TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CUAUTLA

INGENIERÍA EN SISTEMAS

GRAFICACIÓN

SEMESTRE **ENERO JUNIO-2026**

DOCENTE: CABALLERO ALFARO ARÍSTIDES

UNIDAD 1

MARTÍNEZ GALLARDO SANTIAGO

4TO SEMESTRE

GRUPO: 1

25 DE FEBRERO DEL 2026

Reporte de Investigación: Unidad I - Introducción a la Graficación por Computadora

Este documento presenta una investigación documental sobre los fundamentos de la graficación por computadora. Se aborda su evolución histórica, las principales áreas de aplicación, los conceptos matemáticos esenciales, los modelos de color más utilizados, y las técnicas para la representación de primitivas gráficas. Además, se incluyen tutoriales prácticos sobre iluminación en Blender y ejercicios de dibujo de polígonos, así como una introducción al procesamiento de mapas de bits.

1.1. Historia y evolución de la graficación por computadora

La graficación por computadora ha experimentado una evolución fascinante, pasando de ser una herramienta exclusiva de la investigación y la industria militar a convertirse en una tecnología omnipresente en el entretenimiento, el diseño y la vida cotidiana.

Años 1950-1960: Los inicios y el hardware primitivo.

Whirlwind y el CRT: El proyecto Whirlwind en el MIT (Instituto Tecnológico de Massachusetts) utilizaba tubos de rayos catódicos (CRT) para mostrar información gráfica simple, sentando las bases para la visualización interactiva.

Sketchpad (1963): Ivan Sutherland, en su tesis doctoral en el MIT, creó "Sketchpad", un sistema revolucionario que permitía a los usuarios dibujar y manipular objetos geométricos en una pantalla con un lápiz óptico. Se le considera el precursor de los modernos programas de diseño asistido por computadora (CAD).

Años 1970: Rasterización y el nacimiento de los gráficos 3D.

Display de Raster: Se popularizaron las pantallas de raster, que dividen la imagen en una cuadrícula de píxeles. Esto permitió representar superficies sólidas y sombreados, a diferencia de los gráficos vectoriales de línea.

Universidad de Utah: Este centro se convirtió en la cuna de muchos avances. Allí se desarrollaron algoritmos fundamentales como el *renderizado de malla de polígonos*, el *z-buffer* (para eliminar superficies ocultas) y el *sombreado de Gouraud*.

Años 1980: La revolución de la computadora personal y los gráficos 3D en el cine.

PC y GUI: La llegada de computadoras personales como la Apple Macintosh y el uso de Interfaces Gráficas de Usuario (GUI) popularizaron los gráficos 2D para el público masivo.

Efectos especiales: Películas como *Tron* (1982) de Disney y *The Abyss* (1989) de James Cameron demostraron el potencial de los gráficos generados por computadora (CGI) en el cine. En 1986, Pixar comenzó a crear cortometrajes y largometrajes totalmente animados por computadora.

Años 1990 - Presente: Tiempo real, fotorrealismo y democratización.

GPU (Unidad de Procesamiento Gráfico): El desarrollo de tarjetas gráficas aceleradoras por empresas como NVIDIA (con la GeForce 256 en 1999) permitió el renderizado 3D en tiempo real, revolucionando la industria de los videojuegos.

Realismo y Nuevas Técnicas: La evolución de las GPUs ha permitido técnicas cada vez más complejas como el *sombreado por píxel*, la *iluminación global*, el *trazado de rayos (ray tracing) en tiempo real, y la integración de la realidad virtual y aumentada en aplicaciones cotidianas.

1.2. Áreas de aplicación

La graficación por computadora es una disciplina transversal que se aplica en numerosos campos:

Entretenimiento: Es el área más visible. Incluye los videojuegos (desde títulos independientes hasta grandes producciones AAA), la producción de películas animadas (como las de Pixar, DreamWorks o Studio Ghibli) y los efectos visuales (VFX) en cine y televisión.

Diseño y Manufactura (CAD/CAM/CAE): El Diseño Asistido por Computadora (CAD) es esencial para la ingeniería, la arquitectura y el diseño industrial. Permite crear modelos precisos de productos, edificios y piezas mecánicas antes de su fabricación.

Visualización Científica y Médica: Se utiliza para representar datos complejos de forma visual, como modelos moleculares, mapas meteorológicos, o simulaciones físicas. En medicina, es fundamental para la reconstrucción 3D de tomografías, resonancias magnéticas, y para la planificación de cirugías asistidas por computadora.

Simulación y Entrenamiento: Creación de entornos virtuales realistas para el entrenamiento de pilotos (simuladores de vuelo), personal militar, conductores de vehículos pesados, o cirujanos, permitiendo la práctica sin riesgos en el mundo real.

Arte Digital y Diseño Gráfico: Herramientas como Adobe Photoshop, Illustrator o Krita permiten la creación de ilustraciones, pintura digital, diseño de interfaces de usuario (UI), tipografía y composiciones gráficas para medios impresos y digitales.

Educación: Creación de materiales didácticos interactivos, simulaciones educativas, visitas virtuales a museos o recreaciones históricas que facilitan el aprendizaje.

1.3. Aspectos matemáticos de la graficación

Las matemáticas son el lenguaje subyacente de la graficación por computadora. Para manipular y representar objetos en un espacio virtual, se utilizan diversas ramas:

Geometría Analítica: Describe los objetos mediante coordenadas (x, y) en 2D y (x, y, z) en 3D. Las ecuaciones de rectas, planos, circunferencias y esferas son fundamentales.

Álgebra Lineal: Es la herramienta más importante.

Vectores: Representan direcciones, posiciones (como vectores de posición) y propiedades como la velocidad o la normal de una superficie (perpendicular a ella).

Matrices: Permiten aplicar transformaciones geométricas a los objetos de manera eficiente. Una misma matriz puede representar una **traslación** (movimiento), **rotación** (giro) o **escalado** (cambio de tamaño). Combinando matrices (multiplicándolas) se pueden realizar transformaciones complejas en un solo paso.

Trigonometría: Esencial para calcular ángulos de rotación, proyecciones de cámaras, y para entender funciones como seno y coseno en el trazado de curvas y circunferencias.

Cálculo: Se utiliza para modelar superficies curvas complejas (como las curvas y superficies de Bézier o B-spline), para cálculos de iluminación y sombreado

(integrales sobre áreas de luz), y para simular movimientos físicos realistas (cálculo de trayectorias y velocidades).

1.4. Modelos del color: RGB, CMY, HSV y HSL

Los modelos de color son sistemas para representar y especificar los colores de forma numérica.

Modelo RGB (Aditivo):

Fundamento: Se basa en la síntesis aditiva de la luz. Los colores primarios son Rojo (Red), Verde (Green) y Azul (Blue).

Funcionamiento: Al combinar estos tres colores de luz a diferentes intensidades (generalmente de 0 a 255), se crea un amplio espectro de colores. La suma de los tres a máxima intensidad da el blanco, y la ausencia de todos da el negro.

Uso: Es el modelo estándar para pantallas (monitores, televisores, teléfonos), escáneres y cámaras digitales.

Modelo CMY (Sustractivo):

Fundamento: Se basa en la síntesis sustractiva de pigmentos (tintas, pinturas). Los colores primarios son Cian (Cyan), Magenta (Magenta) y Amarillo (Yellow).

Funcionamiento: Se parte de una superficie blanca (que refleja toda la luz). Al aplicar pigmentos, estos *restan* (absorben) ciertas longitudes de onda de la luz. La combinación de todos los pigmentos en teoría da el negro (aunque en la práctica resulta un marrón oscuro, por lo que se añade tinta negra, dando lugar al modelo CMYK).

Uso: Es el modelo utilizado en la impresión y las artes gráficas.

Modelo HSV (Matiz, Saturación, Valor) y HSL (Matiz, Saturación, Luminosidad/Ligereza)

Fundamento: Estos modelos están diseñados para ser más intuitivos para los humanos, ya que separan la información cromática de la luminosidad.

Componentes:

Matiz (Hue): Es el tipo de color puro (ej. rojo, verde, azul). Se representa como un ángulo de 0° a 360° en una rueda de color.

Saturación (Saturation): Es la intensidad o pureza del color. Va desde 0% (gris, sin color) hasta 100% (color puro y vibrante).

Valor (Value - HSV) / Luminosidad (Lightness - HSL): Controla el brillo. En HSV, el valor 0% es siempre negro. En HSL, la luminosidad 50% representa el color puro, 0% es negro y 100% es blanco.

Uso: Son muy utilizados en software de edición de imagen y diseño gráfico (como los selectores de color en Photoshop o GIMP), ya que permiten a los artistas elegir colores de forma más natural.

Tutorial: Iluminación de un cubo en Blender

Este tutorial explica cómo iluminar un cubo y sus caras en Blender para resaltar su volumen.

1. Configuración Inicial:

Abre Blender. La escena por defecto ya contiene un cubo, una cámara y una luz puntual. Selecciona la luz (Light) y elimínala con la tecla `Supr` para empezar desde cero con nuestra propia iluminación.

2. Añadir una Luz de Área:**

Presiona `Shift + A` -> `Luz` -> `Área`. Una luz de área produce sombras suaves y realistas, similar a una ventana o un panel de luz.

3. Posicionar la Luz Principal (Key Light):

Selecciona la luz de área. Con la herramienta de mover (`G`), colócala arriba y a la izquierda del cubo, formando un ángulo de aproximadamente 45 grados con respecto a la cámara. Con la herramienta de rotar (`R`), oriéntala para que apunte hacia el cubo.

En el panel de propiedades de la luz (icono de bombilla), aumenta la *Potencia* a unos 500-1000 W para que ilumine bien el cubo.

4. Añadir una Luz de Relleno (Fill Light)

Esta luz suaviza las sombras del lado opuesto a la luz principal.

`Shift + A` -> `Luz` -> `Área`. Colócala abajo y a la derecha del cubo, con menor potencia (por ejemplo, 100-200 W). Puedes darle un color ligeramente azulado para crear un contraste interesante.

5. Añadir una Luz de Recorte o Efecto (Rim Light):

Esta luz se coloca detrás del objeto para crear un brillo en los bordes que lo separe del fondo.

`Shift + A` -> `Luz` -> `Área`. Colócala detrás del cubo, ligeramente por encima, apuntando hacia la cámara. Ajusta su potencia (ej. 400 W).

6. Visualizar el Resultado:

Cambia el modo de visualización en la ventana 3D a `Vista Sombreada` o, mejor aún, a `Vista Renderizada` (icono en la esquina superior derecha de la ventana 3D) para ver el efecto de las luces en tiempo real.

7. Ajustar el Material del Cubo:

Selecciona el cubo. En el panel de propiedades, ve a la pestaña de material (icono de esfera).

Ajusta el *Color Base* al que desees. Juega con la *Rugosidad* (Roughness): un valor bajo hará que el material sea brillante y refleje las luces de forma más nítida, mientras que un valor alto lo hará mate y difuminará los reflejos.

Con esta configuración de tres puntos de luz (Key, Fill y Rim), las caras del cubo quedarán iluminadas de manera diferencial, resaltando su geometría tridimensional.

1.5. Representación y trazo de líneas y polígonos

En gráficos por computadora, la unidad básica de dibujo es el píxel. Para representar formas geométricas ideales en una cuadrícula de píxeles, se necesitan algoritmos específicos.

Trazo de Líneas: El objetivo es determinar qué píxeles deben activarse para aproximar una línea recta entre dos puntos.

Algoritmo de Bresenham Es uno de los más eficientes. Utiliza solo aritmética de números enteros (sumas, restas y desplazamientos de bits) para decidir qué píxel se aproxima más a la línea teórica en cada columna (o fila), lo que lo hace muy rápido para el hardware.

Trazo de Polígonos: Un polígono es una figura geométrica plana formada por la unión de una secuencia finita de segmentos de línea (sus lados) que encierran una región. La representación de un polígono implica dibujar sus bordes (usando algoritmos de línea) y, lo que es más complejo, **rellenar** el área interior. Para esto último se usan algoritmos como el *Relleno por Escaneo de Líneas (Scanline Fill)*, que determina, para cada línea horizontal de píxeles (scanline), los puntos de intersección con los bordes del polígono y rellena los píxeles entre ellos.

1.5.1 Formatos de imagen

Un formato de imagen define la estructura del archivo que almacena los datos de una imagen digital. Se dividen principalmente en dos categorías:

Mapas de Bits (Raster): Almacenan la imagen como una cuadrícula de píxeles (un mapa de bits). Su resolución es fija; al ampliarlos, se pixelan.

BMP (Bitmap): Formato sin compresión de Windows. Archivos muy grandes pero sin pérdida de calidad.

JPEG/JPG (Joint Photographic Experts Group): Utiliza compresión con pérdida, lo que reduce drásticamente el tamaño del archivo al descartar información de color que el ojo humano no percibe fácilmente. Ideal para fotografías.

PNG (Portable Network Graphics): Utiliza compresión sin pérdida y soporta transparencia (canal alfa). Ideal para gráficos web, logotipos e imágenes con texto.

GIF (Graphics Interchange Format): Soporta animaciones y transparencia, pero tiene una paleta de color limitada a 256 colores.

Gráficos Vectoriales: Almacenan la imagen como fórmulas matemáticas que describen líneas, curvas, formas y colores. Son escalables infinitamente sin pérdida de calidad.

SVG (Scalable Vector Graphics): Formato estándar para la web.

AI (Adobe Illustrator), CDR (CorelDRAW): Formatos nativos de software de diseño vectorial.

Ejercicio Práctico: Dibujo de un Polígono y la Flor de la Vida

A continuación, se describen los ejercicios prácticos realizados para comprender el trazo de primitivas gráficas.

Práctica 1: Dibujo de un Polígono Regular (Pentágono)

Objetivo:Trazar un pentágono regular utilizando un algoritmo de línea (como el de Bresenham) o, en un entorno de programación, conectando puntos calculados matemáticamente.

Procedimiento (Conceptual):

1. Calcular las coordenadas de los 5 vértices de un pentágono. Si el centro es `(cx, cy)` y el radio es `r`, la posición del vértice `i` es:

   `x_i = cx + r * cos(2 * π * i / 5)`

   `y_i = cy + r * sen(2 * π * i / 5)`

2. Utilizar una función `dibujarLinea(x1, y1, x2, y2)` para conectar cada vértice con el siguiente (`v0 -> v1`, `v1 -> v2`, ..., `v4 -> v0`), formando así el contorno del polígono.

Práctica 2: Dibujo de la Flor de la Vida

Objetivo: Dibujar la "Flor de la Vida", un símbolo geométrico compuesto por múltiples círculos superpuestos con una simetría hexagonal.

Procedimiento (Conceptual):

1. Definir un radio base `r` para los círculos.

2. Dibujar un círculo central con centro en `(0,0)`.

3. Dibujar una capa de 6 círculos alrededor del central. Sus centros se sitúan a una distancia `r` del centro, en ángulos de 0°, 60°, 120°, 180°, 240° y 300°. Sus coordenadas son `(r * cos(angulo), r * sen(angulo))`.

4. Para una flor más completa, dibujar una segunda capa de 12 círculos alrededor de la primera. Los centros se sitúan sobre una circunferencia de radio `2r` y en ángulos que son múltiplos de 30°.

Estos ejercicios permiten comprender la transición de una definición matemática (coordenadas, ecuaciones de círculos y líneas) a su representación visual en una pantalla.

1.6. Procesamiento de mapas de bits

El procesamiento de mapas de bits (o imágenes raster) se refiere a la manipulación de una imagen digital píxel por píxel, o en grupos de píxeles, para lograr un efecto deseado o extraer información. Esto se logra mediante filtros y operaciones.

Operaciones Punto a Punto: Se modifica el valor de cada píxel de forma independiente, basándose únicamente en su valor original. El resultado es una nueva imagen.

Ajustes de Brillo y Contraste: Sumar o restar un valor constante a todos los píxeles (brillo) o expandir/contrastar el rango de valores (contraste).

Umbralización (Thresholding): Convertir una imagen a blanco y negro puro. Si el píxel está por encima de un cierto valor de gris, se vuelve blanco; si está por debajo, negro. Muy útil en visión por computadora para segmentar objetos del fondo.

Mapa de Colores (Look-Up Table - LUT):** Reemplazar el color de cada píxel por otro según una tabla predefinida, para crear efectos de filtro o falsos colores.

Operaciones de Vecindad (Convoluciones o Filtros Espaciales): El valor del píxel de salida se calcula a partir de los valores de sus píxeles vecinos. Se utiliza una pequeña matriz de números llamada **kernel** o máscara, que se desliza sobre toda la imagen.

Filtro de Desenfoque (Blur): El kernel promedia los valores de los píxeles vecinos, suavizando la imagen y reduciendo el ruido. El *Filtro Gaussiano* es un tipo común de desenfoque.

Filtro de Enfoque (Sharpen): El kernel está diseñado para acentuar las diferencias entre un píxel y sus vecinos, haciendo que los bordes y detalles sean más nítidos.

Detección de Bordes (Edge Detection): Kernels como los de **Sobel** o Canny calculan el gradiente de intensidad de la imagen, resaltando las zonas donde hay cambios bruscos de color, que suelen corresponder a los bordes de los objetos.

Bibliografía (Formato APA)

Foley, J. D., van Dam, A., Feiner, S. K., & Hughes, J. F. (1996). *Computer Graphics: Principles and Practice* (2nd ed. in C). Addison-Wesley.

Hearn, D., Baker, M. P., & Carithers, W. (2014). *Computer Graphics with OpenGL* (4th ed.). Pearson Education.

Shirley, P., Ashikhmin, M., & Marschner, S. (2009). *Fundamentals of Computer Graphics* (3rd ed.). A K Peters/CRC Press.

Blender Foundation. (s.f.). *Blender Manual*. Recuperado el 20 de octubre de 2023, de https://docs.blender.org/manual/en/latest/

González, R. C., & Woods, R. E. (2018). *Digital Image Processing* (4th ed.). Pearson.