Вычислительная геометрия и алгоритмы компьютерной графики

Лекция №2: Графический конвейер. Структура графического приложения

к.ф.-м.н.
Рябинин Константин Валентинович

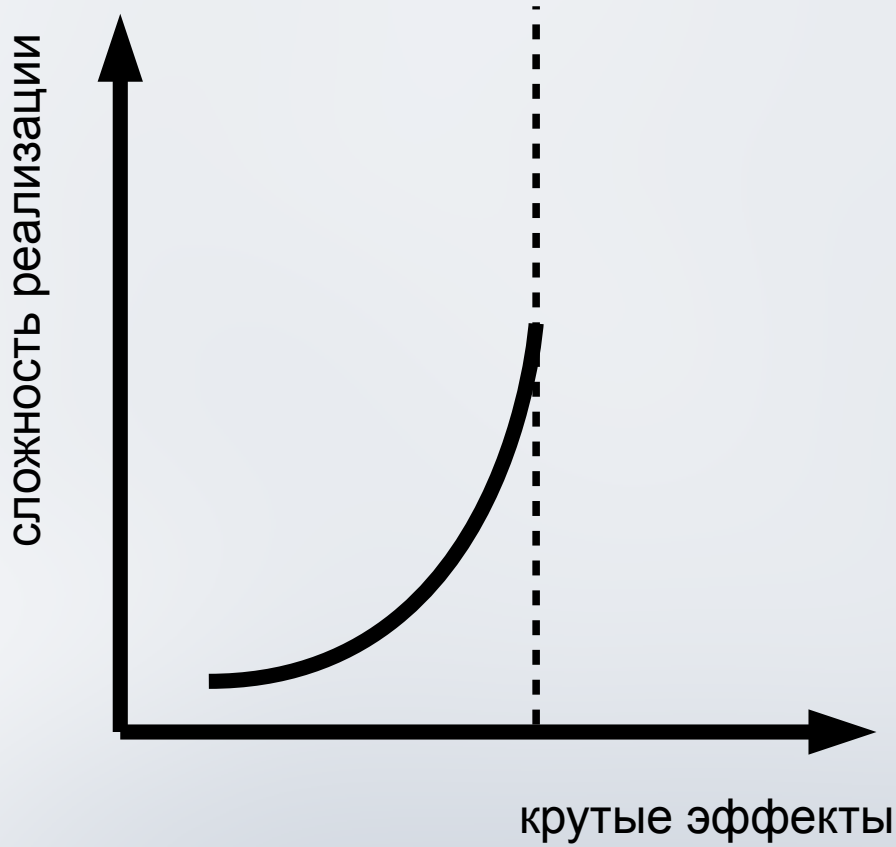
e-mail: kostya.ryabinin@gmail.com

Пермь, 2016

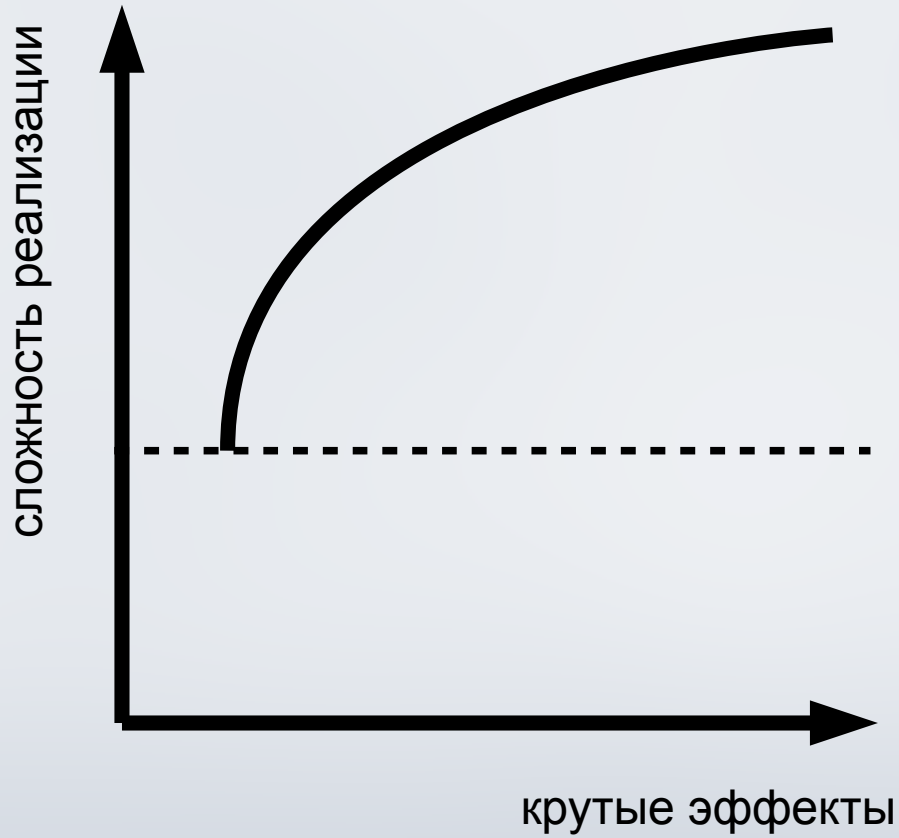
Рубим хвосты :)

- Грубо говоря, есть два класса версий OpenGL

OpenGL 1.x – «Old School»



OpenGL 3.3+ – «Brand New»



Структура объектов

- **Объекты в полигональной 3D- графике представляют собой поверхности, аппроксимированные множеством многоугольников**

Структура объектов

- Атомарная управляемая единица геометрии – вершина
- Вершина имеет набор атрибутов (типа float), интерпретация которых, вообще говоря, лежит на программисте:
  - Координаты в пространстве
  - Координаты нормали
  - Координаты текстуры
  - Цвет
  - ...
- Вершины объединяются в примитивы, чаще всего – треугольники

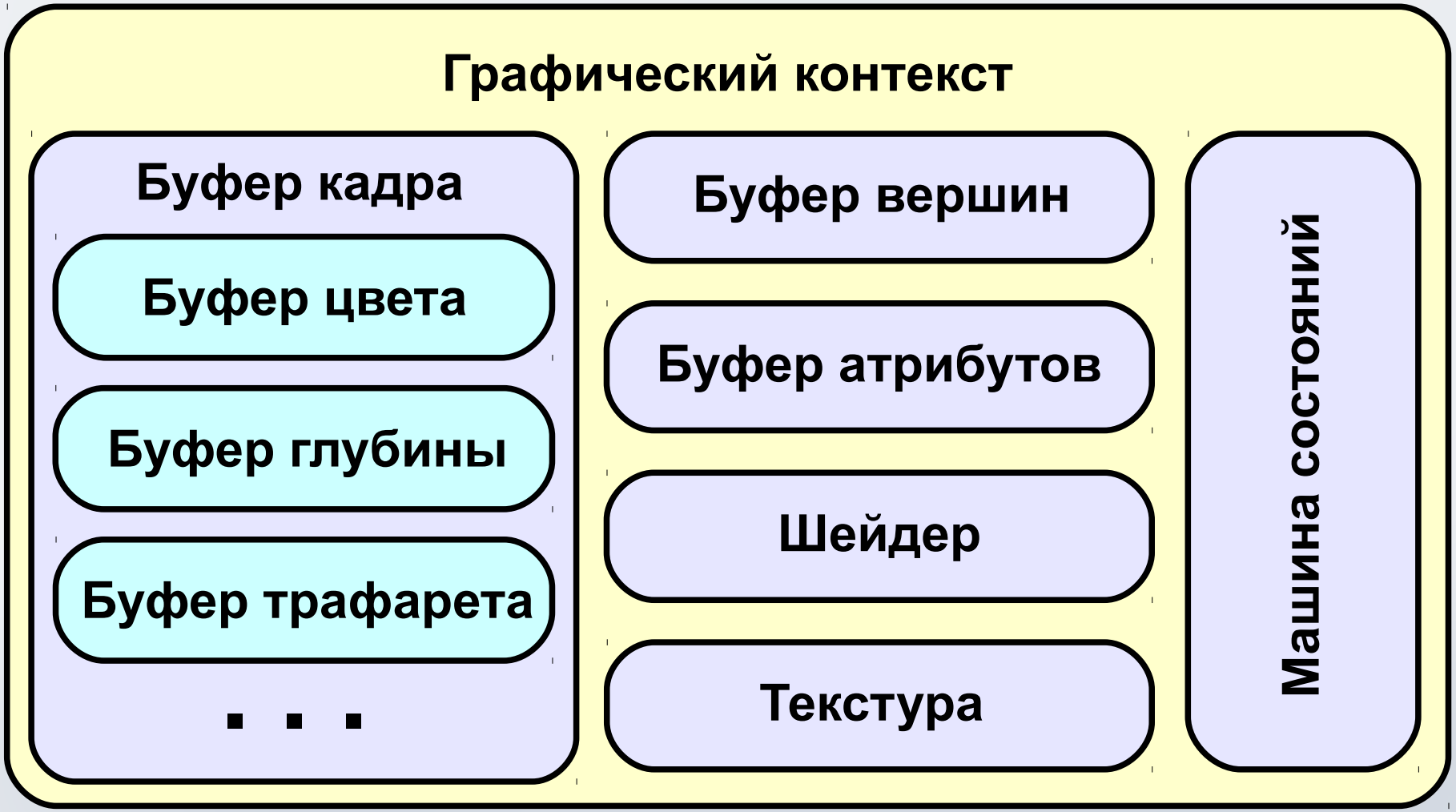
Построение картинки

- Сцена состоит из объектов
- Объекты состоят из примитивов
- Примитивы состоят из вершин
- Вершины имеют 3D-координаты
- Чтобы получить картинку, нужно
  - Разместить объекты в пространстве
  - Спроецировать их примитивы на плоскость экрана
  - Подобрать для проекций пиксели (растеризация)
  - Раскрасить пиксели

Графический конвейер

Графический контекст

- Большая часть графических API имеют следующую концептуальную организацию данных для визуализации:



Графический контекст

- Поверхность рендеринга – объект в составе отображающей (оконной) системы, при помощи которого происходит демонстрация результатов визуализации сцены
- Поверхность рендеринга неразрывна связана с графическим контекстом и является системно-зависимой
- Обычно поверхность рендеринга включает в себя механизм двойной буферизации для обеспечения атомарности обновления экрана
- Смена активного и видимого буферов обычно аппаратно синхронизируется с обновлением монитора (вертикальная синхронизация)

Графическое приложение

1. Инициализировать графический контекст и поверхность рендеринга
2. Создать графические ресурсы
3. Построить сцену
4. Пока не дана команда завершения, повторять:
    - 4.1. Изменить состояние сцены в соответствии с текущим состоянием приложения
    - 4.2. Отправить данные на графический конвейер
    - 4.3. Обновить поверхность рендеринга
5. Удалить графические ресурсы
6. Удалить графический контекст и поверхность рендеринга

Анимация

Анимация – это вдыхание жизни в сцену :)

Анимация – это последовательное отображение кадров с различным содержанием
(конструктивно)

Программно-аппаратная поддержка анимации:
механизм двойной буферизации

→ **Строго говоря, любое свойство объекта может быть анимировано**

Важное требование к анимации:
сохранение межкадрового соответствия
(***frame-to-frame coherence***)

Скорость анимации

Анимация по таймеру:

- Идея: запуск некоторого таймера (должен поддерживаться системой), который в наперёд заданные моменты времени вызывает функцию обновления состояния
- Способ имеет право на существование лишь в исключительных случаях
- Таймер лучше использовать как триггер анимации, а не как её «драйвер»

Непрерывная анимация:

- Идея: обновление состояния происходит с максимально возможной скоростью, а величина изменения вычисляется на основе желаемой и фактической скорости

Формула линейной анимации

$$p^{(i)} = p^{(i-1)} + \Delta_p \Delta_t$$

p – значение анимируемого параметра
i – номер итерации графического цикла
Δ_p – желаемое изменение параметра p за секунду
Δ_t – время в секундах, прошедшее с момента предыдущего изменения p

Буфер кадра

- **Буфер** – структура данных для хранения результатов работы графического конвейера
- За некоторыми буферами чётко закреплено их назначение, другие же используются для сохранения произвольной информации
- **Буфер цвета** – буфер для хранения итогового изображения
- Буферы произвольного назначения называются **целями рендеринга**
- **Буфер кадра** – структура данных (часть графического контекста), агрегирующая буферы