

---

# Advanced Modern Graphics: Pipeline and Performance

Based on OpenGL 4.5, created on February 28, 2026

The logo for AKDAB is a blue speech bubble shape. Inside the bubble, the letters 'AKDAB' are written in white. The 'A' is stylized with a white lightning bolt or wave-like symbol to its left.

AKDAB

*Trusted Software Excellence*

---

- What is OpenGL?
- Terminology
- Qt helper classes

No part of this publication may be made available to others than the named licensee which is shown on every page by any means, electronic, mechanical, photocopying, recording or otherwise, or translated into any language, without the prior written permission of the publisher.

## Performance, Profiling and Debugging

- Introduction to Graphics Optimization
- Finding Performance Issues
  - Debugging OpenGL
  - Synchronization Mechanisms
  - Measuring
- Geometry Processing Performance
  - Geometry limitations
  - Culling
  - Instanced Rendering
  - Multi-draw Indirect
- Fragment Processing Performance
- API Usage Performance
  - API Usage Performance
  - Shaders and Shader Programs introspection
  - Uniform Buffer Objects
  - Shader Subroutines

## Advanced Buffer Usage

- Advanced Context Usage
- Optimising Memory Transfers
- Threading
- Buffering Schemes
- Image Load/Store
- Transform Feedback
  - Capturing Vertex Data
  - Setting Up Buffers
  - Using Transform Feedback
  - Feedback Rendering
  - Transform Feedback Object

## Advanced Pipeline - Geometry

- Geometry Shaders
  - OpenGL Pipeline Revisited
  - Geometry Shader Basics
  - Shader Interface Blocks
  - Removing Geometry
  - Modifying Geometry
    - Augmenting Geometry Data
    - Changing the Primitive Type
- Tessellation
  - Introduction to Tessellation
  - Tessellation in Detail
  - Bezier Curves and Patches
  - Mesh Refinement

## The OpenGL Memory Model

- Introduction
- Memory coherence
- Memory barriers
- Atomics

## Advanced Pipeline - Computing

- Shader Storage Buffer Objects
- Compute Shaders

## Modern Graphics APIs

- Modern Graphics APIs
  - Introduction
  - Simple Graphics Application
    - Initialize The Rendering Hardware
    - Building a pipeline
    - Render Loop
  - Feeding the pipeline
  - Controlling the pipeline
  - KDGpu Render Loop
  - Resources
    - Memory Buffers
    - Textures
    - Shader Resources
  - Synchronization
  - The Compute Pipeline

**Using apitrace**

**Shaders and Shader Programs  
introspection**

**Recommended Books**