

Lentes mágicas: una herramienta de apoyo para la visualización en 3D

Carlos E. Ortiz
(ceo@puj.edu.co)

José Ricardo Popó
(Ricardo@puj.edu.co)

Francisco J. Herrera Botero
(herbo@puj.edu.co)

Andrés A. Navarro Newball
(anavarro@puj.edu.co)
Grupo DESTINO - Pontificia Universidad Javeriana – Cali, Colombia

Fecha de recepción: 22-1-2007

Fecha de selección: 8-10-2007

Fecha de aceptación: 30-8-2007:

ABSTRACT

Magic lenses are tools which allow changing a work area according to specific selections and properties. This paper describes the work done aimed to extend the concept of magic lenses to allow seeing inside 3D models which are uploaded from an example application. In order to implement the lenses a ray tracing algorithm for image synthesis was used.

KEYWORDS

Ray tracing, magic lenses, algorithm, 3D model, computer graphics, human computer interaction.

RESUMEN

Las lentes mágicas son herramientas que permiten modificar parte de un

área de trabajo de acuerdo con una selección específica y a la propiedad que dicha herramienta posee. Este artículo describe el trabajo realizado para extender el concepto de lente mágica, para permitir ver dentro de modelos en tres dimensiones (3D) que se cargan en una aplicación desarrollada como ejemplo. Para la implementación de las lentes se utilizó un algoritmo para la síntesis de imágenes denominado trazador de rayos.

PALABRAS CLAVE

Trazador de rayos, lente mágica, algoritmo, modelos 3D, computación gráfica, interacción humano computadora.

Clasificación Colciencias: Tipo 1

I. INTRODUCCIÓN

El estudio de la interacción humano computadora promueve el desarrollo de interfaces gráficas amigables y fáciles de utilizar; además, involucra la creación de técnicas y herramientas innovadoras que permiten un mejor uso del espacio y la información dentro de una interfaz gráfica de usuario (GUI). Una de estas herramientas innovadoras son las lentes mágicas. La aplicación desarrollada por los autores de este artículo extiende el concepto de lentes mágicas presentado por Luis Garreta¹ al enfocar la aplicación de las mismas a la visualización de imágenes constituidas por mallas poligonales y a sobre como deben manipularse dichas mallas utilizando la herramienta que se propone en este documento, para mostrar su contenido o capas interiores.

La técnica computacional de generación de imágenes denominada trazador de rayos es un método que permite obtener escenas realistas de un escenario virtual que puede estar conformado por mallas. Dicha técnica fue propuesta en los años setenta, pero dado su costo computacional solo fue hasta finales de los años noventa que comenzó a considerarse su uso para la generación de imágenes.² Esta técnica está basada en un algoritmo de iluminación global que determina la visibilidad de las superficies trazando rayos de luz imaginarios desde el punto de visión (cámara u observador) hasta los objetos que se encuentran dentro de la escena. Después, éste aplica un modelo de iluminación que determina el color de la superficie que se interceptó de acuerdo con las luces de la escena y a las propiedades de dicha superficie.

Las mallas poligonales son estructuras que se usan para representar objetos del mundo real como un conjunto de aristas, vértices y polígonos conectados entre sí, de manera que cada arista comparte a lo sumo dos polígonos. Una arista es la conexión entre dos vértices y un polígono es una secuencia cerrada de aristas. Esta estructura de datos posibilita representar objetos de cualquier forma y es una de las formas más comunes de representación de modelos en computación gráfica.³

Este artículo presenta primero una serie de antecedentes relacionados con lentes mágicas. Sigue una explicación de las mallas 3D y de cómo se aplican al trazador de rayos; continúa explicando la forma en que se implementó una lente semi-volumétrica y su arquitectura y termina describiendo una serie de resultados y conclusiones.

2. ANTECEDENTES

Las lentes mágicas pueden ser utilizadas para mostrar la información de un área específica, filtrar cierta información, manipular el área seleccionada, o para mostrar diferentes partes de un objeto gráfico. Para dar mayor claridad, a continuación se citarán cuatro ejemplos de lentes mágicas.

En referencia 4 Eric Bier presenta un tipo de herramienta denominada *lentes mágicas*, las cuales funcionan sobre una interfaz transparente. Las herramientas transparentes son elementos de interfaz ubicados sobre un área de aplicación que permite que se vea la imagen del objeto que se encuentra por debajo de las mismas, sin sufrir cambio alguno. Este tipo de

lentes aparece de manera invisible en los objetos en que se superponen, sin ocupar un espacio extra en la pantalla. La Figura 1 ilustra este concepto.

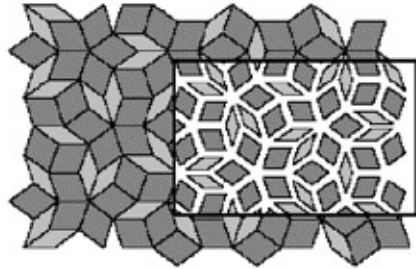


Figura 1. Lente Mágica para minimización de objetos⁴

En referencia 5 John Viega propone un trabajo en el cual extienden el concepto de lentes mágicas a ambientes tridimensionales: *3D Magic Lenses*. Aquí se proponen dos tipos de lentes que permiten la visualización de las capas internas de los objetos sobre los que se superponen: lentes planas y lentes volumétricas. En las lentes planas el usuario puede visualizar cambios en una región limitada del objeto superpuesto, mientras mantiene su contexto. Las lentes volumétricas aplican sus acciones directamente en la estructura interna del objeto superpuesto y dan una sensación de volumen. La Figura 2 muestra la aplicación de un lente volumétrico sobre un mundo 3D.

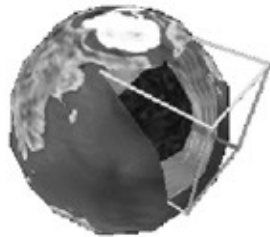


Figura 2. Lente volumétrico que revela la estructura interna de la tierra.⁵

También, en edición de gráficos se encuentran lentes que pueden ser usadas para modificar las propiedades visuales de un objeto seleccionado, por ejemplo: el color, las sombras y los patrones de las líneas. Las lentes mostradas en la Figura 3 son una lente acromática y una lente que muestra las sombras de los elementos, las lentes se encuentran superpuestas una sobre la otra.

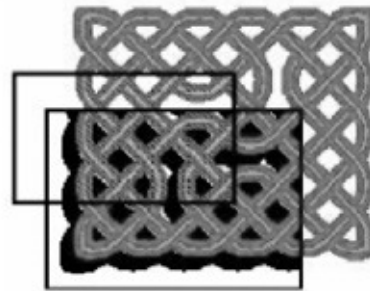


Figura 3. Dos lentes planas superpuestas para la edición de imágenes.⁶

Por otra parte, Luis Garreta¹ propone una arquitectura de software para el diseño de aplicaciones que integran lentes mágicas. Esta extiende la tradicional arquitectura Modelo Vista Control (MVC) a la que él denomina Modelo Vista Control Lente (MVCL) y presenta las lentes mágicas como una importante herramienta de interacción en dos dimensiones.

3. ALGORITMO DE TRAZADOR DE RAYOS⁷

La técnica de trazado de rayos parte de una abstracción del mundo real al plasmar los conceptos de luz, sombras, punto de vista y características de los cuerpos al interactuar con la luz (refracción y reflexión) de manera que puedan ser representados computacionalmente. Además,

usa como herramientas conceptos del álgebra lineal como vectores y soluciones matemáticas para la determinación de intersecciones entre estos y las superficies. Los rayos emitidos por la luz natural son considerados dentro de la computación gráfica como una línea infinita que está delimitada por uno de sus lados. Un rayo posee un origen (un punto en el espacio) y una dirección (Un vector 3D). El punto de vista puede compararse con una cámara u observador del mundo real que sirve para captar una escena; esta cámara puede modelarse como un objeto que posee una posición dentro de la escena y una orientación que indica hacia dónde apunta.

El funcionamiento de esta técnica puede describirse como rayos que describen las trayectorias que siguen los fotones de luz incidiendo en la cámara que capta la imagen. Así, los rayos son emitidos desde las fuentes de luz que conforman la escena y generan trayectorias complejas, ya que deben seguirse todos los rayos que inciden sobre los objetos hasta que alguno de estos intercepta el punto de vista. En contraste, el “trazador de rayos hacia atrás” tiene en cuenta sólo los rayos que contribuyen con la imagen al analizar sólo aquellos que llegan a la cámara. Es decir, se realiza un proceso inverso de trazado, el cual sí es posible implementar, ya que la cantidad de rayos a analizar se reduce a solo aquellos que contribuyen con la generación de la escena. La Figura 4 ilustra este concepto (ver Anexo 1. Cálculos en trazado de rayos).

La Figura 5 muestra el pseudocódigo del algoritmo implementado basado en el trazador de rayos hacia atrás. Aquí se usa el modelo de ilumina-

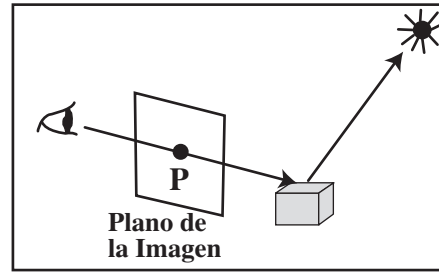


Figura 4. Concepto básico del trazado de rayos hacia atrás.

ción de Phong y se implementan los fenómenos denominados reflexión y refracción.

4. ARQUITECTURA DEL SISTEMA

Para el desarrollo de este trabajo la estructura de datos que se empleó para la representación y manipulación de las mallas poligonales fue la de apuntadores a una lista de vértices.⁸ Dicha estructura sólo almacena una vez cada vértice de la malla poligonal en el arreglo de vértices. Un polígono está dentro de una lista de índices a la lista de vértices, los modelos usados dentro de este proyecto utilizan polígonos representados por tres vértices. Esta representación permite la aplicación del algoritmo trazador de rayos a través de la implementación de rutinas que determinan cuándo un rayo intercepta un polígono. Cuando se presenta dicha intersección la técnica procede a aplicar un modelo de iluminación que determina el color de la superficie interceptada basada en aproximaciones empíricas de los componentes de iluminación difusa, especular y ambiental que se presentan en la realidad. A continuación se explica brevemente cada uno de estos componentes:³ La iluminación difusa puede definirse como la iluminación que reciben los objetos según su

```

Dibujar
  Para cada línea de barrido
    Para cada píxel en la línea de barrido
      Llamar a TrazarRayo

{PRE: Arreglo de objetos que conforman la escena y la instancia de un objeto tipo
rayo}
{POST: Retorna un color}
TrazarRayo
  Para cada objeto de la escena
    Verificar si se intercepta con el rayo
      Calcular la distancia entre el objeto y la cámara
      Si es la menor distancia guardar objeto
      Si hay intersección y es menor a la profundidad
        Llamar a Sombra
      Si no
        Retomar el color del fondo

Sombra
  Para cada fuente de luz
    Para cada objeto de la escena
      Crear un rayo entre la fuente de luz y la intersección
      Si existe la intersección retornar el valor del fondo
      Si el color es reflejante o de refracción, calcular la dirección de los rayos
      y llamar a TrazarRayo
    Retornar el color del objeto

```

Figura 5. Algoritmo de trazador de rayos

posición con respecto a las luces que iluminan la escena. La iluminación especular depende del punto de vista del usuario, ya que es un reflejo directo de la luz que rebota en los objetos y que incide sobre éstos. Por último, la iluminación ambiente es una cantidad de energía constante para todos los elementos de la escena y se produce al rebotar la luz en los átomos de la atmósfera y difundirse uniformemente.

En este trabajo el trazador de rayos se constituye en el elemento básico de una nueva arquitectura que se propone para la implementación de lentes mágicas en 3D. Además, ofrece

una alternativa a implementaciones basadas en algoritmos de recorte que determinan un volumen de aplicación de un lente volumétrico basado en seis planos.⁵ Se propone esta nueva arquitectura para tratar de superar los inconvenientes de desempeño relacionados con los algoritmos de recorte. Adicionalmente, se evita el tener que proponer una técnica para la generación de la imagen y otra para el desarrollo de la lente. Aquí se define una ventana plana de visualización que es recorrida y se utiliza en el trazador de rayos para generar el volumen que es visualizado a través de la lente.

La Figura 6 muestra el diseño de la arquitectura. Aquí la clase Escena permite una visualización de la escena utilizando una instancia de RayTracer. RayTracer contiene la implementación del algoritmo de trazador de rayos (Figura 5). La clase cámara representa el punto de vista del usuario. La clase rayo abstrae el concepto matemático de semirrecta y define un rayo como un vector que posee un origen y una dirección. La jerarquía de clases Primitivas contiene los métodos para calcular intersecciones con los objetos del tipo Malla, Plano y Esfera (sección A.1.2). La clase Lente permite crear un rectángulo superpuesto sobre una instancia de un objeto del tipo Pantalla; está junto con la clase RayTracer,

contiene los métodos necesarios para implementar una lente por capas. La clase Luz permite calcular los colores que son desplegados en la escena visualizada a través de la lente mágica (sección A.1.1). La clase Caja Envolvente implementa un cubo que envuelve un objeto más complejo y hace que el proceso de detección de intersecciones sea más rápido (sección A.1.2.3). Finalmente, las clases Main, Interfaz y Barra de Estado conforman la interfaz gráfica y la aplicación principal. Cabe anotar que dadas las particularidades de un trazador de rayos, se prefirió proponer una arquitectura totalmente nueva para la creación de las lentes 3D y se desistió de extender la arquitectura MVCL¹

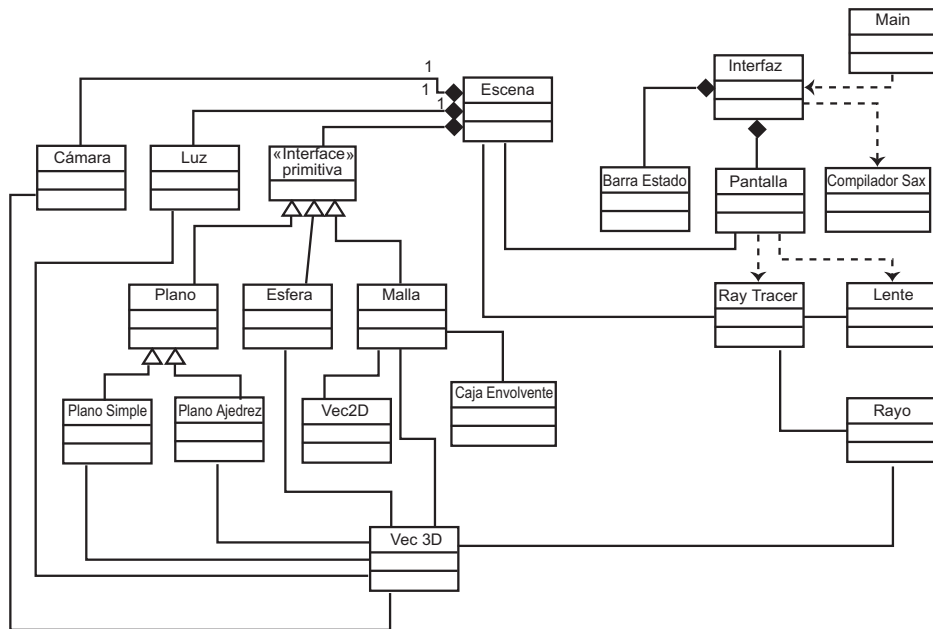


Figura 6. Diagrama de clases para la arquitectura de la lente mágica.

Adicionalmente las mallas debieron ser adaptadas para poder introducir el concepto de capas. Así, cada malla dentro del entorno está definida por un número de capas que entra a relacionarse directamente con la lente mágica. Para la descripción de mallas poligonales como modelos por capas se definieron varias mallas que se encuentran una dentro de otra para así simular el efecto de tener un modelo compuesto por varias capas. La técnica desarrollada para la implementación de la lente permitió simular el efecto de ver “dentro de” los modelos 3D.

5. RESULTADOS⁷

Se obtuvo un nuevo tipo de lente híbrida entre lentes planas y volumétricas. Estas lentes, que se denominaron semi-volumétricas, poseen la forma de una lente plana que está relacionada con la ventana de visualización del trazador de rayos, pero tal como un trazador de rayos, permite la visualización de perspectiva y volumen dentro de una escena.

La lente desarrollada está basada en una variante del algoritmo explicado en la sección 2. El trazador de rayos en su estado original permite generar la imagen más cercana a la cámara o punto de vista. Al mismo tiempo, la variante implementada permite generar una imagen de acuerdo con una distancia específica. Esto con el fin de implementar un efecto que permite ver dentro de los modelos 3D que se encuentren en la escena y así permitir ver varias de las capas de los modelos que componen la escena. La Figura 7 ilustra este concepto. Aquí se ve que un rayo es generado desde el observador y cruza por el área

seleccionada que se encuentra a una distancia d ; allí, siguiendo la dirección del rayo, es posible que alguno de los objetos de la capa seleccionada sea interceptado.

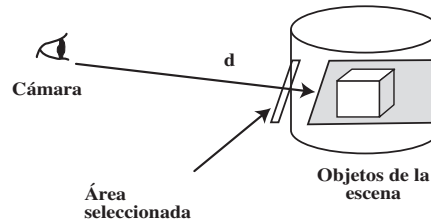


Figura 7. Generar una parte de la escena que se encuentra a una distancia “ d ” de la cámara y que pertenece a una determinada capa.

En la Figura 8 se muestra el pseudocódigo del algoritmo que permite la aplicación de una lente sobre un modelo 3D para así ver sus capas internas de acuerdo con una distancia.

A continuación (Figuras 9, 10, 11), se muestran los resultados gráficos de aplicar la lente semi-volumétrica, representada por el área rectangular de visualización, sobre tres modelos diferentes, desarrollados todos a 500 x 500 píxeles utilizando un procesador Pentium de 1.8 MHz con 256 MB de RAM y una tarjeta de video estándar OnBoard Intel 828456. Aquí se aplicó la técnica de “trazado de rayos hacia atrás” empleando la modificación descrita en el algoritmo de la Figura 8.

El primer modelo representa un cubo.⁸ Dentro de este cubo se encuentra otro cubo que representa una capa interna (Figura 9). Esta escena tomó 6,95 centésimas de segundo en ser generada.

```

(PRE: Arreglo de objetos que conforman la escena y una instancia de un objeto tipo
rayo, además de una distancia específica)
(POST: Retorna un color)
TrazarRayo
  Para cada objeto de la escena
    Verificar si se intersecta con el rayo
    Calcular la distancia entre el objeto y la cámara
    Si es la menor distancia con respecto a una distancia d
      Guardar esta distancia
      Guardar el índice de este objeto
    Si hay intersección y es menor a la profundidad
      Llamar a Sombra
  Si no
    Retornar el color del fondo

```

Figura 8. Algoritmo para la lente semi-volumétrica.

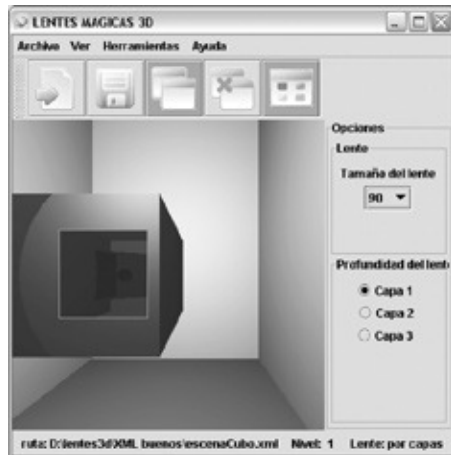


Figura 9. Aplicación del lente sobre un modelo que representa un cubo.

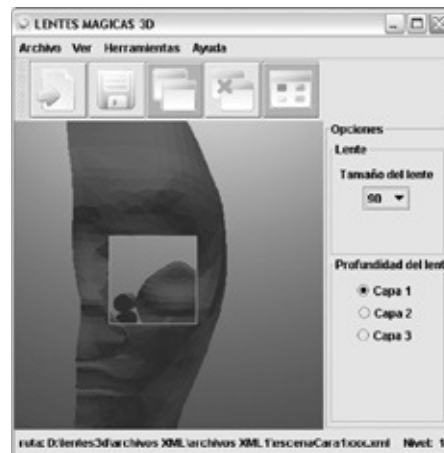


Figura 10. Aplicación del lente sobre un modelo más complejo.

El siguiente modelo es tomado del simulador WESST-OT.^{9,10} Este modelo está conformado por una cara que posee uno de los senos paranasales y las turbinas media e inferior. La prueba realizada con la lente por capas permitió ver la capa interior de esta cara donde se encuentran los senos y la turbina media e inferior

(Figura 10). Esta escena tomó 4,46 segundos en ser generada.

El último modelo representa una mano compuesta por su tejido epitelial, al aplicar el lente se pueden observar los componentes de la capa interna correspondientes al esqueleto⁸ (Figura 11). Esta escena tomó 3,4 segundos en ser generada.

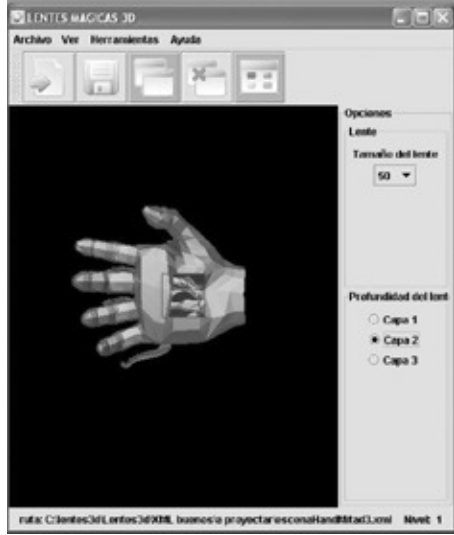


Figura 11. Visualización de la capa interior de una mano.

La Figura 12 muestra el número de polígonos utilizados en cada uno de los escenarios descritos.

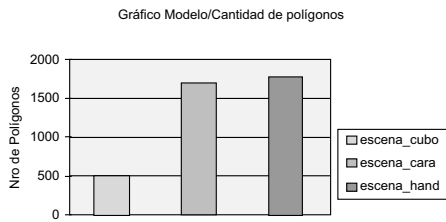


Figura 12. Número de polígonos por escena.

6. CONCLUSIONES

Se ha logrado una fusión entre una técnica para la generación de imágenes conocida como el trazador de rayos y el concepto de lente mágica, lo que ha permitido la creación de una lente que posee efectos semi-volumétricos. Este híbrido entre lentes planas y volumétricas nació debido a una de las características especiales que brinda la técnica, ya que permite

dibujar secciones de los objetos que se encuentran en una escena a partir de su distancia con respecto al punto de vista (cámara). Al implementar esta condición los resultados obtenidos sobre una porción de la escena mostraban cómo se dibujaba un elemento interno dependiendo de una distancia conservando una perspectiva volumétrica.

No se tuvo en cuenta el patrón propuesto por la arquitectura MVCL. La razón principal fue que este proyecto estuvo enfocado en la implementación de una herramienta basada en el algoritmo de trazador de rayos. Aquí las mejoras que se realicen sobre el trazador de rayos pueden traer beneficios directos sobre la lente semi-volumétrica.

De acuerdo con las pruebas, el costo de tiempo de la técnica implementada depende de la cantidad de polígonos de los objetos que componen la escena. En contraste, se cree que la técnica disminuye el tiempo de ejecución y las necesidades de hardware de los lentes volumétricos que se implementan con algoritmos de recorte. Sin embargo, aunque solo existen evidencias antiguas sobre el trabajo de lentes volumétricos y los autores no tuvieron acceso directo a su implementación, se sugiere comparar las dos técnicas en igualdad de condiciones. De hecho se sugiere la utilización de computadoras de última tecnología, pues estos pueden realizar un mayor procesamiento.

Este artículo presenta por primera vez la implementación de una lente mágica sobre una arquitectura basada en un trazador de rayos. Así, el desarrollo propuesto se constituye en una innovación para el campo de la

interacción humano computadora y a futuro puede contribuir a la visualización de objetos 3D definidos por capas, como por ejemplo, la anatomía presentada en el simulador WESST-OT. Aunque los lentes presentados aquí sirven únicamente para visualizar elementos, la arquitectura puede ser fácilmente extendida para permitir lentes que permitan realizar operaciones interactivas sobre los objetos que se están visualizando. A futuro se espera promover este desarrollo en contextos internacionales.

BIBLIOGRAFÍA

1. Garreta, L. E. Arquitectura Orientada a Objetos para el Modelamiento de Filtros de Lentes Mágicas como Elementos de Visualización dentro de las Interfaces Gráficas de Usuario Actuales. Trabajo de grado, Pontificia Universidad Javeriana Cali, 2001.
2. Glassner, A. S. An Introduction to Ray Tracing. Academic Press Inc. San Diego, CA, USA, 1993.
3. Watt, A. H. 3D computer graphics. - 3ed., Estados Unidos: Addison-Wesley, 2000.
4. Stone, M. C.; Fishkin, K.; Buxton, W.; Baudel, T.; Bier, E.A. . A Taxonomy of See-Through Tools. En: <http://www.parc.xerox.com/istl/projects/MagicLenses/Taxonomy.html>. 1999.
5. Conway, M. J.; Williams, G.; Pausch, R.; Viega, J. 3d magic lenses. en: <http://www-2.cs.cmu.edu/stage3/publications/96/conferences/uist/lenses/>. 1996.
6. Stone, C.; Pier, M.; Buxton, K.; Derose, W.; Bier, T.; Eric, A. Toolglass and Magic Lenses: The See-Through Interface. En: Computer Graphics Proceedings, Annual Conference Series. 1993.
7. Ortiz, C. E., Popó, J. R., Lentes Mágicas 3D: Una Herramienta para la Visualización 3D. Trabajo de grado, Pontificia Universidad Javeriana, Cali, 2006.
8. Van Dam, A.; Feiner, K.; Hughes, S.; Foley, J.; James, D. Introducción a la Graficación por Computador. Addison Wesley Iberoamericana, segunda edición, 1996.
9. Navarro N., A. A.; Hernández, C. J.; Vélez B., J. A.; Múnera S., L. E.; García, G. B.; Gamboa, C. A.; Reyes, A. J. Virtual surgical tel-esimulations in otolaryngology. Studies in Health Technology and Informatics, 2005; 111:353-355.
10. Cardona, A.; Mazuera, E.; Herrera, F. J.; Navarro N., A. A.; Gamboa, C. A.; Vélez, J. A. Incorporación de un Módulo para la Práctica de la Habilidad de Ubicación Espacial al Simulador de Otorrinolaringología – WESST – OT. Cali: Sistemas & Telemática, 2006;7:43-53.

CURRÍCULOS

Carlos E. Ortiz. Ingeniero de Sistemas y Computación egresado de la Pontificia Universidad Javeriana en Cali. Perteneció al grupo de investigación *Destino* de la Facultad de Ingeniería durante el desarrollo de su trabajo de grado.

José R. Popó. Ingeniero de Sistemas y Computación egresado de la

Pontificia Universidad Javeriana en Cali. Perteneció al grupo de investigación *Destino* de la Facultad de Ingeniería durante el desarrollo de su trabajo de grado.

Francisco J. Herrera Botero.

Ingeniero de Sistemas y Computación egresado de la Pontificia Universidad Javeriana en Cali, tiene una especialización en Redes y Comunicación (CCNA CISCO) de la Pontificia Universidad Javeriana - Cali. Se desempeñó como Asistente de Investigación (Programa Jóvenes Investigadores de COLCIENCIAS) en la Pontificia Universidad Javeriana - Cali. Además es fundador y gerente de la empresa de base tecnológica Soluciones Virtuales. Actualmente hace parte del grupo de investigación *Destino* y es profesor hora cátedra de la Universidad ICESI.

Andrés A. Navarro Newball. Ingeniero de Sistemas y Computación egresado de la Pontificia Universidad Javeriana en Cali, tiene un MSc in Computer Graphics and Virtual Environments de la Universidad de Hull en Inglaterra y una especialización en Redes y Comunicación de la Universidad ICESI en Cali. Además, formó parte del Centro de Telemedicina de Colombia y fue profesor hora cátedra de la Universidad ICESI. Actualmente se desempeña como profesor de la Pontificia Universidad Javeriana - Cali, donde coordina el grupo de investigación *Destino* y realiza estudios de PhD en la Universidad de Otago, Nueva Zelanda.

Anexo I. Cálculos en trazado de rayos.

A.1.1. Cálculos de iluminación⁷

El cálculo para obtener las contribuciones de cada componente de la luz sobre una superficie se puede expresar como muestra la ecuación 1:

$$color = A + I * \frac{[P * (L \cdot N) + K_s * (N \cdot H)^{P_s}]}{D} \quad (1)$$

Donde:

A, se refiere al coeficiente de la luz ambiental.

I, valor de la luz que incide sobre el objeto.

P, componente difuso.

N, vector normal a una superficie.

L, luz emitida por la fuente de luz.

Ks, coeficiente de la luz especular.

Ps, potencial especular.

D, distancia de la luz al objeto.

H, distancia de la luz al objeto.

H, es la suma del vector luz (L) y el vector de visualización (V).

La reflexión de la luz se define como la cantidad de luz que ciertos objetos pueden transmitir a su entorno. Matemáticamente se contempla como un rayo incidente el cual al ser aplicado sobre una superficie genera un rayo reflejado. El ángulo que forman ambos vectores con respecto a un vector normal a la superficie es el mismo. La Figura 13 describe el comportamiento de este fenómeno.

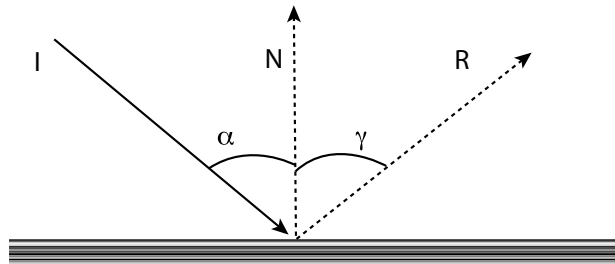


Figura 13. Ángulo respecto a la normal.

Para calcular el vector que representa al rayo reflejado en función del rayo incidente y el vector normal a la superficie se emplea la ecuación 2.

$$R = I - 2 * (N \cdot I) \cdot N \quad (2)$$

Donde:

I, rayo de luz incidente.

N, vector normal a la superficie.

R, rayo reflejado.

A.1.2. Intersección de un rayo con un plano

Con el fin de especializar el trazador de rayos se define un número de objetos que pueden ser puestos en la escena y se indica cómo interceptar rayos con estos objetos y cómo se calcula la normal en el punto de incidencia a estos objetos. Para comenzar, se describe el rayo que va del centro de proyección u observador $[X_0, Y_0, Z_0]$ al centro del píxel en la pantalla $[X_1, Y_1, Z_1]$, de la forma paramétrica mostrada en la ecuación 3:

$$x = x_0 + t \cdot x_d; y = y_0 + t \cdot y_d; z = z_0 + t \cdot z_d \quad (3)$$

A.1.2.1. Intersección rayo - plano

Un plano se define en términos de $[A \ B \ C \ D]$ como muestra la ecuación 4:

$$\text{Plano} = A * x + B * y + C * z + D = 0 \quad (4)$$

$$\text{Donde } A^2 + B^2 + C^2 = 1$$

El vector unitario del plano es definido como:

$$P_{\text{normal}} = P_n = [A \ B \ C]$$

Y la distancia desde el origen del sistema de coordenadas $[0 \ 0 \ 0]$ al plano es determinada por D.

El cálculo de la intersección se realiza sustituyendo la ecuación del rayo (ecuación 4) en la del plano (ecuación 6).

$$A * (X_0 + X_d + t) + B * (Y_0 + Y_d + t) + C * (Z_0 + Z_d + t) + D = 0 \quad (6)$$

Y resolviendo para t (ecuación 7)

$$t = \frac{-(A * X_0 + B * Y_0 + C * Z_0 + D)}{A * X_d + B * Y_d + C * Z_d} \quad (7)$$

En este punto se evalúa t, si su valor es menor que cero o menor que algún valor establecido como límite significa que la intersección ocurre detrás del observador y por lo tanto no es de nuestro interés. Caso contrario el rayo sí se interfecta con el plano. Por último cabe anotar que se requiere que la normal se encuentre apuntan-

do hacia nuestro rayo, por lo cual es necesario hacer un análisis basado en el denominador de la fórmula. Si se presenta un valor no negativo, significa que el sentido de la normal debe invertirse.

A.1.2.2. Triángulo

Un triángulo está definido por una serie de vértices. La intersección de un rayo con un triángulo se calcula de la siguiente manera:

Encontrar la normal al triángulo con el producto cruz de dos de sus lados.

Encontrar la intersección del rayo con el plano definido por el triángulo.

Revisar si los puntos están dentro del triángulo

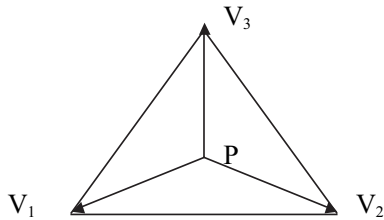


Figura 14. Cálculo de puntos dentro de un triángulo

En la Figura 14 si P es el punto de intersección; primero se obtienen los vectores V1, V2 y V3, que van del punto P a los vértices del triángulo; luego se obtienen las normales (ecuación 8):

$$N_1 = V_1 \times V_2; N_2 = V_2 \times V_3; N_3 = V_3 \times V_1 \quad (8)$$

Si los productos punto de la ecuación 9

$$N_1 \cdot N_2 > 0; N_2 \cdot N_3 > 0; N_3 \cdot N_1 > 0 \quad (9)$$

El punto está dentro del triángulo

A.1.2.3. La malla

Para calcular la intersección con una malla; primero, el rayo se intercepta con el cubo que encierra la malla; luego, si el rayo intercepta este cubo, el rayo se debe interceptar con cada triángulo dentro de la maya. ☀