POLITECNICO
MILANO 1863

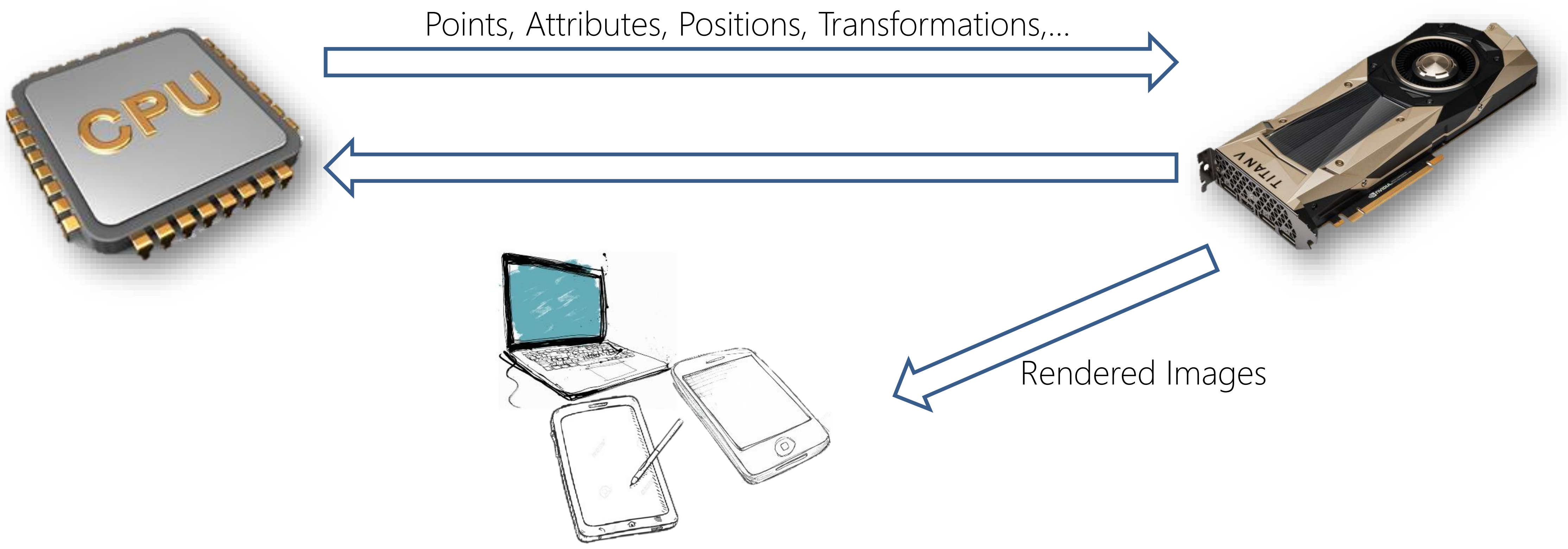
Introduction to OpenGL and WebGL

Computer Graphics 2021

Erica Stella (erica.stella@polimi.it)

OpenGL introduction

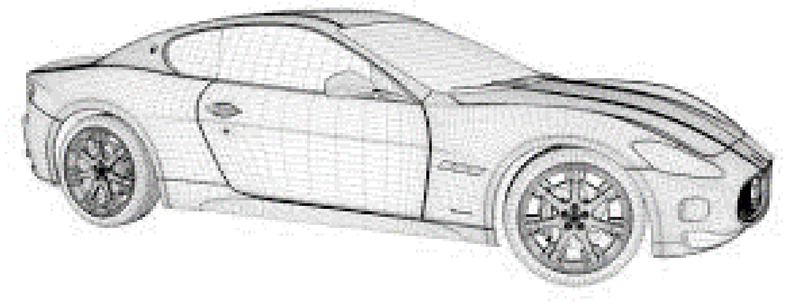
OpenGL *is "a software interface to graphics hardware"*

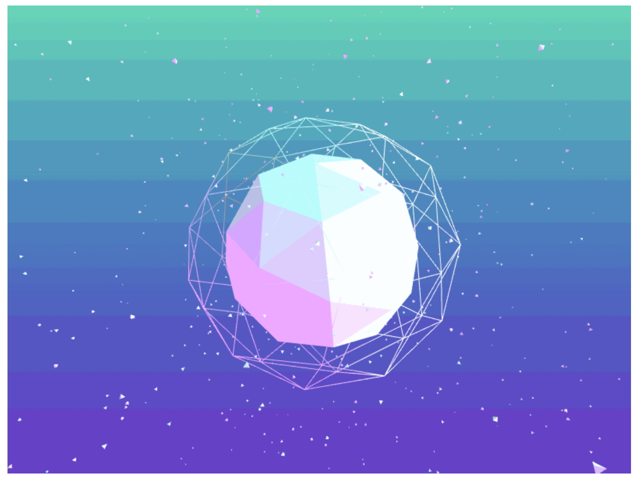


OpenGL introduction

OpenGL is an API:

- Not a programming language like C or C++:
- A program can use OpenGL API for
  - Defining or manipulating objects
  - Applying textures and lighting
  - Moving objects in the scene



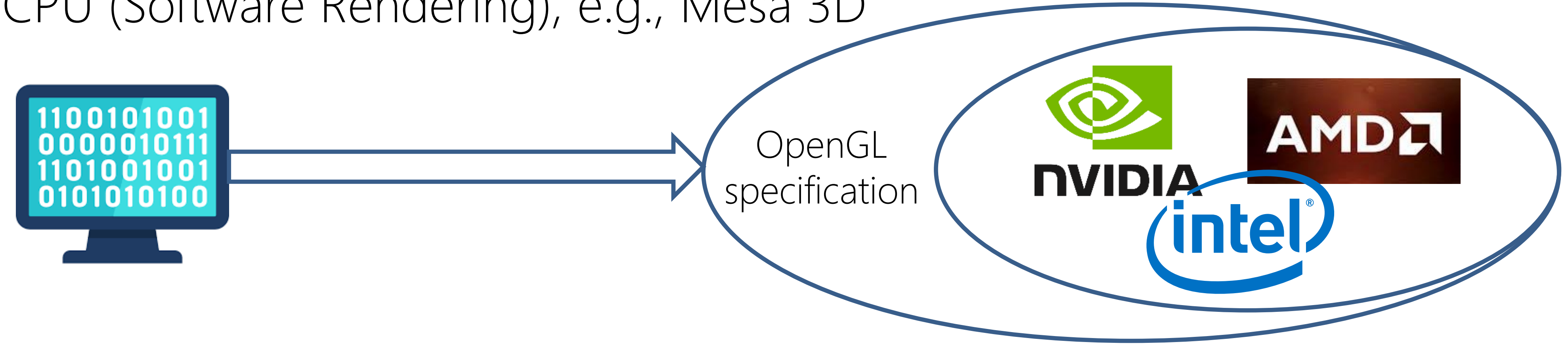


OpenGL introduction

Differently from **Microsoft DirectX** (available only for Windows PC, Phones & Tablets, and Xbox), **OpenGL** is available for a wide range of devices, including mobile phones, tablets and supercomputers.

The **OpenGL Specification** is implemented as the **OpenGL library** by hardware vendors (NVIDIA ,AMD, Intel, or Apple…) making OpenGL effectively portable.

OpenGL may be even implemented without a GPU by moving rendering operations into the host CPU (Software Rendering), e.g., Mesa 3D



OpenGL architecture

OpenGL is implemented as a client-server system:

Client-side:
- The application code in the main CPU memory
- Defines and sends the OpenGL commands

OpenGL commands ⇨

Server-side:
- Hardware and memory on the graphic card
- Executes the commands to render the final image

OpenGL is a large **state machine**:
- A collection of variables defines how OpenGL operates.
- The state is commonly referred to as the **OpenGL context**.
- We change OpenGL state by setting options, manipulating buffers and then render using the current context.
- When a state value is set, it remains set until some other function changes it.
- **The state value will influence subsequent steps until changed**.

Context and Windows

OpenGL specification defines a hardware-independent interface:

- No commands for handling user input
- **No commands for performing windowing tasks**

OpenGL context needs a windowing system for rendering on screen

The windowing system relies on the OS!

Every OS exposes different APIs $\Longrightarrow$ Cross-Platform libraries *(GLFW, GLEW, GLUT…) are* required to manage contexts

… but sometimes things get tricky …

WebGL

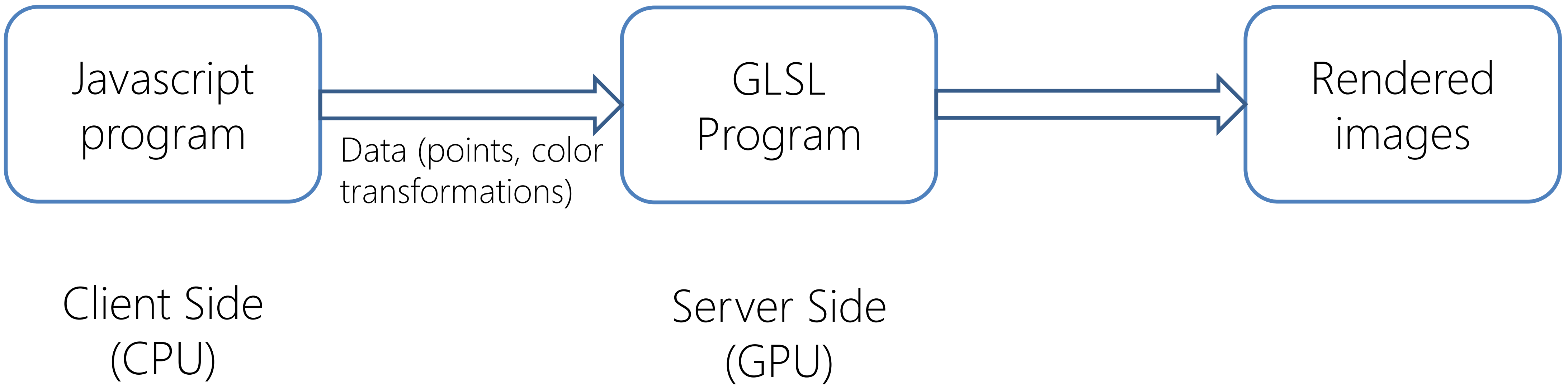
WebGL: OpenGL-style rendering within Web browsers:

- Works natively in all modern Web browsers (Mozilla Firefox, Google Chrome, Opera, Edge)
  - Check if your browser supports it https://get.webgl.org/webgl2/
- Uses **HTML5** <canvas> + WebGLRenderingContext for rendering.
- OpenGL ES functionalities are accessed using **JavaScript**.

In this course, to test the concepts of the theoretical part we will use WebGL with Mozilla Firefox or Google Chrome.

WebGL pipeline

- In WebGL, you must use **GLSL** to specify rendering operations
  - The CPU "just sends data to the GPU"
- GLSL is a programming language (similar to C/C++) that defines how the GPU handles the data and how the final rendered image has to be computed
  - GLSL tells the GPU how to use the data received from the CPU


Javascript program
Data (points, color transformations)
GLSL Program
Rendered images
Client Side (CPU)
Server Side (GPU)