Lecture 01: Definition of computer graphics
Instructor: Mejbah Ahammad

Intelligence Academy

Contents

Definition of Computer Graphics

1.1 Formal Definitions

Computer Graphics is the field that studies the mathematical, algorithmic, and systems principles for *representing, manipulating, generating, and displaying* visual content on digital devices. Formally, we model a scene

$$\mathcal{S} = \{\, G_i,\ \mathcal{M}_i,\ \mathcal{L},\ \mathcal{C}\,\}$$

where G_i are geometric objects, $\mathcal{M}_i$ material/appearance models, $\mathcal{L}$ illumination, and $\mathcal{C}$ camera/display model. A *renderer* R maps $(\mathcal{S}, \Theta)$ to pixels $\mathbf{I}$ given algorithmic parameters Θ (e.g., rasterization vs. ray tracing):

$$\mathbf{I} \;=\; R(\mathcal{S}; \Theta).$$

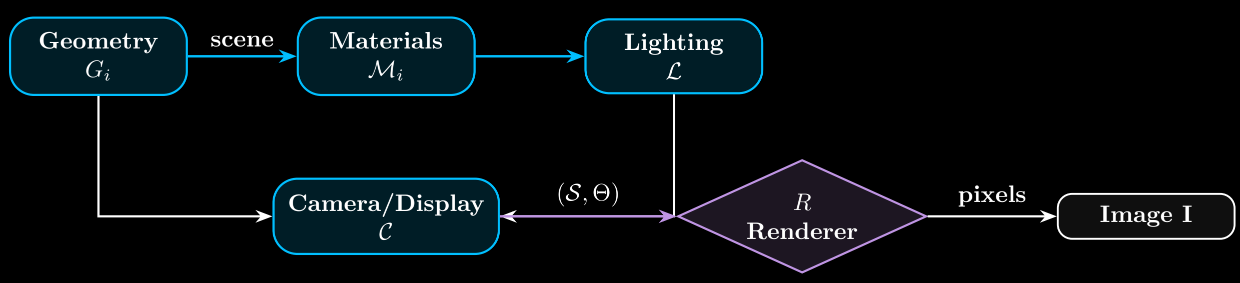

Figure 1: Formal view: scene elements → renderer R → image.

1.2 Scope and Boundaries

In scope: 3D/2D modeling, transformations, rendering (rasterization, ray/path tracing), shading, texturing, animation, image synthesis, GPU systems. **Adjacent but distinct:** *Image processing/computer vision* (analyze existing images), *HCI/Visualization* (interaction/insight). Boundaries often overlap via shared math and pipelines.

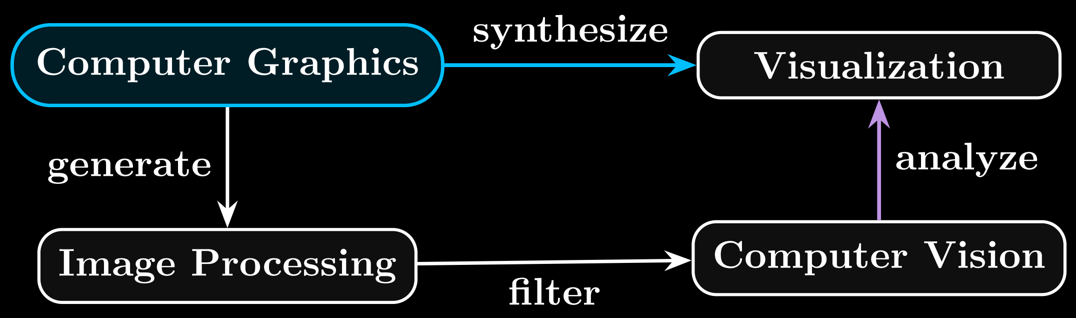

Figure 2: Scope and boundaries with adjacent fields and directional relationships.

1.3 Core Objectives

Key objectives: **(1) Realism** (physical accuracy, PBR), **(2) Speed** (real-time interactivity), **(3) Control** (artistic direction), **(4) Efficiency** (memory/compute), **(5) Fidelity** (perceptual quality).

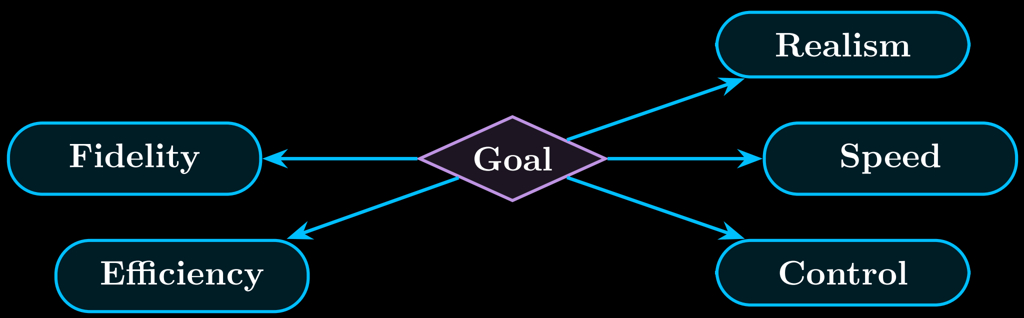

Figure 3: Core objectives radiating from the central goal (directional emphasis).

1.4 Terminology and Key Concepts

Essential terms: *Primitives* (points/lines/triangles), *Transformations* (model–view–projection), *Shading* (Phong, microfacet/BRDF), *Textures/UV*, *Sampling* (Nyquist, anti-aliasing), *Frame buffer*, *Z-buffer*, *Shader stages*.

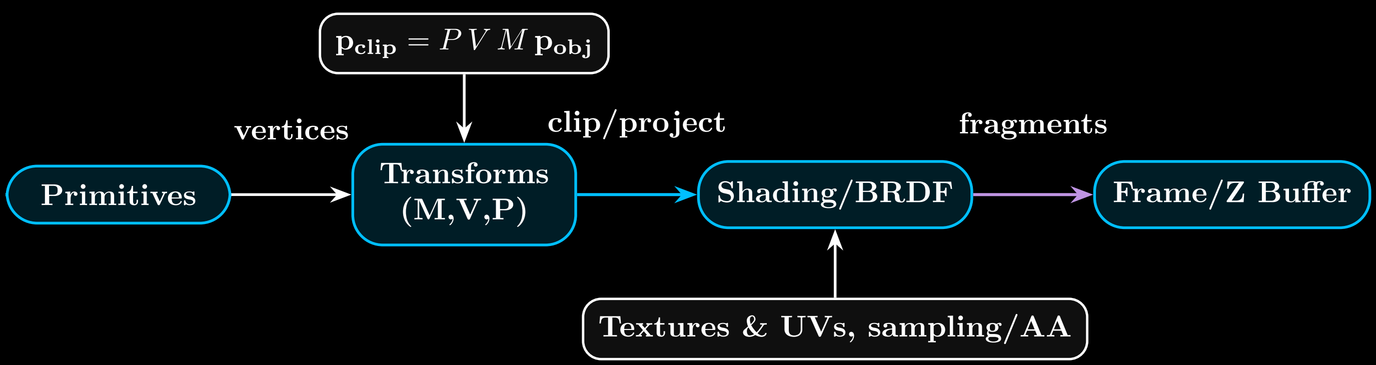

Figure 4: Left-to-right concept flow: primitives → transforms → shading → buffers.

1.5 Relationship to Visualization and Imaging

Graphics *produces* images from models; **Visualization** aims to *reveal structure/insight* from data (often using graphics); **Imaging** and **Vision** *capture/analyze* real signals. Many pipelines are bi-directional (e.g., *image-based lighting*, *neural rendering*).

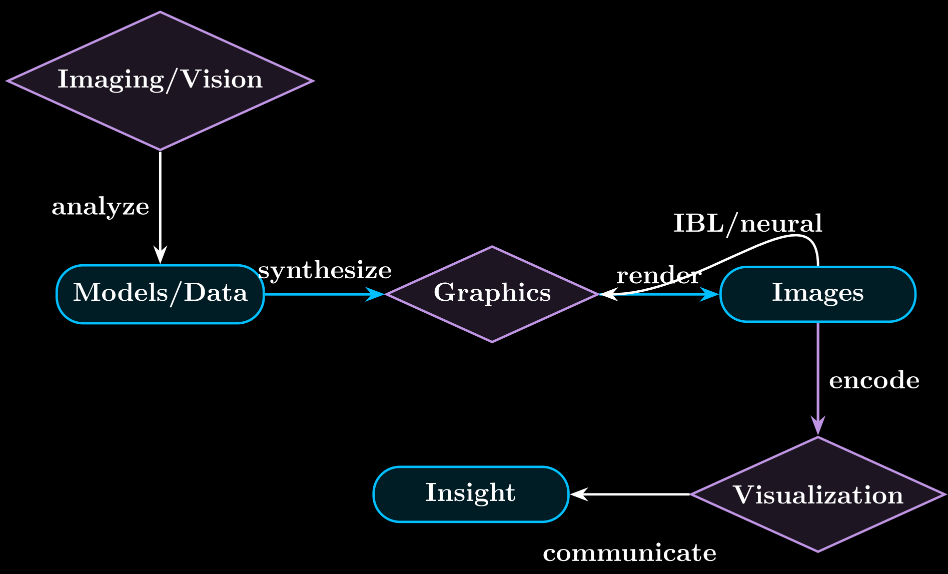

Figure 5: Directional relationships among modeling, graphics, imaging/vision, and visualization.

Historical Context

2.1 Early Mechanical and Vector Displays

Essence. Pre-raster graphics used analog/mechanical plotters, oscilloscopes, and early CRT vector displays. Images were drawn as *continuous strokes* (lines/curves) directed by deflection signals—excellent for crisp lines, limited for filled/tonal imagery.

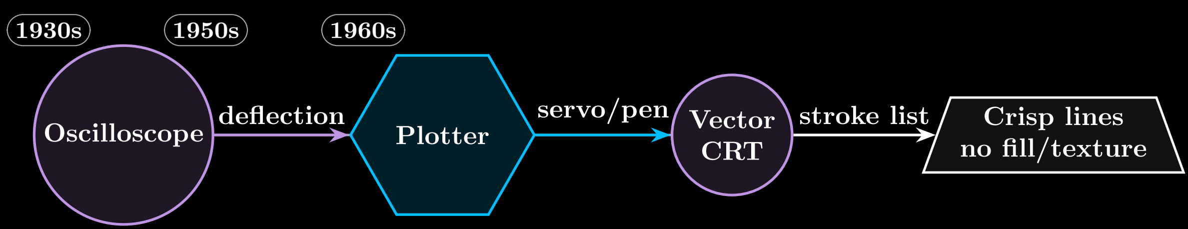

Figure 6: Vector era: draw strokes directly; great lines, limited shading.

2.2 Raster Revolution and Framebuffers

Essence. The shift to *raster* sampled images into pixels with a *framebuffer*. Enabled filled polygons, textures, and scan conversion; performance tied to memory bandwidth and rasterization efficiency.

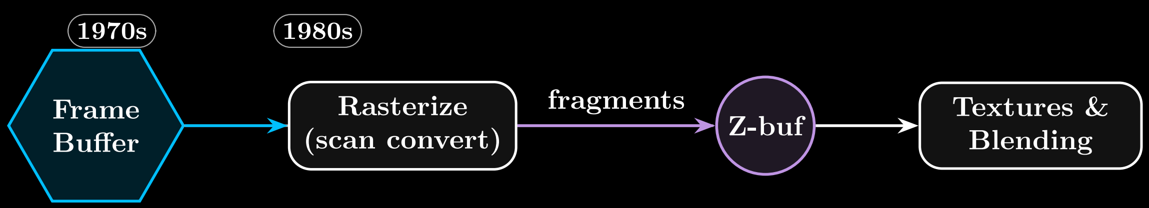

Figure 7: Raster pipeline: framebuffer → rasterization → depth & compositing.

2.3 GUI Era and Workstations

Essence. Bitmapped displays and windows/icons/menus/pointers (WIMP) redefined interaction. Dedicated *workstations* (graphics accelerators) standardized APIs and toolkits.

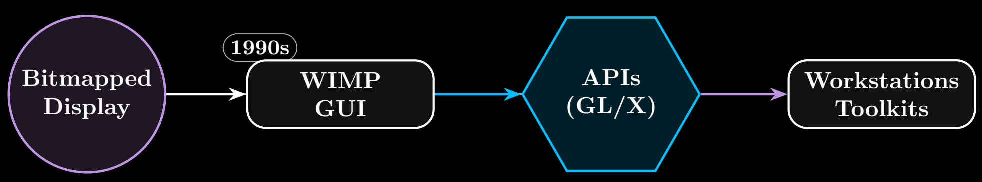

Figure 8: GUI era: bitmaps enable WIMP; APIs + workstations scale productivity.

2.4 Real-Time 3D and GPUs

Essence. Fixed-function pipelines evolved into programmable shaders; GPUs delivered massive parallelism for real-time 3D (games, simulators, CAD). The pipeline became configurable (vertex/fragment → geometry/compute).

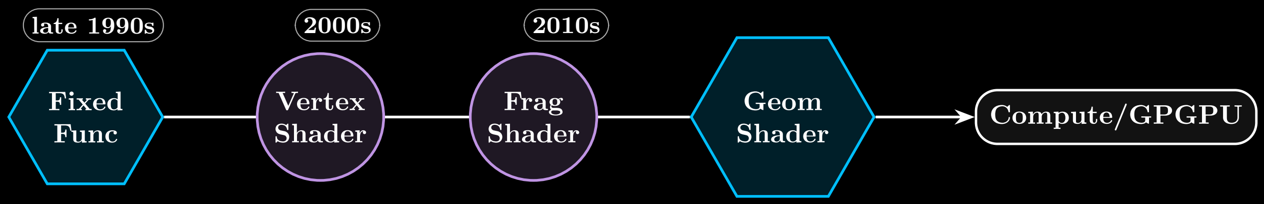

Figure 9: GPU evolution: fixed-function → programmable shaders → compute.

2.5 Modern Trends: Physically Based & Neural Rendering

Essence. *Physically Based Rendering* (PBR) targets energy-consistent light transport (microfacet BRDFs, importance sampling). *Neural methods* learn appearance, radiance fields, denoisers, and super-resolution; hybrid real-time ray/path tracing emerges.

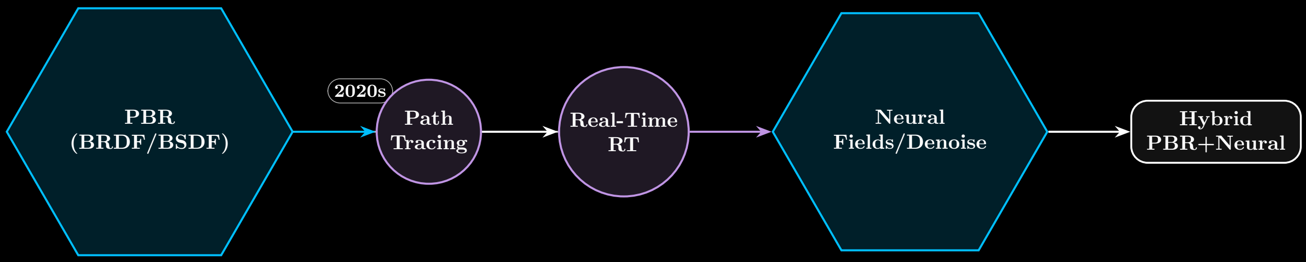

Figure 10: Modern stack: PBR foundations + neural components → hybrid renderers.