UNIVERSIDAD
Panamericana

Asignatura y grupo:

Computación Gráfica 4856

Integrantes del equipo:

López Moreno Rodrigo 0249950 7º semestre

Orihuela Araiza Juan Pablo 0256860 7º semestre

“Tarea 2”

Profesor: César Adrián Victoria Ramírez

Fecha de entrega: 09/10/2025

Introducción

Este programa implementa un renderizador 3D utilizando OpenCV y un modelo de cámara con proyección perspectiva. El objetivo es visualizar un conjunto de triángulos en un espacio 3D, aplicando dos tipos de sombreado: Vertex Color y Phong shading. El sistema permite rotar la escena y ajustar la posición de las luces, simula una cámara con vista en perspectiva, y muestra los resultados en tiempo real en una ventana gráfica.

Proceso de ejecución

Paso 1: Inicialización de la escena y parámetros de cámara

El programa comienza con la definición de una serie de vértices y triángulos que conforman un cubo. Además, se define la posición de la cámara y las luces en el espacio. Se genera la matriz de vista a partir de la posición de la cámara y la dirección hacia el centro de la escena. La matriz de proyección se define usando un campo de visión de 60 grados y una relación de aspecto basada en las dimensiones de la ventana.

Paso 2: Construcción de las matrices de transformación

Se construyen varias matrices para transformar los puntos del mundo a la cámara y proyectarlos en el plano 2D de la pantalla:

- **Matriz de Vista (V):** Utiliza la función `lookAt` para crear una matriz que alinee la cámara en el espacio 3D.
- **Matriz de Proyección (P):** Utiliza la función `perspective` para realizar la proyección de perspectiva, transformando las coordenadas del mundo en coordenadas de cámara y luego en coordenadas proyectadas en 2D.

Paso 3: Transformaciones y cálculo de iluminación

Para cada triángulo de la escena, el programa realiza una serie de transformaciones:

- **Transformación de la cámara:** Los vértices de cada triángulo se transforman a coordenadas de cámara utilizando la matriz de vista.
- **Proyección:** Los puntos transformados en coordenadas de cámara se proyectan en el plano 2D utilizando la matriz de proyección.
- **Sombreado:** Dependiendo del modo de sombreado (Phong o Vertex), se calcula la iluminación de cada triángulo. Para el modo Phong, se aplica un modelo de iluminación de Phong, considerando las luces y las normales de cada superficie.

Paso 4: Rasterización y renderizado

El proceso de rasterización se encarga de llenar la imagen final, convirtiendo los vértices y las caras de los triángulos proyectados en píxeles en la pantalla. Durante la rasterización, se realiza una interpolación de los colores y la profundidad (z-buffering) para evitar el sobreposicionamiento de los triángulos. Finalmente, los resultados se muestran en una ventana gráfica utilizando OpenCV.

Paso 5: Interactividad

El programa permite interactuar con la escena en tiempo real. Se puede:

- **Rotar la escena** usando las teclas W, A, S, D, Q, E para controlar los ángulos de rotación sobre los ejes X, Y, Z.
- **Mover la luz** en el espacio utilizando las teclas I, J, K, L, Z, X.
- **Cambiar el modo de sombreado** entre Phong y Vertex Color con las teclas 1 y 2.

- **Mostrar ayuda** con la tecla H.

Resumen del Flujo del Programa

Transformación de los Vértices: Los vértices del cubo se transforman del espacio de mundo al espacio de la cámara mediante la matriz de vista. Luego, se proyectan a 2D utilizando la matriz de proyección.

Iluminación: Se calcula la iluminación para cada triángulo utilizando el modelo de iluminación Phong, sumando componentes difusa, especular y ambiental.

Rasterización y Z-buffering: Cada triángulo es rasterizado y su color y profundidad son calculados. Se usa el Z-buffer para garantizar que los objetos más cercanos cubran a los más lejanos.

Librerías utilizadas

- **OpenCV:** Para la visualización de la imagen, la creación de ventanas, y el manejo de gráficos 2D.
- **cmath:** Funciones matemáticas para operaciones vectoriales y trigonométricas.
- **string:** Para la construcción de mensajes en pantalla.

Funciones utilizadas

- `lookAt`: Genera una matriz de vista que ubica la cámara en un espacio 3D.
- `perspective`: Genera una matriz de proyección en perspectiva.
- `mul`: Multiplica matrices o matrices por vectores.
- `nrm`: Normaliza un vector.

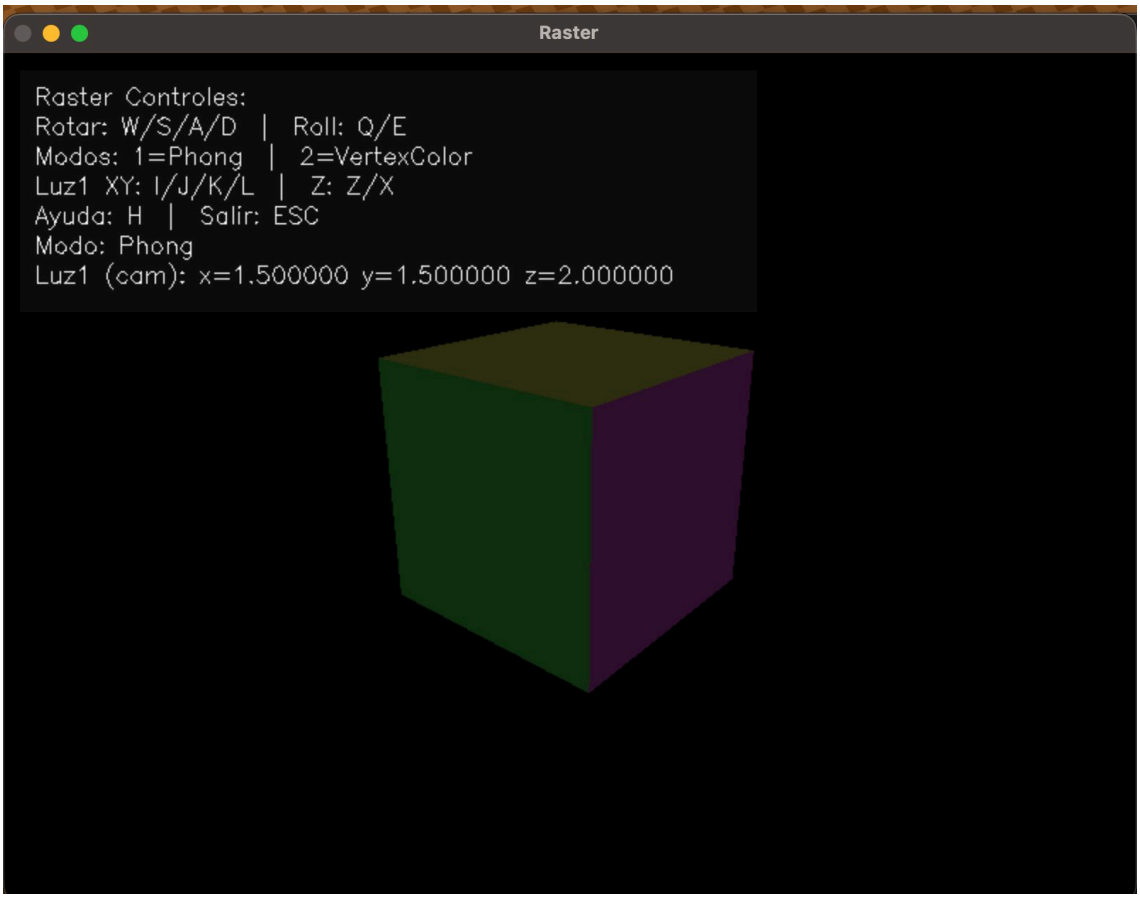
- `cross`: Calcula el producto cruzado entre dos vectores.
- `dot`: Calcula el producto punto entre dos vectores.
- `edge`: Calcula el área de un triángulo proyectado en la pantalla para la rasterización.
- `shade`: Calcula la iluminación utilizando el modelo de iluminación Phong.

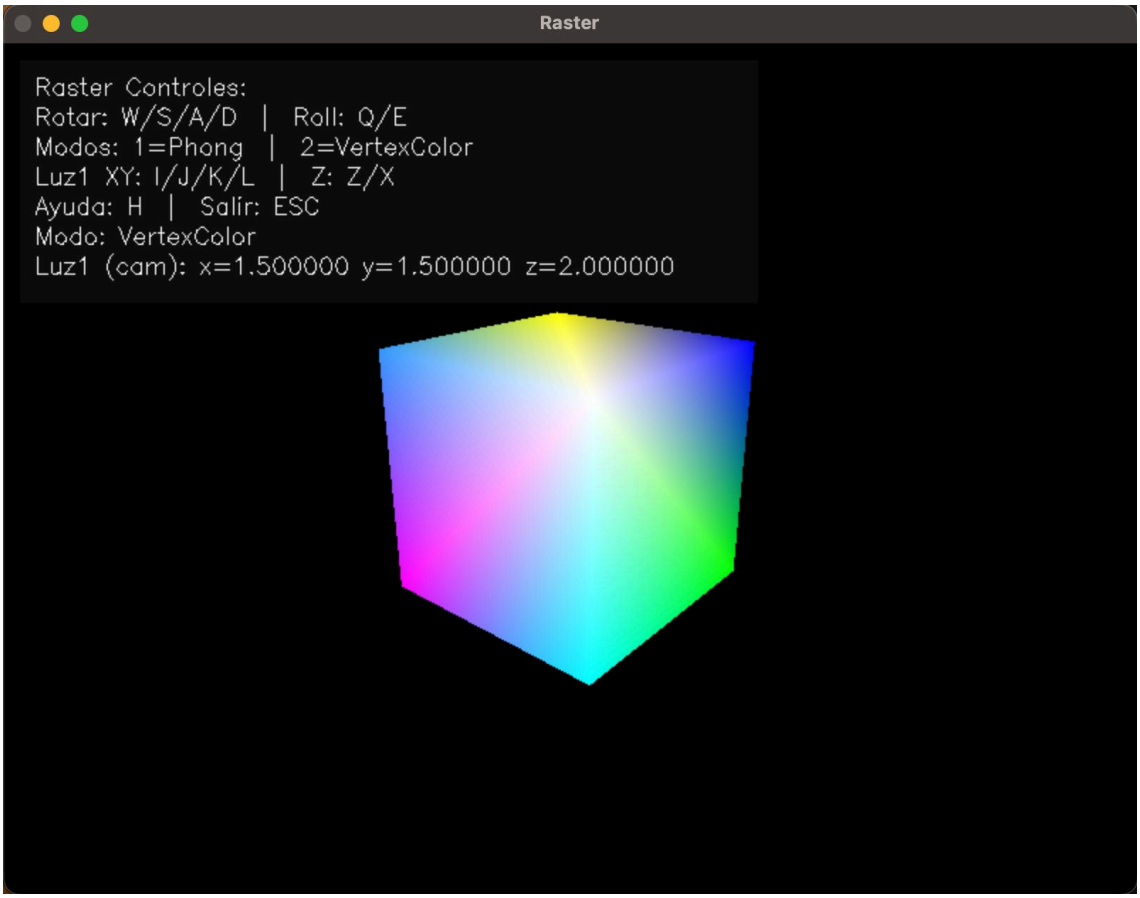
Ejemplo de ejecución

Supongamos la siguiente entrada:

- Posición de la cámara: (3.5f, 2.2f, 4.2f)
- Luces: L1 = (1.5f, 1.5f, 2.0f), L2 = (-2.0f, -1.0f, 1.5f)
- Modo de sombreado: Phong

La salida es una ventana gráfica donde se puede ver un cubo 3D con iluminación calculada en tiempo real. Los triángulos del cubo se iluminan de acuerdo al modelo de Phong, con una luz ambiental, difusa y especular. El usuario puede mover la cámara, rotar la escena, y ajustar la posición de las luces para observar los efectos de iluminación en tiempo real. A continuación dejamos imágenes de cómo se visualiza la ventana con el renderizado:


Raster
Raster Controles:
Rotar: W/S/A/D | Roll: Q/E
Modos: 1=Phong | 2=VertexColor
Luz1 XY: I/J/K/L | Z: Z/X
Ayuda: H | Salir: ESC
Modo: Phong
Luz1 (cam): x=1.500000 y=1.500000 z=2.000000


Raster
Raster Controles:
Rotar: W/S/A/D | Roll: Q/E
Modos: 1=Phong | 2=VertexColor
Luz1 XY: I/J/K/L | Z: Z/X
Ayuda: H | Salir: ESC
Modo: VertexColor
Luz1 (cam): x=1.500000 y=1.500000 z=2.000000