IEEE Comput. Soc., páginas 49 a 58 se refiere a hardware virtual para aplicaciones gráficas que utilizan disposiciones de puertas programables de campo (FPGA). Un enfoque sencillo para usar dispositivos FPGA para implementar operaciones gráficas es simplemente reemplazar el coprocesador gráfico con una red FPGA. Muchas variaciones interconectadas son posibles. Se presenta una matriz de dispositivos FPGA como una analogía de la memoria física. Un medio de almacenamiento secundario puede ser un sistema basado en ROM local que almacena una gran cantidad de configuraciones de FPGA. Esta organización abre una nueva metodología para diseñar un hardware. En lugar de diseñar una biblioteca de circuitos gráficos independientes, se propone el diseño de un enorme circuito gráfico virtual que realiza una gran cantidad de operaciones gráficas.

Styles H. et al: "Customising Graphics Applications: Techniques and Programming Interface", Field-Programmable Custom Machines, Abril 17, 2000, IEEE Symposium, páginas 77 a 87, ISBN: 0-7695-0871-5 se relaciona con la personalización de aplicaciones gráficas. Se afirma que una arquitectura de representación FPGA se puede personalizar rápidamente para acelerar los efectos de asignación de textura no soportados. Además, se revela que los renderizadores ASIC desactivan circuitos para algoritmos extraños, mientras que una arquitectura de representación basada en FPGA puede instanciar una unidad de asignación de textura personalizada para cada aplicación. Por lo tanto, es más probable que este último soporte un conjunto de efectos de textura con mayor diversidad, ya que no es necesario tener hardware para el conjunto completo de efectos al mismo tiempo. Para cada

etapa de una canalización de gráficos, existen varios algoritmos que implementan la funcionalidad requerida, pero ofrecen características de rendimiento diferentes. Para maximizar el rendimiento de una aplicación específica, el algoritmo instanciado en hardware de gráficos debe coincidir mejor con las características de rendimiento de la aplicación. El procesador de gráficos basado en FPGA se puede modificar fácilmente para mejorar el equilibrio de carga específico de la canalización de gráficos. En la técnica descrita, los recursos de hardware atribuidos a cada etapa de rasterización se optimizan para ajustarse mejor a los requisitos de rendimiento de hardware y software. El renderizador de hardware está personalizado para maximizar el rendimiento de cada aplicación, y el FPGA se configura antes de la ejecución. El uso de la reconfiguración para mejorar el rendimiento está motivado por la diferencia en el rendimiento de diferentes aplicaciones gráficas, dependiendo de los efectos gráficos aplicados durante la representación, y del comportamiento de la aplicación en la presentación de primitivas gráficas a la línea de representación. Se han implementado dos rasterizadores: uno está optimizado para un alto rendimiento poligonal y el otro para una tasa de llenado.

El documento US 2003/169269 A1 se refiere a una técnica de optimizar el procesamiento de gráficos para un sistema multiprocesador. Los procesadores de geometría están conectados a procesadores gráficos mediante canales de comunicación y memorias intermedias configurables. Los canales de comunicación configurables y las memorias intermedias permiten a todos y cada uno de los procesadores de geometría enviar datos a todos y a cada uno de los procesadores gráficos. En operación, los objetos 3D definidos se pasan a los procesadores de geometría para convertirlos en listas de visualización 2D de primitivas de gráficos. Las listas de visualización se pasan luego a los procesadores gráficos a través de los canales de comunicación y memorias intermedias configurables. Los procesadores gráficos representan las listas de visualización en 2D de primitivas gráficas y pueden aplicar textura y sombra al representar. Los procesadores gráficos pasan los datos de rendimiento a un procesador de control de representación. Estos datos de rendimiento se procesan dentro del procesador de control de representación. Los resultados se envían por un canal de retroalimentación. Esta retroalimentación se utiliza para realizar la asignación del procesador y la configuración de los canales de comunicación configurables y las memorias intermedias.

Sumario de la invención

El objetivo de la presente invención es mejorar el rendimiento de los canales gráficos de la técnica anterior.

Este objetivo se resuelve mediante el objeto de las reivindicaciones independientes. Realizaciones preferidas se definen mediante las reivindicaciones dependientes.

En consideración de los inconvenientes antes identificados de la técnica, la presente invención proporciona sistemas y procedimientos para optimizar el uso de hardware en la canalización de gráficos teniendo un núcleo común, que puede funcionar como un sombreador de píxeles, un sombreador de vértices o un sombreador de geometría, donde el sombreador de geometría es un sombreador programable que puede operar en primitivas, incluyendo triángulos, como entrada y salida para que los datos gráficos puedan funcionar de forma más directa sin transformar los datos en vértices u otro formato intermedio. Además, con la invención, se pueden generar nuevos vértices dentro del canal, ya sea en la etapa de teselación o mediante un sombreador de geometría. En una realización, se puede utilizar un teselador para generar la geometría dentro del canal programáticamente después de que un sombreador de procedimiento haya operado en datos gráficos y se puede usar un sombreador de geometría para complementar la geometría existente de una manera programable para lograr la alteración sobre la marcha de las formas y los efectos gráficos aplicados dentro del canal.

Ventajosamente, el núcleo común es programable o dinámicamente configurable como un sombreador de píxeles, un sombreador de vértices, o un sombreador de geometría, por lo que una canalización de gráficos puede emplear todos los sombreadores de píxeles, todos los sombreadores de vértices, o todos los sombreadores de geometría, o una combinación de sombreadores basados en la optimización de las necesidades computacionales para una aplicación específica. Los núcleos comunes se pueden configurar dinámicamente (y reconfigurar) dependiendo de las demandas en la canalización de gráficos hechas por el servidor.

Con respecto a un sombreador de geometría, a través de una salida de secuencia, las API de gráficos y el hardware asociado de la invención permiten a los desarrolladores de aplicaciones recircular de forma programable datos una vez dentro de la canalización de gráficos sin implicar la ayuda de la CPU en la lectura de la memoria de vídeo y volver a empaquetar los datos para su reutilización mediante el canal. En una realización, la invención incluye un índice único para cada primitiva que entra en la canalización de gráficos que ayuda en la programación de elementos de trabajo que alcanzan la GPU, y la precisión en cómputos de punto flotante y entero se ajusta a los estándares IEEE. El sombreador de geometría también puede procesar información sobre primitivas o píxeles adyacentes de una secuencia de datos de gráficos que procesa la GPU.

Otras ventajas y características de la invención se describen a continuación.

Breve descripción de los dibujos

Los sistemas y procedimientos para una canalización de gráficos mejorada, de acuerdo con la presente invención, se describen adicionalmente con referencia a los dibujos adjuntos, en los que:

La figura 1A es un diagrama de bloques que ilustra una implementación de la técnica anterior de una canalización de gráficos;
La figura 1B es un diagrama de bloques que ilustra una implementación de la técnica anterior de cálculos de paso múltiple;
La figura 2A es un diagrama de bloques que representa un entorno de red ejemplar que tiene una variedad de dispositivos informáticos en los que se puede implementar la presente invención;
La figura 2B es un diagrama de bloques que representa un dispositivo informático no limitativo a modo de ejemplo en el que se puede implementar la presente invención;
La figura 2C es un diagrama de bloques que representa un ordenador remoto en red, no limitativo, a modo de ejemplo, en el que puede implementarse la presente invención;
La figura 3A es un diagrama de bloques que representa una implementación ejemplar no limitativa de los componentes inventados en una canalización de gráficos;
La figura 3B es un diagrama de bloques que representa una implementación ejemplar no limitativa del cálculo de paso múltiple que usa la funcionalidad de salida de secuencia;
La figura 4A es un diagrama de bloques que representa los componentes de hardware y software del núcleo común;
La figura 4B es un diagrama que ilustra la funcionalidad ejemplar del sombreador de geometría que puede recibir y emitir múltiples vértices desde una única topología;
La figura 4C es un diagrama que ilustra la generación de valores de índice por píxel o por primitiva para secuencias de gráficos procesadas de acuerdo con la canalización de gráficos de acuerdo con la invención; y
La figura 5 ilustra un diagrama de flujo ejemplar no limitativo para utilizar las capacidades mejoradas de la canalización de gráficos de la presente invención.

Descripción detallada de la realización ilustrativa

Sumario

Como se ha mencionado, la presente invención mejora una canalización de gráficos mediante la adición de una mayor funcionalidad para presentar hardware mediante el uso de un núcleo de sombreador común dinámicamente configurable para sombreadores de píxeles, sombreadores de vértices, y sombreadores de geometría. El núcleo común de los sombreadores permite una optimización simplificada, ya que unidades de hardware idénticas para los diferentes sombreadores proporcionan equilibrio de carga al reconfigurar o deshabilitar un sombreador como parte del canal cuando no es necesario, liberando recursos para etapas que permanecen activas. La invención introduce un sombreador de geometría que puede operar sobre primitivas que incluyen puntos, líneas y triángulos, y ventajosamente, se puede generar nueva geometría dentro del canal mediante un teselador que opera en uno de un número limitado de modos preestablecidos, o de manera relativamente arbitraria mediante un sombreador de geometría programable. Otra realización de la invención incluye una secuencia de salida donde los datos primitivos pueden transmitirse a una memoria intermedia para su reutilización en cualquier parte del canal, permitiendo la aplicación de algoritmos programáticos recursivos dentro del canal. Características adicionales incluyen un índice exclusivo para cada primitiva que pasa a través de la canalización de gráficos, y la realización de cálculos enteros y de coma flotante precisos dentro de la canalización de gráficos.

Entornos ejemplares en red y distribuidos

Un experto en la técnica puede apreciar que la invención se puede implementar en relación con cualquier ordenador u otro dispositivo cliente o servidor, que puede ser desplegado como parte de una red de ordenadores, o en un entorno informático distribuido. A este respecto, la presente invención se refiere a cualquier sistema o entorno informático que tenga cualquier cantidad de unidades de memoria o almacenamiento, y cualquier cantidad de aplicaciones y procesos que se produzcan en cualquier cantidad de unidades o volúmenes de almacenamiento, que puedan usarse en conexión con procesos para una implementación no limitativa de una canalización de gráficos mejorada, de acuerdo con la presente invención. La presente invención se puede aplicar a un entorno con ordenadores servidores y ordenadores cliente desplegados en un entorno de red o entorno informático distribuido, que tiene un almacenamiento remoto o local. La presente invención también se puede aplicar a dispositivos informáticos independientes, que tienen funcionalidades de lenguaje de programación, interpretación y ejecución para generar, recibir y transmitir información en conexión con servicios remotos o locales. En un entorno de juegos, una canalización de gráficos es particularmente relevante para los dispositivos informáticos que operan en una red o entorno informático distribuido, y así las técnicas de canalización gráfica mejoradas de acuerdo con la presente invención se pueden aplicar con gran eficacia en esos entornos.

La informática distribuida proporciona el intercambio de recursos y servicios informáticos por intercambio entre dispositivos y sistemas informáticos. Estos recursos y servicios incluyen el intercambio de información, almacenamiento en caché y almacenamiento en disco para archivos. La informática distribuida aprovecha la conectividad de red, lo que permite a los clientes aprovechar su potencia colectiva para beneficiar a toda la empresa. A este respecto, una variedad de dispositivos puede tener aplicaciones, objetos o recursos que pueden implicar los procesos de canalización de gráficos de la invención.

La figura 2A proporciona un diagrama esquemático de un entorno informático distribuido o en red ejemplar. El entorno informático distribuido comprende objetos 10a, 10b, etc. informáticos y objetos o dispositivos 110a, 110b, 110c, etc. informáticos. Estos objetos pueden comprender programas, procedimientos, almacenes de datos, lógica programable, etc. Los objetos pueden comprender porciones del mismo o diferentes dispositivos tales como PDA, dispositivos de audio/vídeo, reproductores de MP3, ordenadores personales, etc. Cada objeto puede comunicarse con otro objeto a través de la red 14 de comunicaciones. Esta red puede comprender otros objetos informáticos y dispositivos informáticos que proporcionan servicios al sistema de la figura 2A, y puede en sí misma representar múltiples redes interconectadas. De acuerdo con un aspecto de la invención, cada objeto 10a, 10b, etc. o 110a, 110b, 110c, etc. pueden contener una aplicación que podría hacer uso de una API u otro objeto, software, firmware y/o hardware, solicitar el uso de los procesos de canalización de gráficos de acuerdo con la invención.

También se puede apreciar que un objeto, tal como 110c, puede estar alojado en otro dispositivo 10a, 10b, etc., o 110a, 110b, etc. informático. Por lo tanto, aunque el entorno físico representado puede mostrar los dispositivos conectados como ordenadores, tal ilustración es meramente a modo de ejemplo y el entorno físico puede representarse o describirse alternativamente comprendiendo diversos dispositivos digitales tales como PDA, televisores, reproductores de MP3, etc., objetos de software tales como interfaces, objetos COM y similares.

Hay una variedad de sistemas, componentes y configuraciones de red que soportan entornos informáticos distribuidos. Por ejemplo, los sistemas informáticos pueden estar conectados entre sí mediante sistemas por cable o inalámbricos, redes locales o redes ampliamente distribuidas. Actualmente, muchas de las redes están conectadas a Internet, lo que proporciona una infraestructura para la informática ampliamente distribuida y abarca muchas redes diferentes. Cualquiera de las infraestructuras se puede usar para comunicaciones ejemplares hechas incidentales a una canalización de gráficos mejorada de acuerdo con la presente invención.

En los entornos de red domésticos, hay al menos cuatro medios de transporte de red dispares que pueden apoyarse en un protocolo único, tales como línea de alimentación, datos (tanto inalámbricos y por cable), voz (por ejemplo, teléfono) y medios de entretenimiento. La mayoría de los dispositivos de control domésticos, como los interruptores de luz y los electrodomésticos, pueden usar líneas eléctricas para conectividad. Los servicios de datos pueden ingresar al hogar como banda ancha (por ejemplo, DSL o módem por cable) y se puede acceder desde el hogar mediante conexión inalámbrica (por ejemplo, HomeRF o 802.11B) o conectividad por cable (por ejemplo, PNA residencial, Cat 5, Ethernet e incluso línea eléctrica). El tráfico de voz puede ingresar en la casa ya sea por cable (por ejemplo, Cat 3) o inalámbrico (por ejemplo, teléfonos celulares) y puede distribuirse dentro de la casa usando cableado Cat 3. Los medios de entretenimiento u otros datos gráficos pueden ingresar a la casa ya sea por satélite o por cable y, por lo general, se distribuyen en la casa mediante un cable coaxial. IEEE 1394 y DVI también son interconexiones digitales para agrupaciones de dispositivos multimedia. Todos estos entornos de red y otros que pueden surgir como estándares de protocolo pueden estar interconectados para formar una red, tal como una intranet, que puede estar conectada al mundo exterior a través de Internet. En resumen, existe una variedad de fuentes dispares para el almacenamiento y la transmisión de datos y, en consecuencia, para avanzar, los dispositivos informáticos requerirán formas de compartir datos, como los datos accedidos o utilizados como incidentes con los objetos del programa, que utilizan la canalización de gráficos mejorada según la presente invención.

Internet comúnmente se refiere a la colección de redes y pasarelas que utilizan el conjunto de protocolos TCP/IP, que son bien conocidos en la técnica de las redes informáticas. TCP/IP es un acrónimo de "Protocolo de control de transmisión/Protocolo de Internet". Internet se puede describir como un sistema de redes informáticas remotas distribuidas geográficamente e interconectadas por ordenadores que ejecutan protocolos de red que permiten a los usuarios interactuar y compartir información a través de la(s) red(es). Debido a este intercambio de información ampliamente difundido, las redes remotas, como Internet, hasta ahora han evolucionado hasta convertirse en un sistema abierto para el cual los desarrolladores pueden diseñar aplicaciones de software para realizar operaciones o servicios especializados, esencialmente sin restricciones.

Por lo tanto, la infraestructura de red permite una multitud de topologías de red como cliente/servidor, entre iguales, o arquitecturas híbridas. El "cliente" es un miembro de una clase o grupo que utiliza los servicios de otra clase o grupo con el que no está relacionado. Por lo tanto, en informática, un cliente es un proceso, es decir, aproximadamente un conjunto de instrucciones o tareas, que solicita un servicio proporcionado por otro programa. El proceso del cliente utiliza el servicio solicitado sin tener que "conocer" ningún detalle operativo sobre el otro programa o el servicio en sí. En una arquitectura de cliente/servidor, particularmente un sistema en red, un cliente suele ser un ordenador que accede a los recursos de red compartidos proporcionados por otro ordenador, por ejemplo, un servidor. En el ejemplo de la figura 2A, los ordenadores 110a, 110b, etc. pueden considerarse como clientes y los ordenadores 10a, 10b, etc. pueden considerarse como el servidor, donde el servidor 10a, 10b, etc. mantiene los datos que entonces se replican en los ordenadores cliente 110a, 110b, etc., aunque cualquier ordenador puede considerarse un cliente, un servidor o ambos, dependiendo de las circunstancias. Cualquiera de estos dispositivos informáticos puede estar procesando datos o solicitando servicios o tareas que pueden implicar las técnicas de programación de gráficos específicas para una implementación de la canalización de gráficos mejorada en la invención.

Un servidor es típicamente un sistema informático remoto accesible a través de una red local o remota, tal como

Internet. El proceso del cliente puede estar activo en un primer sistema informático, y el proceso del servidor puede estar activo en un segundo sistema informático, comunicándose entre sí a través de un medio de comunicación, proporcionando funcionalidad distribuida y permitiendo que múltiples clientes aprovechen las capacidades de recopilación de información del servidor. Cualquier objeto de software utilizado de acuerdo con las técnicas de programación de gráficos de la canalización de gráficos mejorada se puede distribuir a través de múltiples dispositivos u objetos informáticos.

Los clientes y los servidores se comunican entre sí utilizando la funcionalidad proporcionada por la(s) capa(s) de protocolo. Por ejemplo, el protocolo de transferencia de hipertexto (HTTP) es un protocolo común que se usa junto con la World Wide Web (WWW) o "la Web". Normalmente, una dirección de red de ordenador, como una dirección de protocolo de Internet (IP) u otra referencia, como un localizador universal de recursos (URL), se puede usar para identificar el servidor o los ordenadores cliente entre sí. La dirección de red puede ser referida como una dirección URL. La comunicación se puede proporcionar a través de un medio de comunicación, por ejemplo, cliente(s) y servidor(es) se pueden acoplar entre sí a través de conexión(es) TCP/IP para comunicación de alta capacidad.

Por lo tanto, la figura 2A ilustra un entorno en red o distribuido, con un servidor en comunicación con los ordenadores cliente a través de una red/bus, en el que puede emplearse la presente invención. Con más detalle, varios servidores 10a, 10b, etc. están interconectados a través de una red/bus de comunicaciones 14, que puede ser una LAN, WAN, intranet, Internet, etc., con una cantidad de dispositivos informáticos remotos o cliente 110a, 110b, 110c, 110d, 110e, etc., tales como un ordenador portátil, un ordenador de mano, un cliente ligero, un dispositivo conectado en red u otro dispositivo, tal como una vídeograbadora, televisor, horno, luz, calentador y similares de acuerdo con la presente invención. Por lo tanto, se contempla que la presente invención se pueda aplicar a cualquier dispositivo informático en conexión con el cual sea deseable implementar una interfaz gráfica que emplee una canalización de gráficos mejorada de la invención.

En un entorno de red en el que la red/bus de comunicaciones 14 es Internet, por ejemplo, los servidores 10a, 10b, etc., pueden ser servidores Web con los que los clientes 110a, 110b, 110c, 110d, 110e, etc. se comunican a través de cualquiera de una serie de protocolos conocidos, como HTTP. Los servidores 10a, 10b, etc. también pueden servir como clientes 110a, 110b, 110c, 110d, 110e, etc., como puede ser característico de un entorno informático distribuido. Las comunicaciones pueden ser por cable o inalámbricas, según corresponda. Los dispositivos de cliente 110a, 110b, 110c, 110d, 110e, etc. pueden comunicarse o no a través de la red/bus de comunicaciones 14, y pueden tener comunicaciones independientes asociadas con los mismos. Por ejemplo, en el caso de un televisor o vídeograbadora, puede haber o no un aspecto en red para su control. Cada ordenador 110a, 110b, 110c, 110d, 110e, etc. cliente y el ordenador 10a, 10b, etc. servidor pueden estar equipado con diversos módulos u objetos 135 del programa de aplicación y con conexiones o acceso a diversos tipos de elementos u objetos de almacenamiento, a través de los cuales se pueden almacenar archivos o secuencias de datos o a qué porción(es) de archivos o secuencias de datos se pueden descargar, transmitir o migrar. Cualquiera o más de los ordenadores 10a, 10b, 110a, 110b, etc. pueden ser responsables del mantenimiento y actualización de una base 20 de datos u otro elemento de almacenamiento, tal como una base de datos o memoria 20 para almacenar datos procesados de acuerdo con la invención. Por lo tanto, la presente invención se puede utilizar en un entorno de red informática que tenga ordenadores 110a, 110b, etc. cliente que puedan acceder e interactuar con una red/bus 14 de ordenadores y ordenadores 10a, 10b, etc. servidores que puedan interactuar con los ordenadores 110a, 110b, etc. cliente y otros dispositivos similares, y bases 20 de datos.

Dispositivo informático de ejemplo

La figura 2B y la siguiente descripción pretenden proporcionar una breve descripción general de un entorno informático adecuado en relación con el cual se puede implementar la invención. Sin embargo, debe entenderse que dispositivos informáticos de mano, portátiles y de otro tipo y objetos informáticos de todo tipo se contemplan para su uso en relación con la presente invención, es decir, en cualquier lugar donde exista una GPU en un entorno informático. Aunque a continuación se describe un ordenador de propósito general, este no es más que un ejemplo, y la presente invención se puede implementar con un cliente ligero que tenga interoperabilidad e interacción de red/bus. Por lo tanto, la presente invención puede implementarse en un entorno de servicios alojados en red en el que están implicados recursos de cliente muy pequeños o mínimos, por ejemplo, un entorno de red en el que el dispositivo cliente sirve simplemente como una interfaz a la red/bus, como objeto colocado en un dispositivo. En esencia, en cualquier lugar donde se puedan almacenar datos o desde el cual se puedan recuperar o transmitir datos a otro ordenador, es un entorno deseable o adecuado para la operación de la técnica de optimización de gráficos de acuerdo con la invención.

Aunque no se requiere, la invención puede implementarse a través de un sistema operativo, para su uso mediante un desarrollador de servicios para un dispositivo u objeto, y/o incluido dentro de software de aplicación que opera en conexión con las técnicas de programación de gráficos para una canalización de gráficos en la invención. El software se puede describir en el contexto general de las instrucciones ejecutables por ordenador, tal como módulos de programa, que se ejecutan en uno o más ordenadores, como estaciones de trabajo, servidores u otros dispositivos. Generalmente, los módulos de programa incluyen rutinas, programas, objetos, componentes, estructuras de datos, y similares, que realizan tareas particulares o implementan tipos particulares de datos abstractos. Típicamente, la funcionalidad de los módulos de programa se puede combinar o distribuir como se desee

en diversas realizaciones. Además, los expertos en la materia apreciarán que la invención se puede poner en práctica con otras configuraciones y protocolos de sistemas informáticos. Otros sistemas informáticos, entornos y/o configuraciones bien conocidos que pueden ser adecuados para su uso con la invención incluyen, pero no se limitan a, ordenadores personales (PC), cajeros automáticos, ordenadores de servidor, dispositivos de mano o portátiles, sistemas de múltiples procesadores, sistemas basados en microprocesadores, electrónica de consumo programable, PC de red, aparatos, luces, elementos de control ambiental, miniordenadores, ordenadores centrales y similares. La invención también puede ponerse en práctica en entornos informáticos distribuidos, donde las tareas son realizadas por dispositivos de procesamiento remoto que están enlazados a través de una red/bus de comunicación u otro medio de transmisión de datos. En un entorno informático distribuido, los módulos de programa pueden ubicarse tanto en medios de almacenamiento informático local como remoto, incluidos los dispositivos de almacenamiento de memoria, y los nodos de cliente pueden a su vez comportarse como nodos de servidor.

La figura 2B ilustra así un ejemplo de un entorno 100 de sistema informático adecuado en el que puede implementarse la invención, aunque como se ha aclarado anteriormente, el entorno 100 del sistema informático es solo un ejemplo de un entorno informático adecuado y no está destinado a sugerir cualquier limitación en cuanto al alcance del uso o la funcionalidad de la invención. Tampoco se debe interpretar que el entorno 100 informático tiene alguna dependencia o requisito relacionado con uno cualquiera o una combinación de componentes ilustrados en el entorno 100 operativo ejemplar.

Con referencia a la figura 2B, un sistema ejemplar para implementar la invención incluye un dispositivo informático de propósito general en la forma de un ordenador 110. Los componentes del ordenador 110 pueden incluir, pero no se limitan a, una unidad 120 de procesamiento, una memoria 130 del sistema y un bus 121 del sistema que acopla diversos componentes del sistema que incluyen la memoria del sistema a la unidad 120 de procesamiento. El bus 121 del sistema puede ser cualquiera de varios tipos de estructuras de bus que incluyen un bus de memoria o controlador de memoria, un bus periférico y un bus local usando cualquiera de una variedad de arquitecturas de bus. A modo de ejemplo, y no de limitación, tales arquitecturas incluyen el bus de arquitectura estándar de la industrial (ISA), el bus de arquitectura de microcanal (MCA), el bus ISA mejorado (EISA), el bus local VESA (Asociación de estándares electrónicos de vídeo) y el bus de interconexión de componentes periféricos (PCI) (también conocido como bus Mezzanine).

El ordenador 110 incluye típicamente una variedad de medios legibles por ordenador. Los medios legibles por ordenador pueden ser cualquier medio disponible al que se pueda acceder mediante el ordenador 110 e incluye medios volátiles y no volátiles, medios extraíbles y no extraíbles. A modo de ejemplo, y no de limitación, los medios legibles por ordenador pueden comprender medios de almacenamiento informático y medios de comunicación. Los medios de almacenamiento informático incluyen medios volátiles y no volátiles, extraíbles y no extraíbles implementados en cualquier procedimiento o tecnología para el almacenamiento de información, tal como instrucciones legibles por ordenador, estructuras de datos, módulos de programa u otros datos. Los medios de almacenamiento informático incluyen, entre otros, RAM, ROM, EEPROM, memoria flash u otra tecnología de memoria, CDROM, discos versátiles digitales (DVD) u otro tipo de almacenamiento en disco óptico, casetes magnéticos, cinta magnética, almacenamiento en disco magnético u otros dispositivos magnéticos de almacenamiento o cualquier otro medio que pueda usarse para almacenar la información deseada y a la que se pueda acceder por ordenador 110. Los medios de comunicación típicamente incorporan instrucciones legibles por ordenador, estructuras de datos, módulos de programa u otros datos en una señal de datos modulada, tal como una onda portadora u otro mecanismo de transporte e incluye cualquier medio de suministro de información. El término "señal de datos modulada" significa una señal que tiene una o más de sus características establecidas o cambiadas, de tal manera que codifican información en la señal. A modo de ejemplo, y no de limitación, los medios de comunicación incluyen medios cableados, tales como una red cableada o conexión de cableado directo, y medios inalámbricos tales como medios acústicos, de RF, infrarrojos y otros medios inalámbricos. Combinaciones de cualquiera de lo anterior también deben incluirse dentro del alcance de los medios legibles por ordenador.

La memoria 130 del sistema incluye medios de almacenamiento de ordenador en forma de memoria volátil y/o no volátil, tal como memoria 131 de sólo lectura (ROM) y memoria 132 de acceso aleatorio (RAM). Un sistema 133 básico de entrada/salida (BIOS), que contiene las rutinas básicas que ayudan a transferir información entre los elementos dentro del ordenador 110, como durante el arranque, se almacena típicamente en la ROM 131. La RAM 132 contiene típicamente módulos de datos y/o programas que son inmediatamente accesibles y/o están siendo operados actualmente por la unidad 120 de procesamiento. A modo de ejemplo, y no de limitación, la figura 2B ilustra el sistema 134 operativo, programas 135 de aplicación, otros módulos 136 de programa y datos 137 de programa.

El ordenador 110 también puede incluir otros medios extraíbles/no extraíbles, volátiles/no volátiles de almacenamiento de ordenador. A modo de ejemplo solamente, la figura 2B ilustra una unidad 141 de disco duro que lee o escribe en medios magnéticos no volátiles no removibles, una unidad 151 de disco magnético que lee o escribe en un disco 152 magnético no volátil extraíble, y una unidad 155 de disco óptico que lee o escribe en un disco 156 óptico no volátil extraíble, tal como un CDROM u otro medio óptico. Otros medios de almacenamiento informático volátiles/no volátiles extraíbles/no extraíbles que se pueden usar en el entorno operativo ejemplar incluyen, pero no se limitan a, casetes de cinta magnética, tarjetas de memoria flash, discos versátiles digitales, cinta de vídeo digital, RAM de estado sólido, ROM de estado sólido y similares. El disco 141 duro está típicamente

conectado al bus 121 del sistema a través de una interfaz de memoria no extraíble tal como la interfaz 140, y el disco 151 magnético y el disco 155 óptico están típicamente conectados al bus 121 del sistema mediante una interfaz de memoria extraíble, tal como la interfaz 150.

Las unidades y sus medios de almacenamiento informático asociados comentados anteriormente e ilustrados en la figura 2B proporcionan almacenamiento de instrucciones legibles por ordenador, estructuras de datos, módulos de programa y otros datos para el ordenador 110. En la figura 2B, por ejemplo, el disco 141 duro se ilustra como el almacenamiento del sistema 144 operativo, los programas 145 de aplicación, otros módulos 146 de programa y datos 147 de programa. Debe tenerse en cuenta que estos componentes pueden ser iguales o diferentes del sistema 134 operativo, los programas 135 de aplicación, otros módulos 136 de programa y los datos 137 de programa. El sistema 144 operativo, los programas 145 de aplicación, otros módulos 146 de programa y los datos 147 de programa se dan diferentes números aquí para ilustrar que, como mínimo, son copias diferentes. Un usuario puede ingresar comandos e información en el ordenador 110 a través de dispositivos de entrada tales como un teclado 162 y un dispositivo 161 señalador, comúnmente referido como ratón, bola de desplazamiento o almohadilla táctil. Otros dispositivos de entrada (no mostrados) pueden incluir un micrófono, una palanca de mando, una almohadilla para juegos, una antena parabólica, un escáner o similar. Estos y otros dispositivos de entrada a menudo están conectados a la unidad 120 de procesamiento a través de una interfaz 160 de entrada de usuario que está acoplada al bus 121 del sistema, pero pueden estar conectados por otras estructuras de interfaz y bus, tales como un puerto paralelo, puerto de juego o bus serie universal (USB). Una interfaz 182 gráfica, tal como Northbridge, también puede estar conectada al bus 121 del sistema. Northbridge es un conjunto de chips que se comunica con la CPU, o la unidad 120 de procesamiento del servidor, y asume la responsabilidad de las comunicaciones del puerto de gráficos acelerados (AGP). Una o más unidades 184 de procesamiento de gráficos (GPU) pueden comunicarse con la interfaz 182 de gráficos. A este respecto, las GPU 184 incluyen generalmente el almacenamiento en memoria en chip, tal como el almacenamiento de registros y las GPU 184 se comunican con una memoria 186 de vídeo, en el que las variables de aplicación de la invención pueden tener impacto. Las GPU 184, sin embargo, son solo un ejemplo de un coprocesador y, por lo tanto, se pueden incluir una variedad de dispositivos de coprocesamiento en el ordenador 110, y pueden incluir una variedad de sombreadores de procedimiento, tales como sombreadores de píxeles y vértices. Un monitor 191 u otro tipo de dispositivo de visualización también está conectado al bus 121 del sistema a través de una interfaz, tal como una interfaz 190 de vídeo, que a su vez puede comunicarse con la memoria 186 de vídeo. Además del monitor 191, los ordenadores también pueden incluir otros dispositivos de salida periféricos tales como altavoces 197 e impresora 196, que pueden estar conectados a través de una interfaz 195 periférica de salida.

El ordenador 110 puede operar en un entorno de red o distribuido utilizando conexiones lógicas a uno o más ordenadores remotos, tales como un ordenador 180 remoto. El ordenador 180 remoto puede ser un ordenador personal, un servidor, un enrutador, un PC en red, un dispositivo par u otro nodo de red común, y típicamente incluye muchos o todos los elementos descritos anteriormente con relación al ordenador 110, aunque solo un dispositivo 181 de almacenamiento de memoria se ha ilustrado en la figura 2B. Las conexiones lógicas representadas en la figura 2B incluyen una red 171 de área local (LAN) y una red 173 de área extensa (WAN), pero también pueden incluir otras redes/buses. Dichos entornos de red son comunes en los hogares, oficinas, redes informáticas de toda la empresa, intranets e Internet.

Cuando se utiliza en un entorno de red LAN, el ordenador 110 está conectado a la LAN 171 a través de una interfaz de red o adaptador 170. Cuando se usa en un entorno de red WAN, el ordenador 110 incluye típicamente un módem 172 u otro medio para establecer comunicaciones a través de la WAN 173, tal como Internet. El módem 172, que puede ser interno o externo, puede conectarse al bus 121 del sistema a través de la interfaz 160 de entrada de usuario u otro mecanismo apropiado. En un entorno de red, los módulos de programa representados con respecto al ordenador 110, o partes de los mismos, pueden almacenarse en el dispositivo de almacenamiento de memoria remota. A modo de ejemplo, y no de limitación, la figura 2B ilustra programas 185 de aplicación remota que residen en el dispositivo 181 de memoria. Se apreciará que las conexiones de red mostradas son ejemplares y se pueden usar otros medios para establecer un enlace de comunicaciones entre los ordenadores.

Marcos de trabajo o arquitecturas de cálculo distribuidos ejemplares

Varios marcos de trabajo informáticos distribuidos han sido y están siendo desarrolladas a la luz de la convergencia de la informática personal e Internet. Tanto a los usuarios individuales como a los profesionales se les proporciona una interfaz interoperable y habilitada para la Web para aplicaciones y dispositivos informáticos, lo que hace que las actividades informáticas sean cada vez más un navegador Web o una red.

Por ejemplo, la plataforma de código administrado de MICROSOFT®, es decir, .NET, incluye servidores, servicios de construcción en bloques, tal como almacenamiento de datos basado en la Web y software de dispositivo descargable. En términos generales, la plataforma .NET proporciona (1) la capacidad de hacer que toda la gama de dispositivos informáticos trabajen en conjunto y tener información de usuario automáticamente actualizada y sincronizada en todos ellos, (2) mayor capacidad interactiva para páginas web, habilitada por mayor uso de XML en lugar de HTML, (3) servicios en línea que ofrecen acceso personalizado y suministro de productos y servicios para el usuario desde un punto de partida central para la administración de diversas aplicaciones, tales como correo electrónico, por ejemplo, o software, tales como Office .NET, (4) almacenamiento de datos centralizado, que

aumenta la eficiencia y la facilidad de acceso a la información, así como la sincronización de información entre usuarios y dispositivos, (5) la capacidad de integrar diversos medios de comunicación, como correo electrónico, faxes y teléfonos, (6) para desarrolladores, la capacidad de crear módulos reutilizables, aumentando así la productividad y reduciendo el número de errores de programación y (7) muchas otras plataformas cruzadas y también características de integración de idiomas.

Aunque algunas realizaciones de ejemplo en el presente documento se describen en conexión con software residente en un dispositivo informático, una o más porciones de la invención pueden también ser implementadas a través de un sistema operativo, interfaz de programación de aplicaciones (API) o un objeto "hombre medio", un objeto de control, hardware, firmware, instrucciones u objetos de lenguaje intermedio, etc., de modo que los procedimientos se puedan incluir, admitir o acceder a través de todos los idiomas y servicios habilitados por el código administrado, como el código .NET y en otros marcos de trabajo informáticos distribuidos también.

Sistemas y procedimientos para proporcionar una canalización de gráficos mejorada

Como se describe en los antecedentes y resumen de la invención, la presente invención mejora el estado de las técnicas de procesamiento de gráficos mediante la introducción de sistemas y procedimientos para una canal gráficos que optimizan el uso de los recursos, equilibran la carga de trabajo en la canalización de gráficos, permiten el acceso a la información calculada con valores enteros o de coma flotante compatibles con IEEE y proporcionan una capacidad de programación adicional. La invención está dirigida al procesamiento de geometría y sombreadores programables. La canalización de gráficos mejorada incluye la introducción de un núcleo común para todos los sombreadores, una salida de secuencia para los cálculos que tienen lugar en el canal y un sombreador de geometría que permite la programación de primitivas, así como la generación de nueva geometría.

La figura 3A muestra una subunidad de gráficos ejemplar, tal como una tarjeta de vídeo, que incluye una unidad 184' de procesamiento de gráficos (GPU) y una configuración de hardware ejemplar para el canal 184'-1 de gráficos asociado. La realización ejemplar no limitativa del canal 184'-1 de gráficos, como se ilustra, incluye una pluralidad de instancias de un elemento 184'-1a de núcleo común, tal como sombreadores 184'-1a_1 y 184'-1a_2 de vértices, un sombreador 184'-1a_3 de geometría que tiene una salida 184'-2 de secuencia asociada y un sombreador 184'-1a_4 de píxeles. Fuera de la GPU 184', esta realización incluye un ensamblador de entrada que envía datos al primer núcleo común, 184'-1a_1. Como se muestra, el primer núcleo común de la realización está configurado como un sombreador de vértices. Los datos se envían a un teselador 184'-1b, que realiza una teselación en los datos. Después del teselador 184'-1b, hay otro núcleo común en la canalización, que se puede usar para realizar sombreado de vértices de teselación posterior en los datos. Las etapas del teselador, en esta realización, son opcionales. El segundo núcleo común es seguido por un tercer núcleo común que está configurado como un sombreador de geometría, 184'-1a_3. El sombreador de geometría toma una primitiva y produce cero, una o múltiples primitivas. Las primitivas de salida pueden contener más datos que un sombreado de vértices o píxeles. Después del tercer núcleo común, puede haber una salida de secuencia, 184'-2, donde las primitivas que alcanzan este punto pueden transmitirse a una o más memorias intermedias de salida. Para los fines de proporcionar una muestra, la salida de secuencia se incluye en la figura 3A. El siguiente componente del canal es un rasterizador 184'-1c, cuyas funciones regulares pueden incluir recorte, divisiones en perspectiva, ventana gráfica o selección e implementación de tijera. Después del rasterizador hay un cuarto núcleo común, 184'-1a_4, y funciona como un sombreador de píxeles donde toma un píxel y emite el píxel en una posición. Después de realizar todas las funciones en los datos, los datos se muestran en un monitor 191.

Otras realizaciones de la invención, tal como se refleja por el núcleo común, el sombreador de geometría, la salida de secuencia y la capacidad para generar nueva geometría dentro del canal de la invención se describen en más detalle a continuación.

Número de sombreador común para sombreadores en el canal

Como se menciona en los antecedentes, un problema de contención de recursos a veces se produce debido a las demandas de los diferentes componentes de la canalización de gráficos, donde se requieren componentes específicos para una tarea específica, que puede conducir a un cuello de botella en esa área del canal. Los canales de gráficos actuales requieren optimización por parte del programador para utilizar los recursos de la manera más efectiva en el canal. Sin embargo, incluso con los mejores esquemas de optimización, el hardware de la técnica anterior es fijo, y hay momentos en que el sombreador de píxeles o el sombreador de vértices permanecen inactivos, por ejemplo, cuando los cálculos implican una manipulación de matriz pesada y el sombreador de vértices no puede calcular los valores lo suficientemente rápido. Por lo tanto, el canal puede tener cuellos de botella debido al procesamiento excesivo de los sombreadores de vértices mientras los sombreadores de píxeles permanecen inactivos, o viceversa. De acuerdo con la invención, dado que los sombreadores pueden funcionar entre sí, donde un sombreador de píxeles puede reconfigurarse para funcionar como un sombreador de vértices, los recursos subyacentes del chip de gráficos pueden optimizarse para las tareas que se solicitan al chip de gráficos. Además, como se mencionó, la invención introduce una nueva clase de sombreador referido aquí como un sombreador de geometría, que proporciona otro conjunto especializado de capacidades de procesamiento. El núcleo común proporcionado por la invención se puede configurar, así como cualquiera de un sombreador de vértices, un sombreador de píxeles y un sombreador de geometría.

En una realización no limitativa a modo de ejemplo, la GPU contiene una granja 184'-1a de unidades, que de este modo se puede programar para diferentes etapas bajo demanda. Este equilibrio de carga significa que los programadores no tienen que preocuparse de utilizar cada etapa. Cualquier etapa puede habilitarse o deshabilitarse dinámicamente y configurarse o reconfigurarse, liberando y respecializando los recursos para las etapas que están activas. El núcleo común es capaz de realizar la etapa de sombreado de vértices, la etapa de sombreado de geometría y la etapa de sombreado de píxeles, dependiendo de su configuración.

La figura 4A muestra una realización ejemplar no limitativa de un núcleo común interno, 184'-1a. Los datos de entrada son suministrados por una etapa previa, que puede ser desde cualquier lugar en el canal. Los registros 500 de entrada pueden ser una matriz dinámicamente indexable o una matriz bidimensional (2D) dependiendo de qué etapa debe realizar el núcleo común. Los datos se envían al código 510 de sombreado y el código gestiona todo el vector de coma flotante y aritmética de números enteros, y la recuperación de memoria o las operaciones de muestra.

El código de sombreador recibe información de los muestreadores 513, las texturas 514 y las memorias 515 intermedias constantes. Los muestreadores 513 definen cómo muestrear texturas. Sin embargo, la memoria también se puede leer sin filtrar y el muestreador no es necesario en cada realización. Debido a que los objetos del muestreador se crean estáticamente, permite que el hardware mantenga referencias de múltiples muestreadores en el vuelo del canal sin tener que rastrear los cambios o purgar el canal (porque los objetos del muestreador mantienen sus definiciones y no se modifican). Las texturas 514 trabajan con el código del sombreador para proporcionar un muestreo de textura. En esta realización particular, los únicos recursos permitidos son aquellos que contienen un formato de elemento único por téxel. El código de sombreado recibe constantes desde las memorias 515 intermedias constantes. En esta realización, los almacenamientos intermedios constantes funcionarían como texturas unidimensionales (1D) que están optimizadas para un acceso de latencia más bajo y una actualización más frecuente. El código 500 sombreador también tiene una comunicación bidireccional con los registros 511 temporales y la pila 512 de direcciones de retorno de la subrutina. Los registros 511 temporales sirven como almacenamiento temporal. En una realización ejemplar, no limitativa, los registros pueden contener cualquier matriz no indexable o indexable de cualquier tamaño y cantidad que se necesite hasta el límite del almacenamiento temporal. La pila 512 de direcciones de retorno de la subrutina, en esta realización ejemplar no limitativa en particular, es de una altura fija. Además, la pila está oculta del acceso directo al sombreador y almacena de forma transparente las direcciones de retorno únicamente. También permite la definición de algoritmos recursivos.

Después de que el código ha pasado a través del código 510 de sombreador, los datos van a los registros 520 de salida. Los registros 520 de salida, en esta realización ejemplar no limitativa, están hechos de una matriz dinámicamente indexable de cuatro salidas de vector. Además, algunas etapas pueden tener productos especializados adicionales. Luego, los datos se envían a la siguiente etapa de sombreado, como una salida de secuencia o a una etapa de representación, dependiendo del canal.

Las estructuras de datos listados se pueden cambiar de una matriz 1D a una matriz 2D o una lista. Todas las estructuras de datos pueden cambiar dependiendo del tamaño y de las capacidades de almacenamiento de la GPU. Dentro de la GPU, pueden producirse cambios en los límites de almacenamiento y las alturas fijas debido a la asignación dinámica y a la aplicación de algoritmos de compresión a la información para ahorrar espacio. Los muestreadores y las memorias intermedias constantes pueden comportarse como las texturas; sin embargo, a medida que cambian los recursos, las definiciones de estos componentes pueden modificarse. Si las texturas se redefinen, tanto los muestreadores como las memorias intermediaos constantes pueden cambiar y no están limitados solo a las funciones de textura. Además, todas las estructuras de datos pueden implementar nuevos algoritmos de optimización para fines de velocidad y utilidad. Las diversas realizaciones descritas en este documento son meros ejemplos de canales de gráficos que usan núcleos comunes, que pueden configurarse dinámicamente para proporcionar la funcionalidad de un sombreador de vértices, un sombreador de píxeles y un sombreador de geometría.

Sombreador de geometría para operar en primitivas

En la actualidad, a medida que los programadores gráficos desarrollan aplicaciones gráficas a través de un conjunto de API gráficas disponibles, el programador generalmente indica un conjunto de vértices a los que se aplica un conjunto de elementos algorítmicos. Una vez especificada la información, los datos se envían a la canalización de gráficos, y cada vértice se transmite a través de un sombreador de vértices y de un sombreador de píxeles, como se ilustra en la figura 1A. Aunque puede especificarse cualquier dato que se ajuste a la estructura de entrada para los sombreadores de vértices y de píxeles, los sombreadores de vértices son generalmente adecuados y se utilizan para operar sobre vértices y los sombreadores de píxeles son generalmente adecuados y se usan para operar sobre píxeles.

A este respecto, un sombreador de geometría de acuerdo con la invención es un nuevo tipo de sombreador para una GPU en un subsistema de gráficos que es capaz de tomar diferentes tipos de entrada "primitiva" incluyendo cualquiera de un vértice, una línea o un triángulo, mientras que los sombreadores de la técnica anterior (concretamente los sombreadores de vértices) están limitados a poder introducir, operar y emitir vértices. En distinción, además de la operación en una secuencia de vértices, el sombreador de geometría de la invención puede

operar sobre primitivas que definen líneas (conjuntos de dos vértices) y triángulos (conjuntos de tres triángulos), que reciben tales primitivas como entrada dentro del canal y salidas de primitivas dentro del canal para una operación adicional de acuerdo con la arquitectura de gráficos de la invención.

Un aspecto adicional de la primitiva de procesamiento de acuerdo con la invención es que el sombreador de geometría permite las operaciones en toda la primitiva no solo por sí misma, sino también en el contexto de algunos vértices cercanos. Un segmento de línea en una polilínea, por ejemplo, puede procesarse con la capacidad de leer los vértices antes y después de ese segmento. Aunque el mecanismo es general (los datos de gráficos no necesitan ser datos de "gráficos", pueden ser datos definidos para su procesamiento por la GPU), un uso frecuente de esta capacidad para procesar vértices adyacentes de una primitiva es que el sombreador de geometría es capaz de tener en cuenta la información sobre los puntos vecinos en el espacio geométrico en 3-D en los cálculos actuales.

En la figura 4B, hay una realización de la funcionalidad del sombreador de geometría donde TV0, TV1, TV2, LV0, y LV1 son datos de entrada. Los datos de salida pueden incluir puntos adicionales que se indican con el prefijo AV. De este modo, un sombreador de geometría puede tomar una primitiva y emitir múltiples primitivas, procesando opcionalmente primitivas adyacentes, tales como vértices adyacentes. Algunos ejemplos del resultado incluyen topologías como una serie de triángulos, una serie de líneas o una lista de puntos. En esta realización ejemplar no limitativa, la salida de un sombreador de geometría va a un rasterizador para representación y/o a la memoria intermedia para la salida de secuencia.

El concepto se ilustra adicionalmente en la figura 4C. De acuerdo con la invención, opcionalmente, a los datos de los datos gráficos que se procesarán mediante la canalización de gráficos también se les puede asignar un índice. Anteriormente, para indexar los datos, la memoria separada tendría que haberse rellenado previamente con información de índice para usar con los datos que se pasan a través del canal. Además de los problemas de sincronización con este enfoque de la técnica anterior (sincronización de los índices y los datos gráficos que se procesan), también hay claras implicaciones de velocidad. Si bien la generación de la información del índice es rápida, la información del índice se escribe en la memoria de vídeo, y la lectura desde la memoria de vídeo que se utilizará con los datos de gráficos en tiempo real es costosa en términos de tiempo.

Por lo tanto, con la invención, el ensamblador de entrada IA emite los datos gráficos para ser operados por el(los) núcleo(s) común(es), junto con cualquier información opcional de adyacencia (información sobre los vecinos en la secuencia de primitivas o datos) y una secuencia correspondiente de índices se pueden generar para cada elemento (primitiva, punto de datos, valor de píxel, etc.) representado por los datos gráficos. En una realización, los índices comienzan en cero y aumentan en uno con cada nuevo elemento de datos, aunque cualquier valor de índice puede usarse como un valor de índice de inicio. Además, para alguna aplicación, puede incluso no ser útil generar índices ordenados, y, en consecuencia, en el sentido genérico, la invención genera valores de identificación (ID) únicos por elemento de datos gráficos, que pueden ordenarse o no.

La capacidad de índice tiene amplias implicaciones, y la gama de usos son numerosos. Por ejemplo, para instancias de geometría, uno puede tener una primera área de memoria con datos de gráficos que define un árbol y otra tabla que describe la posición de los árboles. En función de un recuento o índice procesado por el canal, la GPU puede recuperar porciones de la memoria de vídeo para colocar variaciones del árbol en las posiciones definidas por la tabla. La invención permite el uso de un valor de índice como parte de un esquema de direccionamiento de memoria de vídeo mientras se procesan los datos, es decir, el índice puede indicar dónde y cuándo recuperar (y/o escribir) porciones de la memoria de vídeo para su procesamiento. Como otro ejemplo, puede ser útil aplicar un algoritmo diferente a cada cuarto triángulo de una secuencia de triángulos y ejecutar el algoritmo dos veces en cada octavo triángulo de la secuencia. La capacidad de indexar las primitivas permite precisamente esta capacidad. Los programas descargados al sombreador pueden utilizar de manera programática los valores de índice asociados con los datos gráficos mientras se recuperan o se escriben en la memoria de vídeo, y también al procesar los datos gráficos.

Salida de secuencia de la memoria antes de la rasterización de la memoria intermedia de tramas

Como se describió en los antecedentes, con canales gráficos de hoy en día, la lectura de la memoria de vídeo utilizada por una GPU para ver el resultado del cálculo que se realiza en el canal es casi imposible. Tener el conocimiento de cómo localizar y extraer datos sobre la marcha mientras están siendo procesados por los sombreadores de procedimiento de una GPU, y luego extraer los datos de la memoria de vídeo es una tarea demasiado difícil en las arquitecturas actuales. La figura 1B muestra cómo un programador podría recuperar datos de la salida de la canalización de gráficos, concretamente la memoria intermedia de tramas, e implicar recursos de servidor para enviar de nuevo los datos al canal para su posterior procesamiento. Como se explicó, esta tarea requiere que el programador obtenga los datos de la memoria intermedia de tramas o de la pantalla. En general, si es necesario un procesamiento posterior de los datos después de pasar por el canal, el programador espera a que la información sea recibida por la memoria intermedia de tramas e invoca los recursos de la CPU para lograr el procesamiento posterior.

En consideración de las deficiencias de los canales gráficos de hoy en día con respecto a la recirculación de los datos gráficos dentro del canal, la invención permite a los programadores "tocar" en el canal mientras que los datos

están dentro del canal, en el que un toque se puede encender y apagar. Cuando se enciende, el canal incluye una secuencia de salida de datos que puede leerse desde la memoria para su recuperación por el servidor u otra operación, o recircularse al canal para realizar funciones recursivas o de bucle. Además, si es necesario, la información puede transmitirse mientras los mismos datos van al rasterizador, por lo tanto, no ralentiza la representación de datos, o permite la visualización de la imagen a medida que experimenta una transformación basada en elementos algorítmicos recursivos que operan en los datos.

La figura 3B muestra una realización de una salida de secuencia de acuerdo con la invención. En esta realización, el núcleo común descrito anteriormente, que se comporta como un sombreador 184-1a_3 de geometría, está realizando cálculos o manipulaciones en los datos y hay una salida de secuencia, 184'-2, por lo que, cuando se enciende, los datos se escriben en una memoria intermedia de acuerdo con la invención. En la figura, la salida de secuencia se muestra recirculando los datos al propio sombreador. Este tipo de recirculación es útil para ejemplos donde un programador desea realizar operaciones de múltiples pasos en los datos dados. Sin embargo, según lo especificado por el programa descargado a la GPU, hay otras maneras de usar la salida de secuencia para recircular también los datos. Otra forma de recircular los datos es enviarlos a través del canal nuevamente ingresando los datos al ensamblador de entrada nuevamente. Las capacidades de salida de secuencia de la invención son opcionales, y una aplicación puede simplemente permitir que el canal envíe los datos a través del mismo sin leer los datos en una memoria intermedia de salida de secuencia. Además, el almacenamiento intermedio de salida de secuencia es un ejemplo del tipo de memoria que se puede usar para almacenar los datos transmitidos. Puede haber momentos en los que se puedan usar diferentes tipos de memoria para dicha funcionalidad, como la memoria caché en un microprocesador.

Por último, la memoria está limitada en el espacio y, por lo tanto, la salida de secuencia sigue de la cantidad de datos que dependen del tamaño de la memoria intermedia. Cuando el espacio de almacenamiento de la memoria está lleno, en una realización a modo de ejemplo, el sombreador continúa sus cálculos para todos los datos, mientras se cuenta el número de bytes necesarios para almacenar todos los datos de salida. Esto puede ser útil cuando es impredecible cuántos datos se producirán mediante los cálculos. La salida de secuencia se puede usar para gráficos, pero no está limitada a solo gráficos. Aunque se trata de una mejora de la canalización de gráficos, la GPU puede utilizarse para cálculos matemáticos y puede utilizar la potencia de procesamiento tanto de la GPU como de la CPU. Los usos de la salida de secuencia cubren un amplio rango, ya que la salida de secuencia no detiene las funciones de rasterización de la canalización de gráficos, y simplemente amplifica la potencia del canal proporcionando más potencia programática para el desarrollador en términos de flexibilidad de programas especializados descargados a la GPU. Como se mencionó, un resultado muy útil de poder acceder a la salida de un núcleo común de acuerdo con la invención es que los datos pueden recircularse a otras partes del canal programáticamente, es decir, el programador puede descargar un programa a la GPU, que realiza operaciones recursivas en los datos (recirculando datos a través de los mismos algoritmos de manera recursiva) o de lo contrario recorre los datos un número predeterminado de veces.

Generación de geometría dentro del canal

Como se ha mencionado, en la actualidad, como los programadores gráficos desarrollar aplicaciones gráficas a través de un conjunto de API gráficas disponibles, el programador indica en general un conjunto de vértices para ser operados por un conjunto de elementos algorítmicos. Una vez especificada la información, los datos se envían a la canalización de gráficos, y cada vértice se transmite a través de un conjunto fijo de subunidades de GPU, como un teselador, sombreador de vértices y sombreador de píxeles, como se ilustra en la figura 1A. Aunque puede especificarse cualquier dato que se ajuste a la estructura de entrada para los sombreadores de vértices y de píxeles, los sombreadores de vértices son generalmente adecuados y se utilizan para operar sobre vértices y los sombreadores de píxeles son generalmente adecuados y se usan para operar sobre píxeles. Sin embargo, aparte de la teselación de escenario fijo que puede generar geometría basada en puntos de control predefinidos antes del sombreado de procedimiento de acuerdo con algoritmos predeterminados, hoy no hay ningún lugar en el canal que pueda usarse para generar nuevos vértices, o geometría, para procesamiento adicional.

Por otra parte, el teselador de la invención se pueden usar después del uso de una o más etapas de sombreado de procedimiento para recibir un conjunto de puntos de control generadas dentro de la canal, y operar mediante programación, en función de los programas que configuran el canal descargado por la API de gráficos, para generar un nuevo número arbitrario de puntos de geometría. Esto es útil, por ejemplo, cuando es deseable realizar operaciones gruesas en una malla gruesa, pero luego refinar la malla para operaciones adicionales que requieren un modelo más realista de las formas. Anteriormente, la generación de geometría del teselador se limitaba a la etapa inicial de una canal, y se limitaba a las operaciones prefijadas en las entradas de punto de control especificadas previamente. En consideración de tales limitaciones, la invención permite que los núcleos comunes al comienzo del canal generen arbitrariamente puntos de control dentro del canal (aguas arriba del teselador), para emitir esos puntos de control a un teselador, lo que genera una geometría adicional para ser operada aguas abajo como salida del teselador.

Como se ha mencionado, el sombreador de geometría de la invención también puede ser utilizado para generar nueva geometría de una manera más general y programable, como diferenciadas de un teselador - que es función fija (no programable), pero genera cantidades arbitrarias de datos (malla n x n, sin límite en n) respecto al

sombreador de geometría, que es programable, pero está limitado en la cantidad de geometría nueva que puede crear.

Para algunos escenarios no limitativos a modo de ejemplo, el sombreador de geometría de la invención facilita un programador mediante la creación de vértices de salida utilizando una topología en lugar de un único vértice. Algunos ejemplos de algoritmos incluyen teselado de sprites puntuales, teselado de líneas anchas, generación de aletas, generación de volúmenes de sombras y representación de un solo paso para múltiples caras de cubos de texturas. Para realizar un teselado de sprites puntual, el sombreador tomaría un solo vértice y generaría cuatro vértices, o dos triángulos de salida que representaban cuatro esquinas de un cuadrilátero, al realizar una teselación de líneas anchas, el sombreador recibe dos vértices de línea, como LV0 y LV1 en la figura 4B, y genera cuatro vértices para un cuadrilátero que representa una línea ensanchada. Además, el sombreador de geometría puede utilizar vértices de línea adyacentes, AV0 y AVI, para realizar ingletes en los puntos finales de la línea.

El sombreador de geometría también se puede utilizar para generar piel o aletas, esto no se limita a la generación de piel o aletas, sino que abarca cualesquiera vértices adicionales añadidos en una tercera dirección de una única topología. Los ejemplos incluyen cabello, escamas, césped, etc., donde las primitivas que describen una geometría se alimentan al sombreador de geometría, y el sombreador de geometría aumenta la geometría de forma arbitraria para complementar la forma. Por lo tanto, con el cabello, por ejemplo, basado en la entrada de triángulos a un sombreador de geometría, el sombreador de geometría puede agregar algunos vértices que representan el cabello en cada vértice. Ventajosamente, debido a que una secuencia de triángulos al sombreador de geometría incluye información sobre los vecinos de un vértice, la proximidad y las propiedades (color, profundidad, etc.) de la geometría de los vecinos del vértice se pueden tener en cuenta durante el procesamiento.

Aunque en la actualidad, los sombreadores de geometría de la invención se limitan a poder generar un número máximo de nueva geometría arbitraria, tal como se especifica mediante el programa descargado en el sombreador de geometría por las API de gráficos, cuando se combina con las capacidades recursivas de la invención habilitada por la salida de secuencia definida con más detalle a continuación, la invención permite la generación arbitraria de geometría ilimitada. Por ejemplo, en el caso donde un sombreador de geometría solo puede crear.

Otro uso no limitativo de ejemplo del sombreador de geometría incluye la generación de volumen de sombra donde la información de adyacencia se utiliza para decidir si se debe extrudir. Además, una aplicación puede querer generar algo de geometría, como una aleta o piel y extrudir volúmenes de sombra de eso. En tales casos, la funcionalidad de múltiples pasos del sombreador de geometría se emplearía usando la capacidad de salida de una secuencia de datos y la haría circular nuevamente a través del uso de la salida de secuencia.

La figura 5 ilustra un diagrama de flujo ejemplar no limitativo para mostrar cómo una aplicación haría llamadas de gráficos para lograr los beneficios descritos anteriormente de la arquitectura de la invención. En 550, se especifican los datos de gráficos a operar. Como se mencionó, se necesitan especificar menos datos gráficos en el extremo frontal que en el pasado porque se pueden generar nuevas cantidades arbitrarias de geometría dentro del canal, incluso recursivamente, de acuerdo con la invención. En 560, los programas para la ejecución de la GPU se pueden descargar a la GPU para definir los elementos algorítmicos para la operación de la GPU. En 570, el ensamblador de entrada (véase, por ejemplo, la figura 3A) toma automáticamente los datos de gráficos especificados en 550 y los programas especificados en 560 y optimiza la disposición de núcleos comunes de acuerdo con la invención. Por lo tanto, cuando el cálculo de vértices o píxeles es intensivo como parte de la especificación de elementos programáticos suministrados a la GPU en 560 y los datos gráficos especificados en 550, la invención puede configurar los núcleos de canales para incluir muchos sombreadores de vértices o sombreadores de píxeles, respectivamente. Por otro lado, cuando se desean operaciones complejas en triángulos, el ensamblador de entrada puede reconfigurar los núcleos para incluir una pluralidad de sombreadores de geometría. Además, se pueden agregar opciones tales como la función de salida de secuencia y las etapas de sombreado de procedimiento pre y post teselación. En 580, el ensamblador de entrada revisa los elementos de trabajo especificados para el procesamiento por la canalización de gráficos y asigna el trabajo a los núcleos. En esta etapa, la información del índice se puede agregar a las primitivas que pasan por el canal, por ejemplo, un índice por primitiva o un índice por píxel.

En esencia, se pueden ver rápidamente los beneficios de hardware altamente especializado y rápido que se pueden personalizar de forma dinámica para el trabajo que se deba realizar. Esto incluye poder especificar programas recursivos a través de la salida de secuencia, poder reconfigurar dinámicamente los núcleos, poder operar rápidamente en vértices y al mismo tiempo tomar entradas relacionadas con los vecinos de los vértices, pudiendo generar nueva geometría arbitrariamente en cualquier parte del canal (generación de geometría antes de la teselación, generación de geometría del teselador y generación de geometría posterior a la teselación). En consecuencia, las ventajas de la presente invención son casi ilimitadas desde el punto de vista del desarrollador, que se acerca más a estar limitado solo por la imaginación que por lo que se puede lograr en los datos gráficos de acuerdo con la invención.

La canalización de gráficos mejorada de la invención, por lo tanto, incluye un núcleo común que es dinámicamente configurable para asignar sombreadores de píxeles, sombreadores de geometría y sombreadores de vértices de una manera que es la más adecuada para las tareas que se solicitan del canal. La invención también incluye un nuevo

sombreador de procedimiento denominado sombreador de geometría capaz de operar en secuencias de cualquiera de los puntos, líneas y triángulos. Además, la invención incluye la capacidad de generar nueva geometría dentro del canal, mediante el teselador o mediante un sombreador de geometría. La invención también incluye una salida de secuencia continua, que puede ser aprovechada dentro del canal, antes de que los datos lleguen a la memoria intermedia de tramas para la rasterización, permitiendo algoritmos recursivos y de bucle en los datos gráficos. Además, la invención incluye la capacidad de indexar cada primitiva que entra o se crea en la canalización de gráficos. La canalización de gráficos también puede realizar todos los cálculos de números enteros y coma flotante según los estándares IEEE.

Hay múltiples formas de implementar la presente invención, por ejemplo, una API apropiado, caja de herramientas, código controlador, sistema operativo, control, objeto de software autónomo o descargable, etc., que permiten a las aplicaciones y servicios utilizar los sistemas y los procedimientos de la invención de la canalización de gráficos mejorada. La invención contempla el uso de la invención desde el punto de vista de una API (u otro objeto de software), así como de un objeto de software o hardware que recibe cualquiera de las técnicas antes mencionadas, incluidas técnicas del núcleo común, el sombreador de geometría o la salida de secuencia de acuerdo con la invención. Por lo tanto, varias implementaciones de la invención descrita en este documento pueden tener aspectos que están totalmente en hardware, en parte en hardware y en parte en software, así como también en software.

Como se mencionó anteriormente, aunque las realizaciones ejemplares de la presente invención se han descrito en relación con los dispositivos informáticos y arquitecturas de red diferentes, los conceptos subyacentes pueden aplicarse a cualquier dispositivo informático o sistema en el que es deseable emplear una GPU con una canalización de gráficos mejorada. Por ejemplo, el(los) algoritmo(s) y las implementaciones de hardware de la invención se pueden aplicar al sistema operativo de un dispositivo informático, proporcionado como un objeto separado en el dispositivo, como parte de otro objeto, como un control reutilizable, como un objeto descargable desde un servidor, como un "intermediario" entre un dispositivo u objeto y la red, como un objeto distribuido, como hardware, en la memoria, una combinación de cualquiera de los anteriores, etc. Aunque se eligen lenguajes de programación, nombres y ejemplos ejemplares aquí como representante de varias opciones, estos idiomas, nombres y ejemplos no son limitativos. Un experto en la técnica apreciará que existen numerosas formas de proporcionar código de objeto y nomenclatura que logra la misma funcionalidad similar o equivalente lograda por las diversas realizaciones de la invención.

Como se ha mencionado, las diversas técnicas descritas en el presente documento pueden implementarse en conexión con hardware o software o, en su caso, con una combinación de ambos. Por lo tanto, los procedimientos y aparatos de la presente invención, o ciertos aspectos o partes de los mismos, pueden adoptar la forma de código de programa (es decir, instrucciones) incorporado en medios tangibles, tales como disquetes, CD-ROM, discos duros o cualquier otro medio de almacenamiento legible por máquina, en el que, cuando el código de programa se carga y se ejecuta en una máquina, tal como un ordenador, la máquina se convierte en un aparato para poner en práctica la invención. En el caso de la ejecución de código de programa en ordenadores programables, el dispositivo informático generalmente incluye un procesador, un medio de almacenamiento legible por el procesador (que incluye memoria volátil y no volátil y/o elementos de almacenamiento), al menos un dispositivo de entrada, y al menos un dispositivo de salida. Uno o más programas que pueden implementar o utilizar las técnicas mejoradas de canalización de gráficos de la presente invención, por ejemplo, mediante el uso de una API de procesamiento de datos, controles reutilizables o similares, se implementan preferiblemente en un lenguaje de programación orientado a objetos o procedimientos de alto nivel para comunicarse con un sistema informático. Sin embargo, el(los) programa(s) puede(n) implementarse en ensamblador o lenguaje de máquina, si así lo desea. En cualquier caso, el lenguaje puede ser un lenguaje compilado o interpretado, y combinado con implementaciones de hardware.

Los procedimientos y aparatos de la presente invención puede también ponerse en práctica mediante comunicaciones incorporadas en forma de código de programa que se transmite por algún medio de transmisión, tal como sobre cableado eléctrico o por cable, a través de fibra óptica, o mediante cualquier otra forma de transmisión, en el que, cuando el código del programa es recibido y cargado y ejecutado por una máquina, tal como una EPROM, una matriz de puertas, un dispositivo lógico programable (PLD), un ordenador cliente, etc., la máquina se convierte en un aparato para practicar la invención. Cuando se implementa en un procesador de propósito general, el código de programa se combina con el procesador para proporcionar un aparato único que opera para invocar la funcionalidad de la presente invención. Adicionalmente, cualquier técnica de almacenamiento utilizada en conexión con la presente invención puede ser invariablemente una combinación de hardware y software.

Aunque la presente invención se ha descrito en conexión con las realizaciones preferidas de las diversas figuras, debe entenderse que pueden usarse otras realizaciones similares o pueden realizarse modificaciones y adiciones a la realización descrita para realizar la misma función de la presente invención sin desviarse de la misma. Por ejemplo, aunque los entornos de red ejemplares de la invención se describen en el contexto de un entorno de red, tal como un entorno de red de igual a igual, un experto en la materia reconocerá que la presente invención no se limita a ello, y que los procedimientos, como se describe en la presente solicitud, pueden aplicarse a cualquier dispositivo o entorno informático, como una consola de juegos, ordenador de mano, ordenador portátil, etc., ya sea por cable o de manera inalámbrica, y se puede aplicar a cualquier cantidad de dichos dispositivos informáticos conectados a través de una red de comunicaciones e interactuando a través de la red. Además, debe enfatizarse que se contemplan una variedad de plataformas informáticas, que incluyen sistemas operativos de dispositivos de

mano y otros sistemas operativos específicos de la aplicación, especialmente a medida que la cantidad de dispositivos en red inalámbrica continúa proliferando.

Aunque las realizaciones a modo de ejemplo se refieren a la utilización de la presente invención en el contexto de una canalización de gráficos, la invención no está tan limitada, sino que puede implementarse para proporcionar una segunda unidad de procesamiento. Por ejemplo, si el programador desea mostrar una pantalla y procesar las matemáticas computacionales mientras se realiza otra función usando la CPU, las unidades de procesamiento pueden necesitar ser utilizadas al máximo, ya sea que la pantalla gráfica esté o no incluida en la salida final. Además, la presente invención puede implementarse en o a través de una pluralidad de chips o dispositivos de procesamiento, y el almacenamiento puede efectuarse de manera similar a través de una pluralidad de dispositivos. Por lo tanto, la presente invención no debe limitarse a ninguna realización individual, sino que debe interpretarse en amplitud y alcance de acuerdo con las reivindicaciones adjuntas.

REIVINDICACIONES

1. Un procedimiento de procesamiento de datos, que comprende:

programar un elemento (184'-1a_1, 184'-1a_2, 184'-1a_3, 184'-1a_4) de núcleo común de una unidad (184') de procesamiento de gráficos, de manera que esté configurado para realizar una primera función, en el que la primera función es una de un sombreador de píxeles, un sombreador de vértices y un sombreador de geometría;
ingresar una primera secuencia de datos de gráficos al elemento de núcleo común;
realizar dicha primera función en la primera secuencia de datos gráficos;
reprogramar dinámicamente el elemento de núcleo común de modo que esté configurado para realizar una segunda función, en el que la segunda función es una de un sombreador de píxeles, un sombreador de vértices y un sombreador de geometría, y en el que la segunda función difiere de la primera función;
ingresar una segunda secuencia de datos gráficos al elemento de núcleo común;
realizar dicha segunda función en la segunda secuencia de datos gráficos,
en el que una de la primera y la segunda función es un sombreador de geometría, y
en el que el procedimiento comprende además tomar, cuando el elemento de núcleo común está configurado para realizar el sombreador de geometría, una primitiva de los datos de entrada y emitir múltiples primitivas, comprendiendo la salida una de una serie de triángulos, una serie de líneas y una lista de puntos.

2. El procedimiento de la reivindicación 1, en el que dicha etapa de reprogramación dinámica incluye habilitar una salida de secuencia, por lo que la información calculada en el elemento de núcleo común se envía en un formato no gráfico a dicha salida de secuencia antes de almacenarse en una memoria intermedia de tramas para su representación.

3. El procedimiento de la reivindicación 2, en el que dicha salida de secuencia incluye datos que resultan para realizar dicha primera o segunda función en dichas primera y segunda secuencias de datos de gráficos, respectivamente.

4. El procedimiento de la reivindicación 1, que comprende, además:

descargar (560) un programa de software al elemento de núcleo común que programa el elemento de núcleo común para realizar una de dichas función de sombreado de píxeles, función de sombreado de vértices y función de sombreado de geometría de una manera programática.

5. El procedimiento de la reivindicación 4, en el que el elemento de núcleo común incluye puertas de entrada, almacenamiento de registros, subelementos de unidad de procesamiento y puertas de salida.

6. El procedimiento de la reivindicación 4, en el que la manera programática incluye recursión.

7. El procedimiento de la reivindicación 4, en el que dicha descarga incluye instrucciones de descarga que permiten una salida de secuencia, por lo que la información a calcular en el elemento de núcleo común se envía a una memoria de vídeo recuperable antes de que la información sea almacenada en una memoria intermedia de tramas rasterizable.

8. Un medio legible por ordenador que comprende instrucciones ejecutables por ordenador para realizar, cuando dichas instrucciones son ejecutadas por un ordenador, el procedimiento de una de las reivindicaciones 1 a 7.

9. Una señal de datos modulada que lleva instrucciones ejecutables por ordenador para realizar, cuando dichas instrucciones son ejecutadas por un ordenador, el procedimiento de una de las reivindicaciones 1 a 7.

10. Una unidad de procesamiento de gráficos adaptada para realizar el procedimiento de una de las reivindicaciones 1 a 7.

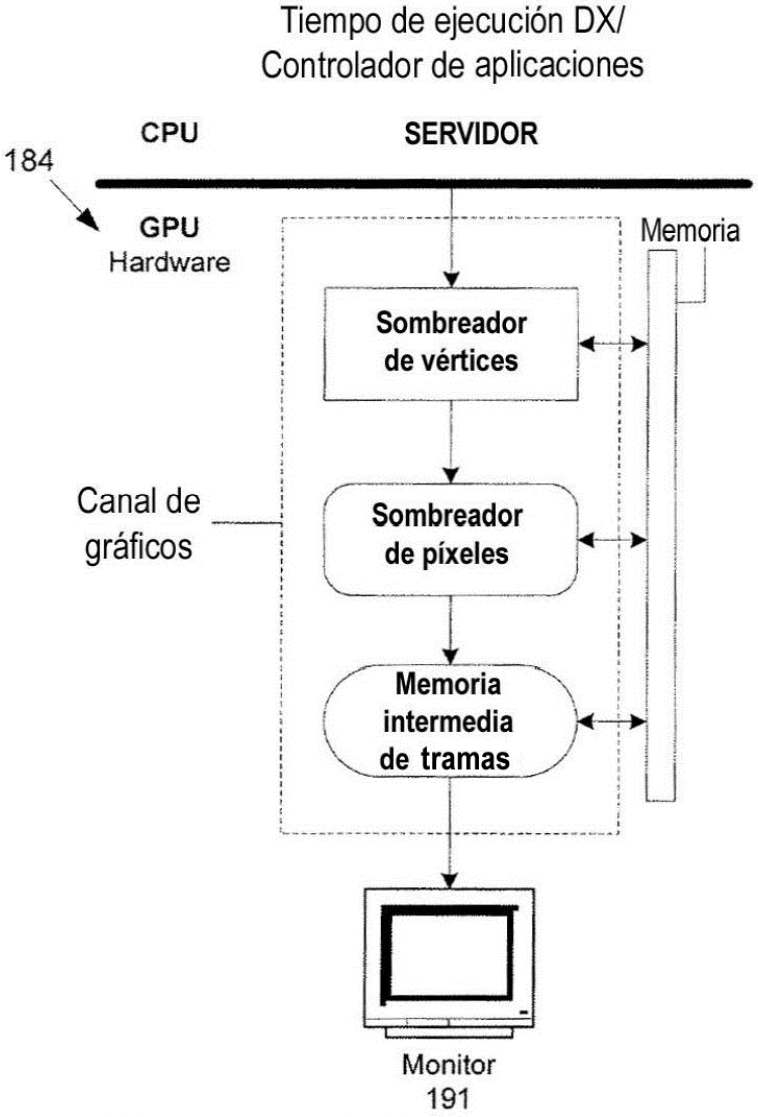

FIG. 1A – Técnica anterior

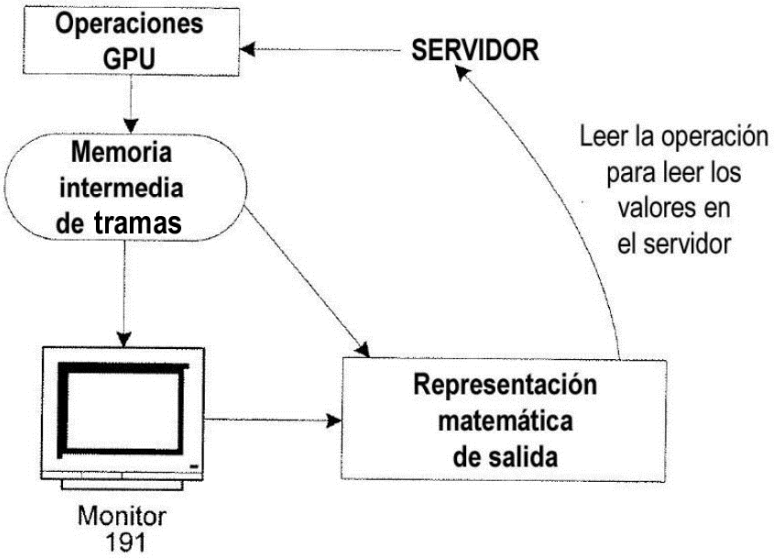

FIG. 1B – TÉCNICA ANTERIOR

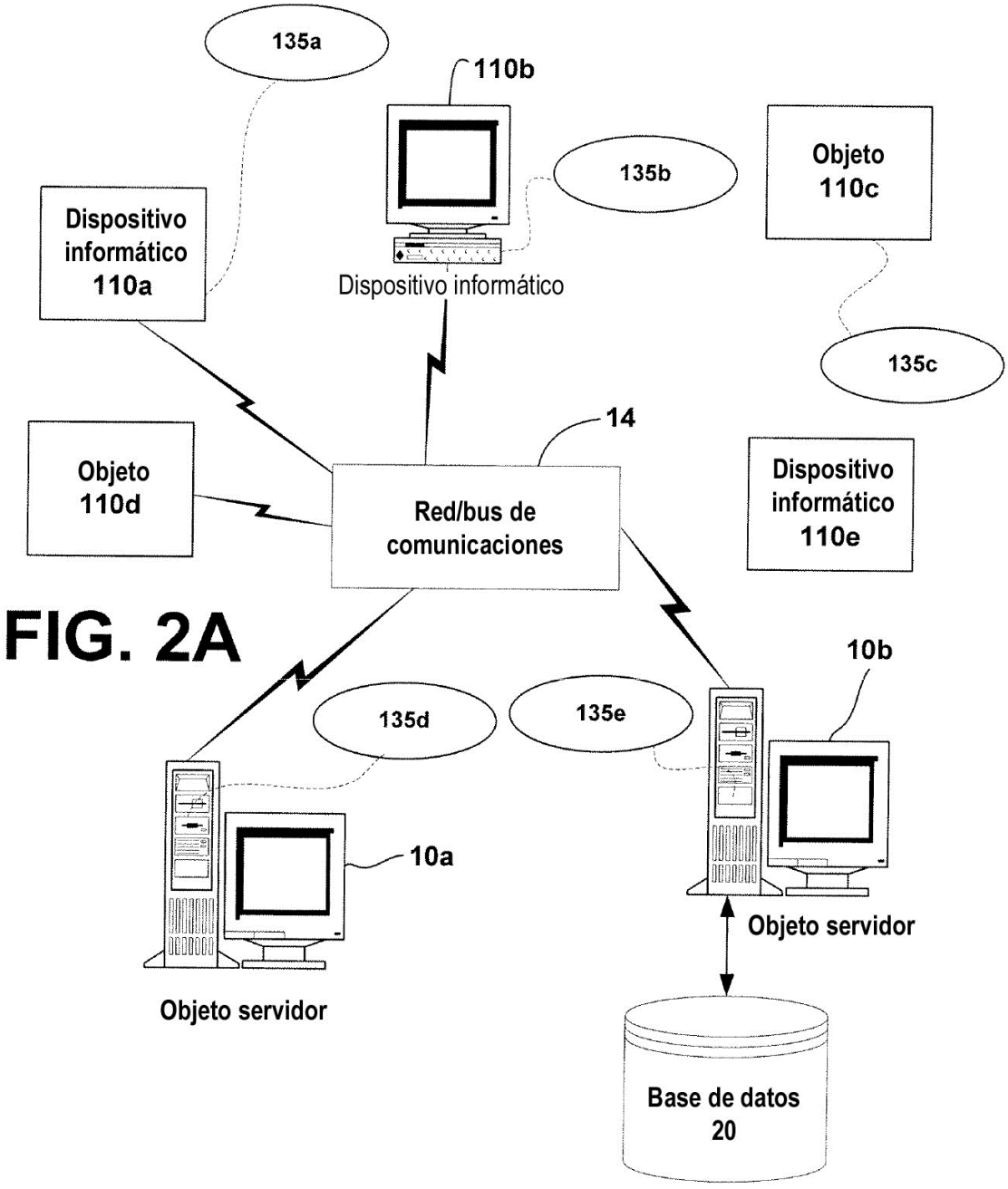
135a
110b
135b
Objeto
110c
Dispositivo
informático
110a
Dispositivo informático
135c
14
Objeto
110d
Red/bus de
comunicaciones
Dispositivo
informático
110e
10b
135d
135e
10a
Objeto servidor
Objeto servidor
Base de datos
20

FIG. 2A

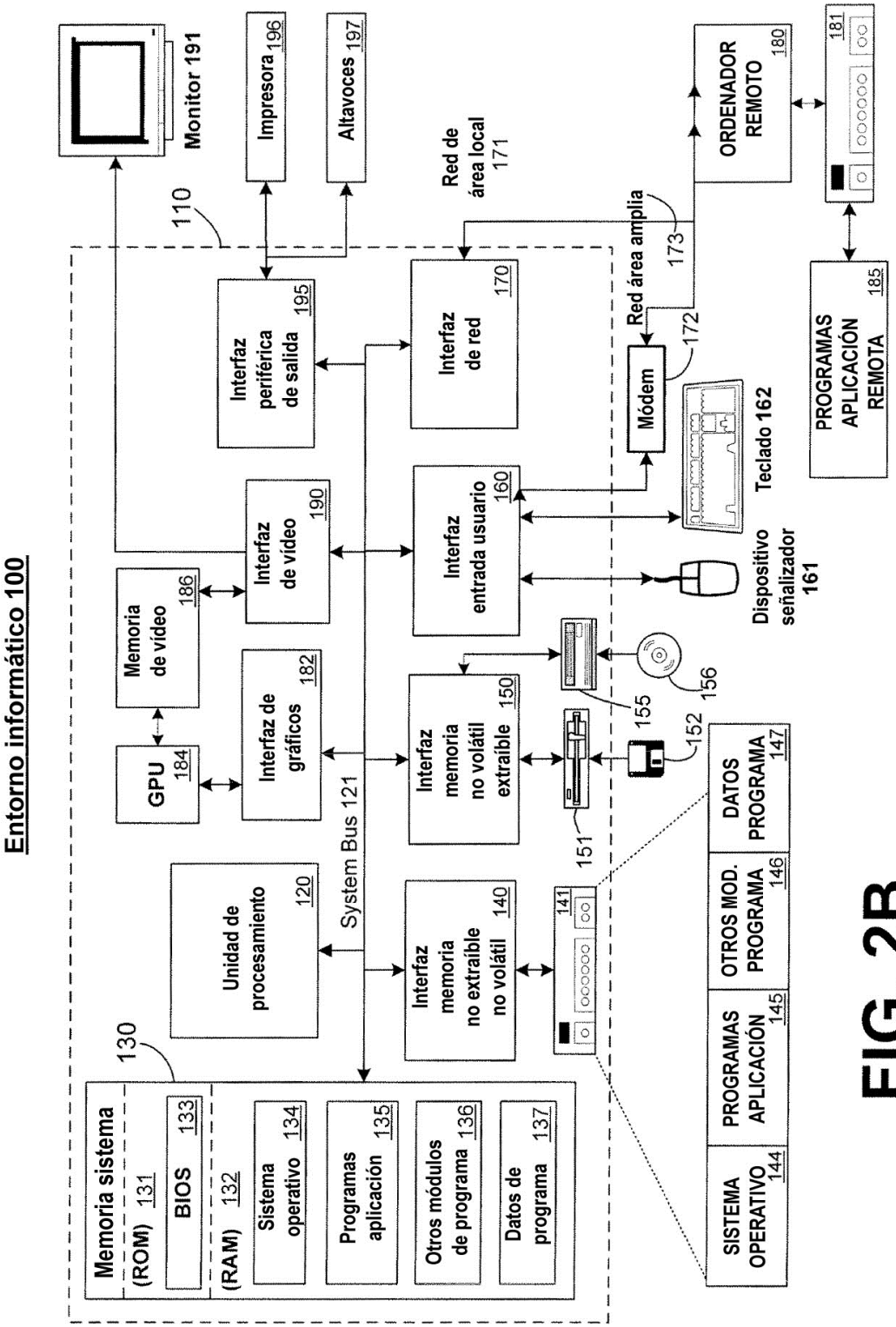
Entorno informático 100
Monitor 191
Impresora 196
Altavoces 197
110
Red de área local 171
Interfaz periférica de salida 195
Interfaz de red 170
Red área amplia 173
ORDENADOR REMOTO 180
181
Módem 172
PROGRAMAS APLICACIÓN REMOTA 185
Memoria de video 186
Interfaz de video 190
Interfaz entrada usuario 160
Teclado 162
Dispositivo señalizador 161
GPU 184
Interfaz de gráficos 182
Interfaz memoria no volátil extraíble 150
155
156
152
151
System Bus 121
Unidad de procesamiento 120
Interfaz memoria no extraíble no volátil 140
141
130
Memoria sistema
(ROM) 131
BIOS 133
(RAM) 132
Sistema operativo 134
Programas aplicación 135
Otros módulos de programa 136
Datos de programa 137
SISTEMA OPERATIVO 144
PROGRAMAS APLICACIÓN 145
OTROS MOD. PROGRAMA 146
DATOS PROGRAMA 147

FIG. 2B

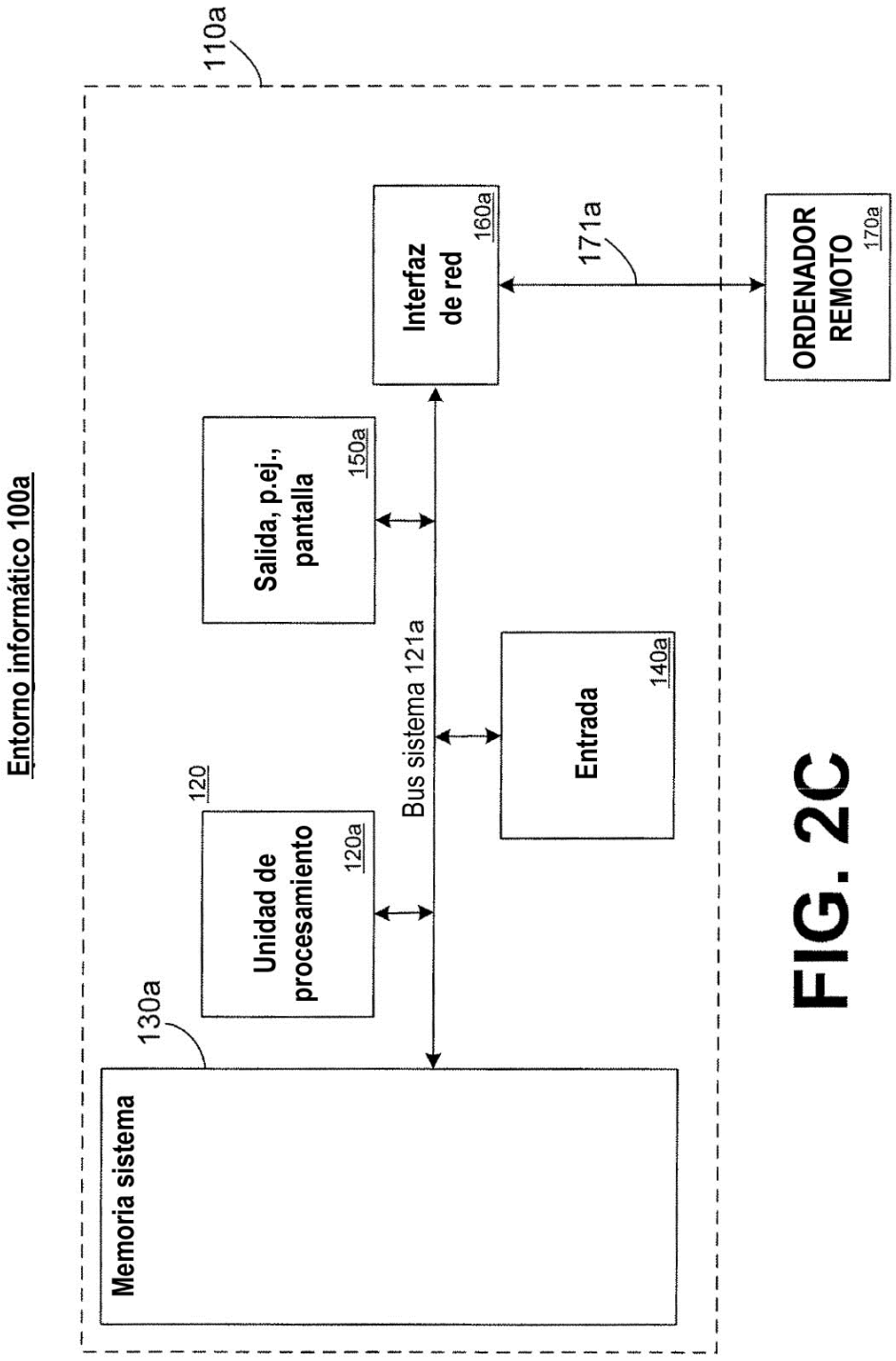
Entorno informático 100a
110a
Interfaz de red
160a
171a
ORDENADOR REMOTO
170a
Salida, p.ej., pantalla
150a
Bus sistema 121a
Entrada
140a
120
Unidad de procesamiento
120a
130a
Memoria sistema

FIG. 2C

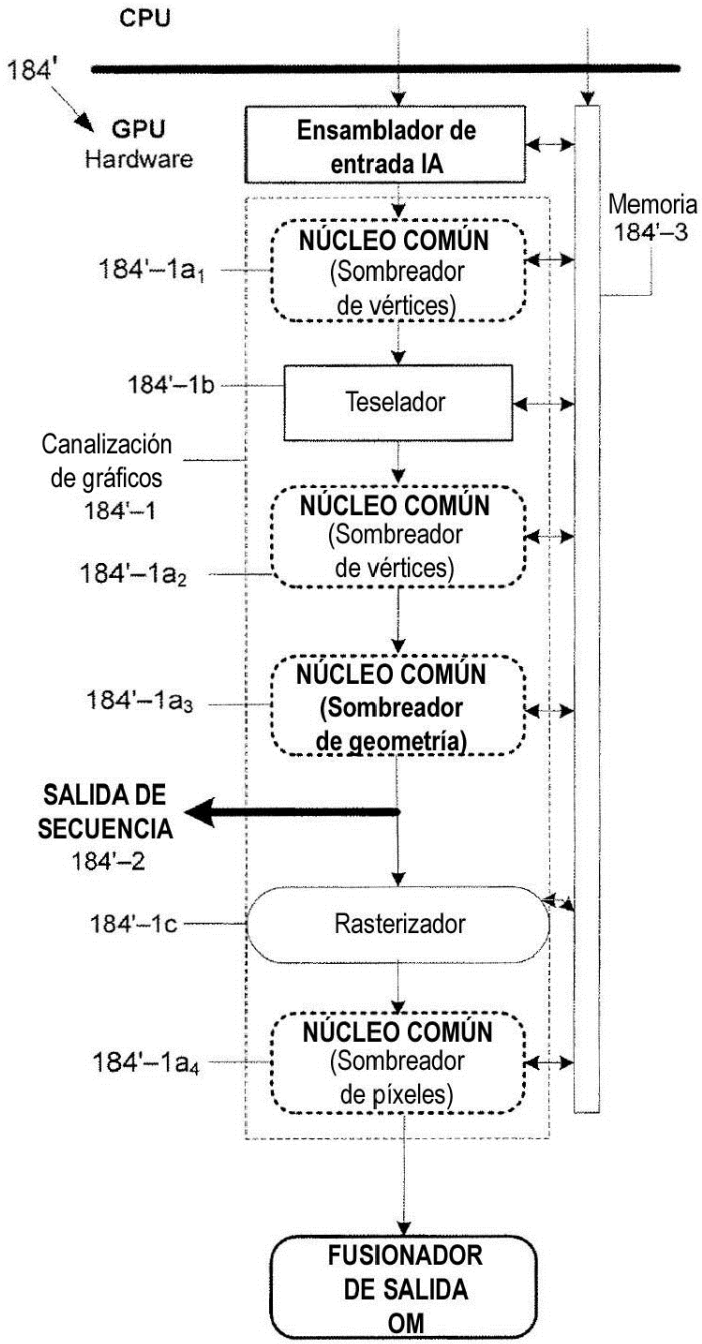
CPU
184'
GPU
Hardware
Ensamblador de entrada IA
Memoria 184'–3
184'–1a1
NÚCLEO COMÚN (Sombreador de vértices)
184'–1b
Teselador
Canalización de gráficos 184'–1
184'–1a2
NÚCLEO COMÚN (Sombreador de vértices)
184'–1a3
NÚCLEO COMÚN (Sombreador de geometría)
SALIDA DE SECUENCIA 184'–2
184'–1c
Rasterizador
184'–1a4
NÚCLEO COMÚN (Sombreador de píxeles)
FUSIONADOR DE SALIDA OM

FIG. 3A

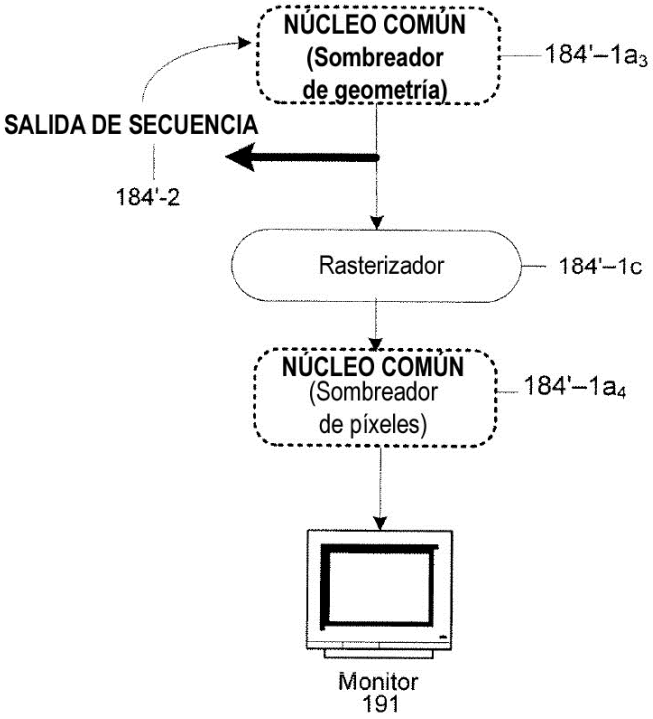
NÚCLEO COMÚN
(Sombreador
de geometría)
184'–1a3
SALIDA DE SECUENCIA
184'-2
Rasterizador
184'–1c
NÚCLEO COMÚN
(Sombreador
de píxeles)
184'–1a4
Monitor
191

FIG. 3B

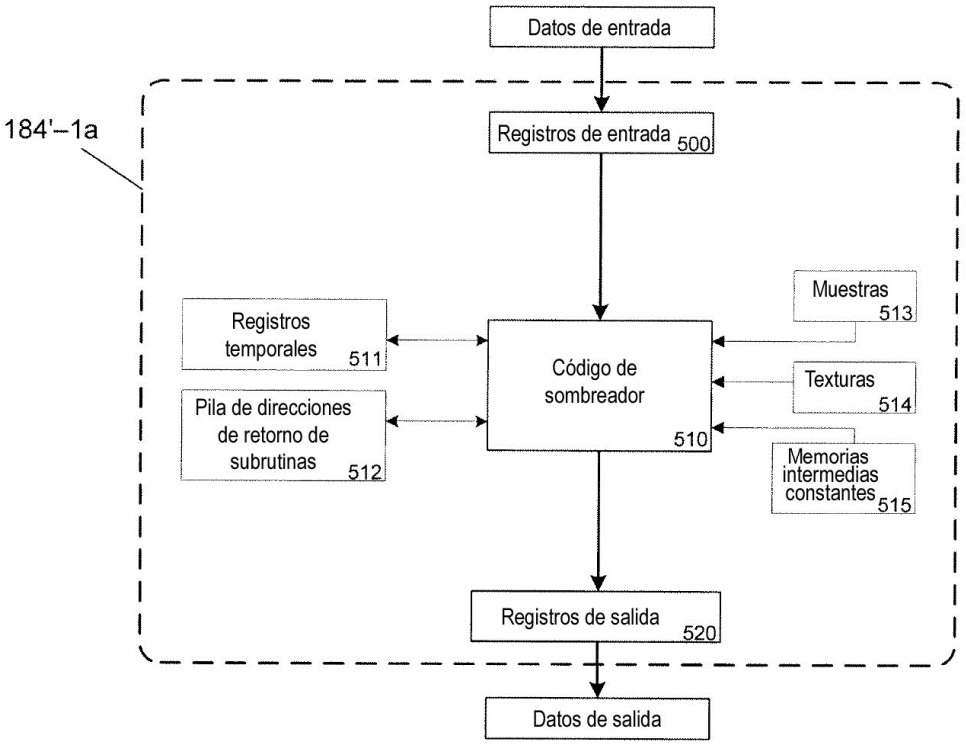
Datos de entrada
184'–1a
Registros de entrada 500
Registros temporales 511
Pila de direcciones de retorno de subrutinas 512
Código de sombreador 510
Muestras 513
Texturas 514
Memorias intermedias constantes 515
Registros de salida 520
Datos de salida

FIG. 4A

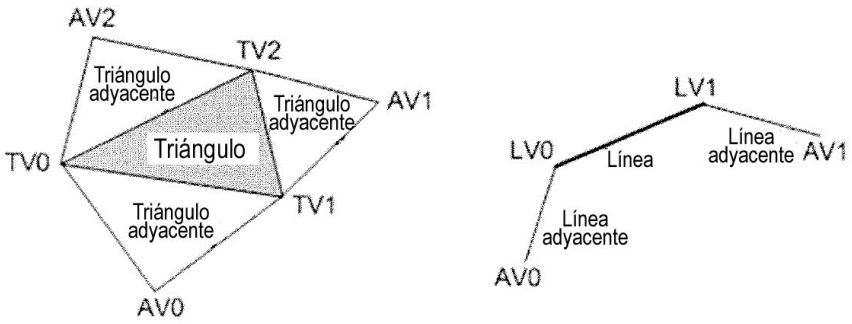
AV2
TV2
Triángulo adyacente
AV1
Triángulo adyacente
TV0
Triángulo
TV1
Triángulo adyacente
AV0
LV1
LV0
Línea
Línea adyacente
AV1
Línea adyacente
AV0

FIG. 4B

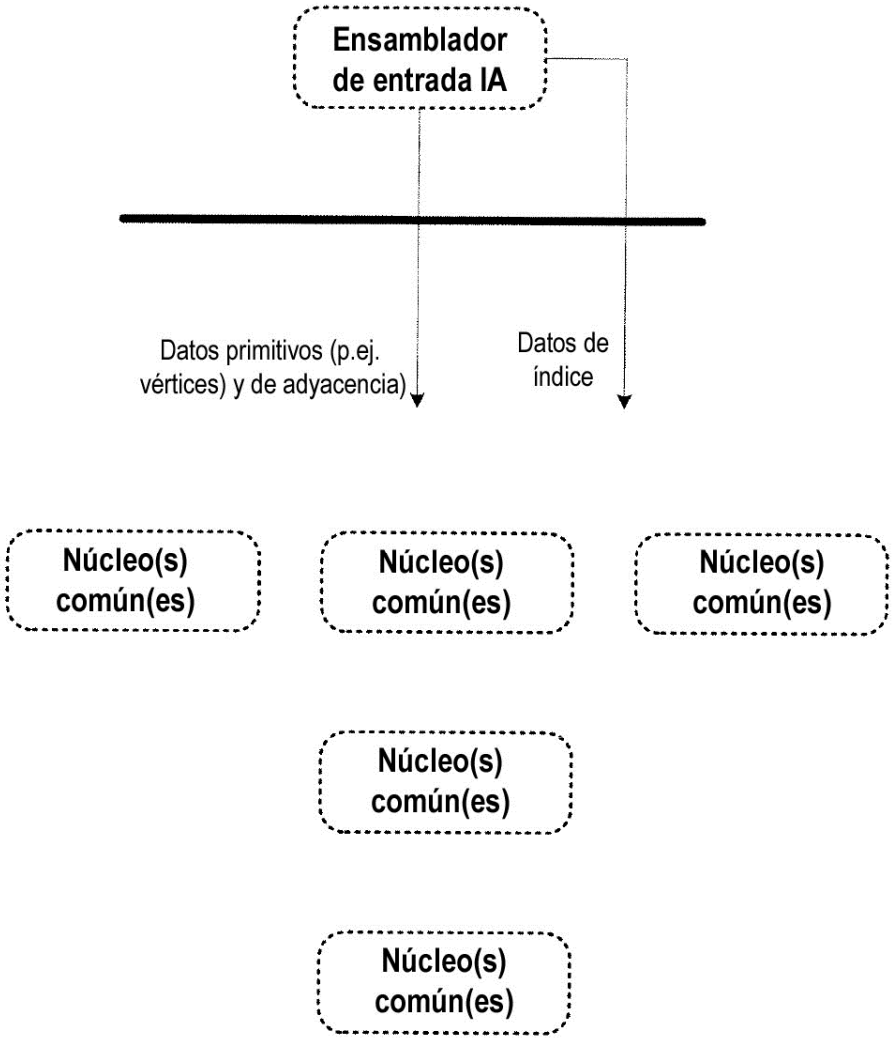
Ensamblador de entrada IA
Datos primitivos (p.ej. vértices) y de adyacencia)
Datos de índice
Núcleo(s) común(es)
Núcleo(s) común(es)
Núcleo(s) común(es)
Núcleo(s) común(es)
Núcleo(s) común(es)

FIG. 4C

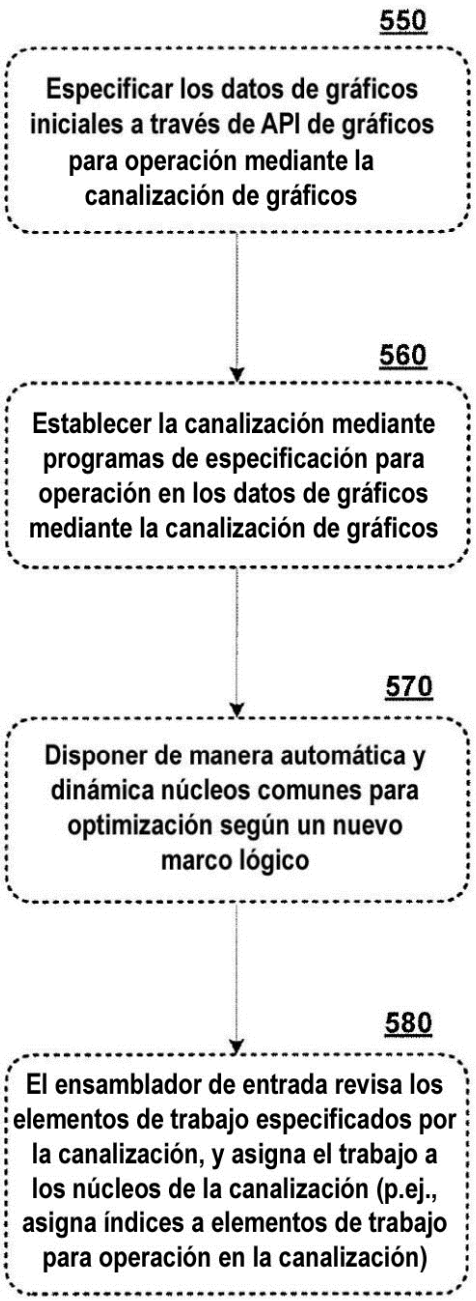
550
Especificar los datos de gráficos iniciales a través de API de gráficos para operación mediante la canalización de gráficos
560
Establecer la canalización mediante programas de especificación para operación en los datos de gráficos mediante la canalización de gráficos
570
Disponer de manera automática y dinámica núcleos comunes para optimización según un nuevo marco lógico
580
El ensamblador de entrada revisa los elementos de trabajo especificados por la canalización, y asigna el trabajo a los núcleos de la canalización (p.ej., asigna índices a elementos de trabajo para operación en la canalización)

FIG. 5