G06T 15/00 20060101CFI20060101BMMX A63F 13/00 20060101CLI20060101BMMX H04N 13/00 20060101CLI20060101BMMX

Instituto Mexicano de la Propiedad Industrial

(11) **MX PA06007015 A**

(12) **SOLICITUD de PATENTE**

(43) Fecha de publicación: **30/01/2007**

(22) Fecha de presentación: **19/06/2006**

(21) Número de solicitud: **PA06007015**

(86) Número de solicitud PCT: **MX 03/00112**

(87) Número de publicación PCT: **WO 2005/059842 (30/06/2005)**

(71) Solicitante:
TDVISION CORPORATION S.A. DE C.V.
Pina 201-A 02800 Distrito Federal MX

(72) Inventor(es):
Manuel Rafael Gutierrez Novelo
Pina 201-A Distrito Federal 02800 MX

(74) Representante:
MIGUEL ANGEL GARCIA LADRON DE GUEVARA.*
Montecito 38 piso 10 oficinas 29-32 Distrito Federal 03810 MX

(54) Título: **SISTEMA DE VIDEO JUEGOS 3D.**

(54) Title: **3D VIDEOGAME SYSTEM.**

(57) Resumen

Sistema de videojuegos 3D capas de desplegar una secuencia izquierda-derecha por distinto canal independiente VGA o de video, con un dispositivo de presentacion que comparte una memoria en forma inmersa. El sistema posee un motor de videojuego que controla y valida las perspectivas de la imagen, asigna texturas, iluminacion, posiciones, movimientos y aspectos asociados con cada objeto que participa en el juego; crea los backbuffer izquierdo-derecho, crea las imagenes y presenta la informacion en los frontbuffer. Permite el manejo de informacion de datos asociados a las coordenadas xyz de la imagen del objeto en tiempo real, incrementa la RAM para el backbuffer izquierdo-derecho, con la posibilidad de discriminar y tomar el backbuffer correspondiente, cuya informacion es enviada al frontbuffer o dispositivo de presentacion adicional, e independiente que comparte una memoria en forma inmersa.

(57) Abstract

The invention relates to a 3D videogame system that can display a left-right sequence through a different independent VGA or video channel, with a display device sharing a memory in an immerse manner. The system has a videogame engine that control and validates image perspectives, assigns textures, lighting, positions, movements and aspects associated with each object participating in the game; it creates left-right backbuffers and the images and displays the information on the frontbuffers. The system also enables the handling of data associated with the xyz coordinates of the object image in real time. It increases RAM for the left-right backbuffer and makes it possible to discriminate and use the corresponding backbuffer, the information being sent to the frontbuffer or additional, independent display device that shares a memory in an immerse manner.

SISTEMA DE VIDEOJUEGOS 3D

CAMPO DE LA INVENCIÓN

La presente invención se relaciona con la presentación de imágenes de televisión en tercera dimensión, más específicamente al diseño de hardware y software para la visión de imágenes en tercera dimensión (3D) y la fácil integración en los equipos existentes de televisión, computadoras personales y sistemas de juegos de video.

ANTECEDENTES

La interfase visual máquina-hombre siempre trata de mejorar las imágenes para la amplia gama de aplicaciones, ya sean militares, investigación biomédica, imagenología médica, manipulación genética, seguridad de aeropuertos, entretenimiento, videojuegos, computación y algunos otros sistemas de visualización.

La información en tres dimensiones (3D) es la clave para alcanzar el éxito en misiones críticas que requieren de imágenes reales en tres dimensiones que permitan tener información confiable al usuario.

Los sistemas de visión estereoscópica se basan en la

capacidad del ojo humano de ver el mismo objeto desde dos perspectivas diferentes (izquierda y derecha). El cerebro mezcla ambas imágenes y el resultado es una percepción de profundidad y volumen que el cerebro traduce en distancia, superficie y volúmenes.

En el estado de la técnica se han hecho varios intentos para lograr imágenes 3D, por ejemplo, se han usado las siguientes tecnologías:

-Polarización rojo-azul

-Polarización horizontal vertical

-Anteojos con imágenes multiplexadas

-Sistemas 3D de realidad virtual

-Despliegues volumétricos

-Despliegues auto-estereoscópicos

Todas las tecnologías mencionadas tienen incompatibilidades de presentación, así como efectos colaterales nocivos y falta de compatibilidad con la infraestructura de la tecnología actual, a saber:

Los sistemas de polarización rojo-azul requieren para ser observados de un proyector especial y una pantalla blanca de gran tamaño, después a los pocos minutos aparece un efecto colateral, como dolor de cabeza, mareos y otros síntomas asociados con la presentación de las imágenes con el efecto de tercera dimensión. Esta tecnología se uso por largo tiempo

en los sistemas de presentación de películas en los cines, pero debido a los problemas mencionados, poco a poco se fue retirando del mercado. Los síntomas colaterales se deben a la diferencia considerable de contenido que reciben el ojo izquierdo y el ojo derecho (uno recibe la información polarizada en azul y el otro en rojo), lo cual hace que el nervio óptico y el cerebro sufran de tensión excesiva, además se están presentando dos imágenes en forma simultánea. Esta tecnología requiere de una pantalla externa para ser vista y el uso de unos lentes de color polarizados. Si el usuario no tiene los lentes azul-rojo no puede ver el efecto de la tercera dimensión y sólo ve imágenes borrosas y dobles.

El sistema de polarización horizontal-vertical mezcla dos imágenes tomadas por una cámara estereoscópica con dos lentes, las imágenes izquierda y derecha tienen una polarización horizontal y vertical de manera respectiva. Estos sistemas se usan en algunos cinemas nuevos, como los teatros de Disney e IMAX3D. Esta tecnología requiere de sistemas de producción caros de manera considerable y están restringidos a una audiencia dirigida y seleccionada, lo cual reduce el mercado y campo de acción. Un interés especial en el contenido de tres dimensiones (3D) mostrado con esta tecnología ha ido en aumento durante los últimos tres años como las producciones de Tom Hanks y Titanic se han producido con contenido 3D por la tecnología IMAX3D. Esta tecnología

también presenta efectos laterales para el usuario después de unos cuantos minutos de exhibición, requiere de una pantalla externa con el uso de lentes polarizados, si el usuario no tiene los lentes entonces sólo ve imágenes borrosas.

Sistemas con la tecnología de lentes con disparos de imágenes multiplexadas, estos sistemas alternan las imágenes izquierda y derecha por medio del bloqueo de una de ellas para que no pueda llegar al ojo correspondiente en un corto periodo de tiempo. Este bloqueo es sincronizado con la presentación de las imágenes (en un monitor o equipo de televisión). Si el usuario no tiene los anteojos, entonces ve las imágenes borrosas y se presentan efectos laterales después de unos pocos minutos de uso. Esta tecnología la proporciona actualmente (entre otras) BARCO SYSTEMS para las empresas Mercedes Benz®, Ford® y Boeing® que les provee una especie de "cuarto" para crear imágenes de 3D por multiplexión (shutter glasses) para elaborar sus prototipos antes de que sean ensamblados en la línea de producción.

Los sistemas de realidad virtual 3D (VR3D), son sistemas basados en computadoras que crean escenas de computadora que interactúan con el usuario final por medio del uso de interfases de posición, como guantes de datos y detectores de posición. Las imágenes son creadas por

computadora y usan imágenes basadas en un vector, polígonos y reproducción de una profundidad monocular para simular profundidad y volumen calculado por medio de software, pero las imágenes se presentan usando un casco con un dispositivo de exhibición colocado en frente de los ojos, el usuario está inmerso en un escenario generado por computadora, este escenario puede existir sólo en la computadora y no en el mundo real. El nombre de este escenario generado por computadora es la "Realidad Virtual". Este sistema requiere de computadoras muy caras como SGI Oxygen® o SGI Onyx Computers® que están fuera del alcance del usuario común. Juegos serios y simulaciones se crean con esta tecnología, que genera secuencias izquierda-derecha por el mismo canal VGA o de Video, el software incluye instrucciones específicas para alternar las imágenes de video al momento de la representación en la pantalla a una frecuencia de 60 Hz. El software o programa de videojuegos interacciona directamente con la tarjeta gráfica.

Existe una tecnología denominada I-O SYSTEMS, que presenta imágenes multiplexadas en pantallas binoculares; pero lo hace a través de un sistema de multiplexión izquierda-derecha y alternando las imágenes a una frecuencia de entre 80 a 100 Hz, aún así se percibe el parpadeo.

Sistemas de exhibición volumétrica son creados sólo por unos pocos fabricantes, como Perspectra Systems®. Usan la

capacidad del ojo humano para retener una imagen por unos pocos milisegundos y por la rotación de una exhibición circular a una velocidad muy alta, Entonces, de acuerdo al ángulo de visión el dispositivo muestra la imagen correspondiente prendiendo y apagando el color de los pixeles, y debido a que la exhibición gira muy rápido el ojo puede recibir una "imagen flotante". Estos sistemas son muy caros (la "esfera" cuesta aproximadamente 50 000 USD) y requieren de software y hardware específicos y apropiados. Esta tecnología tiene en la actualidad aplicaciones militares.

Las presentaciones auto-estereoscópicas son monitores con líneas semi-cilíndricas que corren de arriba hacia abajo y se aplican solamente a las imágenes frontales y posteriores, esto no es una tercera dimensión verdadera, es solamente una simulación en dos planos de perspectiva. Philips® ha trabajado actualmente en esta tecnología de tres dimensiones del mismo modo que SEGA® con el fin de obtener una ventaja tecnológica. Los resultados son muy pobres y la resolución se pierde en un 50%. Esta tecnología no es compatible con la infraestructura tecnológica actual y requiere el reemplazo completo del monitor del usuario. Las aplicaciones que no son creadas de manera original con esta tecnología se ven borrosas, lo cual la hace completamente incompatible con los inconvenientes de la infraestructura

actual. Para poder observar la imagen en 3D es necesario colocarse a una distancia aproximada de (16") 40.64 cm. Que es variable en función del tamaño del monitor, se debe mirar hacia el centro en forma perpendicular y fijar la vista a un punto focal más lejano al de la pantalla real. Un ligero movimiento de la vista o cambio de ángulo de visión, se pierde el efecto tridimensional.

Existen varias patentes en el estado de la técnica que están involucradas en el desarrollo de esta tecnología, a saber:

Las patentes USP No. 6,593,929, del 15 de julio de 2003 y USP No. 6,556,197, del 29 de abril de 2003, otorgadas a Timothy Van Hook y colaboradores, que se refieren a un sistema para juego de video de bajo costo que puede moldear un mundo en tres dimensiones y lo proyecta sobre una pantalla en dos dimensiones, las imágenes se basan en puntos de vista intercambiables y en tiempo real por un usuario a través de los controladores del juego.

La USP No. 6,591,019, del 8 de julio de 2003, otorgada a Claude Comair y colaboradores, se usa la técnica de compresión y descompresión para la transformación de matrices en sistemas gráficos de 3D que son generados por computadora, la técnica consiste en convertir matrices de números reales en matrices de enteros en la búsqueda de los ceros de la matriz. Las matrices comprimidas ocupan mucho menos espacio

en el sitio de memoria y se pueden descomprimir en tiempo real las animaciones en 3D de manera eficiente.

La patente USP No. 6,542,971, del 1 de abril de 2003, otorgada a David Reed, proporciona un sistema de acceso a memoria y un método que emplea un lugar de memoria auxiliar, un sistema con un lugar de memoria anexo a una memoria que escribe y lee una vez los datos introducidos por uno o más dispositivos periféricos.

La patente USP No. 6,492,987, del 10 de diciembre de 2002, otorgada a Stephen Morein, describe un método y aparato para procesar los elementos de los objetos que son representados; comienza por comparar las propiedades geométricas de al menos un elemento de un objeto con propiedades geométricas representativas por un grupo de pixeles. Durante la representación de los elementos del objeto, se determina una nueva propiedad geométrica representativa y se actualiza con un nuevo valor.

La patente USP No. 6,456,290, del 24 de septiembre de 2002, otorgada a Vimal Parikh y colaboradores proporciona una interfase de un sistema gráfico para la aplicación de un programa de uso y aprendizaje. La característica incluye la representación única de un vértice que permite a la línea gráfica retener la información del estado del vértice, se establecen comandos de matriz de proyección y del cuadro de buffer inmerso.

Cualquier videojuego es un programa de software escrito en algún lenguaje de computación. Tiene la finalidad de simular un mundo inexistente y de llevar a un jugador o usuario a dicho mundo, la mayor parte de ellos están enfocados a ampliar la destreza visual, manual, análisis de patrones y de toma de decisiones, en un ambiente de competencia y superación (nivel de dificultad), se presentan en grandes escenarios de alto contenido artístico. La mayor parte de los videojuegos se dividen como motor de juego en la siguiente estructura: videojuego, biblioteca de juego que tiene asociados los motores gráfico y de audio, el motor gráfico contiene el código fuente 2D y el código fuente 3D; el motor de audio contiene el código de efectos y el código de música.

Todos los bloques que se mencionan en el motor de juego son ejecutados de forma cíclica en lo que se llama game loop (loop de juego), estos motores y bibliotecas se encargan cada uno de operaciones diferentes, por ejemplo:

Motor gráfico: despliega imágenes en general

Código fuente 2D: imágenes estáticas, "backs" y "sprites" que aparecen en una pantalla de videojuego.

Código fuente 3D: imágenes dinámicas, manejadas vectorialmente en tiempo real, procesándose como entidades independientes y con coordenadas de posición xyz dentro del mundo generado por computadora.

Motor de audio: reproducción de sonidos

Código de efectos: cuando suceden eventos especiales como explosiones, choques, saltos, etc.

Código de música: música de fondo que normalmente está siendo reproducida en función del ambiente del videojuego.

La ejecución de todos estos bloques en forma cíclica permite validar posiciones actuales, condiciones, métricas de juego, en función de esta información se afectan los elementos que forman un videojuego.

La diferencia entre programas de juego creados para consolas de videojuego o para computadora tenemos que originalmente la PC IBM no fue creada para jugar con ella, irónicamente, ahora muchos de los mejores juegos corren bajo una tecnología compatible la PC IBM. Comparando las PCs de antes con los videojuegos y capacidades actuales de procesamiento podríamos decir que las PCs eran completamente arcaicas, y fue solo a través de un manejo a bajo nivel (ensamblador) que se comenzaron a crear juegos, haciendo uso directo de la tarjeta gráfica y de la bocina de la computadora. Las cosas han cambiado, el poder de procesamiento de los CPU y las capacidades gráficas, así como la creación de tarjetas especiales diseñadas para acelerar los procesos gráficos (GPUs) han evolucionado a un grado en el que sobrepasan muchas de las características de las denominadas supercomputadoras en los años ochenta.

En 1996 apareció un sistema de aceleración gráfica denominado HARDWARE ACELERATION, que consistía en incluir procesadores gráficos que pueden realizar operaciones matemáticas y matriciales a gran velocidad, quitándole la carga al CPU mediante una comunicación y un lenguaje de programación propio de la tarjeta, encontrado en una capa denominada HAL (Hardware Abstraction layer), que permite el manejo de información de datos asociados a coordenadas xyz en tiempo real, mediante matrices de coordenadas y operaciones matemáticas propias de las matrices como suma, multiplicación por un escalar, comparación de matrices con precisión de punto flotante.

BREVE DESCRIPCIÓN DE LA INVENCIÓN

Un objeto de la presente invención es resolver los problemas de incompatibilidad en las tecnologías en la presentación de imágenes de tres dimensiones.

Otro objeto de la presente invención es proporcionar una tecnología multi-propósito que permita al usuario final ver imágenes de video, gráficos de computadora, videojuegos y simulaciones con el mismo dispositivo.

Es también otro objeto de la presente invención

proporcionar una tecnología que elimina los efectos laterales producidos, después de ver las imágenes en tres dimensiones proporcionadas por las tecnologías actuales, hasta por espacio de horas de uso.

Es todavía un objeto de la presente invención proporcionar una integración de alta tecnología en software con la creación de un par de buffers correspondientes al ojo izquierdo y al ojo derecho, hardware con un dispositivo de presentación adicional, independiente que comparte la memoria en forma inmersa, procesadores de imagen de video digital.

Es otro objeto de la presente invención desplegar la imagen físicamente en la pantalla por medio de dos frontbuffer creados mediante unidades de proceso gráfico o GPU.

Es aún otro objeto de la presente invención proporcionar las percepciones del cerebro de profundidad y volumen con imágenes de gran realismo, aún si son creadas por software para gráficos de computadora.

Es todavía otro objeto de la presente invención proporcionar un algoritmo TDVision® para crear imágenes de computadora con alto grado de realismo.

Es otro objeto de la presente invención el hacer cambios en la base tecnológica actual para crear un nuevo proceso digital de imagen con técnicas ópticas para lograr una percepción de imagen real al establecer la vista de una

cámara derecha.

Es también otro objeto de la presente invención lograr la convergencia digital de medios, donde se puede usar una computadora para ver DVD, la Lap Top para producir cine, la capacidad de Internet para transmitir imágenes de video, la PC y las consolas de video juegos en la estructura de internet.

Es otro objeto de la presente invención proporcionar un nuevo algoritmo de software ensamblador, hardware analógico y digital para obtener la mejor adaptación a los equipos de 3D de las tecnologías existentes.

Es todavía otro objeto de la presente invención proporcionar sistemas visuales de cómputo en tercera dimensión para la generación de imágenes estereoscópicas por animación, representación y modelado en software.

BREVE DESCRIPCIÓN DE LAS FIGURAS

La figura 1, representa el mapa de tecnología de TDVision® para videojuegos.

La figura 2, representa la estructura principal de un videojuego de la técnica previa.

La figura 3, representa el elemento tridimensional esencial de la técnica previa, para construir una objeto en una posición en el espacio.

La figura 4, representa el esquema de desarrollo de un programa de videojuegos basado en las tecnologías de las funciones de API de OpenGL y DirectX.

La figura 4 a, representa el diagrama de bloques del algoritmo para crear los buffers izquierdo y derecho, además de discriminar si es tecnología TDVision.

La figura 4 b, representa el diagrama de bloques de la subrutina para establecer la vista de la cámara derecha después de dibujar la imagen en el backbuffer derecho en función del vector de la cámara derecha, también discrimina si se trata de formato de tecnología TDVision.

La figura 5, representa el diagrama de bloques del esquema computacional de las modificaciones al adaptador gráfico para que pueda compilar la tecnología TDVision, por otro lado, permite la comunicación y contiene el lenguaje de programación y permite el manejo de información de datos asociados con el conjunto de imágenes.

La figura 6, representa el diagrama de bloques del algoritmo que permite dibujar la información en el backbuffer TDVision y presentarla en pantalla en formato DirectX 3D.

La figura 7, representa la secuencia de representación utilizando el algoritmo en formato OpenGL.

La figura 8, representa el diagrama de bloques de la presentación de información en pantalla por medio de los backbuffers izquierdo y derecho, utilizando el algoritmo

OpenGL.

La figura 9, se representan los cambios necesarios a la tarjeta de video usada en la tecnología TDVision.

DESCRIPCIÓN DETALLADA DE LA INVENCIÓN

Los videojuegos son procesos que comienzan por proveer una pluralidad de estados lógicos que se relacionan de manera independiente y que incluyen un conjunto de opciones de programación, donde cada opción de programación corresponde a diferentes características de imagen. Las instrucciones de programa genéricas pueden ser compiladas en un código por varios dispositivos de computación sin tener que generar de manera independiente los códigos objeto para cada uno de los dispositivos.

Los dispositivos de computación tales como computadoras personales, laptops, videojuegos, etc., incluyen una unidad central de proceso, sistemas de memoria, circuitos de proceso gráfico de video, circuitos de proceso de audio y puertos periféricos. Típicamente la unidad central procesa un software para generar datos geométricos que se refieren a la imagen que se va a representar y proporciona los datos geométricos al circuito gráfico de video que genera los datos de pixeles que se almacenan en un cuadro de memoria de donde la información pasa al dispositivo de despliegue, lo

anterior en conjunto se llama motor del videojuego (Fig. 2).

Algunos motores de juego se licencian a terceros, como el caso del programa Quake III Arena que cuenta con el motor de juegos denominado QUAKE ENGINE y que se licencio al programa VOYAGER ELITE FORCE, que utiliza el quake engine. De este modo los desarrolladores de juegos pueden concentrarse en las métricas del juego, en vez de desarrollar un motor de juego desde cero. Originalmente los videojuegos utilizaban únicamente imágenes en dos dimensiones, denominados "sprites" que eran los protagonistas del juego.

La mayor parte de los videojuegos y tecnologías evolucionaron y permitieron el trabajo de objetos simulados en un ambiente o mundo tridimensional, otorgando a cada objeto propiedades de posición en xyz, rodeados de otros objetos con las mismas características y actuando juntos en un mundo con un origen (0,0,0).

En un inicio las consolas de videojuegos separadas del mundo de la computación dieron el primer paso para incorporar gráficos 3D como una capacidad física gráfica de dichos dispositivos, técnicas que después fueron adoptadas por el hardware que se utiliza en las PCs. También se incluye un elemento de análisis de circunstancias que se denomina comúnmente como inteligencia artificial aplicada a un videojuego, este proceso analiza la situación, posiciones, choques, riesgos de juego, ventajas, y en base a este

análisis genera una acción de respuesta para cada objeto que participa en el videojuego.

Se hace uso de un backbuffer, que es un lugar en la memoria donde temporalmente se "dibuja" la imagen que va a ser presentada sin dar salida a la tarjeta de video. Si se hace directamente en la memoria de video se obtendría un parpadeo en la pantalla, por lo que se dibuja y procesa la información en el backbuffer de manera rápida. Este backbuffer se encuentra por lo general dentro de la memoria física RAM de la tarjeta de video o de aceleración de gráficos.

Una secuencia típica dentro del algoritmo de un videojuego sería:

1) Desplegar la pantalla de títulos
2) Cargar los personajes, objetos, texturas y sonidos en memoria
3) Crear un lugar de memoria para proceso temporal, denominado doublebuffer o backbuffer
4) Desplegar la imagen de fondo
5) Grabar la imagen debajo de cada elemento que participa en el juego
6) Borrar todos los elementos de la memoria (doublebuffer)

7) Verificar la entrada del usuario y actualizar la posición del jugador
8) Procesar la posición de los enemigos utilizando inteligencia artificial (IA)
9) Mover cada objeto participante a su nueva posición
10) Verificar colisiones entre objetos
11) Incrementar el cuadro de animación
12) Dibujar los objetos en la memoria backbuffer
13) Transferir los datos de backbuffer a pantalla
14) Regresar al paso 5, excepto que el usuario quiera terminar el juego (paso 15)
15) Borrar los objetos de memoria
16) Terminar el juego.

Los dispositivos más comunes utilizados en una consola de video son El CPU o Unidad Central de Proceso, que se encarga del loop de juego, de la entrada del usuario vía teclado, del mouse, de los dispositivos de juego como (gamepad, joystick) y del proceso de la inteligencia artificial del juego.

El GPU o Unidad de Proceso Gráfico se encarga de modelado de polígonos, mapeos de textura, transformaciones y de la simulación de iluminación.

El DSP o Procesador de Señales Digitales de Audio se encarga de reproducir la música de fondo, los efectos de sonido y el sonido de posición en 3D.

El Motor de Gráficos es la sección del juego en software que se encarga de controlar y validar las perspectivas, asignar texturas (metálicas, tipo piel, etc.) iluminación, posiciones, movimientos y demás aspectos asociados a cada objeto que participa en el videojuego, ya sea para consola de videojuegos o para PC. Este conjunto de imágenes es procesado con respecto al origen asignado y calculando las perspectivas de distancia, profundidad y posición. Esto se realiza mediante dos pasos; pero es complicado por las operaciones matemáticas involucradas. A saber, proceso de traslación de objetos (offset desde el origen), proceso de rotación de objetos (ángulo de giro respecto a la posición actual).

Es importante mencionar que las unidades mínimas de imagen (fig. 3) se componen de unidades mínimas de control llamadas "vertex", que representan un punto en el espacio xyz. La unidad geométrica mínima permitida es el triángulo, que se construye con un mínimo de tres puntos en el espacio, a partir de la unidad base triángulo se forman objetos más grandes que se componen de miles de pequeños triángulos, como el personaje de Mario Sunshine. Esta representación se le denomina "Mesh" y a cada mesh e incluso cada triángulo se le puede asociar una textura, un color e incluso características de presentación gráfica. A esta información se le denomina gráficos en 3D. Algo muy importante de mencionar es que aún

cuando se le denomina gráfico en 3D por su naturaleza constructiva por medio de vectores xyz, la presentación final para el usuario suele ser en 2D, en un motor plano con contenido basado en vectores 3D pero el usuario final los ve como si los objetos estuvieran enfrente del mismo, sólo parecen tener ciertas características inteligentes de profundidad y de iluminación, pero para el cerebro no parecen tener un volumen en el espacio.

Originalmente era necesario que los programas de videojuego se comunicaran directamente con las tarjetas gráficas para realizar operaciones de aceleración y de manejo matemático complejo, lo cual representaba prácticamente rescribir gran parte del juego para soportar una tarjeta de video diferente. Ante este problema, Silicon Graphics® se enfocó en desarrollar una capa de software (OpenGL®) que se comunicara directamente con el hardware, con una serie de funciones y subrutinas útiles que de manera independiente del hardware se pueden comunicar con el mismo, pero solo para la parte gráfica. Microsoft® por su lado también realizó este mismo grupo de funciones denominado DirectX 3D®, muy similar al OpenGL®, pero más completa porque agrega áreas de control de sonido, de juego en red, entre otros.

Este conjunto de funciones y de subrutinas recibe el nombre de Interfase de programación de aplicaciones de gráficos (GRAPHICS API). Estas API se pueden acceder desde

diferentes lenguajes de programación, como C. C++, Visual Net, C#, Visual Basic, entre otros.

Todos los sistemas de realidad virtual mencionados utilizan actualmente un esquema de secuencia izquierda-derecha por el mismo canal VGA o de Video, estos sistemas requieren que el software incluya instrucciones específicas para alternar las imágenes de video al momento de la representación de pantalla en el backbuffer, aplicando un algoritmo conocido de desfase por offsets y ángulo a manera de simulación.

Además de las funciones que otorgan las API de OpenGL® y de DirectX® se cuenta con una serie de funciones de manejo gráfico dentro de una interfase de programación de aplicación que proporciona Windows®, denominada WINDOWS API.

El esquema de desarrollo de un programa de videojuegos basado en éstas tecnologías es el representado en la figura 4, en el que se incluye la implementación del software para videojuego desarrollado en la presente solicitud por TDVision® Corp. En la fig. 4 se esquematiza el diagrama de flujo que inicia con la implementación del software con la métrica apropiada para el videojuego (40), el software se desarrolla en cualquier lenguaje de programación apropiado como (C, C++, Visual Basic, Otros) (41), se introduce el código fuente del videojuego (42), la lógica del juego y las características de los objetos, sonido, eventos, etc. (43),

en
(44) se encuentra el selector de eventos que lo puede hacer por medio de las API
de Windows (45), de OpenGL (46) o DirectX (47) que finalmente se manda a la
representación de video (48).

Todo esto se refiere al software y algo interesante es que DirectX cuenta con una gran cantidad de funciones, y Microsoft® logró que aún cuando inicialmente algunas requerían de hardware específico la misma API de DirectX pudiera emular las características por software como si existiera realmente el hardware.

La presente invención hace uso máximo y óptimo de éstas tecnologías OpenGL® y DirectX®, para obtener un software con ciertas características específicas , algoritmos y procesos digitales para cumplir con las especificaciones establecidas por TDVision que se usan el la presente solicitud.

Con respecto al hardware, se podrá analizar el HAL y la interfase directa mediante drivers para cada una de las tarjetas, y para efectos de la implementación de tecnología TDVision será necesario analizar las especificaciones mínimas y requerimientos, así como algunos posibles cambios en la tecnología que permitan obtener 3D real en los 3Dvisors de TDVision.

Respecto a los sistemas de despliegue o representación,

se tiene que la información generada por el software y que se coloca en el Graphic Device Context o Image Surface se transmite directamente a la última etapa de la tarjeta gráfica, que se encarga de convertir la señal de video digital a señales analógicas o digitales (en función del monitor de representación), y entonces se presenta la imagen en la pantalla.

Los métodos de despliegue actuales son:

Monitor analógico con señal digital de computadora

Monitor digital

Monitor analógico con señal de televisión

Sistemas de realidad virtual 3D.

El tipo de salida (s) depende de la tarjeta de video, que deberá estar conectada a un monitor compatible.

En la figura 4 a, se representa la creación de los lugares de memoria para proceso temporal de gráficos (backbuffer) izquierdo y derecho, en el que básicamente agrega un lugar de memoria extra, esto es, establece un buffer derecho en (400) y discrimina si es tecnología TDVision en (401), en caso afirmativo, establece bufffer izquierdo en (402) y termina en (403), cuando no es tecnología TDVision el proceso termina en (403) puesto que no hubo que discriminar.

En la figura 4 b, se representa el diagrama de flujo de la discriminación y representación de la imagen de la cámara

izquierda y la cámara derecha, en efecto, se establece la vista izquierda en (410), se dibuja la imagen en el backbuffer izquierdo (411) en función de la posición de la cámara, se presenta la imagen en la pantalla izquierda (412), se discrimina si tiene formato TDVision en (413) y en caso afirmativo se calculan las coordenadas de posición de la vista derecha (414), se dibuja la imagen en el backbuffer derecho en función de la posición de la cámara izquierda (415), presentación de la información en la pantalla derecha (416), el proceso termina en (417), si no es necesario discriminar en (413) porque la imagen proviene de un formato de la técnica actual, la subrutina se va a la etapa final (417) y termina, puesto que no hubo necesidad de calcular otras coordenadas y presentar una información paralela. La parte novedosa de la presente solicitud se refiere a la unidad de proceso gráfico representado en la figura 5, (GPU HARDWARE) y el motor de gráficos (GRAPHICS ENGINE, SOFTWARE).

Las modificaciones en hardware son:

-Incremento de la RAM para backbuffer izquierdo y derecho

-Implementar un dispositivo de representación adicional en el buffer de representación, de manera independiente a que comparta la memoria en forma inmersa para que tome el backbuffer correspondiente.

En este caso es necesario que la memoria RAM del

backbuffer y el frontbuffer de la tarjeta de video sean suficientemente amplios para soportar los canales izquierdo y derecho en forma simultánea. Esto nos obliga a tener un mínimo de 32 MB para soportar cuatro buffers con una profundidad de 1024x768x4 bytes de profundidad de color cada uno. Además, la salida de video es dual (dos puertos VGA), o bien tiene la capacidad de manejar monitores múltiples, como es el caso de la tarjeta ATI RADEON 9500®, que tiene dos sistemas de despliegue de salida, uno VGA, otro S-Video, a elegir un puerto de video. Se crea una tarjeta de gráficos que tenga doble salida únicamente para cumplir con la presentación a 60 cuadros por segundo por canal izquierdo-derecho para ser conectados a un 3Dvisor, estas salidas son del tipo SVGA, S-Video, RCA o DVideo.

El esquema computacional se presenta con modificaciones para la compilación TDV, como se describe en la figura 5, un CPU (50), el controlador de memoria (51) y la memoria ampliada (52), esta memoria alimenta al controlador de audio (53) y a las bocinas (54); también al controlador de salida y entrada (55) que a su vez controla los puertos de disco (56) y a los elementos interactivos con el usuario (57) como (ratón) mouse, KBD, gamepad, joystick; por otro lado, el controlador de gráficos interacciona directamente con el monitor (59) y los visores tridimensionales 3DVISORS (59 b).

En lo que se refiere específicamente al hardware de la

parte de gráficos (HAL) se necesitan cambios para que se pueda compilar con la tecnología TDVision, en efecto se tiene la aplicación (500) que envía la información a los controladores de gráficos (501) que realizan su función debido al apoyo del hardware para gráficos (502) que necesitan de cambios físicos para que compilen la tecnología TDVision. Para implementar la tecnología TDVision mediante OpenGL y DirectX es necesario realizar modificaciones en partes de la sección de software de un videojuego y también como ya se mencionó en algunas secciones del hardware.

En el caso del software es necesario agregar algunas características especiales dentro de un algoritmo típico de trabajo, así como una llamada a una subrutina TDVision, a saber, como se representa en la figura 6:

-Cargar información de superficies (600)

-Cargar información de meshes (601)

-Crear backbuffer TDVision (602), en el que se crea un backbuffer izquierdo en memoria, si es tecnología TDVision entonces, crea backbuffer derecho en memoria.

-Aplicar coordenadas iniciales (603)

-Aplicar lógica de juego (604)

-Validación e inteligencia artificial (605)

-Cálculo de posiciones (606)

-Verificar colisiones (607)

-Dibujar información en backbuffer TDVision y presentar en pantalla (608), en la que se debe establecer la vista de cámara derecha, dibujar la imagen en el backbuffer derecho en función del vector actual de la cámara derecha, presentar imagen en la pantalla (front buffer) derecha. Si es tecnología TDVision, entonces: calcular las coordenadas del par izquierdo, establecer vista de cámara izquierda, dibujar la imagen en el backbuffer izquierdo en función del vector actual de la cámara izquierda, presentar la información en pantalla (front buffer) derecha que requiere de modificación en hardware.

Creando así un par de buffers correspondientes al ojo izquierdo y al ojo derecho, que al ser evaluados en el loop de juego "game loop" obtendrán las coordenadas vectoriales correspondientes a la visualización de cada cámara derecha (actual) y cámara izquierda (complemento calculado con la función SETXYZTDV) que se muestra abajo.

Es importante mencionar que dichos buffers de salida a pantalla o front buffers son asignados desde el inicio a la superficie de representación de video (device context) o a la superficie en cuestión (surface), pero para efectos de desplegar la información en un 3Dvisor de TDVision es necesario que se presenten físicamente dos salidas de video, la derecha (normal VGA) y la izquierda (VGA adicional o complemento digital o Svideo) para ser compatible con

TDVision. En el ejemplo se utilizó DirectX, pero el mismo proceso y concepto se aplica para el formato de OpenGL.

En la figura 7, se bosqueja el algoritmo (70) que conduce una línea de representación de la interfase de comunicación de aplicaciones gráficas, en efecto, por medio de trigonometría (72) con las llamadas operaciones de vértices (77) se construye la imagen (71) y por medio de operaciones con pixeles o elementos de imagen (75) a través de los comandos (73), la lista de despliegues (74) y una memoria que le asigna una textura a la imagen (76) se llega a la representación (78) que por medio de las operaciones (79) las envía al cuadro de memoria (70F). El software de windows (700) se comunica con (702) y con la tarjeta de lenguaje gráfico (701), que a su vez contiene una librería de información gráfica útil para alimentar a (703), (704).

La figura 8, representa la tecnología TDVision usando el algoritmo (80) OpenGL para desplegar la imagen izquierda y derecha del objeto, limpia el backbuffer (81), adquiere el apuntador para el backbuffer (82), cierra el backbuffer (83), vuelve a hacer la escena (84), abre el backbuffer (85), libera el apuntador al backbuffer (86), envía la imagen a la superficie de representación izquierda; en (800) discrimina si se trata de tecnología TDVision y en caso afirmativo limpia la memoria (801) y adquiere un apuntador para el backbuffer (802), cierra el

backbuffer (803), adquiere las coordenadas de la nueva perspectiva (804), vuelve a
hacer la escena (805), abre la memoria (806) y libera el apuntador al backbuffer
(807), envía la imagen a la superficie de representación derecha (808).

La figura 9, representa los cambios (90) de la tarjeta de video que es necesario hacer para que se pueda compilar con la tecnología TDVision, en efecto, se tiene el backbuffer normal izquierdo (91) que antecede al backbuffer primario izquierdo normal (92) el cual se conecta a la salida VGA del monitor (95) el cual debe poseer otra salida VGA para que reciba el backbuffer primario derecho (94) que a su vez tiene como antecedente el backbuffer derecho propio de la tecnología TDVision. Ambos backbuffers izquierdo y derecho se pueden conectar a un 3Dvisor (96) que tiene doble entrada VGA para recibir y representar la información enviada por los backbuffer (91) y (93).

Estas modificaciones en software hacen uso de las siguientes funciones API en DirectX:

Crear backbuffer TDVision:

```
FUNCTION CREATE BACKBUFFERTDV()
buffer izquierdo
  Set d3dDevice = d3d.CreateDevice(D3DADAPTER_DEFAULT,_
    D3DDEVTYPE_HAL,hWndL,_
```

```
            D3DCREATE_SOFTWARE_VERTEXPROCESSING, d3dpp)
        If GAMEISTDV then
         Buffer derecho
        Set                    d3dDeviceRight                    =
5   d3d.CreateDevice(D3DADAPTER_DEFAULT,_
            D3DDEVTYPE_HAL,hWndR,_
            D3DCREATE_SOFTWARE_VERTEXPROCESSING, d3dpp2)
        Edif
        END SUB
10      Dibujar imagen en backbuffer TDVision:
        FUNCTION DRAWBACKBUFFERTDV()
         DIBUJA ESCENA IZQUIERDA
        d3dDivice.BeginScene
        d3dDivece.SetStreamSource0, poly 1_vb, Len(poly1.v1)
15      d3dDevice.DrawPrimitive D3DPT_TRIANGLELIST,0,1
        d3dDevice.EndScene

              Copiar backbuffer a frontbuffer, pantalla
              D3dDivice.Present By Val 0,By Val 0, 0, By Val 0
20            'VERIFICAMOS SI ES UN PROGRAMA TDVISION POR FLAG
               IF GAMEISTDV THEN
                   'CALCULAR COORDENADAS CAMARA DERECHA
                    SETXYZTDV ()
                     ' dibuja escena derecha
25                    d3dDevice2.BeginScene
```

```
                    d3dDevice2.Set StreamSource 0, poly2_vb,
Len(poly1,v1)
                     d3dDevice2.DrawPrimitive
D3DPT_TRIANGLELIST,0,1
                      d3dDevice2.EndScene
                       d3dDevice2.Present ByVal 0, ByVal 0,
0, ByVa
      END SUB.
      Las modificaciones del vector de cámara xyz:
            VecCameraSource.z = posición z
            D3DXMatrixLook AtLH matView, vecCameraSource,_
                VecCameraTarget, CreateVector (0,1,0)
            D3dDevice 2.SetTransform D3DTS_VIEW, matView

      VecCameraSource.x = posición x
            D3DXMatrixLook AtLH matView, vecCameraSource,_
                VecCameraTarget, CreateVector (0,1,0)
            D3dDevice 2.SetTransform D3DTS_VIEW, matView

      VecCameraSource.y= posición y
            D3DXMatrixLook AtLH matView, vecCameraSource,_
                VecCameraTarget, CreateVector (0,1,0)
            D3dDevice 2.SetTransform D3DTS_VIEW, matView

      Creando así un par de buffers correspondientes al ojo
```

izquierdo y al ojo derecho, de manera respectiva, que al ser evaluados en el loop de juego obtendrán las coordenadas vectoriales correspondientes a la visualización de cada cámara derecha y cámara izquierda (complemento calculado con la función SETXYZTDV) por medio de las ecuaciones usuales de transformación de coordenadas.

Es importante mencionar que dichos buffers de salida a pantalla o frontbuffer son asignados desde un inicio al device context o a la superficie en cuestión, pero para efectos de desplegar la información en un 3Dvisor de TDVision es necesario que se presenten físicamente dos salidas de video, la derecha (normal VGA) y la izquierda (VGA adicional o complemento digital o SVIDEO) para que sea compatible con TDVision.

El ejemplo se hizo utilizando DirectX, pero el mismo proceso y concepto aplica para el formato OpenGL representado en la figura 8.

En este caso es necesario que la memoria RAM del backbuffer y el frontbuffer de la tarjeta de video sean suficientemente amplias para soportar los canales izquierdo y derecho en forma simultánea. Esto nos lleva a tener un mínimo de 32MB para poder soportar cuatro backbuffer con una profundidad de color de 1024x768x4 bytes cada uno. Como ya se indicó arriba, la salida de video debe ser dual (dos puertos VGA), o bien tener la capacidad de manejar monitores

múltiples, como es el caso de la tarjeta ATI RADEON 9500®, que tiene dos sistemas de despliegue de salida, uno VGA, otro S-Video, y a elegir un puerto de Dvideo.

Se crea una tarjeta de gráficos que tenga doble salida únicamente para cumplir con la presentación de 60 cuadros por segundo por canal izquierdo-derecho para ser conectados a un 3Dvisor, estas salidas podrán ser del tipo SVGA, S-Video, RCA o Dvideo.

Así podremos obtener vía software las imágenes correspondientes al punto de vista de la cámara en ambas perspectivas izquierda y derecha y el hardware reconocerá la información para ser presentada en dos salidas de video diferentes e independientes, sin multiplexión y desplegadas en tiempo real. Actualmente todas las tecnologías utilizan multiplexión y simulación de software, en la tecnología propuesta por la presente solicitud se podrá obtener información real y al utilizar los 3Dvisors se podrá visualizar la imagen desde dos perspectivas diferentes y el cerebro le asociará el volumen que ocupa en el espacio, sin que aparezca ningún parpadeo (o flickering) en la pantalla, efecto asociado a las tecnologías del estado de la técnica.

Método para cálculo de coordenadas de cámara secundaria estereoscópica (SETXYZTDV()) que permite obtener sistemas visuales de cómputo en tercera dimensión para la generación de imágenes estereoscópicas por animación, representación y

modelado en programas de software. Este método permite la obtención de coordenadas espaciales (x, y, z) que se deben asignar a dos cámaras de visualización virtuales generadas por computadora para obtener una visión estereoscópica por medio del uso de cualquier programa de software que simulan la tercera dimensión y generan las imágenes mediante polígonos que permiten su visualización en diferentes perspectivas mediante el movimiento del objeto, o bien, mediante el movimiento de la "cámara virtual" que observa en ese instante el objeto generado por computadora, tal como: Autocad, Micrografix simply 3D, 3Dmax Studio, Point, Dark Basic, Maya, Marionette, Blender, Excel, Word, Paint, Power, Corel Draw, Photo paint, Photoshop, etc.; pero todos estos programas están hechos para representar una sola cámara con una sola perspectiva fija o en movimiento.

Se agrega una característica adicional a los anteriores programas de modelado en 3D y animación, en efecto, por medio de las ecuaciones de transformación de coordenadas, a saber:

$$x = x' \cos \phi - y' \operatorname{sen} \phi$$

$$y = x' \operatorname{sen} \phi + y' \cos \phi$$

se calcula la posición exacta de una segunda cámara o cámara secundaria , que está vinculada directamente con la primera cámara y de este modo se obtienen dos imágenes simultáneas desde diferentes perspectivas que simulan la perspectiva visual estereoscópica del ser humano. Este procedimiento,

mediante un algoritmo calcula en tiempo real la posición de la cámara secundaria para ubicarla en la posición adecuada, y de este modo obtener la imagen de modelado y representación de la segunda cámara, que se logra utilizando las ecuaciones de transformación de coordenadas, llevando la cámara al origen, se calcula el ángulo y la distancia entre la cámara secundaria y el objeto u objetivo, enseguida se vuelve a posicionar la cámara primaria, el objetivo y la cámara secundaria en la posición obtenida. Entonces se necesita conocer siete parámetros , a saber, las primeras tres coordenadas de la posición de la cámara primaria (x_p, y_p, z_p) en el sistema original de coordenadas, el cuarto parámetro es la distancia equivalente a la separación promedio de los ojos (6.5 a 7.0 cm), las tres coordenadas de posición del objetivo al ser observado por las cámaras.

Los parámetros de salida serán las coordenadas de la cámara secundaria en observación del mismo punto objetivo, es decir, (x_s, y_s, z_s), que se obtiene siguiendo los pasos:

- Conocer las coordenadas de la cámara primaria en el sistema original de coordenadas (x_p, y_p, z_p),
- Conocer las coordenadas del objetivo (x_t, y_t, z_t)
- Solo se transforma las coordenadas "x" y "z" , puesto que la coordenada y o altura de la cámara se mantiene constante (el

observador no tiene

desviación visual)

- Se llevan las coordenadas de la cámara primaria a la posición $(0, y_s, 0)$
- Se traslada también el objetivo
- Se calcula la pendiente de la recta que une a la cámara y el objetivo
- Se calcula el ángulo entre el eje y el vector que une la cámara primaria con el objetivo
- Se clasifica el cuadrante al que pertenece para aplicar consideraciones especiales en el cálculo del ángulo por medio de la función tangente inversa
- Se obtienen las nuevas coordenadas rotando todo el sistema de coordenadas desde su eje en el mismo ángulo entre el eje y el vector, se obtiene un nuevo sistema de coordenadas en el que se coloca el objeto sobre el eje z' y la cámara primaria quedará en el origen del nuevo sistema de coordenadas
- Se obtienen las coordenadas de la cámara secundaria colocándola en la posición de la distancia promedio entre los ojos del ser humano
- Se rotan estas coordenadas en el mismo ángulo inicial
- Se suman los desfases en "x" y en "z" que se restaron originalmente para llevar la cámara primaria al

origen

- Finalmente estas dos coordenadas nuevas, x_s y z_s se asignan a la cámara secundaria y se mantiene la coordenada y_p que determina la altura para el mismo valor para un punto final de coordenadas (x_s, y_p, z_s) que será asignado a la cámara secundaria.

El procedimiento podrá ser implementado en lenguajes tales como Delphi, C, C++, Visual C++, Omnis, etc., pero el resultado será el mismo.

La aplicación generalizada de este algoritmo será utilizada en cualquier programa que quiera obtener en tiempo real la posición de una cámara secundaria.

Este algoritmo deberá implementarse en cualquier software existente que maneja dos dimensiones, pero se ha desarrollado para aplicaciones de visión estereoscópica.

Se han ilustrado y descrito modalidades particulares de la presente invención, será obvio para aquellos expertos en la técnica que se pueden hacer varias modificaciones o cambios sin salir del alcance de la presente invención. Lo anterior se intenta cubrir con las reivindicaciones agregadas para que todos los cambios y modificaciones caigan dentro del

alcance de la presente invención.

Habiendo descrito la invención que antecede, se reclama como propiedad lo contenido en las siguientes reivindicaciones:

REIVINDICACIONES

1.- Un sistema de videojuegos 3D capaz de procesar un flujo de datos de video por medio de un motor gráfico que procesa el código de requerimientos de gráficos en 3D, caracterizado porque comprende:

tomar las instrucciones de programa de aplicación OpenGL® o DirecTX® dentro del motor de juego;

crear un par de buffers o líneas de memoria física, correspondientes al ojo izquierdo y al ojo derecho;

representar la imagen de la cámara virtual izquierda en el backbuffer izquierdo o establecimiento de la imagen izquierda en función de la posición de la cámara;

calcular de las coordenadas de posición de la vista derecha;

representar la imagen en el backbuffer derecho en función de la posición de la cámara virtual izquierda;

crear lugares separados de memoria para proceso temporal gráfico o backbuffer izquierdo y derecho, en el que se agrega un lugar de memoria extra al establecer un buffer derecho que se encuentra en una localidad de memoria adyacente y diferente al buffer izquierdo;

implementar un dispositivo de representación

independiente adicional en el buffer de representación que comparte la memoria en forma inmersa para que discrimine y tome el backbuffer correspondiente;

una unidad de proceso gráfico junto con el motor de gráficos, en donde se incrementa la RAM para el backbuffer independiente izquierdo o derecho;

una tarjeta de gráficos con salida dual de video ;

generar imágenes izquierda o derecha por diferente canal VGA o de video.

2.- Sistema de videojuegos 3D de acuerdo con la reivindicación 1, caracterizado además porque el backbuffer o lugar de memoria física RAM de la tarjeta de video se incrementa para soportar ambos buffers de salida y dibuja el conjunto de imágenes de la escena de forma temporal y rápida, sin dar salida a la tarjeta de video, crea los pares estereoscópicos dentro de la aplicación antes de la representación.

3.- Sistema de videojuegos 3D de acuerdo con la reivindicación 1, caracterizado además porque al establecer el buffer izquierdo discrimina si la imagen es por tecnología TDVision® y establece el buffer derecho en memoria; establece vista de cámara derecha; dibuja la imagen en el backbuffer derecho en función del vector de la cámara derecha; presenta

la imagen en frontbuffer derecho; calcula las coordenadas del par izquierdo; establece vista de cámara izquierda; dibuja la imagen en el backbuffer izquierdo en función del vector de la cámara izquierda, por medio de rotación y traslación de ejes en 3D; presenta la información en tiempo real en pantalla TDVision® para tener una percepción de profundidad, volumen o distancia y superficie.

4.- Sistema de videojuegos 3D de acuerdo con la reivindicación 1, caracterizado además porque el flujo de datos de video digital es un flujo de video en tiempo real; el motor gráfico incluye un control para desplegar el flujo de video digital en tiempo real, imágenes izquierda y derecha en los monitores respectivos; el flujo de video se despliega en un dispositivo de presentación (frontbuffer) adicional e independiente VGA o de video que comparte una memoria en forma inmersa y es capaz de desplegar una secuencia izquierda y derecha por un canal independiente.

5.- Sistema de videojuegos 3D de acuerdo con la reivindicación 1, caracterizado además porque se incrementa la RAM del backbuffer izquierdo a más de 32 MB.

6.- Sistema de videojuegos 3D de acuerdo con la reivindicación 1, caracterizado además porque se incrementa

la RAM del backbuffer derecho a más de 32 MB.

7.- Sistema de videojuegos 3D de acuerdo con la reivindicación 1, caracterizado además porque tiene la posibilidad de discriminar y tomar el backbuffer correspondiente para presentar la imagen completa y de forma independiente en pantalla.

8.- Sistema de videojuegos 3D, que comprende la implementación del software del videojuego que se programa en cualquier lenguaje de programación utilizando un código y lógica de juego fuente que responde a las acciones y eventos del usuario a través de una serie de funciones de manejo gráfico dentro de una interfase de programación como OpenGL® y DirecTX® que envían las imágenes a la superficie de representación, caracterizado porque en el software del videojuego TDVision® , comprende:

una subrutina que carga información de superficies;

carga información de meshes;

crea backbuffer izquierdo, derecho por medio de una subrutina;

aplica coordenadas iniciales;

aplica lógica de juego;

valida inteligencia artificial;

calcula posiciones;

verifica colisiones;

dibuja información en backbuffer izquierdo, derecho y la presenta en pantalla;

evalúa el loop de juego;

obtiene las coordenadas vectoriales correspondientes a la visualización de cada cámara izquierdo o derecha;

presenta la información en tiempo real por medio de una serie de funciones de manejo gráfico dentro de una interfase de programación como OpenGL®, DirecTX® que envían las imágenes a la superficie de representación.

9.- Sistema de videojuegos 3D de acuerdo con la reivindicación 8, caracterizado además porque los pasos para dibujar en el backbuffer TDVision® consiste en:

crear vista izquierda;

dibujar en el backbuffer izquierdo en función de la posición de la cámara;

presentar la imagen en el frontbuffer izquierdo;

discriminar si es formato de tecnología TDVision® ;

calcular coordenadas del par derecho;

dibujar en el backbuffer en función de la posición de la cámara izquierda;

presentar la información en el frontbuffer derecho.

10.- Sistema de videojuegos 3D de acuerdo con la

reivindicación 8, caracterizado además porque para la presentación de la imagen desde el backbuffer hasta la pantalla usa un algoritmo con los siguientes pasos:

limpiar el backbuffer;

obtener un apuntador para el backbuffer;

cerrar el backbuffer;

rehacer la escena;

abrir el backbuffer;

liberar el apuntador al backbuffer;

discriminar si es formato TDVision®;

presentar la imagen en pantalla izquierda;

si es formato TDVision®, entonces;

limpiar el backbuffer;

obtener un apuntador para el backbuffer;

cerrar el backbuffer;

obtener las coordenadas del punto;

rehacer la escena;

abrir el backbuffer;

liberar el apuntador del backbuffer;

presentar la imagen en pantalla derecha.

11.- Sistema de videojuegos 3D de acuerdo con la reivindicación 8, caracterizado además porque el software calcula las coordenadas de una cámara secundaria estereoscópica que permite obtener sistemas visuales de

cómputo en tercera dimensión para la generación de imágenes estereoscópicas, obtiene las coordenadas espaciales (x,y,z), por medio de las ecuaciones de transformación de coordenadas, que se asignan a dos cámaras virtuales de visualización para la obtención de una visión estereoscópica; calcula la posición exacta de una cámara secundaria vinculada directamente con la primera cámara situada en el origen de coordenadas; obtiene dos imágenes completas simultáneas desde diferentes perspectivas que le dan la visión estereoscópica al ser humano.

RESUMEN

Sistema de videojuegos 3D capaz de desplegar una secuencia izquierda-derecha por distinto canal independiente VGA o de video, con un dispositivo de presentación que comparte una memoria en forma inmersa.

El sistema posee un motor de videojuego que controla y valida las perspectivas de la imagen, asigna texturas, iluminación, posiciones, movimientos y aspectos asociados con cada objeto que participa en el juego; crea los backbuffer izquierdo-derecho, crea las imágenes y presenta la información en los frontbuffer.

Permite el manejo de información de datos asociados a las coordenadas xyz de la imagen del objeto en tiempo real, incrementa la RAM para el backbuffer izquierdo-derecho, con la posibildad de discriminar y tomar el backbuffer correspondiente, cuya información es enviada al frontbuffer o dispositivo de presentación adicional, e independiente que comparte una memoria en forma inmersa.

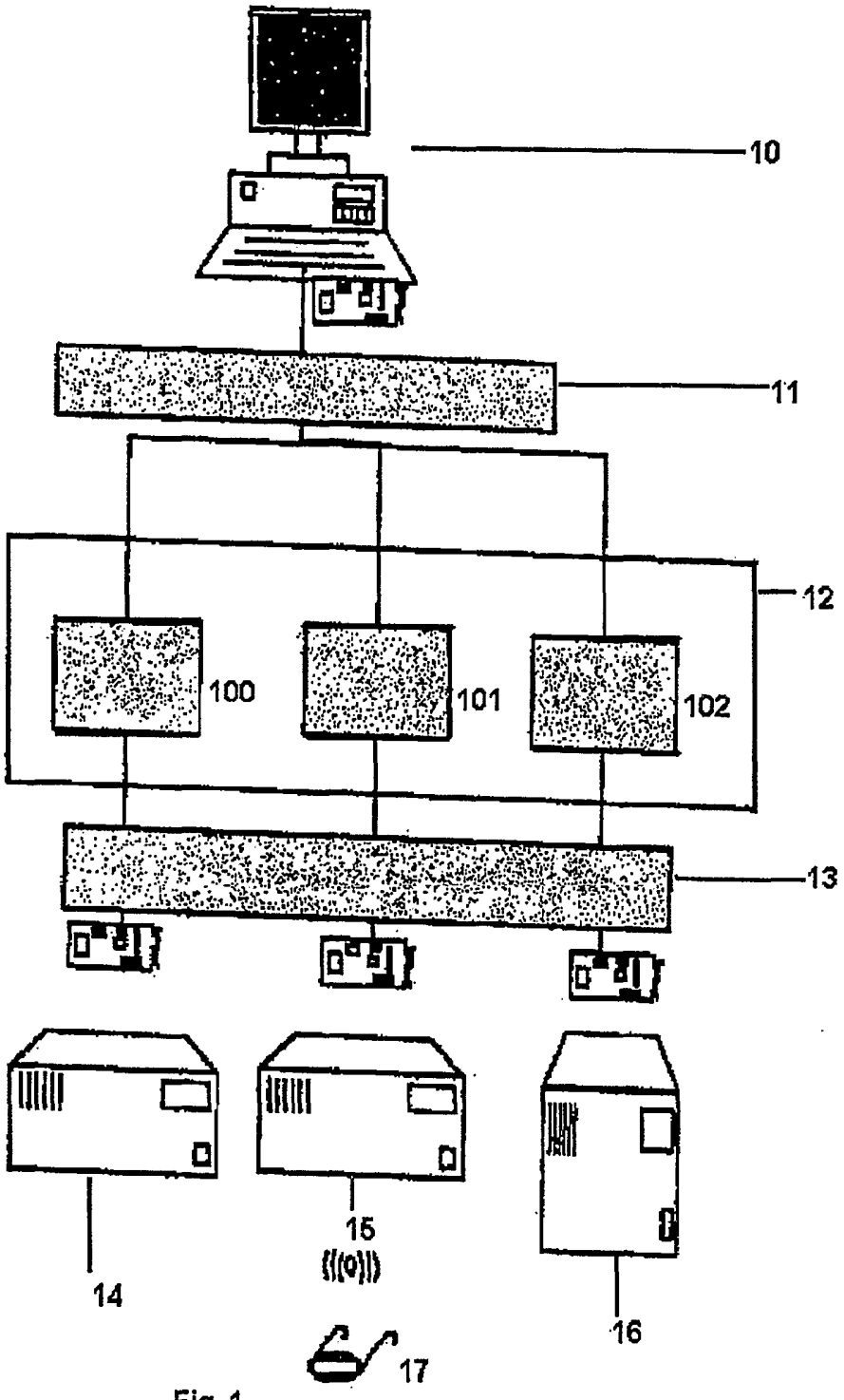

Fig. 1

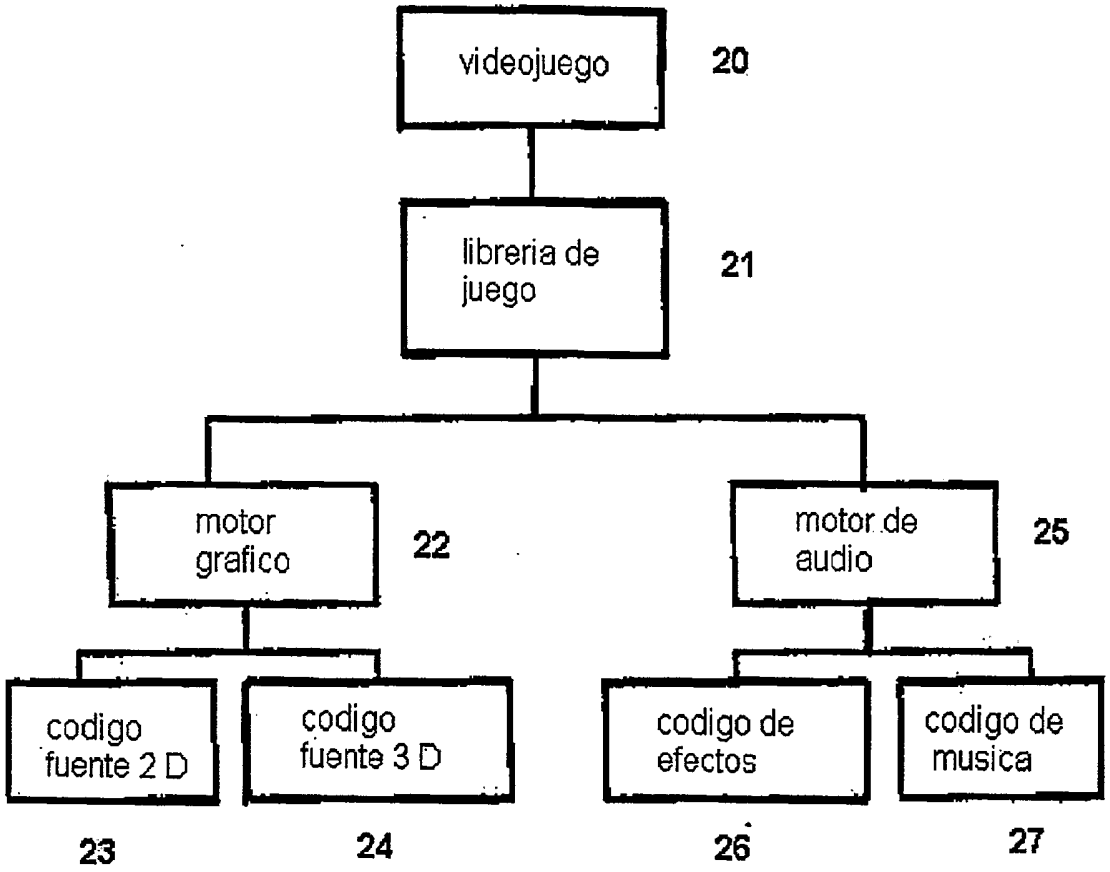
videojuego
20
libreria de juego
21
motor grafico
22
motor de audio
25
codigo fuente 2 D
codigo fuente 3 D
codigo de efectos
codigo de musica
23
24
26
27

Fig. 2

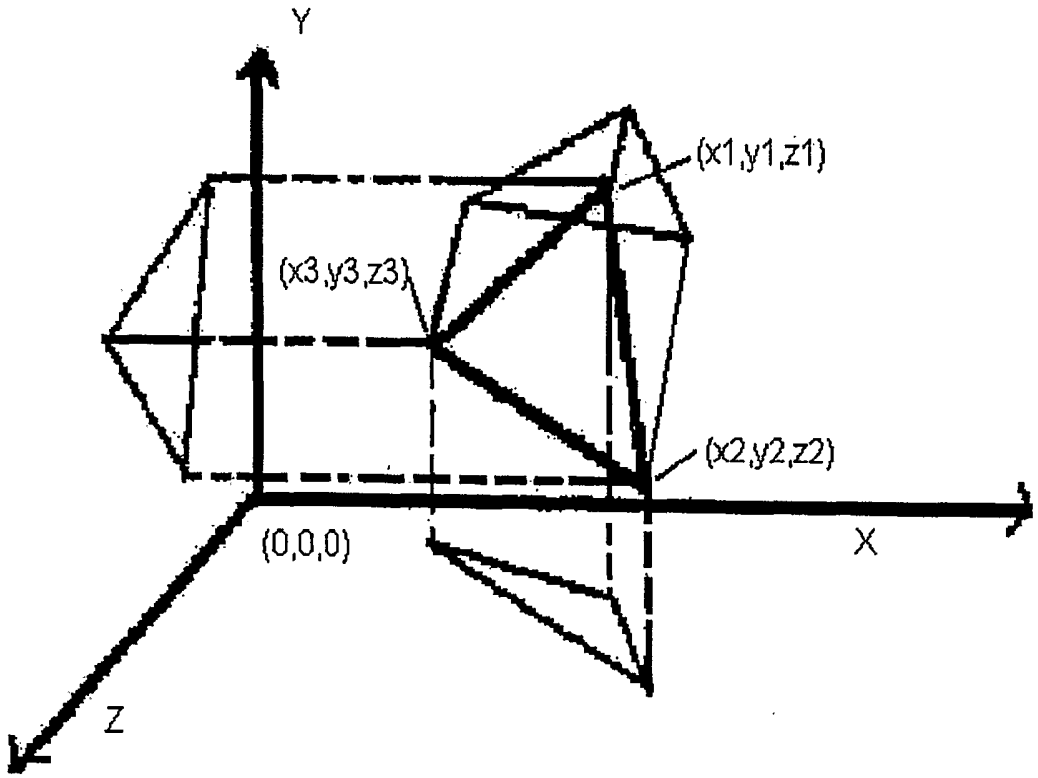

Fig. 3

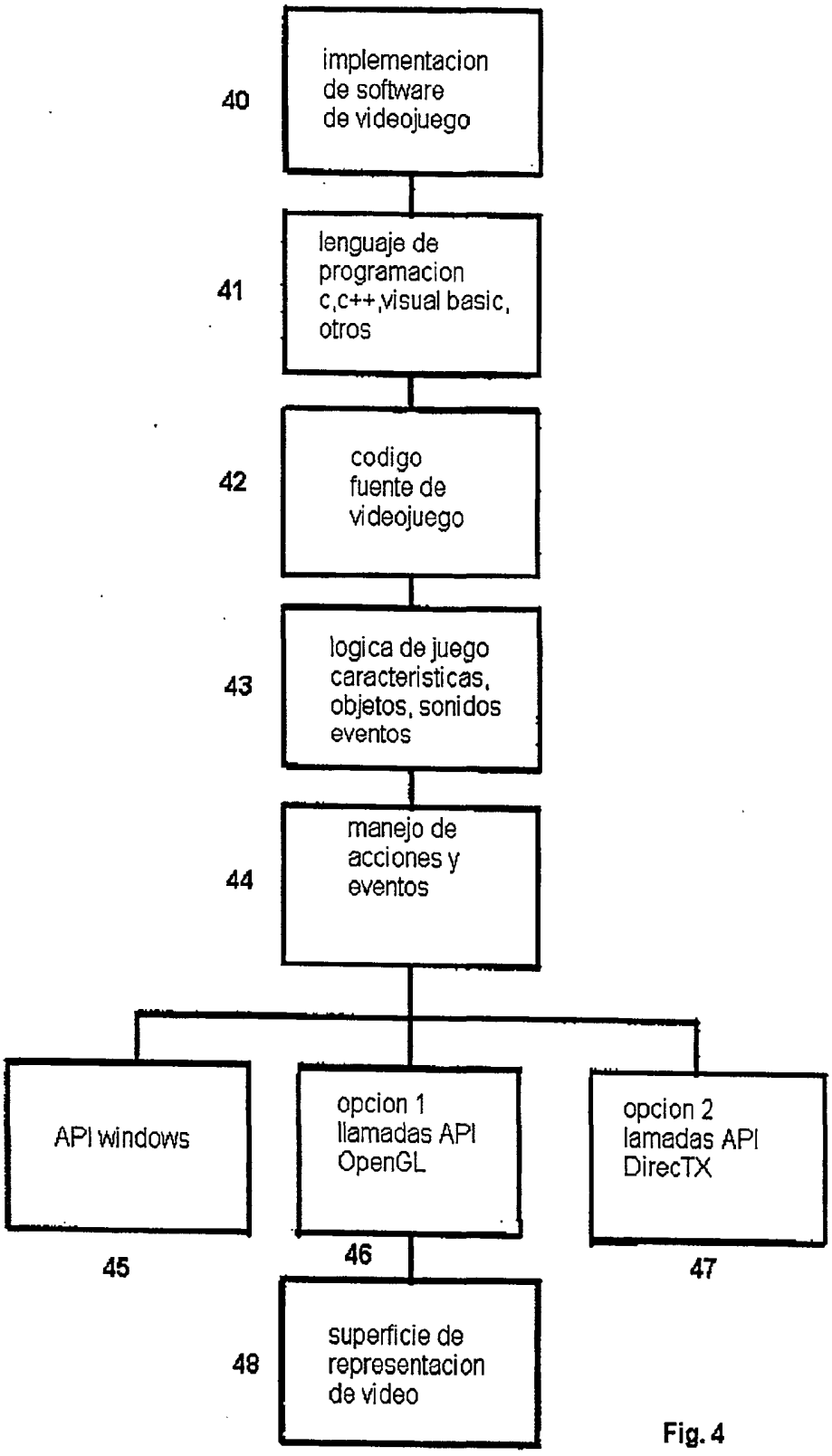
40
implementacion de software de videojuego
41
lenguaje de programacion c,c++,visual basic, otros
42
codigo fuente de videojuego
43
logica de juego caracteristicas, objetos, sonidos eventos
44
manejo de acciones y eventos
API windows
45
opcion 1 llamadas API OpenGL
46
opcion 2 lamadas API DirecTX
47
48
superficie de representacion de video

Fig. 4

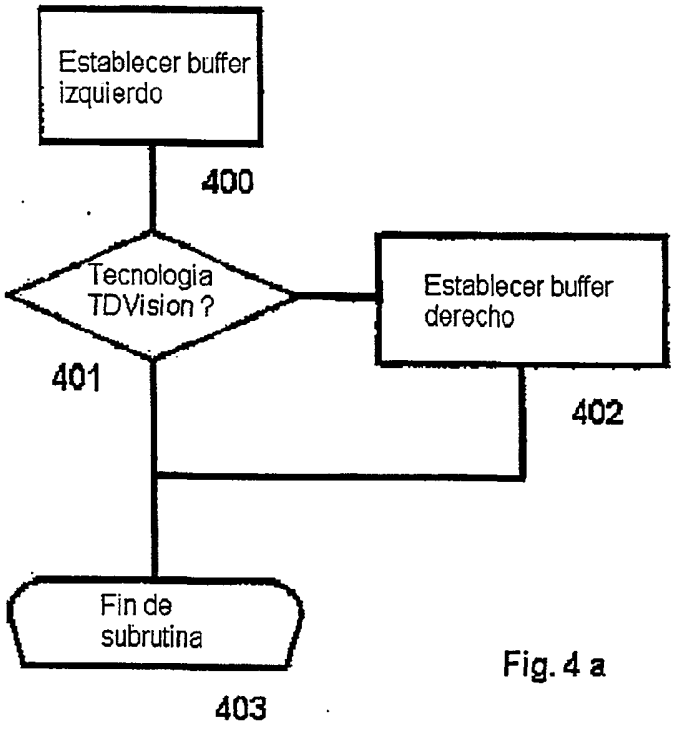
Establecer buffer izquierdo
400
Tecnologia TDVision ?
401
Establecer buffer derecho
402
Fin de subrutina
403

Fig. 4 a

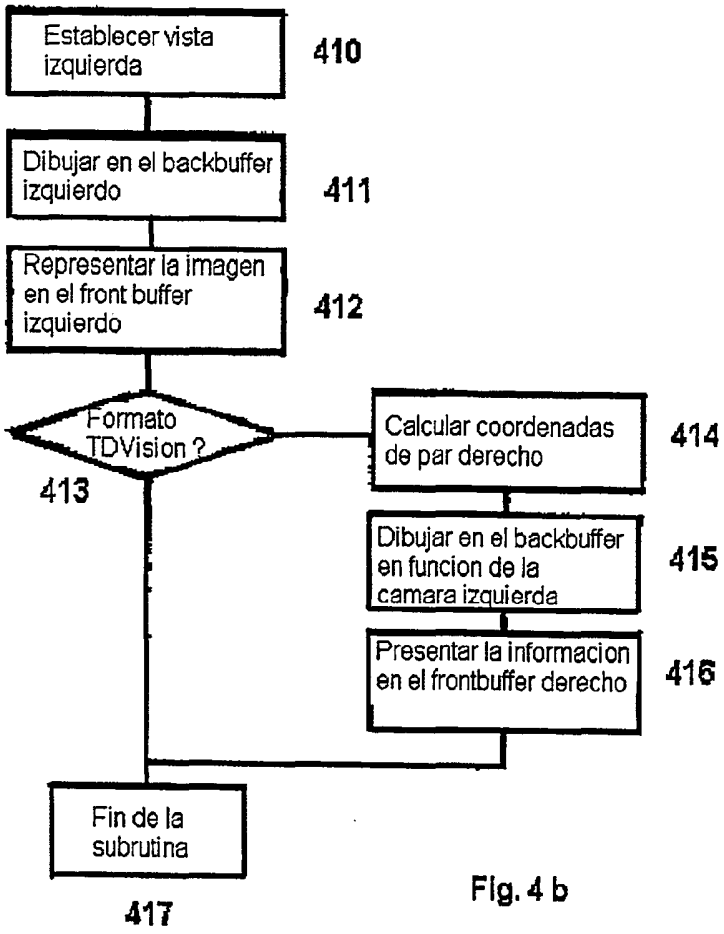
Establecer vista izquierda
410
Dibujar en el backbuffer izquierdo
411
Representar la imagen en el front buffer izquierdo
412
Formato TDVision ?
413
Calcular coordenadas de par derecho
414
Dibujar en el backbuffer en funcion de la camara izquierda
415
Presentar la informacion en el frontbuffer derecho
416
Fin de la subrutina
417

Fig. 4 b

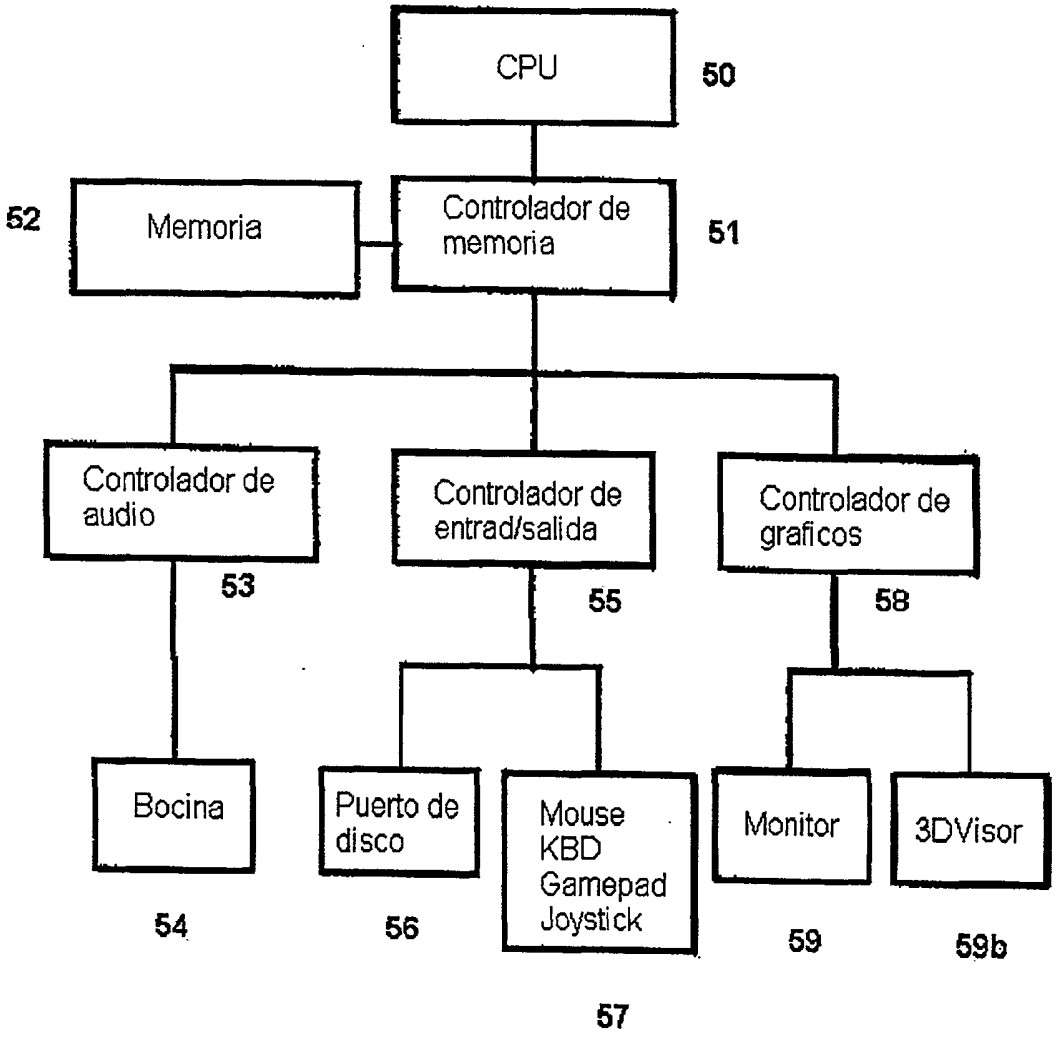
CPU
50
52
Memoria
Controlador de memoria
51
Controlador de audio
53
Controlador de entrad/salida
55
Controlador de graficos
58
Bocina
54
Puerto de disco
56
Mouse
KBD
Gamepad
Joystick
57
Monitor
59
3DVisor
59b

Fig. 5a

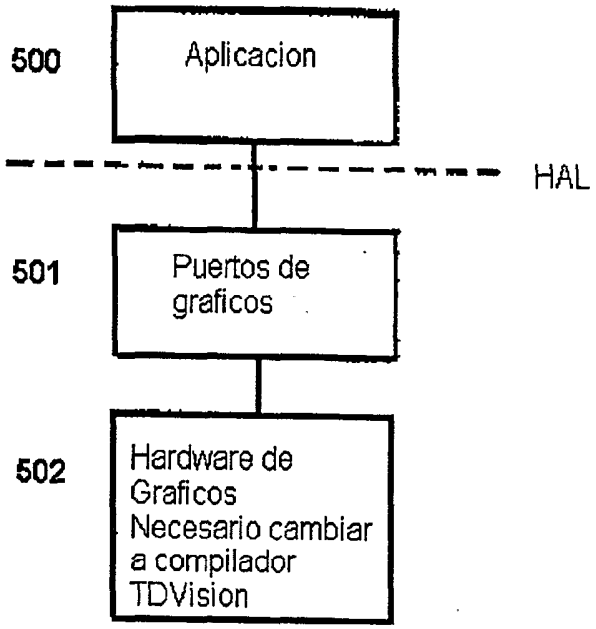
500
Aplicacion
HAL
501
Puertos de graficos
502
Hardware de Graficos Necesario cambiar a compilador TDVision

Fig. 5b

Algoritmo TDVision con DirecTX 3D

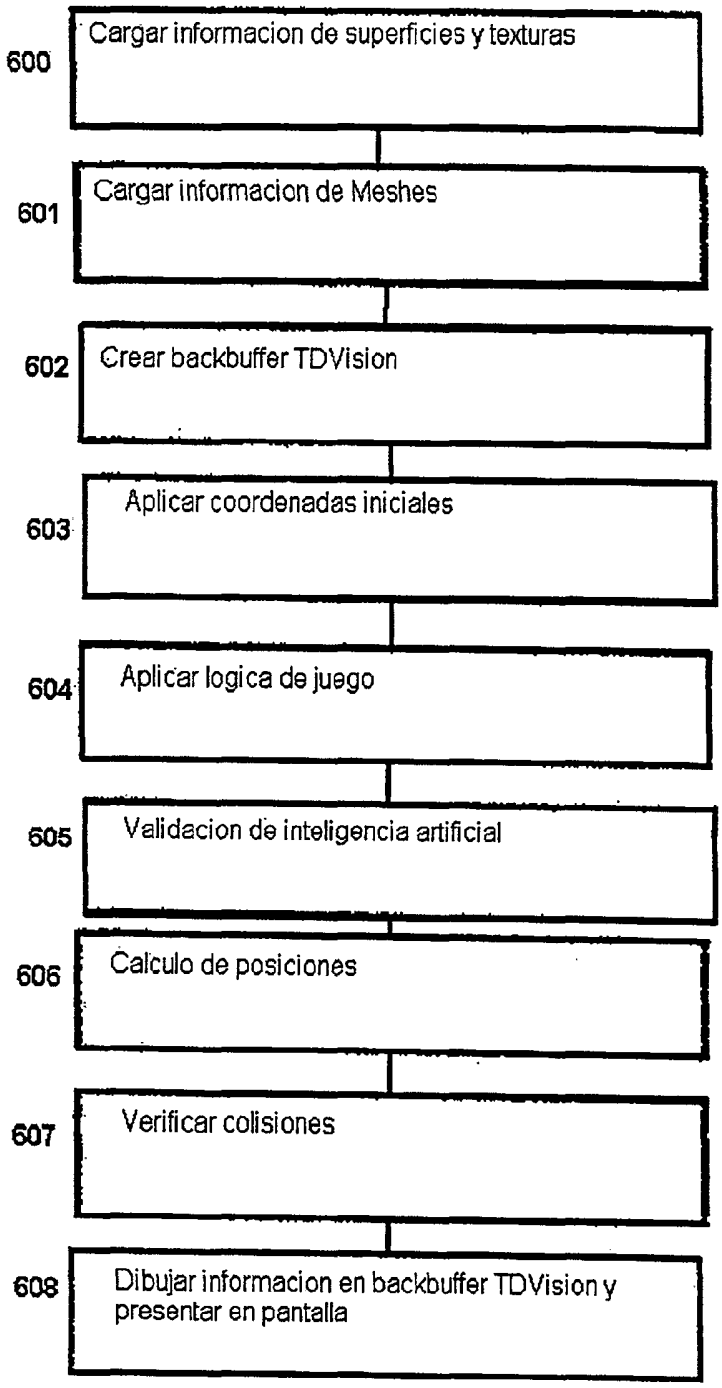

Fig. 6

Algoritmo OpenGL

Flujo de representacion

70

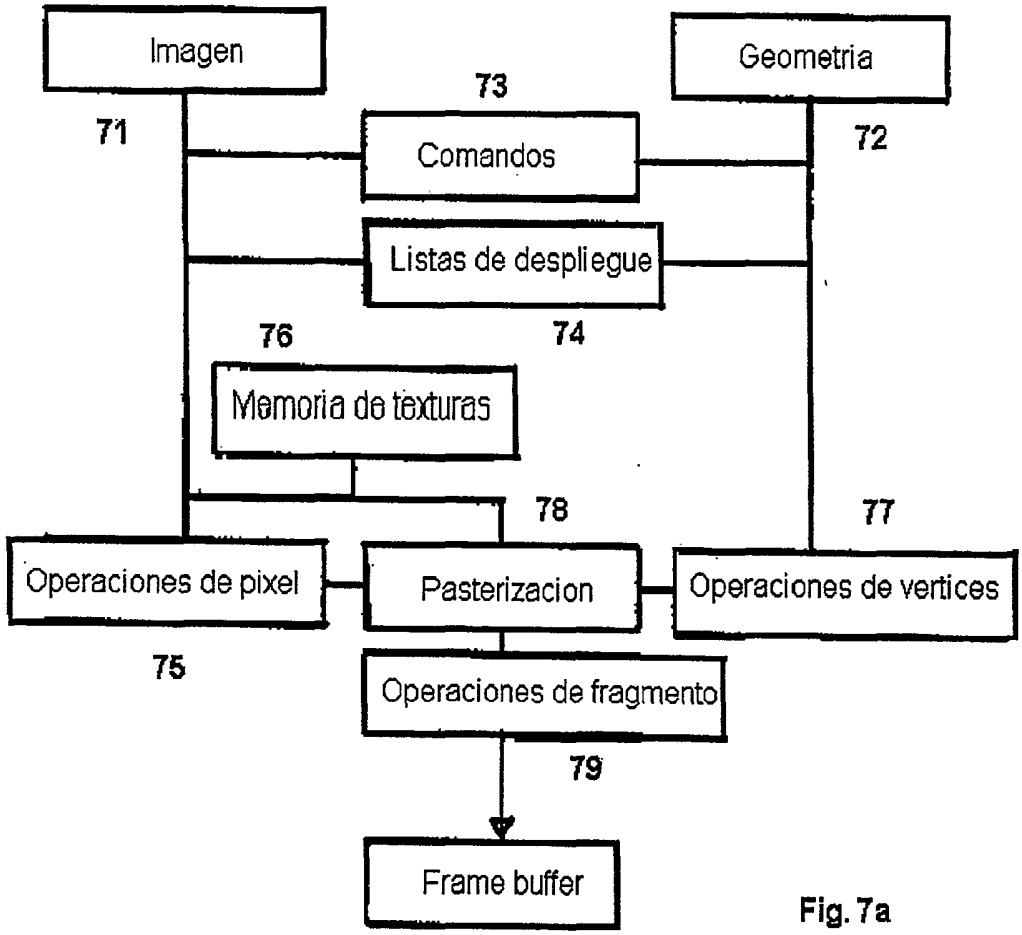

Fig. 7a

70F

Algoritmo

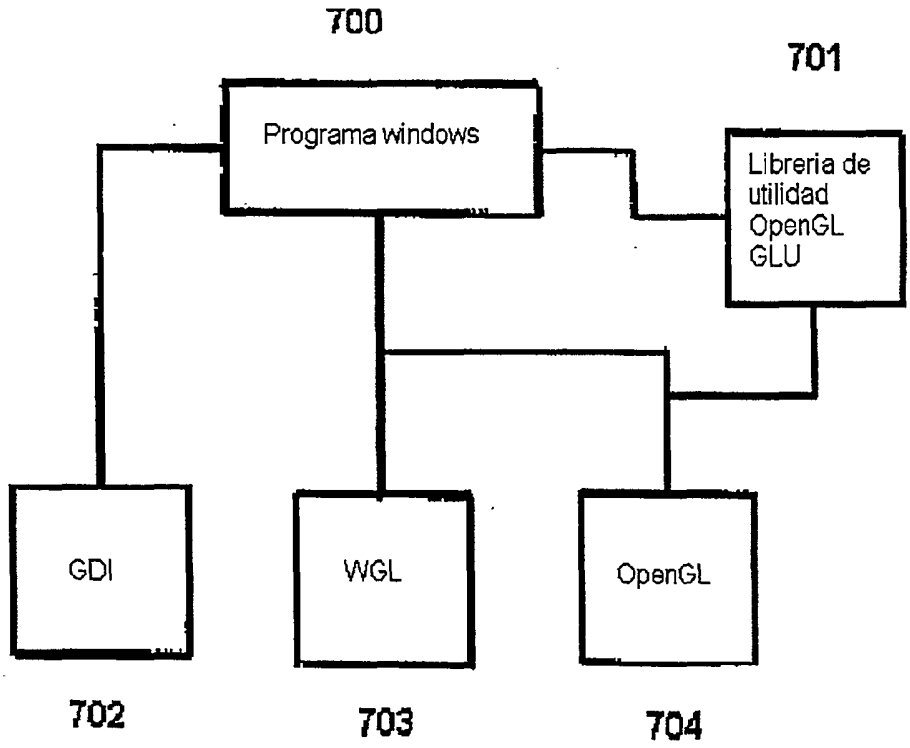

Fig. 7b

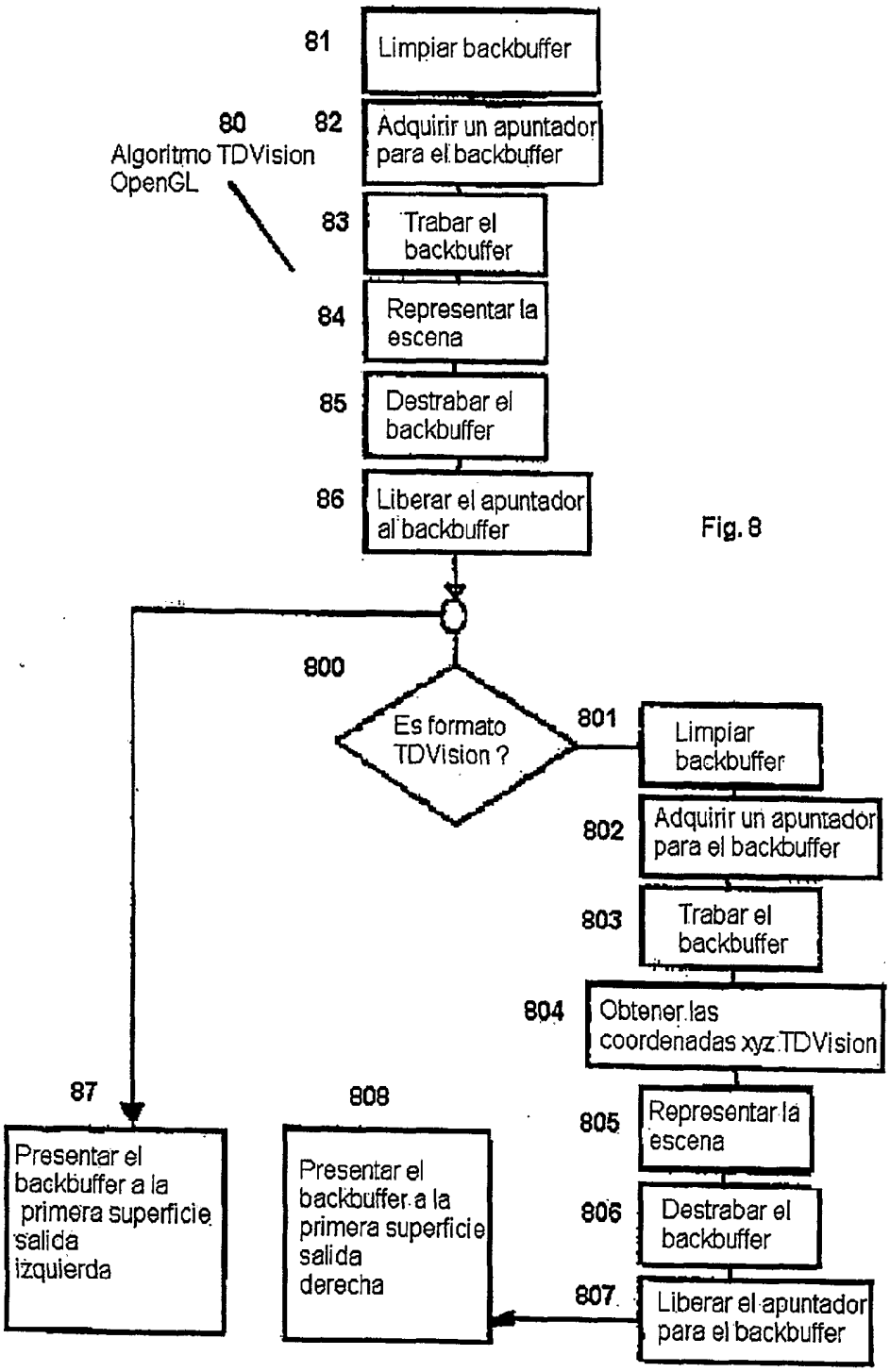
81
Limpiar backbuffer
80
Algoritmo TDVision OpenGL
82
Adquirir un apuntador para el backbuffer
83
Trabar el backbuffer
84
Representar la escena
85
Destrabar el backbuffer
86
Liberar el apuntador al backbuffer
800
Es formato TDVision ?
801
Limpiar backbuffer
802
Adquirir un apuntador para el backbuffer
803
Trabar el backbuffer
804
Obtener las coordenadas xyz TDVision
805
Representar la escena
806
Destrabar el backbuffer
807
Liberar el apuntador para el backbuffer
87
Presentar el backbuffer a la primera superficie salida izquierda
808
Presentar el backbuffer a la primera superficie salida derecha

Fig. 8

Cambios para que compile la Tarjeta de video TDVision 90

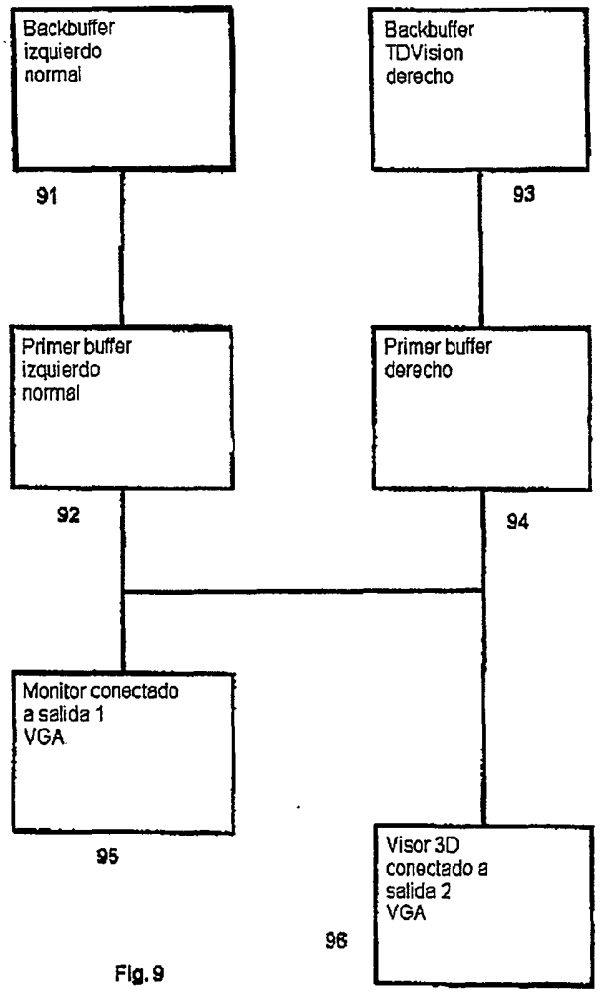

Fig. 9