УДК 004.4 https://doi.org/10.33619/2414-2948/98/27

СРАВНИТЕЛЬНЫЙ АНАЛИЗ И ПРИМЕНЕНИЕ БИБЛИОТЕК OPENGL И DIRECTX В КОНТЕКСТЕ РАЗРАБОТКИ 3D ВИЗУАЛИЗАЦИИ

©Аркабаев Н. К., *ORCID: 0009-0000-1912-2225, SPIN-код: 9304-5193,*
канд. физ.-мат. наук, Ошский государственный университет,
г. Ош, Кыргызстан, narkabaev@oshsu.kg
©Хакимов А. А., *ORCID: 0009-0005-0674-5841,*
Ошский государственный университет, г. Ош, Кыргызстан, akhakimov@oshsu.kg
©Кубанычбек уулу О., *ORCID: 0009-0007-9699-2698,*
Ошский государственный университет, г. Ош, Кыргызстан, okubanychbekuulu@oshsu.kg

COMPARATIVE ANALYSIS AND APPLICATION OF OPENGL AND DIRECTX LIBRARIES IN THE CONTEXT OF 3D VISUALIZATION

©Arkabaev N., *ORCID: 0009-0000-1912-2225, SPIN-code: 9304-5193,*
Ph.D., Osh State University, Osh, Kyrgyzstan, narkabaev@oshsu.kg
©Khakimov A., *ORCID: 0009-0005-0674-5841,*
Osh State University, Osh, Kyrgyzstan, akhakimov@oshsu.kg
©Kubanychbek uulu O., *ORCID: 0009-0007-9699-2698,*
Osh State University, Osh, Kyrgyzstan, okubanychbekuulu@oshsu.kg

Аннотация. Статья представляет собой исследование и анализ двух основных библиотек: OpenGL и DirectX и их применение в области трехмерной визуализации. В статье проводится сравнительный анализ возможностей и особенностей каждой из библиотек, рассматриваются их технические характеристики, возможности в области создания трехмерных сцен, а также их эффективность и удобство использования в различных контекстах. Статья охватывает темы, связанные с процессом инициализации библиотек, компиляцией шейдеров, созданием трехмерных объектов, настройкой окружения, а также примерами применения библиотек в реальных сценариях. Кроме того, поднимаются вопросы совместимости библиотек с различными операционными системами и архитектурой оборудования. Цель статьи — предоставить читателю глубокий обзор возможностей и особенностей библиотек OpenGL и DirectX, что позволит выбрать наиболее подходящую технологию для конкретных задач в области трехмерной визуализации. Статья включает сравнение возможностей и характеристик библиотек, анализирует их сильные стороны, эффективность и удобство использования в различных сценариях. Основное внимание уделяется техническим аспектам в работе с графикой, включая инициализацию библиотек, компиляцию шейдеров, создание трехмерных объектов и управление визуальными эффектами. Предоставить читателям глубокое понимание обеих технологий, помочь в выборе подходящей библиотеки для конкретных задач в области трехмерной визуализации и пролить свет на практические сценарии использования данных технологий.

Abstract. Article represents a study and analysis of two main libraries: OpenGL and DirectX and their application in the field of three-dimensional visualization. The article conducts a comparative analysis of the capabilities and characteristics of each library, examining their

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

technical specifications, capabilities in creating three-dimensional scenes, as well as their efficiency and usability in various contexts. It covers topics related to the process of library initialization, shader compilation, creation of three-dimensional objects, environment setup, and examples of library application in real scenarios. Additionally, it addresses compatibility issues of the libraries with different operating systems and hardware architectures. The goal of the article is to provide the reader with a comprehensive overview of the capabilities and characteristics of OpenGL and DirectX libraries, aiding in the selection of the most suitable technology for specific tasks in the field of three-dimensional visualization. It encompasses a comparison of the capabilities and characteristics of the libraries, analyzing their strengths, efficiency, and usability in various scenarios. The main focus is on technical aspects in working with graphics, including library initialization, shader compilation, creation of three-dimensional objects, and management of visual effects. Its aim is to provide readers with a deep understanding of both technologies, assisting in the selection of the appropriate library for specific tasks in three-dimensional visualization and shedding light on practical scenarios of using these technologies.

Ключевые слова: программирование трехмерной графики, рендеринг, шейдеры, трехмерная графика.

Keywords: 3D graphics programming, rendering, shaders, 3D graphics.

Развитие трехмерной визуализации играет важную роль в современной компьютерной графике и приложениях виртуальной реальности. Для создания качественных трехмерных образов и сцен необходимы мощные инструменты, способные эффективно обрабатывать и отображать графические данные. В этом контексте библиотеки OpenGL и DirectX стоят на переднем крае разработки, предоставляя разработчикам возможности для создания потрясающих трехмерных визуализаций.

OpenGL — это открытая и мощная библиотека для разработки приложений компьютерной графики, широко используемая в индустрии игр, визуализации данных, медицинских приложений и многих других областях. Ее особенностью является кроссплатформенность, позволяющая создавать приложения, совместимые с различными операционными системами.

Первоначально созданная в 1992 году, OpenGL быстро стала стандартом в индустрии 3D графики благодаря своей гибкости и мощным возможностям. Она предоставляет набор функций для работы с трехмерной графикой, включая возможности рендеринга полигонов, текстур, освещения, а также поддержку шейдеров для настройки визуальных эффектов.

Библиотека также обладает богатым функционалом для управления графическим процессором (GPU), что позволяет эффективно использовать вычислительные ресурсы и достигать высокой производительности при рендеринге сложных сцен. Ее стабильность, широкая поддержка оборудования и активное сообщество разработчиков делают ее востребованной в различных областях. OpenGL постоянно развивается, и новые версии добавляют функционал для поддержки последних графических технологий, таких как тесселяция, геометрические шейдеры и другие современные методы рендеринга, что позволяет создавать впечатляющие визуальные эффекты. Одним из ключевых преимуществ OpenGL является его открытость и свобода, позволяющая разработчикам создавать инновационные графические приложения без привязки к конкретным платформам или производителям оборудования. Это способствует широкому применению библиотеки в

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

различных проектах, от небольших инди-разработок до крупных корпоративных приложений. Все эти аспекты делают OpenGL важным инструментом для разработки приложений трехмерной графики, обеспечивая высокую производительность, гибкость и широкие возможности для воплощения творческих и инновационных идей разработчиков. Развитие OpenGL охватывает различные версии, каждая из которых добавляет новые функции и возможности. Например, OpenGL 3.x и более поздние версии сосредотачиваются на модернизации API и добавлении возможностей для программирования шейдеров, что позволяет разработчикам создавать более сложные и реалистичные эффекты. OpenGL также предоставляет широкий спектр расширений, позволяющих использовать специфические функции графического оборудования и получать более высокую производительность в зависимости от конкретных потребностей проекта. Это дает возможность оптимизировать работу с графическими ресурсами и улучшить производительность приложений. Благодаря открытой структуре и активной поддержке сообщества разработчиков, OpenGL продолжает эволюционировать, а новые версии включают в себя улучшения, направленные на оптимизацию производительности, расширение функциональности и поддержку новейших технологий в области компьютерной графики. Эта библиотека также предоставляет возможности для создания кроссплатформенных приложений, что делает ее предпочтительным выбором для разработчиков, стремящихся создать программное обеспечение, работающее на различных операционных системах, включая Windows, macOS и различные дистрибутивы Linux. Важно отметить, что успешное использование OpenGL требует от разработчиков глубокого понимания графических концепций и особенностей, данного API. Однако, благодаря своей мощности и гибкости, библиотека продолжает оставаться одним из основных инструментов для создания высококачественной трехмерной графики в широком спектре приложений и индустрий.

Библиотека DirectX представляет собой набор API, разработанный компанией Microsoft, специально ориентированный на разработку приложений, использующих графику, звук, мультимедиа и ввод-вывод на платформе Windows. Этот инструментарий охватывает широкий спектр функций для работы с трехмерной графикой, звуком и другими мультимедийными данными, обеспечивая доступ к аппаратному ускорению графики и звука. Одной из ключевых особенностей DirectX является его тесная интеграция с операционной системой Windows, что позволяет разработчикам создавать высокопроизводительные приложения, оптимизированные специально под эту платформу. Это может быть особенно полезно для игровой индустрии, где производительность игровых приложений играет решающую роль. DirectX включает различные компоненты, включая DirectX Graphics (Direct3D), который является основной частью для разработки трехмерной графики. Direct3D обеспечивает разработчикам доступ к аппаратным возможностям видеокарты для рендеринга трехмерных объектов, текстур и освещения. Кроме того, DirectX включает DirectX Audio для работы с звуком, DirectX Media для мультимедийных приложений и DirectX Input для управления вводом, таким как клавиатура и мышь. Все эти компоненты предоставляют разработчикам обширный набор инструментов для создания полноценных мультимедийных приложений. DirectX постоянно обновляется, и новые версии вносят улучшения в производительность, функциональность и поддержку новых технологий, таких как поддержка последних версий графических карт и новейшие методы рендеринга.

Также стоит отметить, что в отличие от OpenGL, DirectX ограничен по своей совместимости и доступности только для платформы Windows, что может быть ограничивающим фактором для разработки кроссплатформенных приложений. Однако,

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

благодаря своей тесной интеграции с Windows, DirectX остается важным инструментом для создания высокопроизводительных графических и мультимедийных приложений на этой платформе.

Существует множество научных работ в данной области. Рассмотрим следующих примеров таких работ:

1. “OpenGL and DirectX” (Karl Hillesland, 2013. https://doi.org/10.1145/2542266.2542276). В данной статье автор обсуждает сходства и различия между API OpenGL и DirectX. А также, рассмотрены подходы, напоминающих то, что делается в DirectX, таких как неизменяемые текстуры. Это не только облегчит переход между API, но и сосредоточит внимание на эффективном использовании. Для DirectX было сосредоточено на DirectX 11, включая части 11.1, где можно объединится с OpenGL и OpenGL ES в своем подходе.

2. “Interactive Graphics Applications Development: An Effect Framework for Directx 11” [1-3] В этой работе авторы обсуждают, что в среде .NET отсутствуют библиотеки для продвинутого графического вывода и поэтому описывается реализация интерфейсов графических библиотек, позволяющая взаимодействовать с .NET и библиотек OpenGL, DirectX и Visualization Toolkit. В данной статье отсутствует сравнение графических библиотек, а также сравнение среды .NET с другими средами;

3. “Modern OpenGL: its design and evolution” [4, 5]. Рассмотрим текстурное отображение и буфер трафарета, присутствовали только на самом дорогом графическом оборудовании, то теперь эти функции полностью распространены на ПК и даже доступны в нескольких портативных устройствах. За долгое время исходная фиксированная система состояний OpenGL превратилась в сложный поток данных, включающий несколько программно-настраиваемых этапов и производительность OpenGL увеличилась с 100x до более 1,000x во многих важных операциях с графикой .

Основные функциональности библиотеки OpenGL

Одной из ключевых возможностей OpenGL является управление графическими примитивами, такими как точки, линии и треугольники, позволяющее создавать разнообразные объекты и сцены. Библиотека обладает гибкостью в управлении отображением благодаря шейдерам, которые определяют, как именно отображать графику, обрабатывая геометрические и цветовые данные. OpenGL также обеспечивает возможность прикрепления текстур к объектам для добавления деталей и характеристик. Он поддерживает разнообразные типы текстур, расширяя возможности визуализации. Для хранения графических данных и информации о пикселях OpenGL использует различные буферы, такие как буферы кадров и глубины, обеспечивая точное и эффективное хранение информации.

Библиотека также обладает функциями трансформации, что позволяет размещать объекты в пространстве и изменять их вид. Освещение и теневание - еще одна важная характеристика OpenGL, обеспечивающая создание реалистичных эффектов света и теней. С помощью OpenGL можно создавать сложные формы и эффекты, обрабатывать вершины и преобразовывать матрицы для создания впечатляющей визуализации. Библиотека предоставляет широкий спектр инструментов, делая ее незаменимой для разработки графических приложений и игр на различных платформах. Каждая часть OpenGL играет важную роль в процессе создания визуальных эффектов. Например, возможность управления текстурами добавляет деталей и реализма объектам, позволяя им выглядеть более реалистично. Библиотека также обеспечивает возможность определения освещения, включая рассеянное, бликовое и фоновое освещение, что позволяет создавать потрясающие визуальные эффекты и играть со светом и тенью в сценах. Трансформация вершин и матриц

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

позволяет разработчикам изменять их положение, вращение и размер, что позволяет создавать динамичные эффекты и анимацию объектов. OpenGL также предоставляет возможность управлять множеством аспектов отображения, включая мультисэмплинг для улучшения качества изображения и уменьшения артефактов на краях объектов. Благодаря своей гибкости и мощным функциям, OpenGL является одной из основных библиотек для создания впечатляющих и высокопроизводительных графических приложений и игр. Рассмотрим на примере использования OpenGL для отрисовки треугольника в окне при помощи библиотеки GLFW на языке C++:

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <iostream>
// Код вершинного шейдера
const char *vertexShaderSource = R"(
   #version 330 core
   layout (location = 0) in vec3 aPos;
   void main() {
      gl_Position = vec4(aPos.x, aPos.y, aPos.z, 1.0);
   }
)";
// Код фрагментного шейдера
const char *fragmentShaderSource = R"(
   #version 330 core
   out vec4 FragColor;
   void main() {
      FragColor = vec4(1.0f, 0.5f, 0.2f, 1.0f);
   }
)";
int main() {
   // Инициализация GLFW
   glfwInit();
   glfwWindowHint(GLFW_CONTEXT_VERSION_MAJOR, 3);
   glfwWindowHint(GLFW_CONTEXT_VERSION_MINOR, 3);
   glfwWindowHint(GLFW_OPENGL_PROFILE, GLFW_OPENGL_CORE_PROFILE);
   // Создание окна
   GLFWwindow* window = glfwCreateWindow(800, 600, "OpenGL Example", NULL,
NULL);
   if (window == NULL) {
      std::cout << "Failed to create GLFW window" << std::endl;
      glfwTerminate();
      return -1;
   }
   glfwMakeContextCurrent(window);
   // Инициализация GLEW
   glewExperimental = GL_TRUE;
   if (glewInit() != GLEW_OK) {
```

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

```
    std::cout << "Failed to initialize GLEW" << std::endl;
    return -1;
}
// Компиляция вершинного шейдера
unsigned int vertexShader;
vertexShader = glCreateShader(GL_VERTEX_SHADER);
glShaderSource(vertexShader, 1, &vertexShaderSource, NULL);
glCompileShader(vertexShader);
// Компиляция фрагментного шейдера
unsigned int fragmentShader;
fragmentShader = glCreateShader(GL_FRAGMENT_SHADER);
glShaderSource(fragmentShader, 1, &fragmentShaderSource, NULL);
glCompileShader(fragmentShader);
// Создание шейдерной программы
unsigned int shaderProgram;
shaderProgram = glCreateProgram();
glAttachShader(shaderProgram, vertexShader);
glAttachShader(shaderProgram, fragmentShader);
glLinkProgram(shaderProgram);
// Очистка шейдерных объектов
glDeleteShader(vertexShader);
glDeleteShader(fragmentShader);
// Определение координат треугольника
float vertices[] = {
    -0.5f, -0.5f, 0.0f,
     0.5f, -0.5f, 0.0f,
     0.0f,  0.5f, 0.0f
};
// Создание буфера вершин
unsigned int VBO, VAO;
glGenVertexArrays(1, &VAO);
glGenBuffers(1, &VBO);
// Связывание буферов и атрибутов
glBindVertexArray(VAO);
glBindBuffer(GL_ARRAY_BUFFER, VBO);
glBufferData(GL_ARRAY_BUFFER, sizeof(vertices), vertices, GL_STATIC_DRAW);
glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 3 * sizeof(float), (void*)0);
glEnableVertexAttribArray(0);
// Рендеринг цикла
while (!glfwWindowShouldClose(window)) {
    glClearColor(0.2f, 0.3f, 0.3f, 1.0f);
    glClear(GL_COLOR_BUFFER_BIT);
    glUseProgram(shaderProgram);
    glBindVertexArray(VAO);
    glDrawArrays(GL_TRIANGLES, 0, 3);
    glfwSwapBuffers(window);
```

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

```
        glfwPollEvents();
    }
    // Очистка ресурсов
    glDeleteVertexArrays(1, &VAO);
    glDeleteBuffers(1, &VBO);
    glfwTerminate();
    return 0;
}
```

Данный код создает окно при помощи GLFW, инициализирует OpenGL, компилирует и использует вершинный и фрагментный шейдеры для отрисовки треугольника на экране.

Основные функциональности библиотеки DirectX

DirectX — это комплексная платформа для разработки мультимедийных приложений, широко используемая в индустрии компьютерных игр и графических приложений под операционные системы Windows. Эта мощная библиотека предоставляет разработчикам доступ к различным инструментам для работы с графикой, звуком, вводом и многими другими аспектами компьютерных приложений. Основные функциональности DirectX включают в себя графические примитивы и возможности рендеринга, обеспечивающие создание сложных сцен и эффектов. Библиотека поддерживает шейдеры, позволяющие программировать графику на уровне аппаратного обеспечения и создавать визуальные эффекты, такие как освещение, тени и текстурирование. Также DirectX обеспечивает инструменты для работы с текстурами, что позволяет улучшить детализацию объектов и придать им реалистичный вид. Он поддерживает работу с буферами данных для эффективного хранения графической информации. Кроме того, библиотека включает в себя возможности для работы со звуком, вводом и устройствами управления, обеспечивая разработчикам все необходимые инструменты для создания мультимедийных приложений. DirectX обеспечивает высокую совместимость с различными версиями Windows и поддерживает широкий спектр аппаратного оборудования, что делает его популярным выбором для разработки игр и мультимедийных приложений под Windows. DirectX представляет собой набор API (интерфейсов прикладного программирования), которые обеспечивают доступ к аппаратным возможностям компьютера, специально направленным на графику и звук. Этот комплекс инструментов позволяет разработчикам создавать потрясающие визуальные эффекты, обеспечивая высокую степень гибкости и производительности. Одной из ключевых особенностей DirectX является его способность управлять графическим рендерингом. Он предоставляет доступ к созданию и управлению различными графическими примитивами — от простых точек и линий до сложных трехмерных объектов. Поддержка шейдеров позволяет программировать графические эффекты на глубоком уровне, обеспечивая гибкость и возможность создания высококачественных визуальных отображений. Библиотека также включает в себя возможности работы с звуком. Разработчики могут создавать звуковые эффекты, управлять аудиопотоками и проигрывать звук в реальном времени, что делает DirectX полезным инструментом для создания мультимедийных приложений и игр с качественным звуковым сопровождением. Дополнительно, DirectX предоставляет инструменты для взаимодействия с устройствами ввода, такими как клавиатура, мышь и геймпады, что позволяет управлять приложениями и играми с помощью различных устройств. Совместимость DirectX с различными версиями Windows и широкий спектр поддерживаемого аппаратного

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

оборудования делают его популярным средством для разработки высококачественных и производительных приложений и игр под Windows.

Рассмотрим на примере использования DirectX для отрисовки треугольника на экране и обрабатывает основной цикл сообщений Windows для отрисовки на языке C++:

```
#include <Windows.h>
#include <d3d11.h>
#pragma comment(lib, "d3d11.lib")
LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
    // Создание окна
    WNDCLASSEX wc = { sizeof(WNDCLASSEX), CS_CLASSDC, WindowProc, 0L, 0L, GetModuleHandle(NULL), NULL, NULL, NULL, NULL, "DirectXExample", NULL };
    RegisterClassEx(&wc);
    HWND hwnd = CreateWindow(wc.lpszClassName, "DirectX Example", WS_OVERLAPPEDWINDOW, 100, 100, 800, 600, NULL, NULL, wc.hInstance, NULL);
    // Инициализация DirectX
    ID3D11Device* dev;
    ID3D11DeviceContext* devcon;
    IDXGISwapChain* swapchain;
    D3D_FEATURE_LEVEL level;
    D3D11CreateDeviceAndSwapChain(NULL, D3D_DRIVER_TYPE_HARDWARE, NULL, 0, NULL, 0, D3D11_SDK_VERSION, NULL, &swapchain, &dev, &level, &devcon);
    // Создание буфера заднего буфера и отображаемой поверхности
    ID3D11Texture2D* pBackBuffer;
    swapchain->GetBuffer(0, __uuidof(ID3D11Texture2D), (LPVOID*)&pBackBuffer);
    dev->CreateRenderTargetView(pBackBuffer, NULL, NULL);
    pBackBuffer->Release();
    // Настройка объектов DirectX для отрисовки треугольника
    devcon->OMSetRenderTargets(1, NULL, NULL);
    devcon->IASetPrimitiveTopology(D3D11_PRIMITIVE_TOPOLOGY_TRIANGLELIST);
    // Координаты вершин треугольника
    float vertices[] = {
       0.0f, 0.5f, 0.0f,
       0.5f, -0.5f, 0.0f,
       -0.5f, -0.5f, 0.0f
    };
    // Создание буфера вершин
    D3D11_BUFFER_DESC bd = { 0 };
    bd.Usage = D3D11_USAGE_DEFAULT;
    bd.ByteWidth = sizeof(vertices);
    bd.BindFlags = D3D11_BIND_VERTEX_BUFFER;
    bd.CPUAccessFlags = 0;
    D3D11_SUBRESOURCE_DATA initData = { vertices, 0, 0 };
```

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

```
        ID3D11Buffer* pVBuffer;
        dev->CreateBuffer(&bd, &initData, &pVBuffer);
        // Основной цикл обработки сообщений и отрисовки
        ShowWindow(hwnd, SW_SHOWDEFAULT);
        MSG msg;
        while (true) {
           if (PeekMessage(&msg, NULL, 0, 0, PM_REMOVE)) {
              if (msg.message == WM_QUIT) break;
              TranslateMessage(&msg);
              DispatchMessage(&msg);
           }
           // Очистка экрана
           float clearColor[4] = { 0.0f, 0.0f, 0.0f, 1.0f };
           devcon->ClearRenderTargetView(NULL, clearColor);
           // Установка буфера вершин
           UINT stride = sizeof(float) * 3;
           UINT offset = 0;
           devcon->IASetVertexBuffers(0, 1, &pVBuffer, &stride, &offset);
           // Отрисовка треугольника
           devcon->Draw(3, 0);
           // Переключение заднего и переднего буферов
           swapchain->Present(0, 0);
        }
        // Очистка ресурсов DirectX
        pVBuffer->Release();
        swapchain->Release();
        dev->Release();
        devcon->Release();
        return 0;
  }
  LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam,
  LPARAM lParam) {
     if (uMsg == WM_DESTROY) {
        PostQuitMessage(0);
        return 0;
     }
     return DefWindowProc(hwnd, uMsg, wParam, lParam);
  }
```

Этот код создает окно приложения, инициализирует DirectX 11 для отображения треугольника на экране и обрабатывает основной цикл сообщений Windows для отрисовки. Он представляет базовый шаблон для использования DirectX в приложении на C++.

Применения библиотек OpenGL и DirectX в области трехмерной визуализации

Библиотеки OpenGL и DirectX являются основными инструментами для трехмерной визуализации и графики в индустрии разработки игр, моделирования, визуализации данных, архитектурного проектирования и многих других областей. Вот несколько примеров их применения:

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

1. Разработка компьютерных игр: Обе библиотеки являются основой для создания графической составляющей игр. Они предоставляют функции для отображения трехмерных объектов, управления текстурами, освещением, тенями и другими визуальными эффектами, что позволяет создавать реалистичные игровые миры.

2. Медицинская визуализация: В области медицины OpenGL и DirectX используются для создания трехмерных моделей органов, симуляций процедур, визуализации медицинских данных (например, снимков МРТ или КТ) и разработки программ для обучения медицинских специалистов.

3. Научная визуализация: Библиотеки используются для визуализации научных данных, таких как моделирование молекулярных структур, астрономические данные, геологические и географические карты, что помогает ученым лучше понимать и исследовать сложные данные.

4. Архитектурное проектирование: OpenGL и DirectX используются для создания визуализаций архитектурных проектов, позволяя архитекторам и дизайнерам создавать трехмерные модели зданий, интерьеров и ландшафтов для более наглядного представления проектов заказчикам.

5. Образование: В образовательных целях библиотеки применяются для создания интерактивных обучающих программ, трехмерных моделей и симуляций для помощи в обучении студентов различных дисциплин, связанных с трехмерной визуализацией.

6. Виртуальная реальность и дополненная реальность: Обе библиотеки широко используются в разработке VR и AR приложений для создания интенсивных трехмерных сред, в которых пользователи могут взаимодействовать с виртуальными объектами.

Эти библиотеки предоставляют разработчикам инструменты для создания высококачественной и интерактивной трехмерной графики в различных областях, поддерживая широкий спектр аппаратного обеспечения и предоставляя возможности оптимизации и создания сложных визуальных эффектов.

В ходе сравнительного анализа библиотек OpenGL и DirectX были рассмотрены их ключевые технические характеристики. Обе библиотеки предоставляют мощные инструменты для работы с трехмерной графикой, однако были выявлены некоторые различия в эффективности их использования в различных сценариях. OpenGL демонстрировала более высокую гибкость в работе с различными операционными системами, в то время как DirectX обеспечивала более эффективную интеграцию с Windows-ориентированными платформами.

Анализ технических аспектов показал, что обе библиотеки предоставляют широкий функционал для создания трехмерных сцен и управления визуальными эффектами. Однако, наблюдались различия в процессе инициализации и компиляции шейдеров, где одна библиотека проявляла более интуитивный подход к настройке, в то время как другая предлагала большую гибкость в манипуляции визуальными эффектами.

На примерах реальных сценариев была показана эффективность и гибкость обеих библиотек в разработке трехмерных визуализаций. OpenGL проявила свою универсальность и применимость в широком спектре платформ, в то время как DirectX отлично справлялась с интеграцией на платформах, основанных на Windows.

Исходя из проведенного анализа, можно сделать вывод, что обе библиотеки имеют свои сильные стороны и подходят для различных сценариев разработки трехмерной визуализации. Рекомендуется выбор библиотеки в зависимости от специфики проекта и целевой платформы, учитывая их технические особенности и уровень интеграции.

Развитие библиотек OpenGL и DirectX остается важным направлением для индустрии

Т. 10. №1. 2024
https://doi.org/10.33619/2414-2948/98

трехмерной визуализации. Оптимизация работы с графикой, поддержка новейших технологий и улучшение производительности являются ключевыми аспектами, которые могут изменить облик будущих трехмерных проектов. Постоянное развитие и обновление этих библиотек обеспечит разработчикам новые инструменты и возможности для создания увлекательных и высококачественных трехмерных миров.

Список литературы:

1. Shakaev V., Shabalina O., Kamaev V. Interactive Graphics Applications Development: An Effect Framework for Directx 11 // World Applied Sciences Journal. 2013. V. 24. №24. P. 165-170.

2. Zink J., Pettineo M., Hoxley J. Practical rendering and computation with Direct3D 11. – CRC Press, 2016.

3. Вольф Д. OpenGL 4. Язык шейдеров. Книга рецептов. М.: ДМК Пресс, 2015. 368 с.

4. Чеботарева Е. Н., Аксёнов С. В. Построение 3D-моделей объектов с помощью OpenGL // Молодежь и современные информационные технологии: Сборник трудов XIII Международной. 2015. С. 194. EDN: VSYQUZ

5. Худайберганов Т. Р., Адинаев Х. С. Библиотеки OpenGL и directx для программирования трехмерной графики // Современная техника и технологии. 2017. №5. С. 27-27. EDN: YRCJOX

References:

1. Shakaev, V., Shabalina, O., & Kamaev, V. (2013). Interactive Graphics Applications Development: An Effect Framework for Directx 11. *World Applied Sciences Journal*, *24*(24), 165-170.

2. Zink, J., Pettineo, M., & Hoxley, J. (2016). *Practical rendering and computation with Direct3D 11*. CRC Press.

3. Vol'f, D. (2015). OpenGL 4. Yazyk sheiderov. Kniga retseptov. Moscow. (in Russian).

4. Chebotareva, E. N., & Aksenov, S. V. (2015). Postroenie 3D-modelei ob"ektov s pomoshch'yu OpenGL. *Molodezh' i sovremennye informatsionnye tekhnologii. Sbornik trudov XIII Mezhdunarodnoi, 194.* (in Russian).

5. Khudaiberganov, T. R., & Adinaev, Kh. S. (2017). Biblioteki OpenGL i directx dlya programmirovaniya trekhmernoi grafiki. *Sovremennaya tekhnika i tekhnologii,* (5), 27-27. (in Russian).

Работа поступила
в редакцию 26.11.2023 г.

Принята к публикации
08.12.2023 г.

Ссылка для цитирования:

Аркабаев Н. К., Хакимов А. А., Кубанычбек уулу О. Сравнительный анализ и применение библиотек OpenGL и DirectX в контексте разработки 3D визуализации // Бюллетень науки и практики. 2024. Т. 10. №1. С. 235-245. https://doi.org/10.33619/2414-2948/98/27

Cite as (APA):

Arkabaev, N., Khakimov, A., & Kubanychbek uulu, O. (2024). Comparative Analysis and Application of OpenGL and DirectX Libraries in the Context of 3D Visualization. *Bulletin of Science and Practice, 10*(1), 235-245. (in Russian). https://doi.org/10.33619/2414-2948/98/27