MCA101 : COMPUTER GRAPHICS

Raghav B. Venkataramaiyer

CSED TIET Patiala India.

August 3, 2024

OUTLINE

OUTLINE

SKETCHPAD 1960

Image Courtesy: Youtube

Flatland 1999

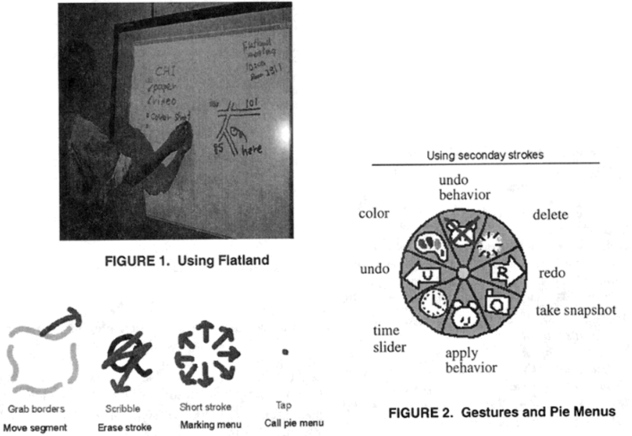


Figure: Flatland [MIEL99]

PLUSHIE 2007

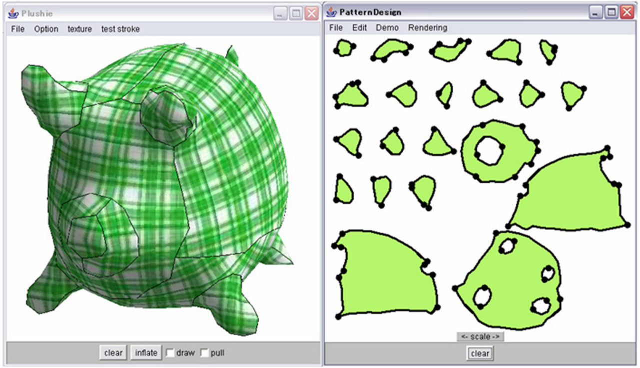


FIGURE: Plushie [MI07]

See Also: Teaser @Youtube

CITYENGINE ESRI 2012

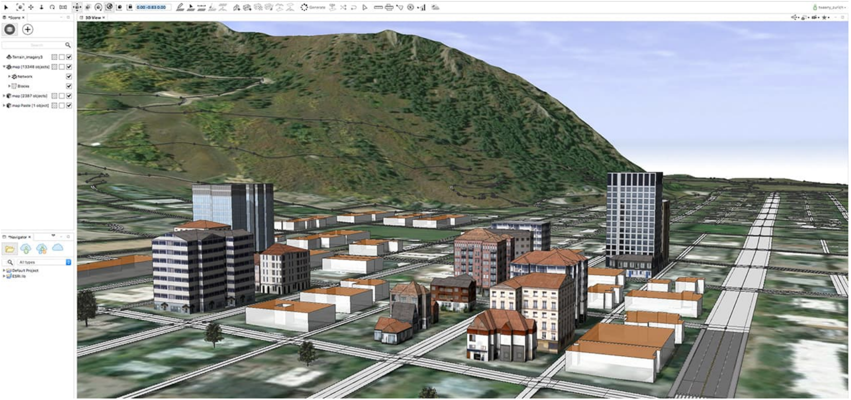


Image Courtesy: Youtube
See Also: Peter Wonka @3DGV

OUTLINE

CATHODE RAY TUBE

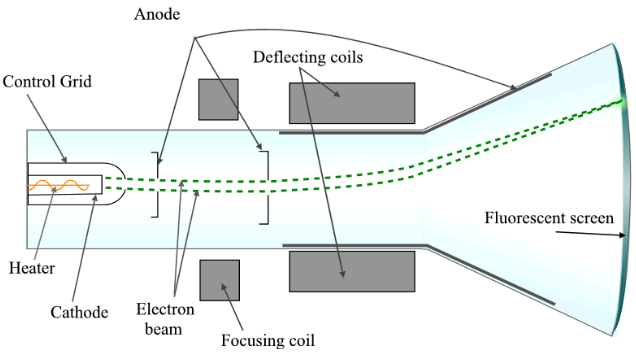


Image Courtesy: *Wikipedia*
Monochrome (Raster), Colour (Raster), Direct-View Bistable Storage (Vector)

(Variant of a Neon Lamp)

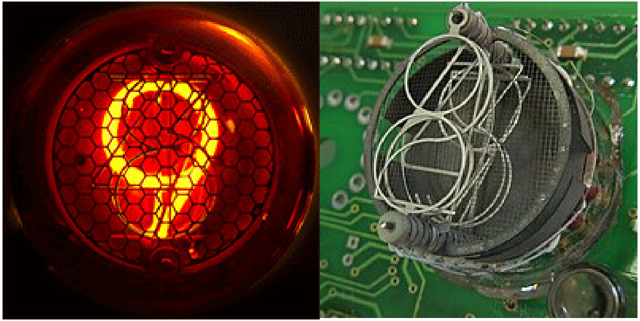


Image Courtesy: Wikipedia

FLIP-DISC DISPLAY

Image Courtesy: *Web*

Recently

- LCD
- Plasma (1995)
- LED
- OLED
- AMOLED
- E-Ink (Kindle)

OUTLINE

RASTER

Prehistoric, Ancient & Medieval

- Clay Tablets
- Wood Block Printing
- Stencils/ Masks
- Seals and Stamps
- Lithography
- Flat-bed Printing Press

Modern

- Rotary Printing Press
- Offset Press
- Screenprinting
- Dot matrix printing (DMP)
- Thermal Printing (Thermochromic)
- Laser Printing

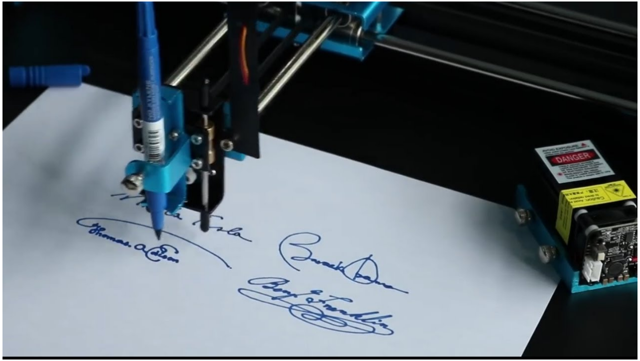


Image Courtesy: Youtube
See Also: Can I teach a robot to replicate a line art? [VKN20]

OUTLINE

REAL TIME

- THROUGHPUT $>=$ 18 frames per second (FPS);
- REFRESH RATE $>=$ 24 Hz;

RESOLUTION

A target of 24 FPS, will leave per pixel computation time of,

- $320 \times 240 \approx 77000$ px $\rightarrow 543$ ns
- $640 \times 480 \approx 0.3$ MP $\rightarrow 135$ ns
- $1024 \times 768 \approx 0.8$ MP $\rightarrow 53$ ns
- $1366 \times 768 \approx 1$ MP $\rightarrow 40$ ns

RESOLUTION

Same with a 384 core GPU,

- $1440 \times 900 \approx 1.3$ MP $\rightarrow 12.3\ \mu s$
- $1920 \times 1280 \approx 2.5$ MP $\rightarrow 6.5\ \mu s$
- $3072 \times 1920 \approx 6$ MP $\rightarrow 2.7\ \mu s$
- $8192 \times 4320 \approx 35$ MP $\rightarrow 0.45\ \mu s$

OUTLINE

REPRESENTATION

The way a concept/ entity is represented in machine code. *e.g.*

- Colour as RGB; is just a mem block of 24-bits;
- Geometry as Polygonal Faces; Array of Vertex Data and Face Relations;
- Illumination as a physical model;
- Textures as 2D images;
- and so forth...

Changing the view-point

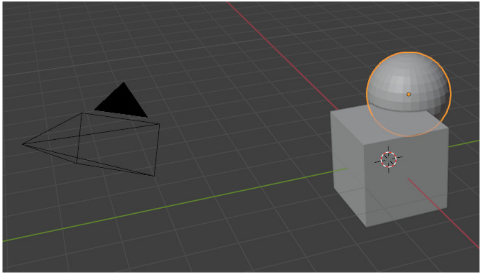


FIGURE: User View

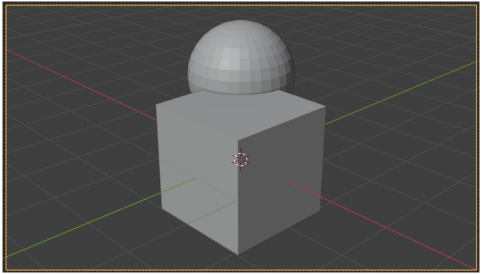


FIGURE: Camera View

INTERPOLATION

Estimating/ Computing the values in between two states.

e.g.

Points on a line segment are intermediate/ interpolated states between its two end points.

VISUALISATION

To create a visible artefact
corresponding to a concept.

OUTLINE

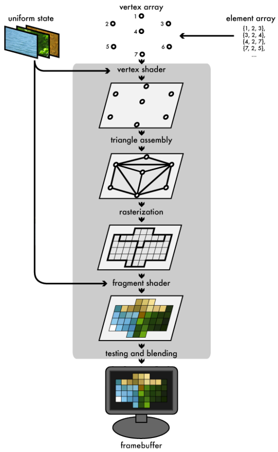


Image Courtesy: Durian Software

- Data Input
  - Uniform State
  - Vertex Array (per-vertex data)
  - Element Array (list of elements)
- Vertex Shader
- Triangle Assembly
- Rasterisation
- Fragment Shader
- Framebuffer

THE GRAPHICS PIPELINE

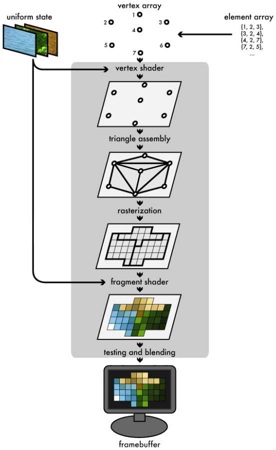


Image Courtesy: Durian Software

- Data Input
- Vertex Shader
  - INPUT
    - Per Vertex Data
    - Uniform State
  - OUTPUT
    - Position
    - For Next Shader
- Triangle Assembly
- Rasterisation
- Fragment Shader
- Framebuffer

THE GRAPHICS PIPELINE

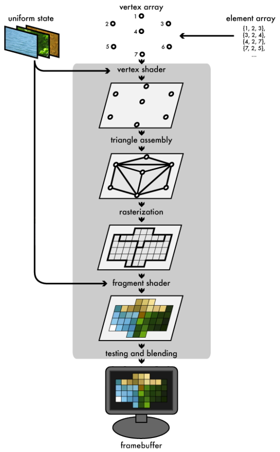
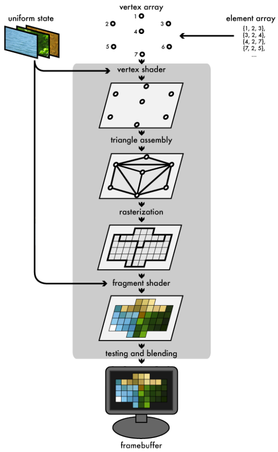


Image Courtesy: Durian Software

- Data Input
- Vertex Shader
- Triangle Assembly
  - Uses Element Array
- Rasterisation
- Fragment Shader
- Framebuffer

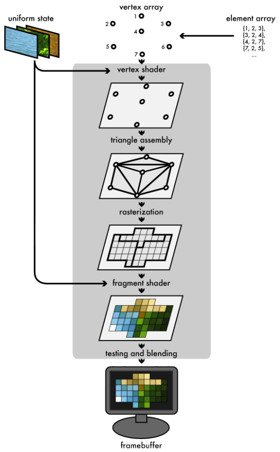
THE GRAPHICS PIPELINE



- Data Input
- Vertex Shader
- Triangle Assembly
- Rasterisation
  - Which element corresponds to which fragment
- Fragment Shader
- Framebuffer

Image Courtesy: Durian Software

THE GRAPHICS PIPELINE



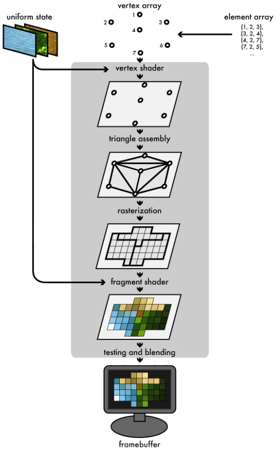
- Data Input
- Vertex Shader
- Triangle Assembly
- Rasterisation
- Fragment Shader

INPUT	■ From Prev Shader
	■ Uniform State
OUTPUT	■ Fragment Colour

- Framebuffer

Image Courtesy: Durian Software

THE GRAPHICS PIPELINE



- Data Input
- Vertex Shader
- Triangle Assembly
- Rasterisation
- Fragment Shader
- Framebuffer

Image Courtesy: Durian Software

OUTLINE

THREE TYPES

- POINT
- STRAIGHT LINE
- PLANAR POLYGON

THREE TYPES

- POINT
- STRAIGHT LINE
- PLANAR POLYGON

THREE TYPES

- POINT
- STRAIGHT LINE
- PLANAR POLYGON
  - Triangle
  - Quad

ATTRIBUTES

ELEMENTS

OUTLINE

OUTLINE

OUTLINE

PROBLEM

Given a straight line with vertex attributes: *a)* position in 2D coordinates; and *b)* temperature value; Draw the line with a heat gradient using a predefined heat map for colour.

[MI07] Yuki Mori and Takeo Igarashi. “Plushie: An Interactive Design System for Plush Toys”. In: *ACM Trans. Graph.* 26.3 (2007) (see p. 6).

[MIEL99] Elizabeth D. Mynatt, Takeo Igarashi, W. Keith Edwards, and Anthony LaMarca. “Flatland: New Dimensions in Office Whiteboards”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’99. New York, NY, USA: Association for Computing Machinery, 1999, pp. 346–353 (see p. 5).

[VKN20] R. B. Venkataramaiyer, S. Kumar, and V. P. Namboodiri. “Can I Teach a Robot to Replicate a Line Art”. In: *WACV*. 2020 IEEE Winter Conference on Applications of Computer Vision (WACV). Aspen, CA, USA: IEEE, 2020, pp. 1922–1930 (see p. 15).