ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Кафедра «Информационно-вычислительные системы»

Презентация по дисциплине «Геометрическое моделирование и компьютерная графика»

Профессор каф.ИВС, д.т.н. Косников Ю.Н.

Векторная, растровая и фрактальная графика

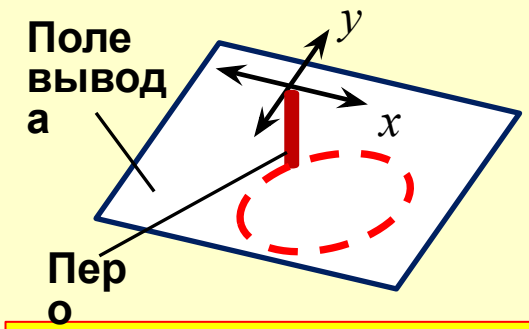
Изображения формируются на информационном поле (поле вывода) графических устройств. Это экран, табло, лист бумаги,… Изображения формирует отображающий орган – электронный луч, перо, узел возбуждения ячейки экрана,… Изображения описываются и формируются по разным законам. По этому признаку выделяют векторную и растровую графику. Иногда к ним добавляют фрактальную графику.

В векторной графике изображение составляется из фрагментов, имеющих математическое описание. Исходными данными являются параметры фрагментов.

Например, для окружности исходные данные: идентификатор, местоположение центра, радиус, цвет, тип линии, … - $Circle, x_c, y_c, R, Color, \ldots$

По идентификатору выбирается программная процедура. Она получает исходные данные, вычисляет промежуточные точки фрагмента и подает их на управление координатами отображающего органа. Она же задает тип и цвет линии

$$x = Rcos(2\pi t/T) + x_c,$$
$$y = Rsin(2\pi t/T) + y_c.$$



Особенность: отображение «на-лету»

Достоинства векторной графики

1. Компактное описание изображений
2. Простота редактирования фрагментов (они остаются независимыми)
3. Сохранение высокого качества при редактировании

Недостаток: невысокий уровень реалистичности

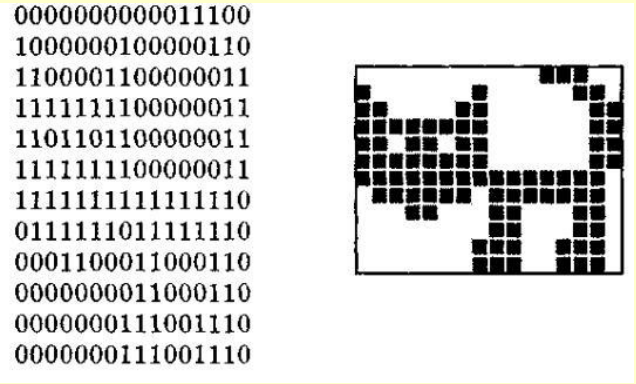
Векторная, растровая и фрактальная графика

В растровой графике информационное поле - набор точек. На экране это пиксели. Они упорядочены в строки и столбцы. Возникает растровая решетка. Точку можно поставить только в узле решетки. Изображение составляется из окрашенных пикселей в процессе адресации (возбуждения) узлов растровой решетки.

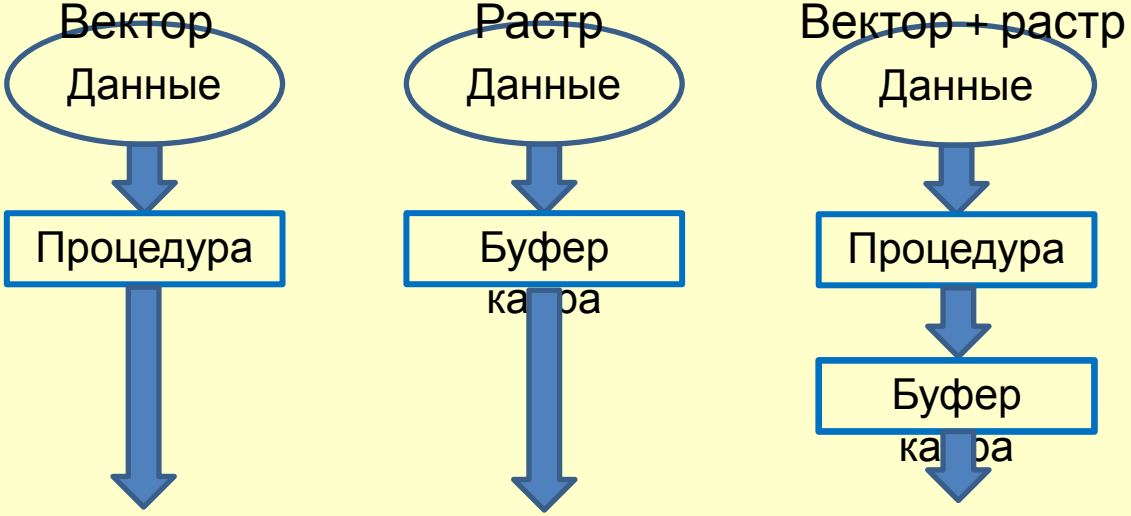
Достоинства растровой графики:

1. Реалистические изображения
2. Стандартные форматы хранения данных

Недостаток: сложность редактирования



Принцип формирования изображения в векторной, растровой и векторно-растровой графике



Особенность: коды пикселей вычисляются по одному закону, а выводятся по другому. Нужна промежуточная память – буфер кадра

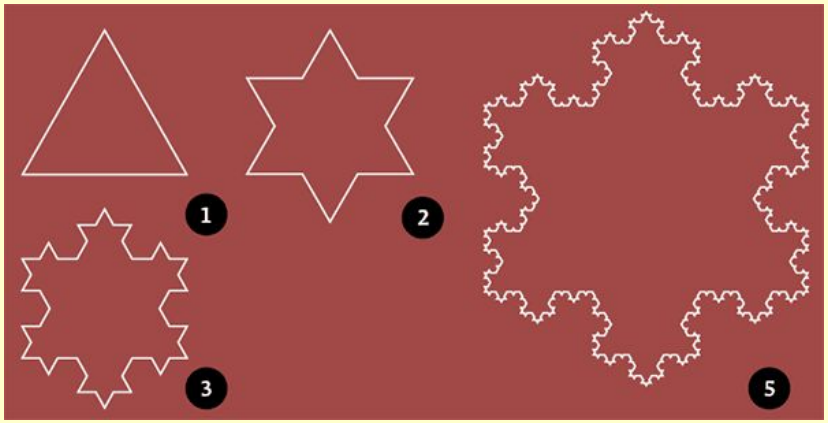
Векторная, растровая и фрактальная графика

Фрактальная графика является векторно-растровой. Название -- от латинского ***fractus*** – раздробленный, состоящий из фрагментов. Автор термина – Бенуа Мандельброт.

Фрактал – самоподобная геометрическая фигура, т.е. ее части подобны целому.

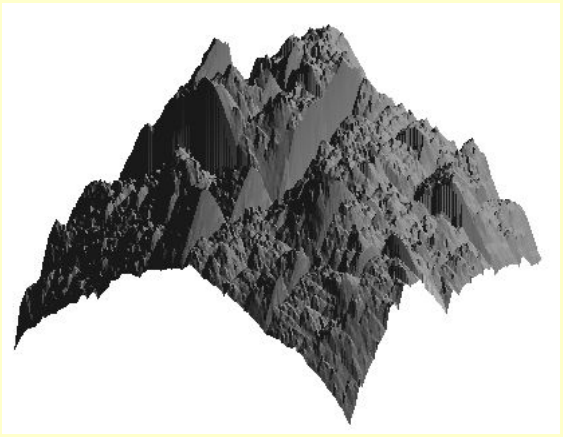
Описание фрактала – математическое выражение простой формы или компактный алгоритм. Например, описание фрактального множества Мандельброта имеет вид

$$z_{k+1} = z_k^2 + c, \qquad k = 0,1,2,\dots, \qquad z_0 = c = x + jy$$



Снежинка Коха

Папоротник Барнсли



Метод смещения средней точки

Вычисления повторяются многократно, в идеале – до бесконечности, а реально – до достижения желаемого качества изображений. Это – отличительная особенность фрактальной графики.

В компьютерной графике фракталы используются для получения изображений узоров, текстур и геометрии природных объектов.

Этапы процесса отображения объектов сцены

Наименование этапа	Применяемые средства
Предварительно:	
Описание объектов и ввод описаний в компьютер	Создание информационных моделей объектов
В режиме РВ	
Определение местоположения объектов	Вычисление координат объектов с учетом их динамики. Геометрические преобразования.
Определение видимости объектов, примитивов	Отсечение объектов (метод оболочек, алгоритм Коэна - Сазерленда). Отсечение примитивов (алгоритм Сазерленда – Ходжмена)
Учет заслонения объектов	Алгоритм буфера глубины
Наложение текстур	Применение функций отображения. Алгоритмы префильтрации (применение субпикселей)
Растеризация	Алгоритмы базовой графики

Разновидности информационных моделей объектов отображения

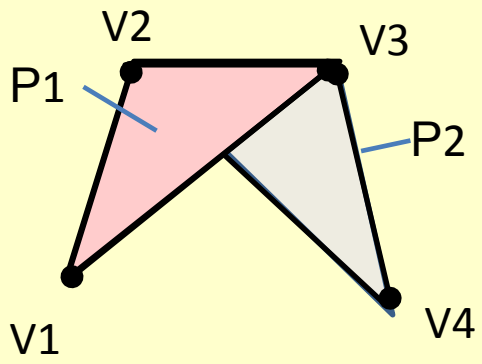
Наименование моделей	Форма представления объекта	Форма описания для ввода в компьютер	Достоинства	Недостатки
Точечные	Множество точек, принадлежащих контуру или поверхности объекта	Пары декартовых координат для каждой точки	Универсальность	Большие затраты ресурсов компьютера
Каркасные	Набор отрезков линий, принадлежащих контуру или поверхности объекта	Координаты точек начала и конца каждого отрезка, уравнения отрезков	Компактность, наличие информации о геометрической форме объекта	Пониженная реалистичность
Поверхностные	Набор поверхностей, принадлежащих поверхности объекта	Координаты характерных точек поверхности и указание на закон их соединения, уравнения поверхностей	Высокая реалистичность	Отсутствие информации о внутреннем строении объекта
Твердотельные, применяются для 3d-моделирования	Множество пространственных точек, принадлежащих поверх-ности и внутреннему объему	Уравнения геометрических тел, множество элементов объема - вокселей	Представление информации о внутреннем строении объекта	Большие затраты ресурсов компьютера

Иерархическая полигональная модель

Поверхность объекта состоит из полигонов, полигон задается вершинами, вершина – тройкой координат. Вариант описания (модели) объекта, показанного на рисунке

$$O = (P_1,P_2,) = (((x_1,y_1,z_1),(x_2,y_2,z_2),(x_3,y_3,z_3)),\ ((x_2,y_2,z_2),(x_3,y_3,z_3),(x_4,y_4,z_4))$$

Такой вариант имеет недостатки: повышенные затраты памяти на хранение вершин, повышенные затраты времени на вычерчивание ребер, возможность «разбегания» вершин.



Многоуровневая иерархическая модель

Нижний уровень – список вершин

$$V = (v_1, v_2, \dots, vi, \dots, vL), \qquad v_i=(x_i,y_i,z_i)$$

Средний уровень – список ребер

$$\mathrm{E} = (e_1, e_2, \dots, e_j, \dots, e_M), \qquad e_j=(pv_{j1},pv_{j2},f_{j1},f_{j2})$$

Верхний уровень – список полигонов

$$O = (P_1, P_2, \dots, P_k, \dots, P_N), \qquad P_k=(pe_{k1},pe_{k2},\dots,pe_{kR})$$

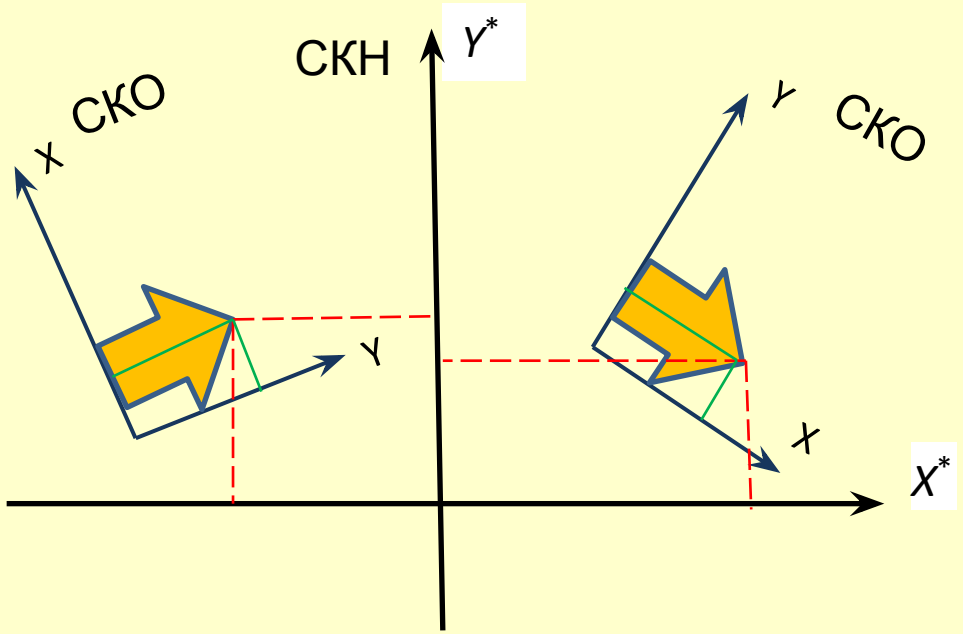
$v_1 = (x_1,y_1,z_1),\ v_2 = (x_2,y_1,z_1),\dots$

$e_1=(pv_1,pv_2,1,0),\ e_2=(pv_2,pv_3,1,0),\dots$

$P_1=(pe_1,pe_2,pe_3),\ P_2=(pe_2,pe_4,pe_5),$

$O = (P_1, P_2)$

Назначение геометрических преобразований



Объект описывается в своей системе координат – СКО, а отображается в системе координат наблюдателя - СКН. Расстановка и динамика объектов представляется как расстановка и динамика СКО в СКН. Для описания расстановки и динамики применяются аффинные геометрические преобразования

Математическое описание аффинных преобразований

$$x^* = t_{11}x + t_{12}y + x_0^*,$$
$$y^* = t_{21}x + t_{22}y + y_0^*,$$

Общая форма координатного описания аффинных преобразований

x, y – координаты точки в «старой», а x^*, y^* – в «новой» системе координат, например в СКО и СКН

$$x^* = x + x_0^*,$$
$$y^* = y + y_0^*,$$

Частное преобразование сдвига

x_0^*, y_0^* – координаты начала СКО в СКН

$$x^* = \cos\varphi \cdot x - \sin\varphi \cdot y,$$
$$y^* = \sin\varphi \cdot x + \cos\varphi \cdot y,$$

Частное преобразование поворота

ϕ – угол поворота СКО в СКН

$$x^* = k_x \cdot x, \quad y^* = k_y \cdot y,$$

Частное преобразование масштабирования

$k_x, k_y \neq 0$ – коэффициенты преобразования по координатным осям

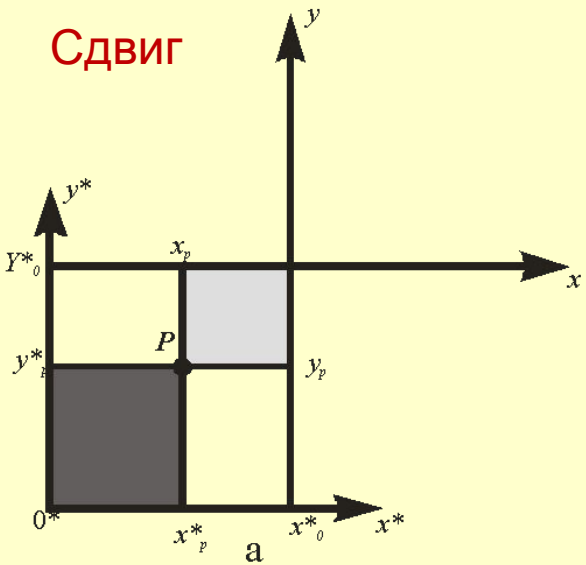
$$x^* = \cos 2\alpha \cdot x + \sin 2\alpha \cdot y,$$
$$y^* = \sin 2\alpha \cdot x - \cos 2\alpha \cdot y.$$

Частное преобразование симметрии

α - угол наклона оси симметрии, проходящей через начало координат

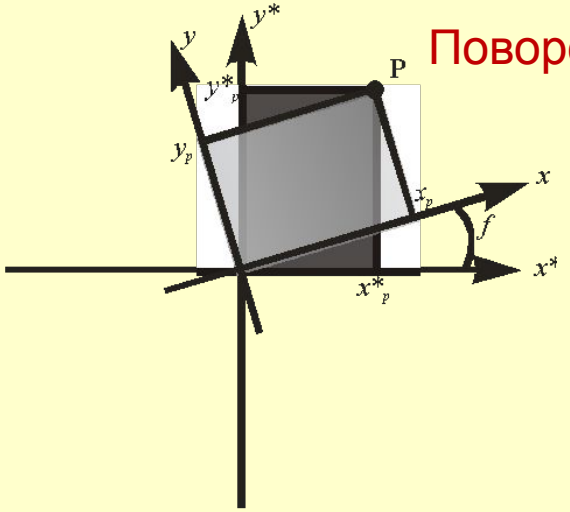
Графическая интерпретация частных аффинных преобразований

Сдвиг



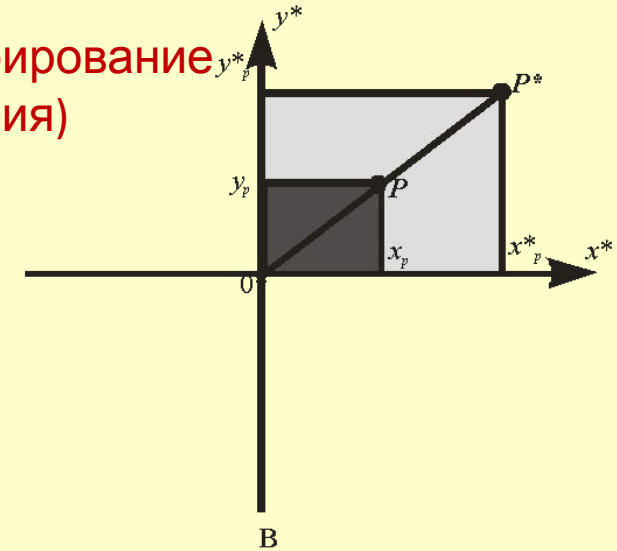
а

Поворот



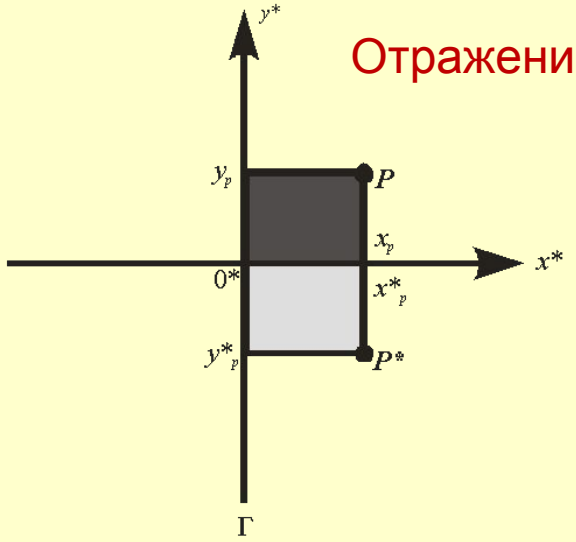
б

Масштабирование (симметрия)



В

Отражение



Г

Немного о матрицах

Математическая матрица – это таблица с числами. Числа упорядочены в строки и столбцы. Пример: **A**= $\begin{vmatrix} 3 & -2 & 0 \\ 1 & 7 & -3 \end{vmatrix}$. В этой матрице **A** 2 строки и 3 столбца, ее размер M×N, где M=2, N=3. Матрица размера 1×N называется матрицей-строкой (вектором-строкой), а матрица размера M×1 называется матрицей столбцом (вектором-столбцом). Матрица, у которой M=N называется квадратной. С матрицами выполняют различные действия: транспонирование, сложение, умножение на число, перемножение.

При умножении на число каждый элемент матрицы умножается на это число

Транспонирование – замена строк столбцами, а столбцов строками. Пример: $A^T = \begin{vmatrix} 3 & 1 \\ -2 & 7 \\ 0 & -3 \end{vmatrix}$ - это транспонированная матрица **A.**

Перемножение матриц

Матрица-строка M1=$|a1 \quad a2 \quad a3|$ умножается на матрицу-столбец M2=$\begin{vmatrix} b1 \\ b2 \\ b3 \end{vmatrix}$ по такому правилу: первый элемент M1 умножается на первый элемент M2 **плюс** второй элемент M1 – на второй элемент M2 **плюс** третий элемент M1 на третий элемент M2 (и так далее, если размер больше 3).

M1·**M2**=a1·b1+a2·b2+a3·b3.

Ясно, что число элементов столбца должно быть равно числу элементов строки.

Матрица-строка **S** размера 1×M умножается на матрицу **A** размера M×N по следующему правилу. Сначала строка умножается на первый столбец **A** по описанному правилу, в результате получается первый элемент результирующей матрицы. Затем строка умножается на второй столбец **A,** в результате получается второй элемент результирующей матрицы. И так далее до окончания столбцов **A.**

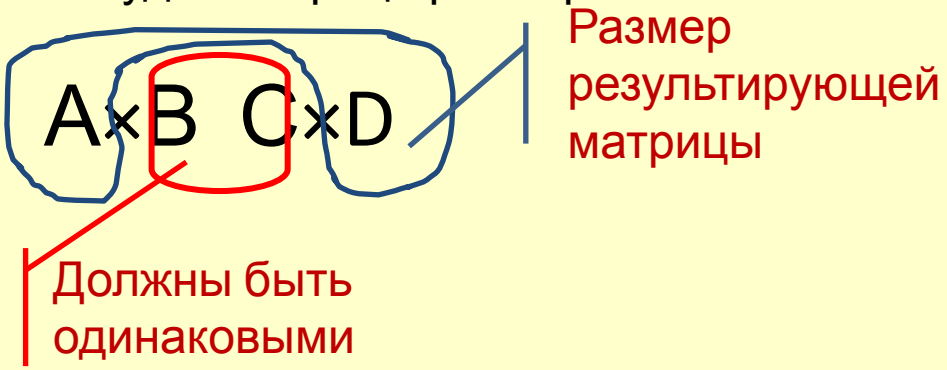
Например: $|a1 \quad a2 \quad a3| \cdot \begin{vmatrix} b1 & c1 & d1 & e1 \\ b2 & c2 & d2 & e2 \\ b3 & c3 & d3 & e3 \end{vmatrix} =$

=|(a1·b1+a2·b2+a3·b3) (a1·c1+a2·c2+a3·c3) (a1·d1+a2·d2+a3·d3) (a1·e1+a2·e2+a3·e3)|

Перемножение возможно, если размер строки **S** и размер столбца матрицы **A** совпадают.

Перемножение матриц

Учитывая правила перемножения, можно сформулировать признак возможности перемножения матриц: если одна матрица имеет размер A×B, а вторая – размер C×D, то для возможности перемножения нужно, чтобы B=C. Тогда результатом будет матрица размером A×D.



Например, при перемножении матриц, имеющих размеры 5×3 и 3×4 получится матрица размера 5×4, матриц, имеющих размеры 1×3 и 3×3 – матрица размера 1×3, матриц, имеющих размеры 1×3 и 3×1 – матрица размера 1×1, т.е. просто число (скаляр).

В общем случае матрица S умножается на матрицу R по следующему правилу. Чтобы получить элемент результирующей матрицы, стоящий на пересечении i-ой строки и j-го столбца, нужно умножить i-ю строку S на j-й столбец R.

Однородные координаты

Однородным представлением *n*-мерного объекта в математике, в общем случае, называют его представление в (*n*+1)-мерном пространстве, полученное добавлением

еще одной координаты – скалярного множителя. Однородные координаты на плоскости определяются следующим образом.

Пусть на плоскости в аффинной системе координат задана точка *P* с координатами

(*x*, *y*). Однородными координатами этой точки называется любая тройка одновременно

не равных нулю чисел [w_1 w_2 w_3], связанных с ко

соотношениями: $\frac{w_1}{w_3} = x, \qquad \frac{w_2}{w_3} = y.$

Безразмерное число w_3 называется скалярным множителем.

Геометрическое преобразование в матричной форме с использованием однородного представления точек имеет вид **K* = K · T,**

где K=$|x\ y\ 1|$ – матрица координат исходной точки, K* =$|x^*\ y^*\ 1|$ - матрица координат точки после преобразования, *T* – матрица преобразования.

Матрицы частных аффинных преобразований имеют следующий вид

сдвиг

$$TR = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_0^* & y_0^* & 1 \end{vmatrix},$$

поворот

$$RT = \begin{vmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{vmatrix},$$

масштабирование

$$SC = \begin{vmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{vmatrix},$$

отражение

$$RF = \begin{vmatrix} \cos 2\alpha & \sin 2\alpha & 0 \\ \sin 2\alpha & -\cos 2\alpha & 0 \\ 0 & 0 & 1 \end{vmatrix}.$$

Матричное описание аффинных преобразований в однородных координатах

Теперь аффинное преобразование в общем виде будет выглядеть следующим образом:

$$\begin{vmatrix} x^* & y^* & 1 \end{vmatrix} = \begin{vmatrix} x & y & 1 \end{vmatrix} \cdot \begin{vmatrix} t_{11} & t_{21} & 0 \\ t_{12} & t_{22} & 0 \\ x_0^* & y_0^* & 1 \end{vmatrix}$$

Перемножи в матрицы, получим ⇒

$$x^* = t_{11}x + t_{12}y + x_0^*,$$
$$y^* = t_{21}x + t_{22}y + y_0^*,$$
$$1 = 1$$

Любое частное аффинное преобразование в матричной форме имеет вид

$$\begin{vmatrix} x^* & y^* & 1 \end{vmatrix} = \begin{vmatrix} x & y & 1 \end{vmatrix} \cdot \mathbf{T},$$ где $\mathbf{T}$ – матрица преобразования

Матрицы преобразований

сдвига

$$TR = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_0^* & y_0^* & 1 \end{vmatrix},$$

поворота

$$RT = \begin{vmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{vmatrix},$$

масштабирования

$$SC = \begin{vmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{vmatrix},$$

отражения

$$RF = \begin{vmatrix} \cos 2\alpha & \sin 2\alpha & 0 \\ \sin 2\alpha & -\cos 2\alpha & 0 \\ 0 & 0 & 1 \end{vmatrix}.$$

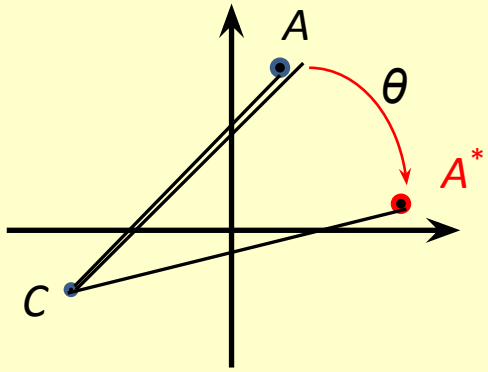
Проверка соответствия для поворота

$$\begin{vmatrix} x^* & y^* & 1 \end{vmatrix} = \begin{vmatrix} x & y & 1 \end{vmatrix} \cdot \begin{vmatrix} \cos\varphi & \sin\varphi & 0 \\ -\sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{vmatrix} \Rightarrow$$

$$x^* = \cos\varphi \cdot x - \sin\varphi \cdot y,$$
$$y^* = \sin\varphi \cdot x + \cos\varphi \cdot y,$$
$$1 = 1$$

Суперпозиция матричных аффинных преобразований

Пример. Пусть требуется описать поворот точки $A(x_A, y_A)$ объекта на угол θ по часовой стрелке вокруг произвольного центра C с координатами x_c, y_c. Результат – точка $A^*(x_A^*, y_A^*)$



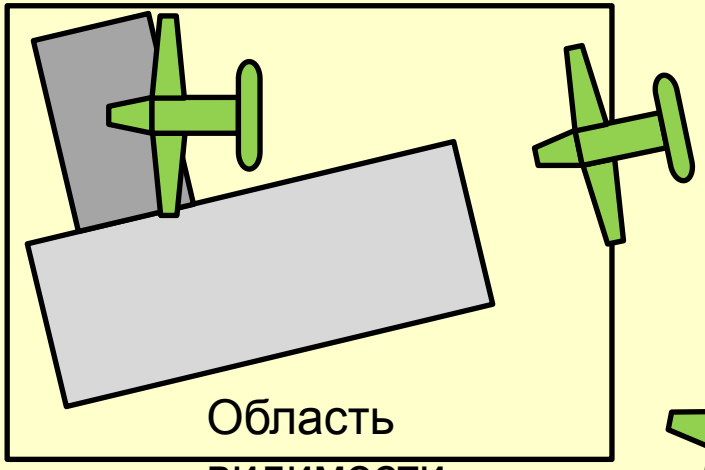
$$\begin{vmatrix} x^* & y^* & 1 \end{vmatrix} = \begin{vmatrix} x & y & 1 \end{vmatrix} \cdot \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_C & -y_C & 1 \end{vmatrix} \cdot \begin{vmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{vmatrix} \cdot \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_C & y_C & 1 \end{vmatrix},$$

Использовав обозначения матриц, получим:

$$\begin{vmatrix} x^* & y^* & 1 \end{vmatrix} = \begin{vmatrix} x & y & 1 \end{vmatrix} \cdot \mathbf{TR1} \cdot \mathbf{RT} \cdot \mathbf{TR2},$$

Произведение матриц, выделенное жирным, называется суперпозицией

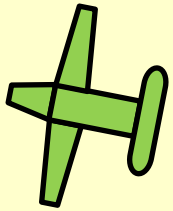
Удаление на изображении невидимых частей объектов



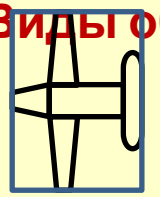
Видимый, невидимый и частично видимый объекты

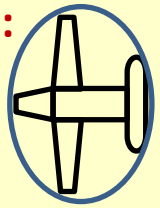
Метод оболочек

Оболочка – геометрическая фигура простой формы, которая описана вокруг объекта. Определение видимости объекта заменяется определением видимости оболочки.



Виды оболочек 2D-объектов:



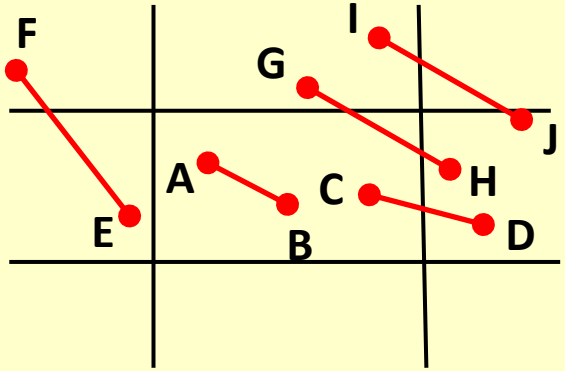


Прямоугольник Многоугольник Эллипс (bounding box)

Определение видимости оболочки с помощью алгоритма Коэна-Сазерленда

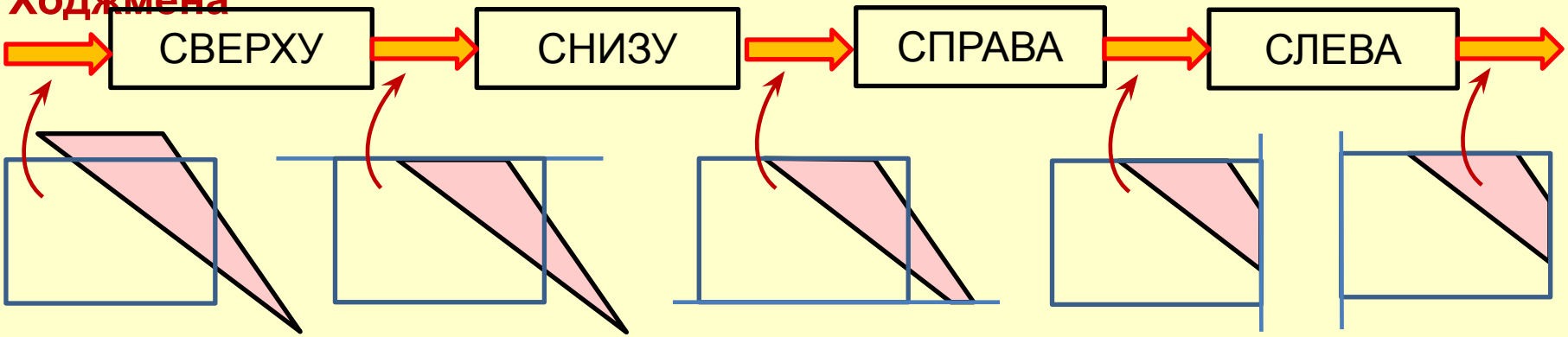
1001	1000	1010
0001	0000	0010
0101	0100	0110

1. **Если коды обеих вершин ребра нулевые, ребро видимо (AB)**
2. **Если нет, то логическое пересечение кодов вершин. Если результат НЕнулевой, отрезок невидим (EF)**
3. **Для остальных – анализ точек пересечения ребер с границами (IJ – невидимо; CD,GH – видимы частично)**



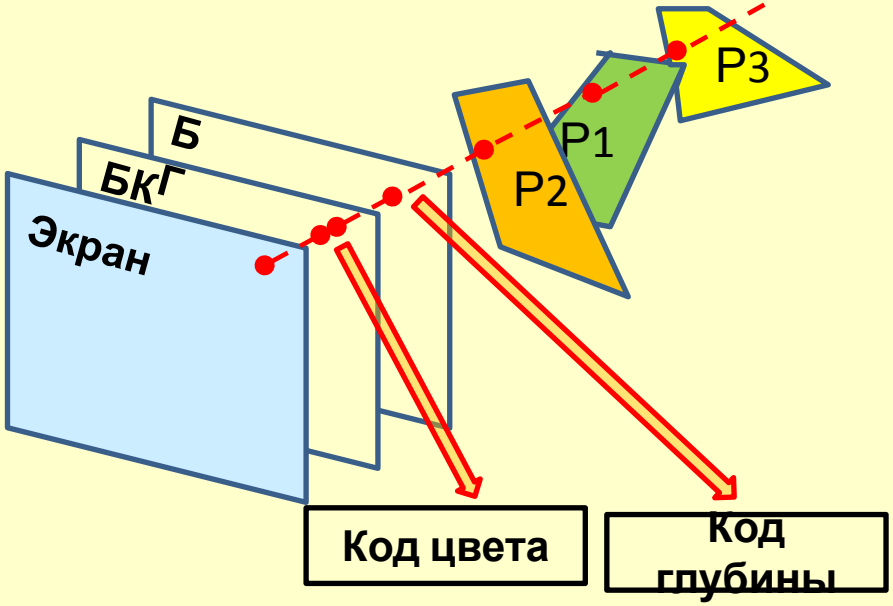
Удаление на изображении невидимых частей объектов

Конвейер отсечения полигонов на основе алгоритма Сазерленда-Ходжмена



Алгоритм буфера глубины (Z-буфера)

1. Занесение в БГ кода максимального удаления, в БК – кода цвета фона
2. Перебор примитивов и их точек
3. Глубина очередной точки, претендующей на засветку пикселя с некоторым адресом, сравнивается с кодом в БГ по этому адресу
4. Если очередная точка ближе точки, занесенной в буферы БГ и БК, их



Наложение проективных текстур

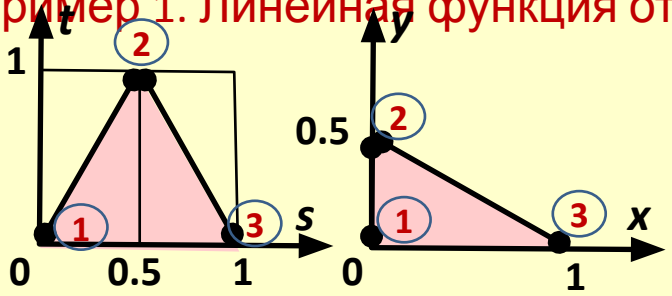
Текстура – это характерный рисунок объекта, узор на его поверхности. Текстура располагается в своей системе координат *s,t* (СКТ) и накладывается на объект, размещенный в его системе координат *x,y* (СКО).

Наложение текстуры (текстурирование) – это установление соответствия между элементами текстуры – **текселями** (texel) – и элементами растра – **пикселями** (pixel).

Другими словами, это установление соответствия между координатами *s,t* и координатами *x,y*, т.е. между координатами точек узора и координатами геометрии объекта. Описание такого соответствия – функция отображения: $s=f(x,y)$, $t=f(x,y)$.

Функции отображения бывают линейными и нелинейными

Пример 1. Линейная функция отображения



$s=ax+by+c$, $t=dx+ey+f$,

s=x+0.5y, t= y

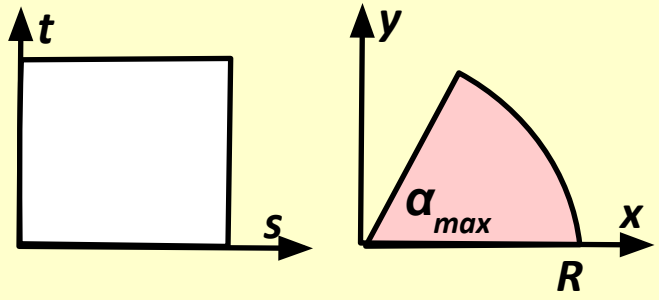
1) $0=a\cdot0+b\cdot0+c \Rightarrow c=0$
2) $0=d\cdot0+e\cdot0+f \Rightarrow f=0$
3) $0.5=a\cdot0+b\cdot1+c \Rightarrow b=0.5$
4) $1=d\cdot0+e\cdot1+f \Rightarrow e=1$
5) $1=a\cdot1+b\cdot0+c \Rightarrow a=1$
6) $0=d\cdot1+e\cdot0+f \Rightarrow d=0$

Применение найденных коэффициентов дает искаженную текстуру. А как не исказить?

Подставив в полученные выражения координаты (*x,y*) любой точки объекта, найдем для нее *s,t*

Наложение проективных текстур

Пример 2. Нелинейная функция отображения



Описание сектора

в полярных координатах

ρ=0..R

α=0..α max

⇒

в декартовых координатах

x=ρcosα

y=ρsinα

Правила текстурирования

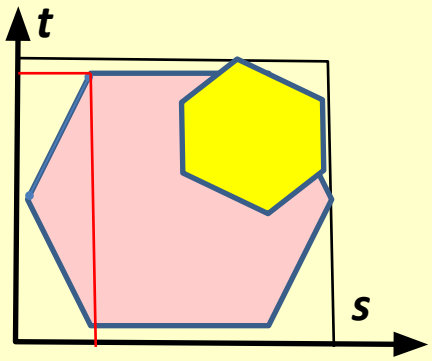
1) Нужно задать закон наложения текстуры, т.е. определить, какой кусок текстуры и как будет наложен на объект. Для этого, например, объект проецируется в СКТ
2) Нужно выбрать размеры проекции объекта в СКТ
3) Нужно определить координаты текстуры для характерных точек объекта
4) Числовое соответствие использовать в процедуре текстурирования

Функции отображения реализуются автоматически, а соответствие координат характерных точек объекта и текстуры задает

Сопоставим $s \to x$, $t \to y$, тогда нелинейные функции отображения:

$$s = \frac{\rho cos\alpha}{R}, \quad t = \frac{\rho sin\alpha}{R}$$

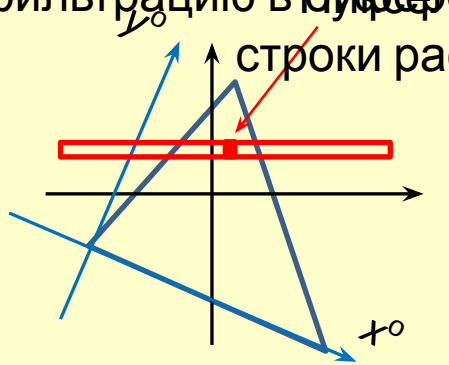
Для каждой пары ρ, α находятся *x,y*

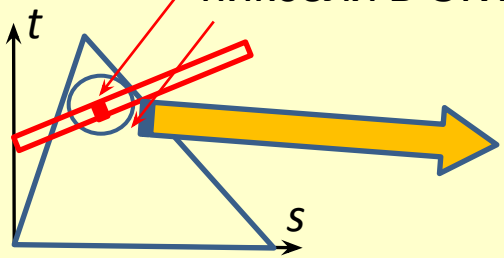


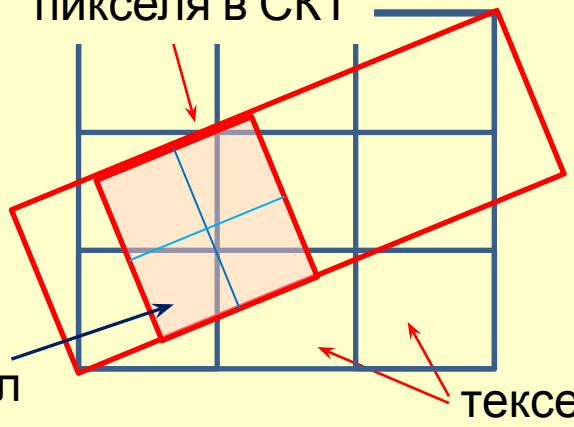
Префильтрация и постфильтрация изображений

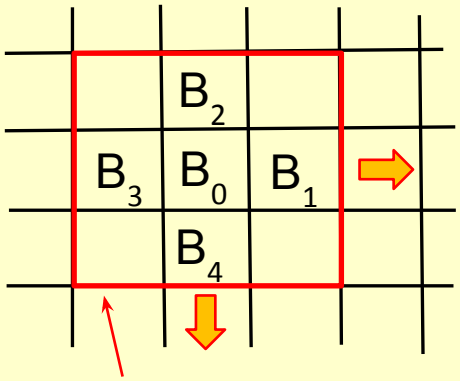
Для улучшения качества изображения применяют фильтрацию. Это усреднение цветояркости пикселей, сглаживание, «размазывание» изображения.

Различают префильтрацию - фильтрацию во время текстурирования, до занесения кодов засветки в буфер кадра (БК), и пост фильтрацию в буфере кадра.







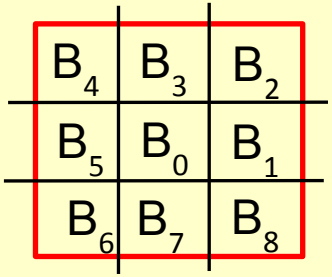


Окно 3×3 ячейки БК

Постфильтрация

Методом 4-соседей:

$$B = \frac{1}{2}B_0 + \frac{1}{8}(B_1 + B_2 + B_3 + B_4)$$

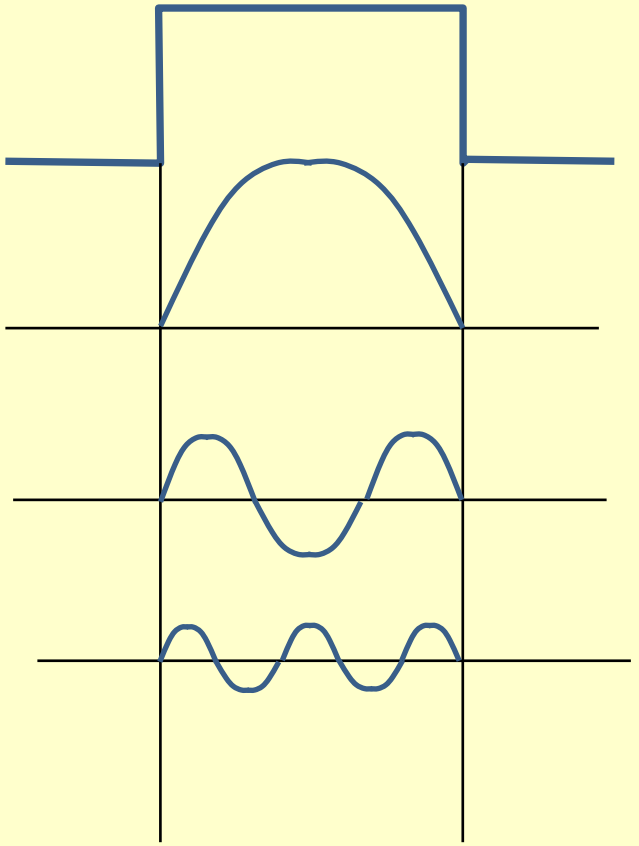


Методом 8-соседей:

$$B = \frac{1}{4}B_0 + \frac{1}{8}(B_1 + B_3 + B_5 + B_7) + \frac{1}{16}(B_2 + B_4 + B_6 + B_8)$$

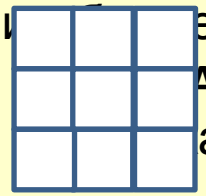
Результат постфильтрации – устранение ступенчатости

Иллюстрация к методу сжатия JPEG



Сжатие (компрессия) изображений и текстур

Запись изображения в память компьютера (в виде кодов цветояркости R-G-B) можно выполнять последовательно – пиксель за пикселем. Возникает цепочка троек кодов R-G-B. Это формат bmp. Буквенная запись показанного примера изображения:



AABBCCC. Сжатия нет. Объем памяти для хранения большой.

атие изображений и текстур выполняется разными методами.

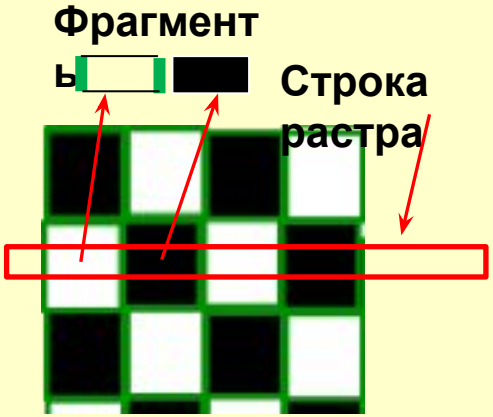
Сжатие изображений

Методы сжатия делятся на две группы: сжатие с потерями и сжатие без потерь (качества, информации). Первыми появились методы сжатия без потерь: метод Хаффмана (Huffman), метод группового кодирования (run length encoding – RLE), метод Лемпеля – Зива - Уэлча (Lempel-Ziv-Welch – LZW).

Метод Хаффмана создан для сжатия текста, применяется в графике как вспомогательный. Для часто встречающихся в тексте символов используются короткие коды. Сжатие идет в 2 этапа: 1) анализ частоты повторения символов в сжимаемом тексте, составление таблицы кодирования; 2) замена символов текста их кодами из таблицы. Декомпрессия текста идет в обратном порядке. Коэффициент сжатия – до 8.

Метод RLE. Последовательность одинаковых по цветояркости пикселей заменяется двумя байтами: первый – количество одинаковых пикселей, второй – код цветояркости. Запись для изображения-примера: **4A2B3C**. Коэффициент сжатия зависит от изображения и может достигать 32.

Сжатие (компрессия) изображений и текстур



Метод LZW. В строке изображения отыскиваются повторя-ющиеся фрагменты, расположение которых (адреса в строке) фиксируется в таблице. Исходное изображение заменяется кодами этих фрагментов и адресами их расположения. При декомпрессии фрагменты расставля-ются в строке по их адресам. Метод асимметричный. Коэффициент сжатия 5 – 7, но

Позднее появился эффективный метод сжатия JPEG (**Joint Photographic Experts Group**). Метод – с потерями, он отбрасывает в сигнале изображения до 50% информации о цвете (кодирует ее с меньшей разрядностью), а яркостный сигнал кодирует с большей разрядностью. Метод заменяет сходные оттенки цвета одним оттенком.

Метод JPEG реализуется по этапам

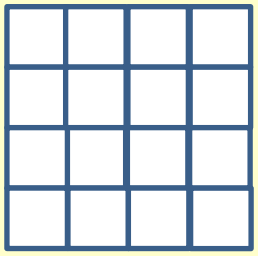
1. Переход от цветовой модели R-G-B к цветовой модели YUV (или - YCrCb).
2. Разбиение изображения на квадраты размером 8×8 пикселей.
3. Для каждого квадрата – переход от цветояркости пикселей к скорости (частоте) изменения цветояркости. Результат – числа, описывающие параметры частотных сигналов. Для этого применяется дискретное косинусное преобразование – ДКП. Компоненты с высокими частотами отбрасываются.
4. Полученные числа сжимаются методом RLE, а результат – еще методом Хаффмана.

При декомпрессии порядок действий обратный. Метод несимметричный. Коэффициент сжатия – до 100 и может задаваться.

Сжатие (компрессия) изображений и текстур

Метод **сжатия текстур** S3TC (Texture Compression фирмы S3)

Особенность работы с текстурами – произвольный доступ. Поэтому при сжатии текстуры разбиваются на фрагменты и сжимаются по этим фрагментам. При сжатии методом S3TC текстура разбивается на блоки размером 4×4 текселя.



Находятся тексели с максимальной (B_{max}) и минимальной (B_{min}) цветояркостью. Они кодируются явно 16-разрядным кодом. Между B_{min} и B_{max} предусматривается (разрешается) еще два уровня цветояркости, итого – 4 уровня: B_{min}, $\left(\frac{2}{3}B_{min} + \frac{1}{3}B_{max}\right)$, $\left(\frac{1}{3}B_{min} + \frac{2}{3}B_{max}\right)$, B_{max}

10	10	10	11
10	01	01	01
00	00	00	00
00	10	01	01

Они соответствуют интерполяционной формуле $\boldsymbol{B = K_1 B_{min} + K_2 B_{max}}$, где K_1 K_2 – интерполяционные коэффициенты: 0, $\frac{1}{3}$, $\frac{2}{3}$, 1. Достаточно знать один коэффициент, второй является дополнением до 1. Для каждого уровня цветояркости коэффициент кодируется двухбитным кодом.

В итоге, сжатый образ фрагмента – два 16-разрядных и шестнадцать 2-разрядных кодов, объем 64 бита. Образ без сжатия – шестнадцать 24-разрядных кодов, объем 384 бит. Коэффициент сжатия равен 6.

Декомпрессия выполняется схемными интерполяторами по приведенной формуле. Восстанавливаются не исходные цветояркости, а разрешенные уровни.

Метод S3TC – метод сжатия с потерями.

Программное обеспечение компьютерной графики

Программное обеспечение графики имеет иерархическую структуру

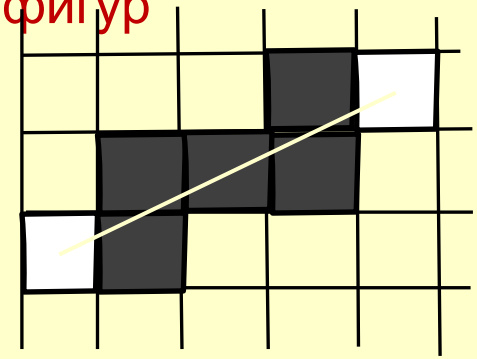
Верхний уровень — **Графические ППП** → Не связаны с тех.средствами. Зависят от предм. области. Работают с объектами,

Средний уровень — **Графические библиотеки** → Не связаны с тех.средствами. Не зависят от предметной области. Работают с примитивами, объектами, сценами

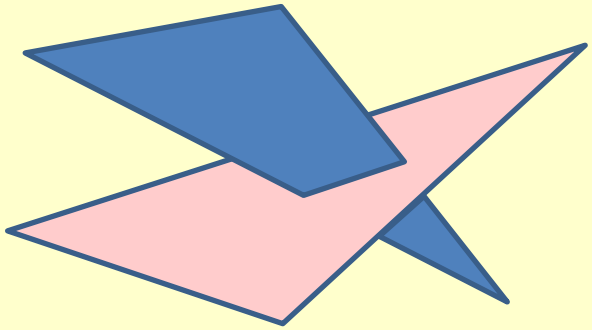
Нижний уровень — **Базовая графика** → Связана с тех.средствами. Не зависит от предметной области. Работает с элементами отображения. Встроена в прикладное ПО

Алгоритмы базовой графики

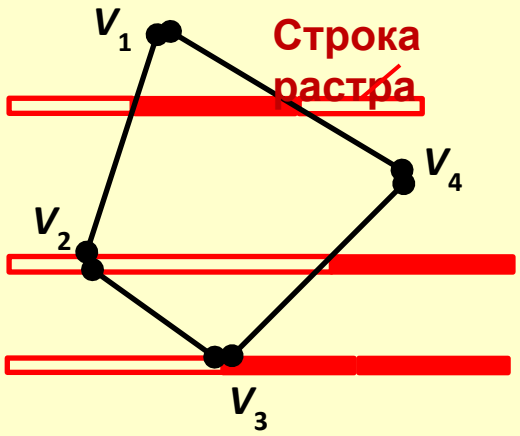
Растрирование линий фигур



Ограничение примитивов



Заполнение



Программное обеспечение компьютерной графики

Графические библиотеки (на примере OpenGL)

Причины появления: повторение процедур построения часто встречающихся графических объектов в различных программах. Около 20 лет назад появилось полтора десятка библиотек. Сейчас по факту – три: OpenGL, DirectX, Vulkan.

Содержание: это набор макропроцедур – команд. Они выполняют конкретные графические операции, параметры которых настраиваются.

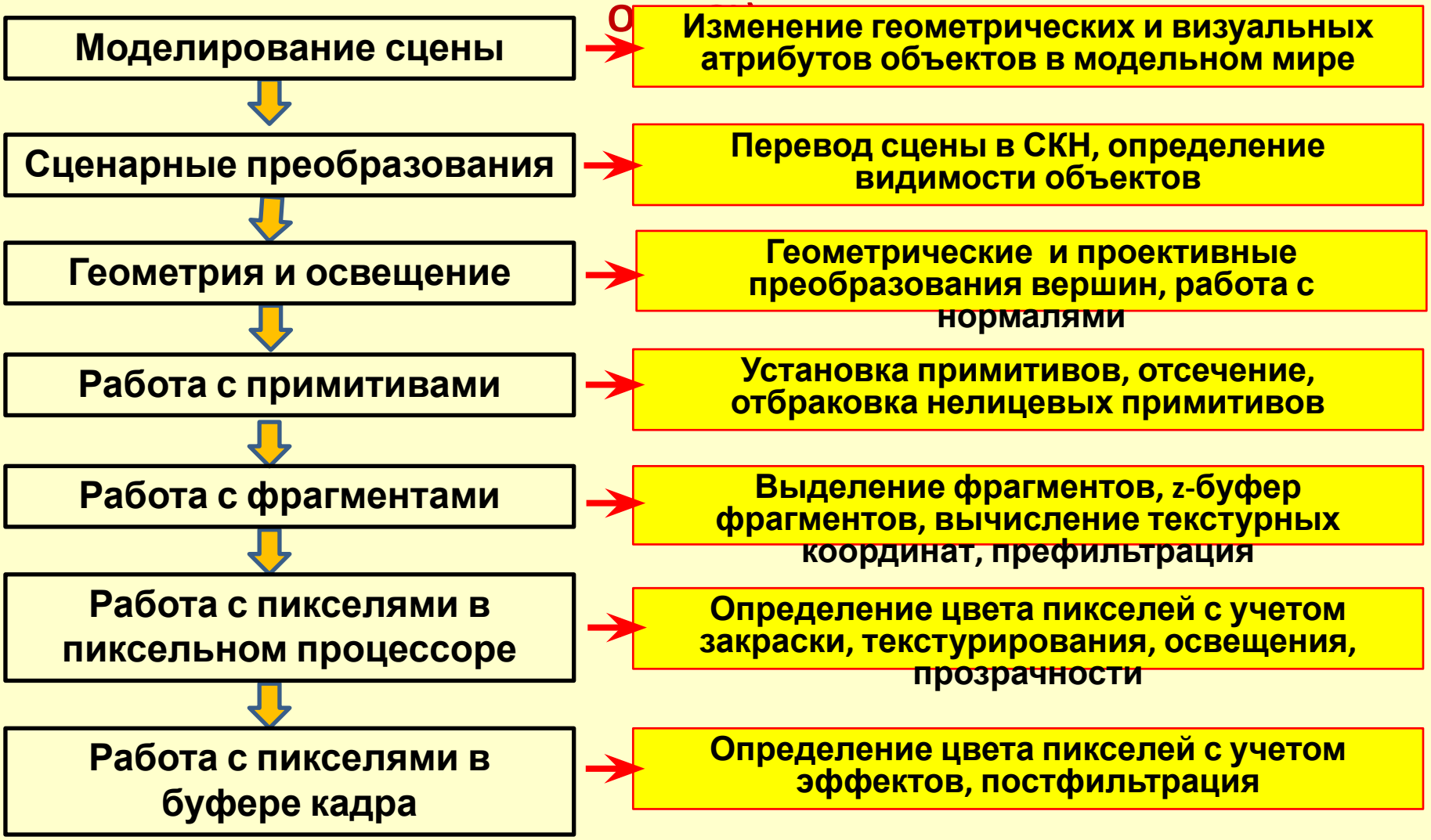
Возможности: создание 2d- и 3d-объектов произвольной формы в виде точечных, «проволочных» и поверхностных моделей; генерация стандартных трехмерных поверхностей (сфера, куб, цилиндр и др.); моделирование освещения 3d-объектов; цветная закраска и наложение текстур; задание шести степеней свободы и масштабирования объектов; улучшение качества изображения с помощью пирамиды текстур, отбраковки нелицевых примитивов, теста глубины, фильтрации, эффектов.

Структура: базовый набор команд (более 300) и несколько расширений (glu, glaux, glut, wgl,…). Группы команд отвечают за геометрию, динамику, закраску и текстурирование, материалы и освещение объектов, пиксельные операции, сценарные преобразования, режим отображения, анализ выполнения команд.

Геометрические примитивы: точки, прямые линии, плоские многоугольники. Есть укрупненные примитивы: поверхности 2-го порядка, многогранники, чайник Юта.

Программное обеспечение компьютерной графики

Графический конвейер (на примере О



Программное обеспечение компьютерной графики

Пакеты прикладных программ

Наименование	Функции	Примеры
Просмотрщики	Просмотр, конвертация формата, редактирование цветов	**ACDSee, IrfanView**
Графические редакторы	Создание и редактирование рисунков, работа с фотографиями	**Corel Draw, Visio, Photoshop, Paint.net**
Иллюстрационные пакеты	Имитация техники работы художника-человека	**MyPaint, Art Rage Studio Pro**
Издательские системы	Проектирование и оформление печатной продукции	**QuarkXPress, Adobe Frame Maker**
Анимационные пакеты	Создание движущихся изображений	**Open Toonz, Moho, Easy GIF Animator Pro**
Моделеры	Создание и редактирование 2d- и 3d-моделей, создание сцен	**3D Studio MAX, Blender, Sculptris**
Графические движки	Создание компьютерных игр	**Unity3D, Unreal Engine, GameMaker: Studio**
Графика профес-сиональных ППП	Визуализация результатов вычислений, проектирования, исследований	**AutoCAD, MatLab, Origin**