Relleno y Recorte de Figuras
Juan Sebastián Perez Rios
Introducción a la Computación Gráfica
Héctor Murcia Forero
Universidad Militar Nueva Granada
2024

Relleno y Recorte en Computación Gráfica

En computación gráfica, el relleno se refiere al proceso de colorear o completar el interior de una figura o un polígono en un área cerrada. El objetivo es dar apariencia sólida a las figuras para visualizarlas mejor y hacerlas más atractivas y comprensibles para el usuario.

El recorte es el proceso de limitar la porción de una figura, imagen o escena gráfica que se muestra en la pantalla. Se utiliza para evitar que los objetos fuera de la zona visible sean renderizados, mejorando la eficiencia al no procesar gráficos innecesarios.

Por otro lado, tienen importancia en el Renderizado, tanto el relleno como el recorte son cruciales para optimizar el renderizado y la presentación gráfica. El relleno permite que los objetos tengan apariencia y solidez, facilitando la visualización y percepción de profundidad. El recorte mejora la eficiencia de los programas gráficos al reducir el número de operaciones necesarias para representar solo las partes visibles, lo cual es especialmente importante en sistemas en tiempo real, como los videojuegos y simulaciones científicas.

Algoritmos de Relleno

1. Flood Fill

Es un algoritmo que rellena un área conectada que comparte un mismo color o valor. Se usa principalmente en aplicaciones de pintura, como el 'bucket fill' en programas como MS Paint.

Ventajas:
- Sencillo de implementar para áreas conectadas.
- Efectivo en figuras con bordes bien definidos.

Desventajas:
- Puede ser ineficiente en áreas grandes.
- Requiere una gran cantidad de memoria en sistemas recursivos.

2. Scanline Fill

Rellena una figura analizando cada línea de escaneo horizontalmente y determinando los puntos de entrada y salida dentro de los límites de la figura.

Ventajas:
- Más eficiente para polígonos complejos.
- Maneja mejor la detección y prevención de relleno fuera de los bordes.

Desventajas:
- Requiere un análisis detallado de los bordes y puede ser más complejo de implementar en figuras irregulares.

Algoritmos de recorte

1. Algoritmo de Cohen-Sutherland

Es un método de recorte rápido que clasifica los puntos de las líneas en una ventana de visualización utilizando regiones (arriba, abajo, izquierda, derecha, etc.). A través de este sistema de códigos de región, el algoritmo determina si la línea está dentro, fuera o parcialmente dentro del área visible. Es altamente eficiente para líneas ya que utiliza un sistema de códigos binarios para verificar si una línea debe ser renderizada o no. Es más usado en recortes de segmentos de línea en sistemas de gráficos 2D.

2. Algoritmo de Liang-Barsky

Utiliza cálculos de intersección para determinar el segmento de línea visible dentro de un área definida. A diferencia de Cohen-Sutherland, el algoritmo de Liang-Barsky es matemáticamente más preciso y realiza menos cálculos en general. Es más rápido en ciertos casos, especialmente para recortes de líneas, porque utiliza menos comparaciones y multiplicaciones. Es adecuado para aplicaciones de recorte de líneas y gráficos en tiempo real donde se necesita eficiencia.

Se pueden comparar en que Cohen-Sutherland es más intuitivo y es fácil de implementar en escenas sencillas, mientras que Liang-Barsky es preferido en situaciones donde la eficiencia es crucial y se necesita mayor precisión en el recorte de segmentos de línea.

Aplicaciones Prácticas

1. Software de Diseño Gráfico: En herramientas como Adobe Illustrator y Photoshop, el relleno se usa para colorear áreas de imágenes y vectores, mientras que el recorte es fundamental para limitar las áreas de edición.

2. Videojuegos: Los videojuegos utilizan técnicas de recorte para mostrar solo los objetos visibles en la pantalla, mejorando el rendimiento gráfico y evitando renderizar

elementos que no están en el campo visual del jugador.

3. Visualización Científica: En aplicaciones científicas, el relleno ayuda a dar color y distinción a datos específicos, mientras que el recorte asegura que se muestren solo las áreas relevantes para la interpretación de datos (como secciones de mapeo de proteínas o imágenes médicas).

Estas técnicas permiten un renderizado más eficiente y preciso, ahorrando recursos del sistema y mejorando la experiencia visual para el usuario. En videojuegos, esto se traduce en una mayor fluidez, mientras que en aplicaciones de diseño gráfico, permite representaciones visuales más atractivas y manejables.

Podemos ahondar dentro del tema si tenemos en cuenta artículos como lo son Survey of Algorithms for Real-Time Graphics Clipping en este artículo académico, se puede buscar en bases de datos como IEEE Xplore o JSTOR, analiza la eficiencia de varios algoritmos de recorte en tiempo real, lo cual es útil para decidir cuál implementar en aplicaciones prácticas. Por otro lado, en Efficient Polygon Scan Conversion using Scanline Algorithms muestra una implementación más avanzada del algoritmo de scanline fill, en este artículo presenta mejoras y optimizaciones en el algoritmo para áreas grandes y polígonos complejos.

Se puede ayudar mediante bibliotecas que cuentan con la documentación para su implementación como lo son Three.js que incluye ejemplos de implementación de relleno de superficies y recorte en escenas 3D, con funciones para manejar polígonos, definir materiales de relleno y aplicar cámaras con vistas limitadas. Además Three.js muestra ejemplos de escenas interactivas también muestran cómo aplicar recorte y crear escenas optimizadas para juegos y visualizaciones científicas.

También tenemos la librería OpenGL que tiene recursos que cubren temas avanzados, como el uso de stencil buffers para rellenar regiones complejas y la configuración de recorte en OpenGL. También ofrece guías de implementación práctica y optimización de recursos. Podemos tomar la biblioteca grafica “The OpenGL Red Book”, que es considerado una referencia estándar, proporciona información detallada sobre técnicas de relleno y recorte en gráficos 3D y 2D, con ejemplos de código que pueden ayudar a profundizar en la implementación.

Por otro lado, la documentación de la API Canvas incluye ejemplos de algoritmos de relleno y recorte que pueden aplicarse en la creación de gráficos 2D en la web. A su vez, para el desarrollo de videojuegos, Unity ofrece herramientas y scripts para manejar recortes y crear shaders de relleno. La documentación de Unity también es útil para trabajar en el relleno de superficies en 3D usando materiales y luces.