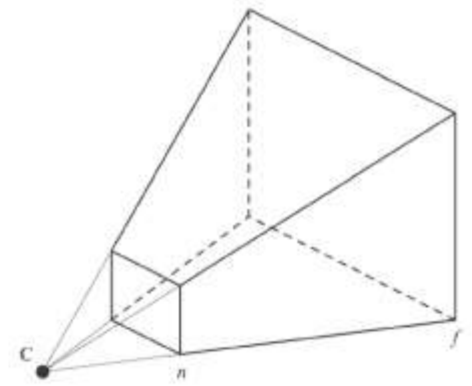
Компьютерная графика и визуализация в реальном времени

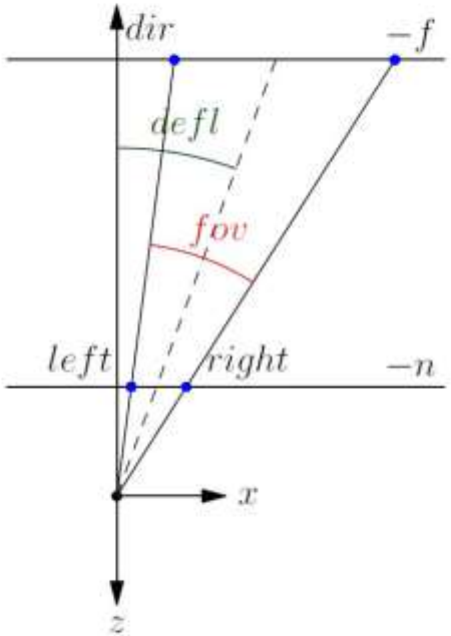
Лекция 1
Пирамида видимости, проективные преобразования, графический конвейер

Алексей Романов

Пирамида видимости

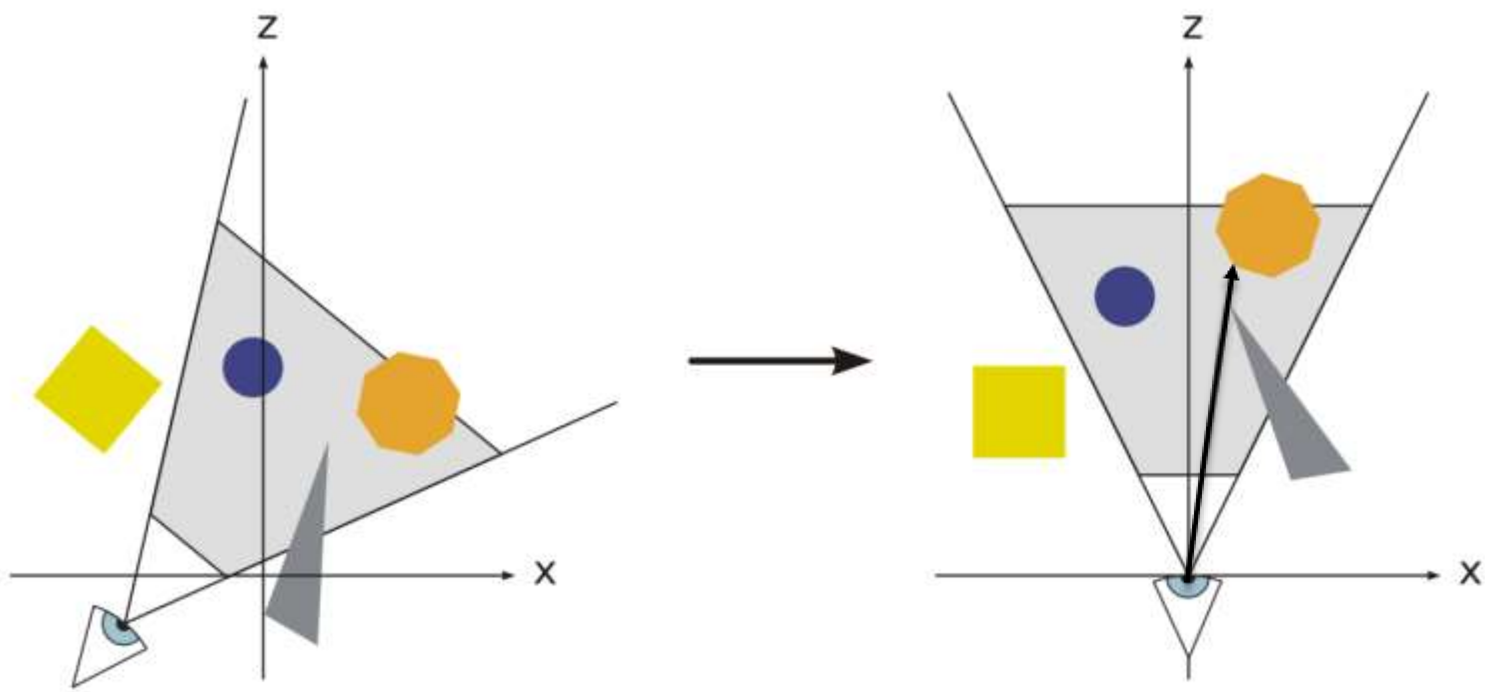
Задается в систем координат камеры





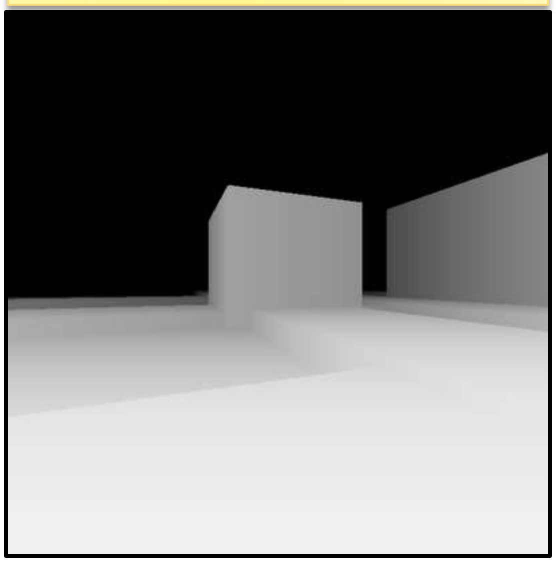
n (near)	Расстояние до ближней плоскости
f (far)	Расстояние до дальней плоскости
fov (field of view)	Угол обзора
defl (deflection)	Угол отклонения

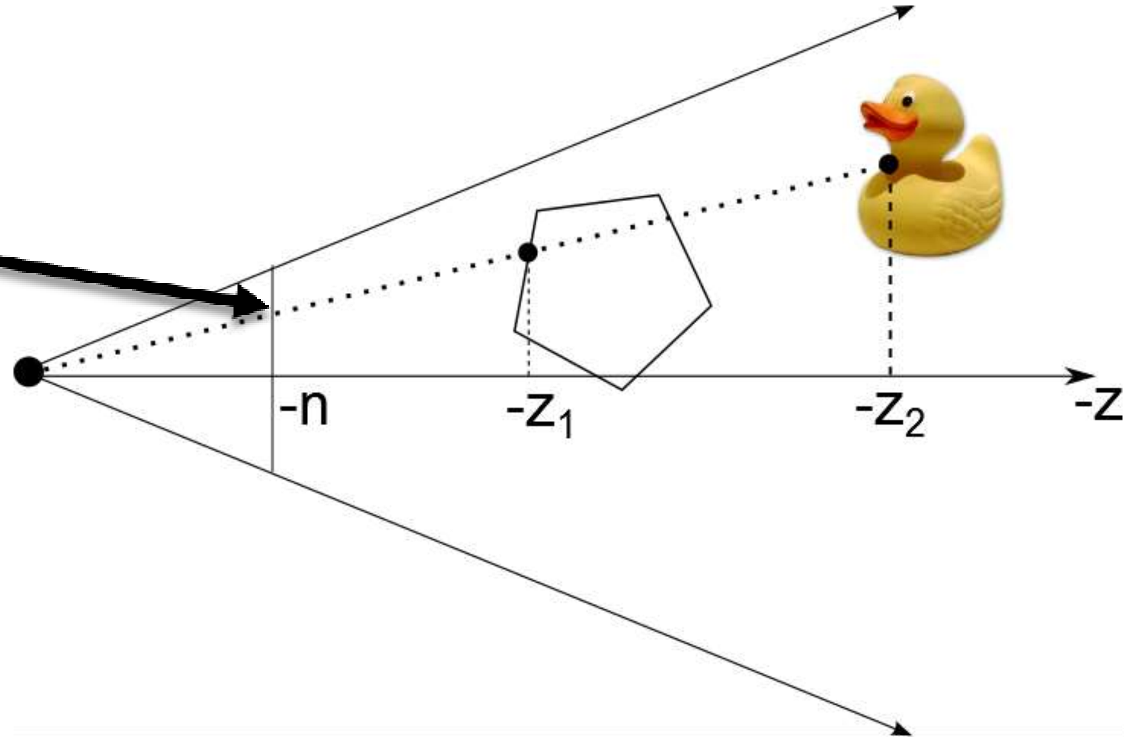
- Перевод в систему координат камеры
- Преобразование проекции

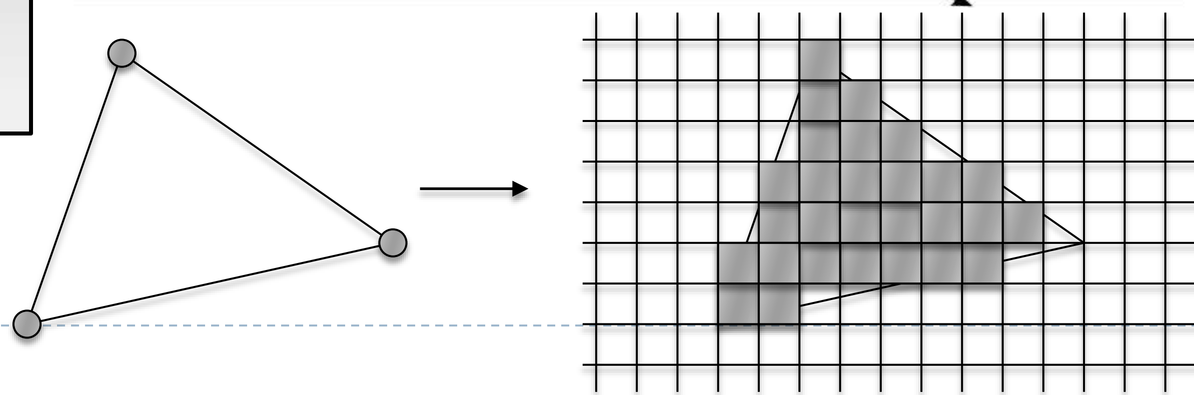


Удаление невидимых поверхностей, *z* тест (тест глубины)

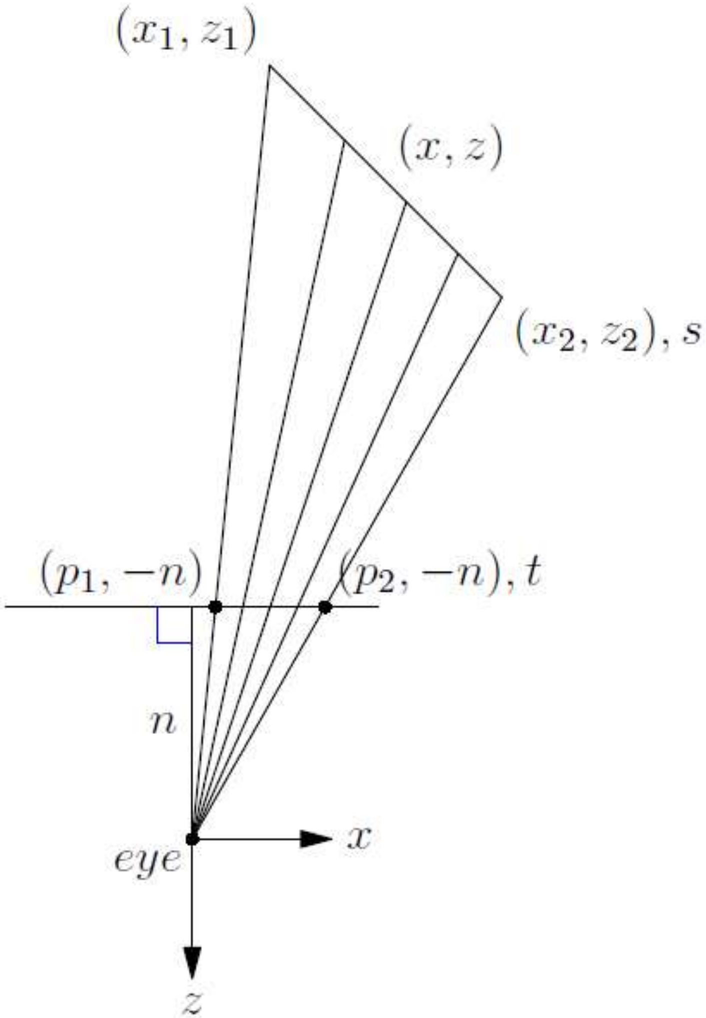
Буфер глубины



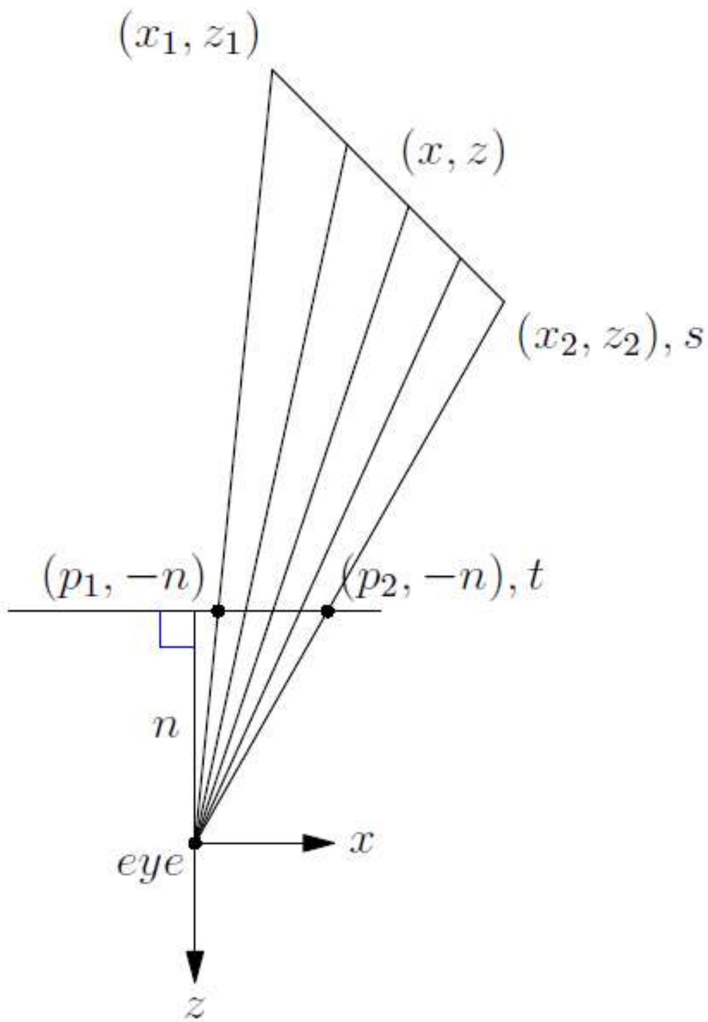




Интерполяция глубины



Интерполяция глубины



$$ax + bz = c, (c \neq 0)$$

$$\frac{p}{x} = \frac{-n}{z}$$

$$(-\frac{ap}{n} + b)z = c$$

$$\frac{1}{z} = -\frac{ap}{cn} + \frac{b}{c}$$

$$\frac{1}{z} = -\frac{ap_1}{cn}(1 - t) - \frac{ap_2}{cn}t + \frac{b}{c}$$

$$\frac{1}{z} = (-\frac{ap_1}{cn} + \frac{b}{c})(1 - t) + (-\frac{ap_2}{cn} + \frac{b}{c})t$$

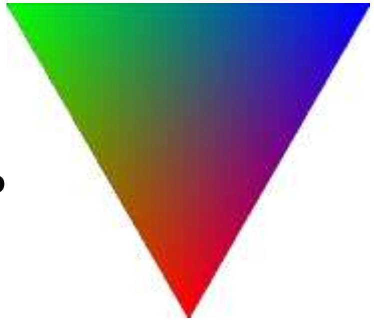
$$\frac{1}{z} = \frac{1}{z_1}(1 - t) + \frac{1}{z_2}t$$

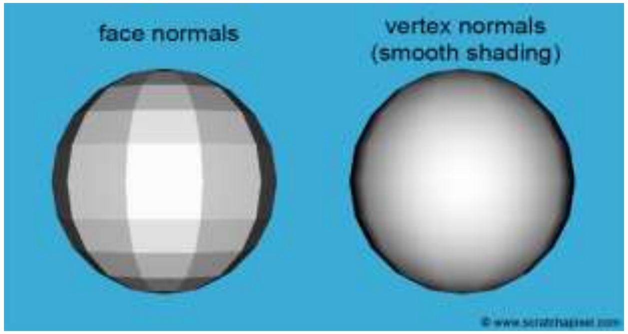
Линейно интерполируется обратная глубина

Интерполяция атрибутов

Интерполяция атрибутов

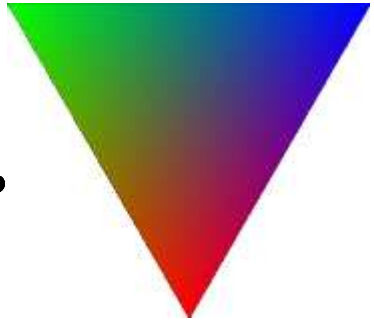
- Цвет
- Нормаль
- Текстурные координать
- …

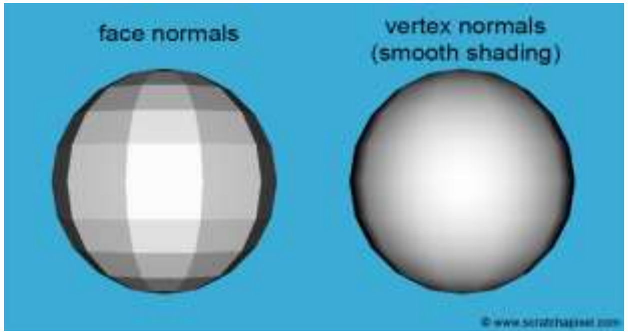


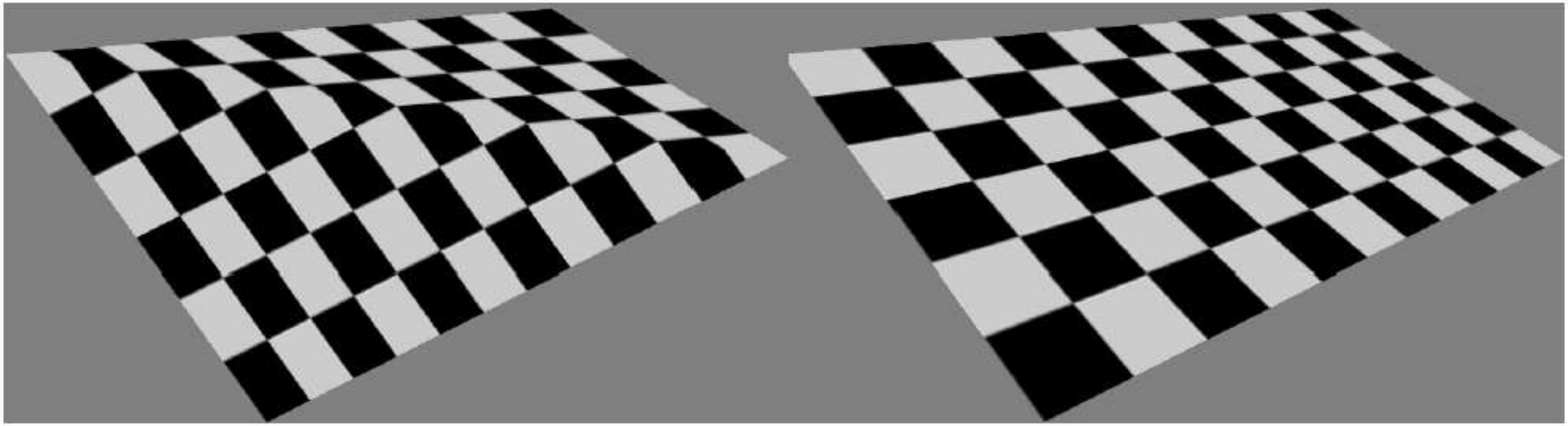


Интерполяция атрибутов

- Цвет
- Нормаль
- Текстурные координать
- …







Линейная (слева) и перспективно-корректная (справа) интерполяция текстурных координат

Интерполяция атрибутов

$$b = b_1(1 - s) + b_2 s$$

$$z = z_1(1 - s) + z_2 s$$

$$s = \frac{tz_1}{tz_1 + (1 - t)z_2}$$

$$\frac{b}{z} = \frac{b_1}{z_1}(1 - t) + \frac{b_2}{z_2} t$$

Линейно интерполируется отношение значения атрибута и глубины

Вывод матрицы перспективной проекции

- Проекция на ближнюю плоскость пирамиды видимости

$$P = (P_x, P_y, P_z, 1)$$

$$x = -\frac{n}{P_z}P_x \qquad y = -\frac{n}{P_z}P_y$$

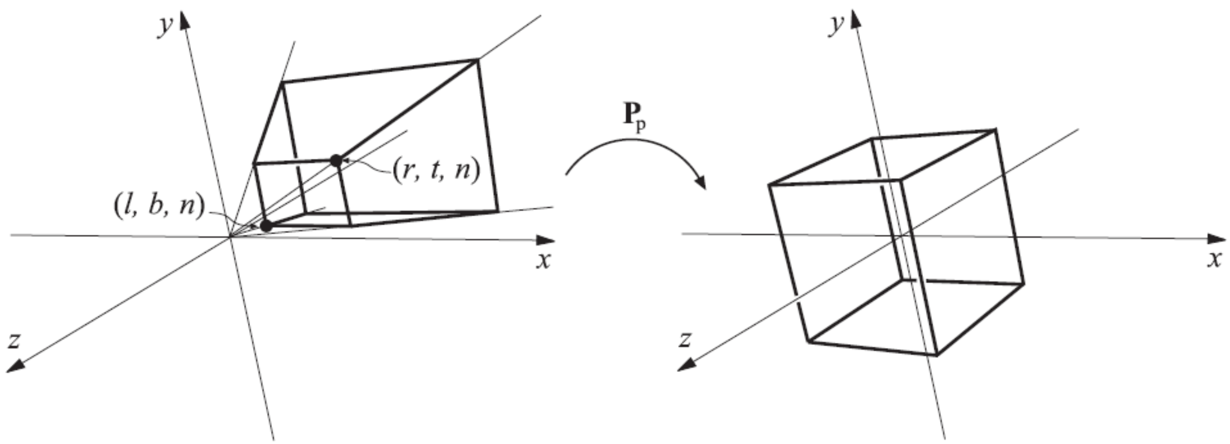
- Масштабирование до границ единичного куба

$$x' = (x - l)\frac{2}{r - l} - 1$$

$$y' = (y - b)\frac{2}{t - l} - 1$$

$$x' = \frac{2n}{r - l}\left(-\frac{P_x}{P_z}\right) - \frac{r + l}{r - l}$$

$$y' = \frac{2n}{t - b}\left(-\frac{P_y}{P_z}\right) - \frac{t + b}{t - b}$$



Вычисление псевдо глубины

В силу линейной интерполяции обратных величин псевдо глубина ищется в виде

$$\boxed{z' = \frac{A}{z} + B}$$

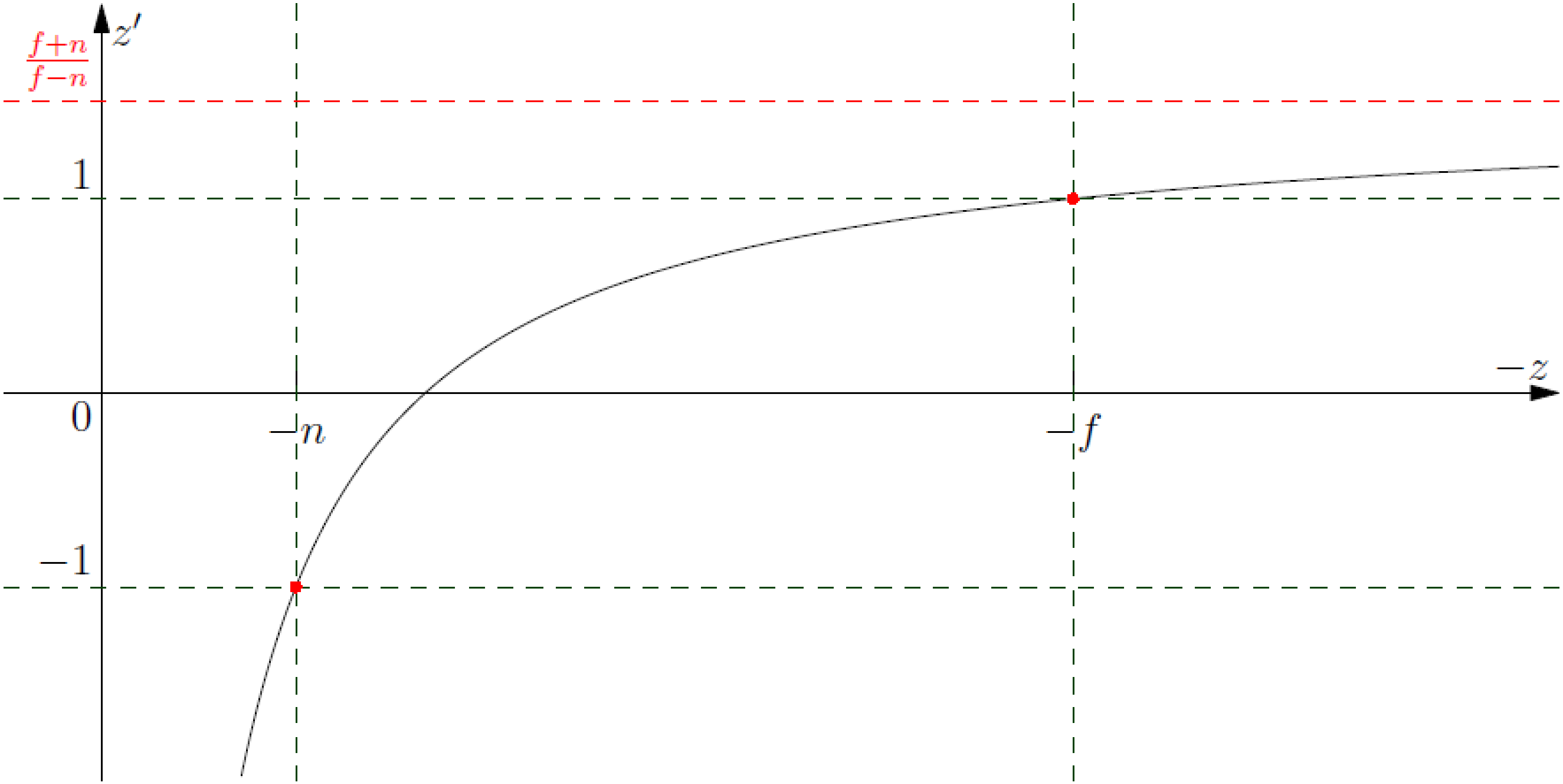
$$-n \to -1 \qquad -f \to 1$$

$$z' = -\frac{2nf}{f-n}\left(-\frac{1}{P_z}\right) + \frac{f+n}{f-n}$$

$$x' = \frac{2n}{r-l}\left(-\frac{P_x}{P_z}\right) - \frac{r+l}{r-l}$$

$$y' = \frac{2n}{t-b}\left(-\frac{P_y}{P_z}\right) - \frac{t+b}{t-b}$$

Псевдо глубина



Переход к однородным координатам

$$P' = (x', y', z') \equiv (x', y', z', 1) \equiv (-x'P_z, -y'P_z, -z'P_z, -P_z)$$

$$-x'P_z = \frac{2n}{r-l}P_x + \frac{r+l}{r-l}P_z$$

$$-y'P_z = \frac{2n}{t-b}P_y + \frac{t+b}{t-b}P_z$$

$$-z'P_z = -\frac{f+n}{f-n}P_z - \frac{2nf}{f-n}$$

Перспективная проекция

$$P' = M_{frustum}P = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2nf}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \\ 1 \end{pmatrix}$$

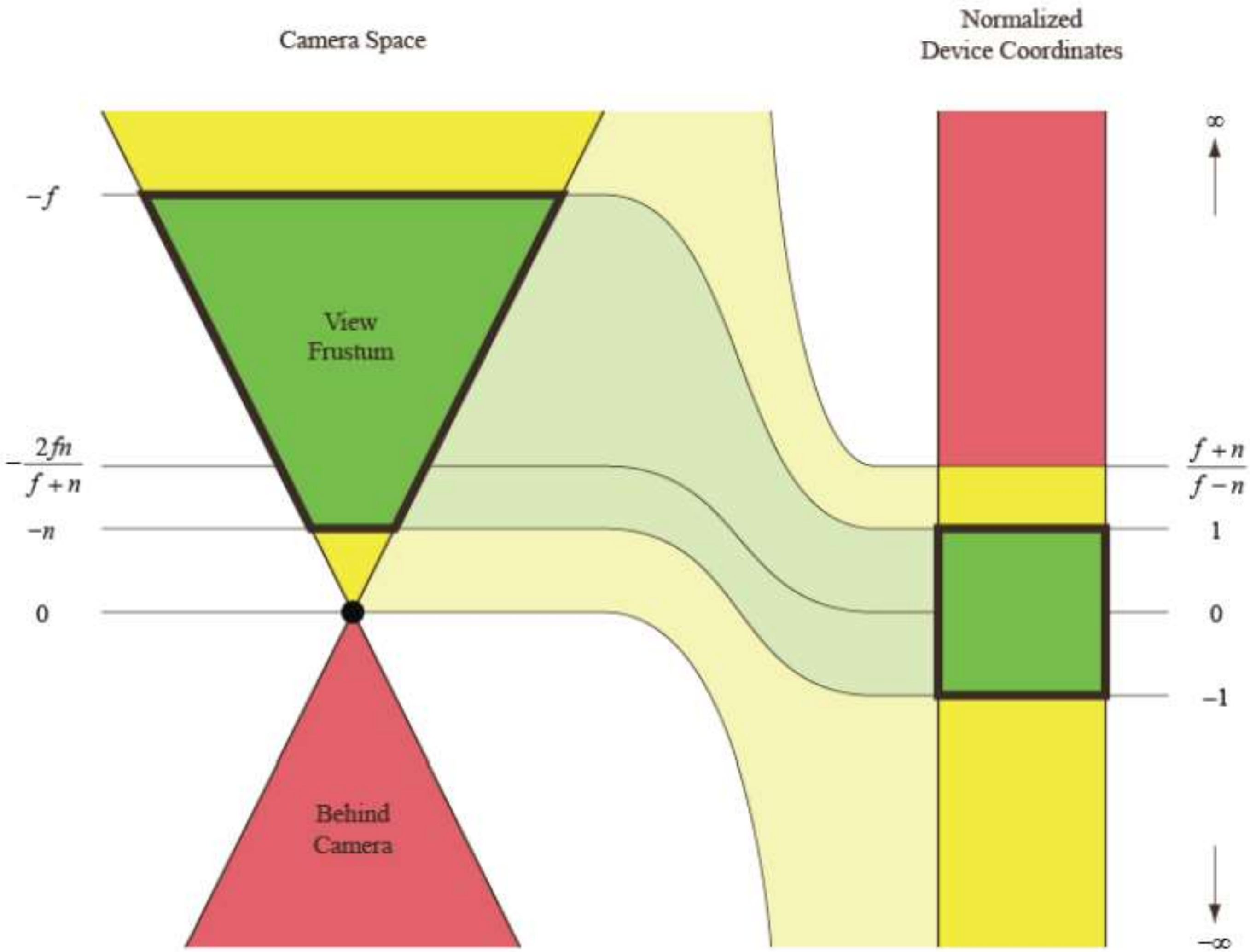
$-f \to \infty$:

$$M_{inf} = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -1 & -2n \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Ортографическая проекция

$$M_{ortho} = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

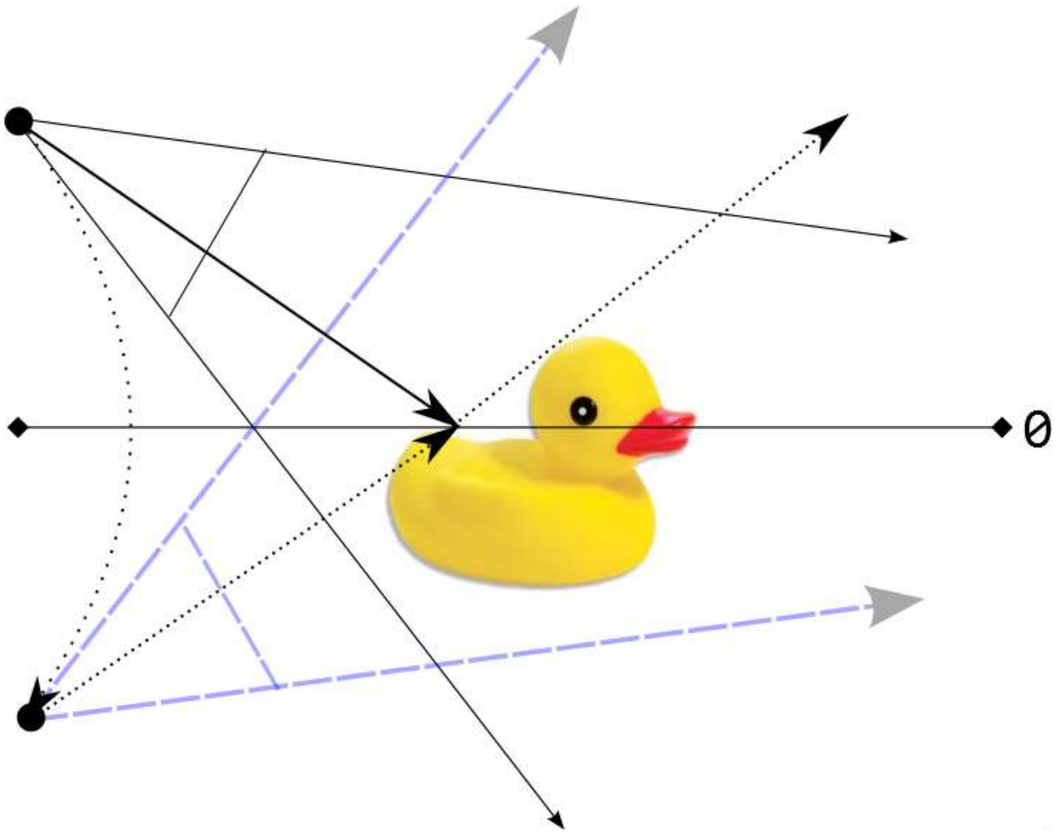
Распределение псевдо глубины пирамиды видимости



Oblique clipping (косое отсечение)

- Возможность модифицировать матрицу проекции для замены отсечение по ближней плоскости отсечению на произвольную
- Применяется при построении карты отражений

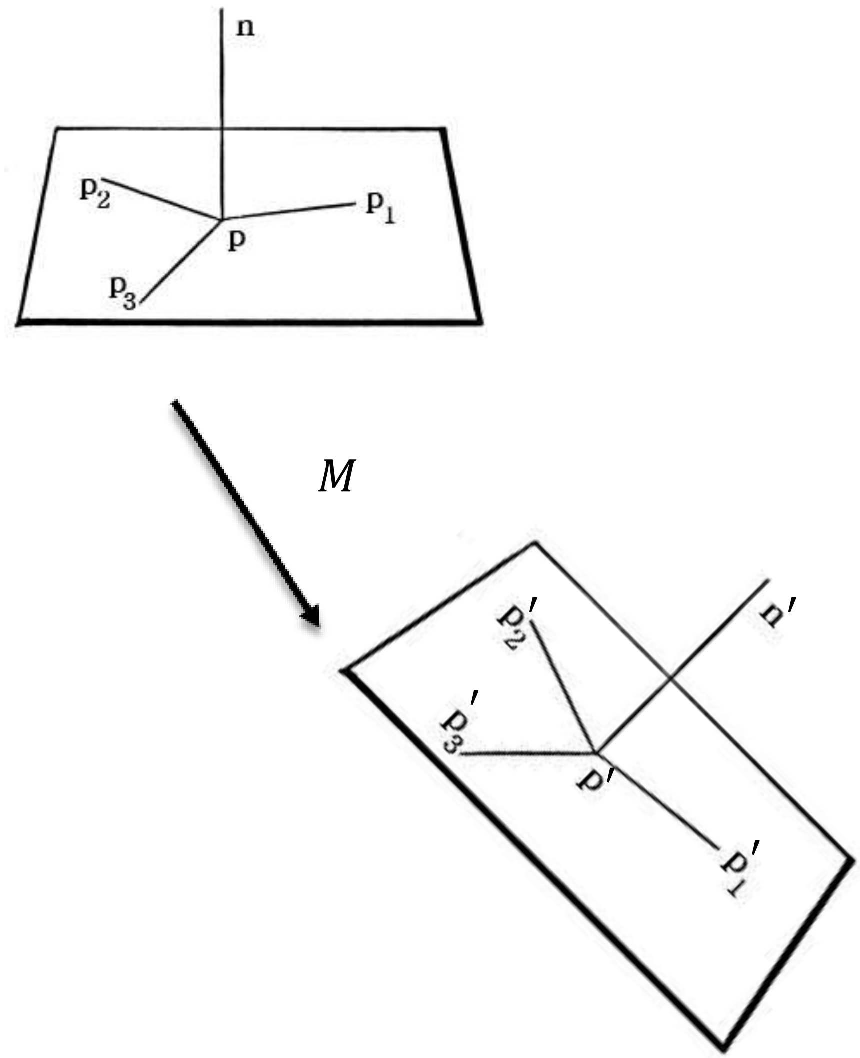
Косое отсечение в отражениях



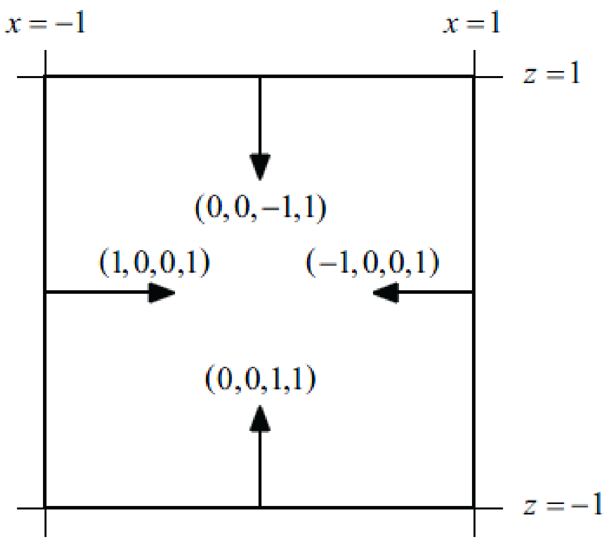
Косое отсечение в отражениях

NB: Преобразование нормалей/плоскостей

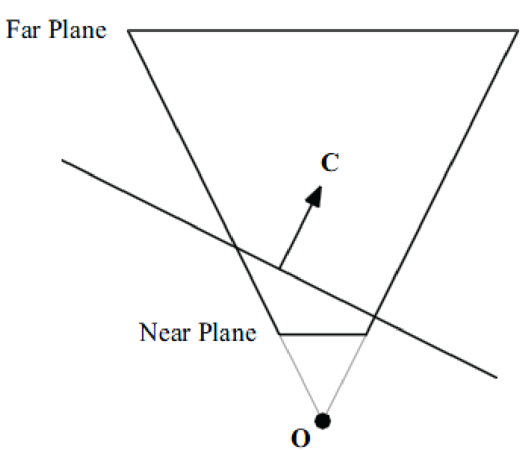
- $(\bar{p} - \overline{p_0})\bar{n} = 0$
- $(p - p_0)^T n = 0$
- $(Mp - Mp_0)^T n' = 0$
- $(p - p_0)^T M^T n' = 0$
- $M^T n' = n\alpha$
- $\alpha = 1$
- $n' = {M^T}^{-1} n$



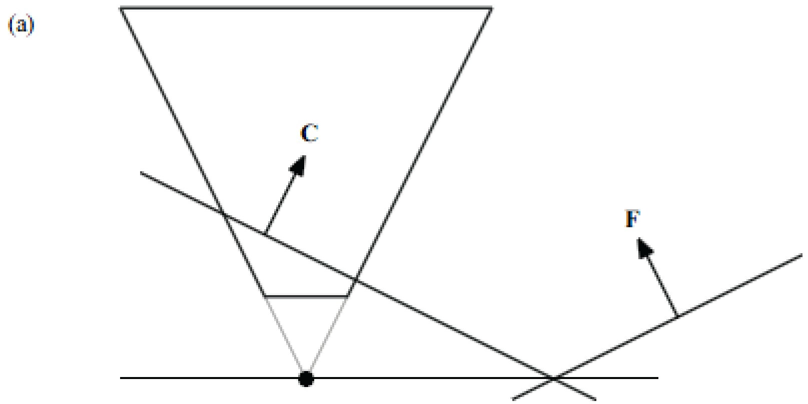
Oblique clipping



Near	$\langle 0,0,1,1 \rangle$	$\mathbf{M}_4 + \mathbf{M}_3$
Far	$\langle 0,0,-1,1 \rangle$	$\mathbf{M}_4 - \mathbf{M}_3$
Left	$\langle 1,0,0,1 \rangle$	$\mathbf{M}_4 + \mathbf{M}_1$
Right	$\langle -1,0,0,1 \rangle$	$\mathbf{M}_4 - \mathbf{M}_1$
Bottom	$\langle 0,1,0,1 \rangle$	$\mathbf{M}_4 + \mathbf{M}_2$
Top	$\langle 0,-1,0,1 \rangle$	$\mathbf{M}_4 - \mathbf{M}_2$



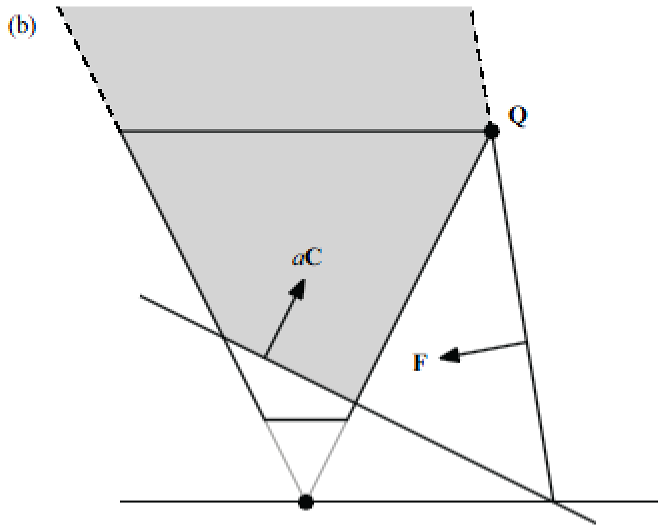
Oblique clipping



$$\mathbf{C} = \mathbf{M}_4 + \mathbf{M}_3.$$

$$\mathbf{M}'_3 = \mathbf{C} - \mathbf{M}_4.$$

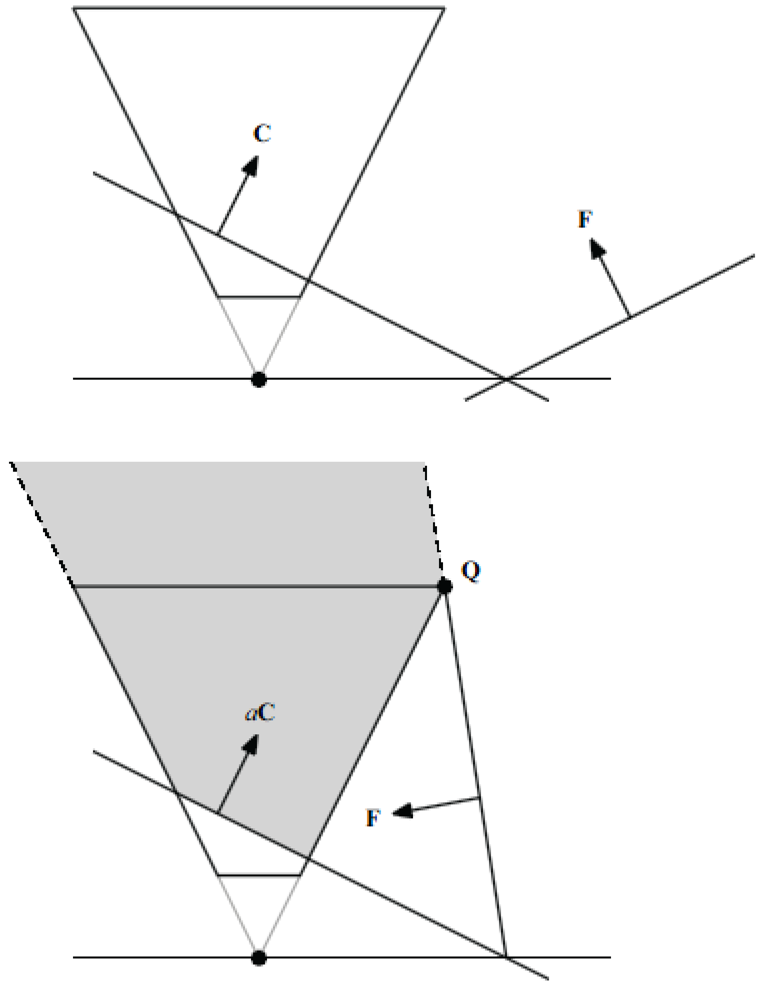
$$\begin{aligned}\mathbf{F} &= \mathbf{M}_4 - \mathbf{M}'_3 \\ &= 2\mathbf{M}_4 - \mathbf{C}.\end{aligned}$$



Oblique clipping

(a)

(b)



$$\mathbf{C} = \mathbf{M}_4 + \mathbf{M}_3.$$

$$\mathbf{M}'_3 = \mathbf{C} - \mathbf{M}_4.$$

$$\mathbf{F} = \mathbf{M}_4 - \mathbf{M}'_3$$
$$= 2\mathbf{M}_4 - \mathbf{C}.$$

$$\mathbf{C}' = \left(\mathbf{M}^{-1}\right)^{\mathrm{T}} \mathbf{C}$$

$$\mathbf{Q}' = \left\langle \operatorname{sgn}(C'_x), \operatorname{sgn}(C'_y), 1, 1 \right\rangle.$$

$$\mathbf{F} = 2\mathbf{M}_4 - a\mathbf{C}.$$

$$a = \frac{2\mathbf{M}_4 \cdot \mathbf{Q}}{\mathbf{C} \cdot \mathbf{Q}}.$$

$$\mathbf{M}'_3 = a\mathbf{C} - \mathbf{M}_4$$

Преобразование проекции

- Перспективно-корректная интерполяция глубины

$$\frac{1}{z} = \frac{1}{z_1}(1-t) + \frac{1}{z_2}t$$

- Перспективно-корректная интерполяция атрибутов

$$\frac{b}{z} = \frac{b_1}{z_1}(1-t) + \frac{b_2}{z_2}t$$

- Матрица перспективной и ортографической проекции

$$\begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2nf}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \quad \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Преобразование нормалей

$$p' = M_{o\to v}p$$

$$n' = {M_{o\to v}^{-1}}^{T} n$$

- Модификация матрицы проекции для косого отсечения

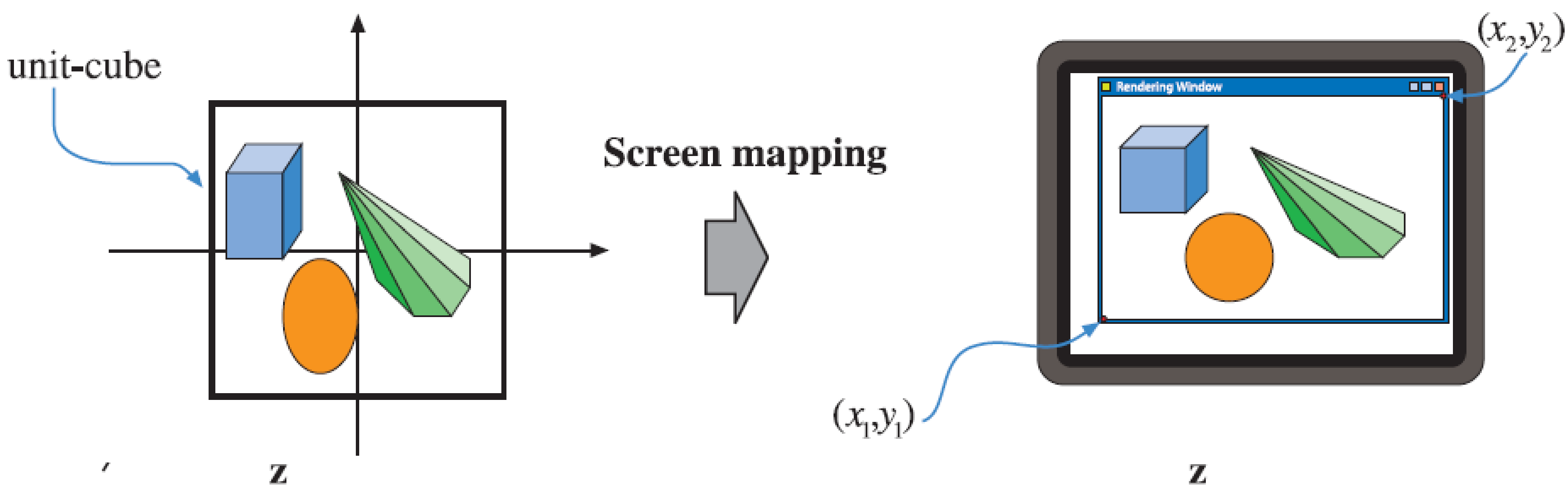
$$a = \frac{2\mathbf{M}_4 \cdot \mathbf{Q}}{\mathbf{C} \cdot \mathbf{Q}}.$$

$$\mathbf{M}'_3 = a\mathbf{C} - \mathbf{M}_4$$

Схема обработки геометрии

Вершины (набор атрибутов) → Вершинный шейдер → Преобразованные вершины → Геометрический шейдер → Примитивы с повершинными атрибутами →

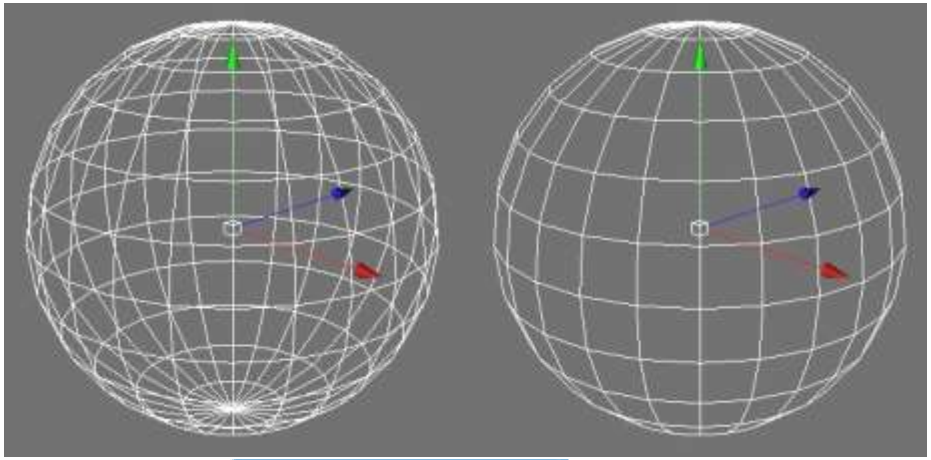
Удаление задних граней → Отсечение по дополнительным плоскостям → Отсечение по пирамиде видимости → Проективное деление → Преобразование порта просмотра

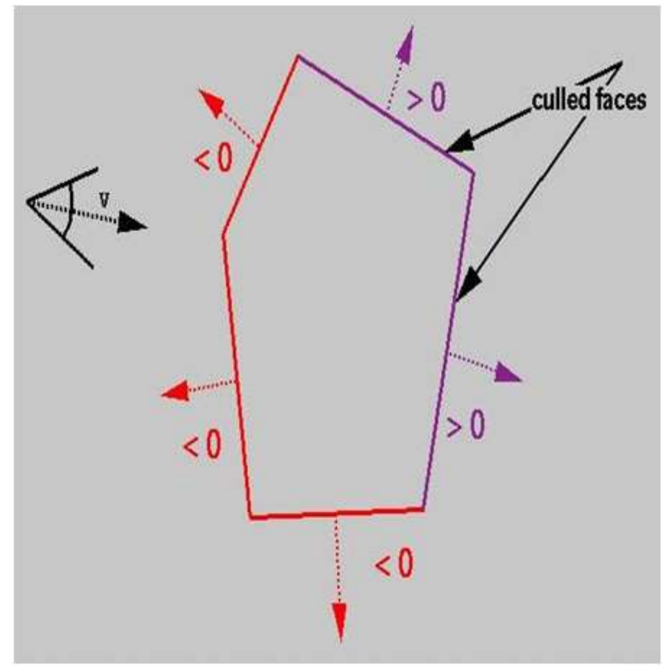


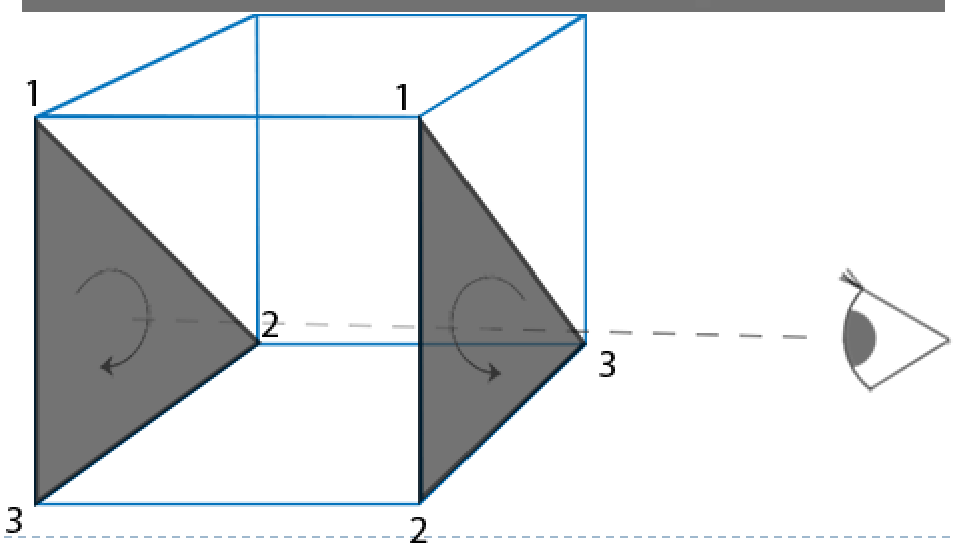
Пример обработки геометрии

Оптимизация: удаление задних граней

- Применяется для замкнутой геометрии







Растеризация в OpenGL

Спроецированный примитив на экран с атрибутам в вершинах → Растеризация треугольника → Фрагментный шейдер → Тест глубины, Scissor тест, … → Композиция в буфер кадра (blending)

Примеры выполнения фрагментного шейдера

Смешивание цветов

Общий вид

$$c_r = \phi_c(c_s f_1, c_d f_2)$$

$$a_r = \phi_a(a_s f_3, a_d f_4)$$

r — результат

s — растеризуемый фрагмент

d — целевой буфер

c — цвет

a — прозрачность

f_i — множитель аргумента

ϕ — функция смешивания

Типы функции смешивания ϕ

$$\phi(a_s, a_d) = a_s + a_d$$

$$\phi(a_s, a_d) = a_s - a_d$$

$$\phi(a_s, a_d) = a_d - a_s$$

$$\phi(a_s, a_d) = max(a_s, a_d)$$

$$\phi(a_s, a_d) = min(a_s, a_d)$$

Типы множителей

$f = 1$	$f = 0$
$f = 1 - s_{[a\|c]}$	$f = 1 - d_{[a\|c]}$
$f = s_{[a\|c]}$	$f = d_{[a\|c]}$

Популярные функции смешивания

Alpha blend

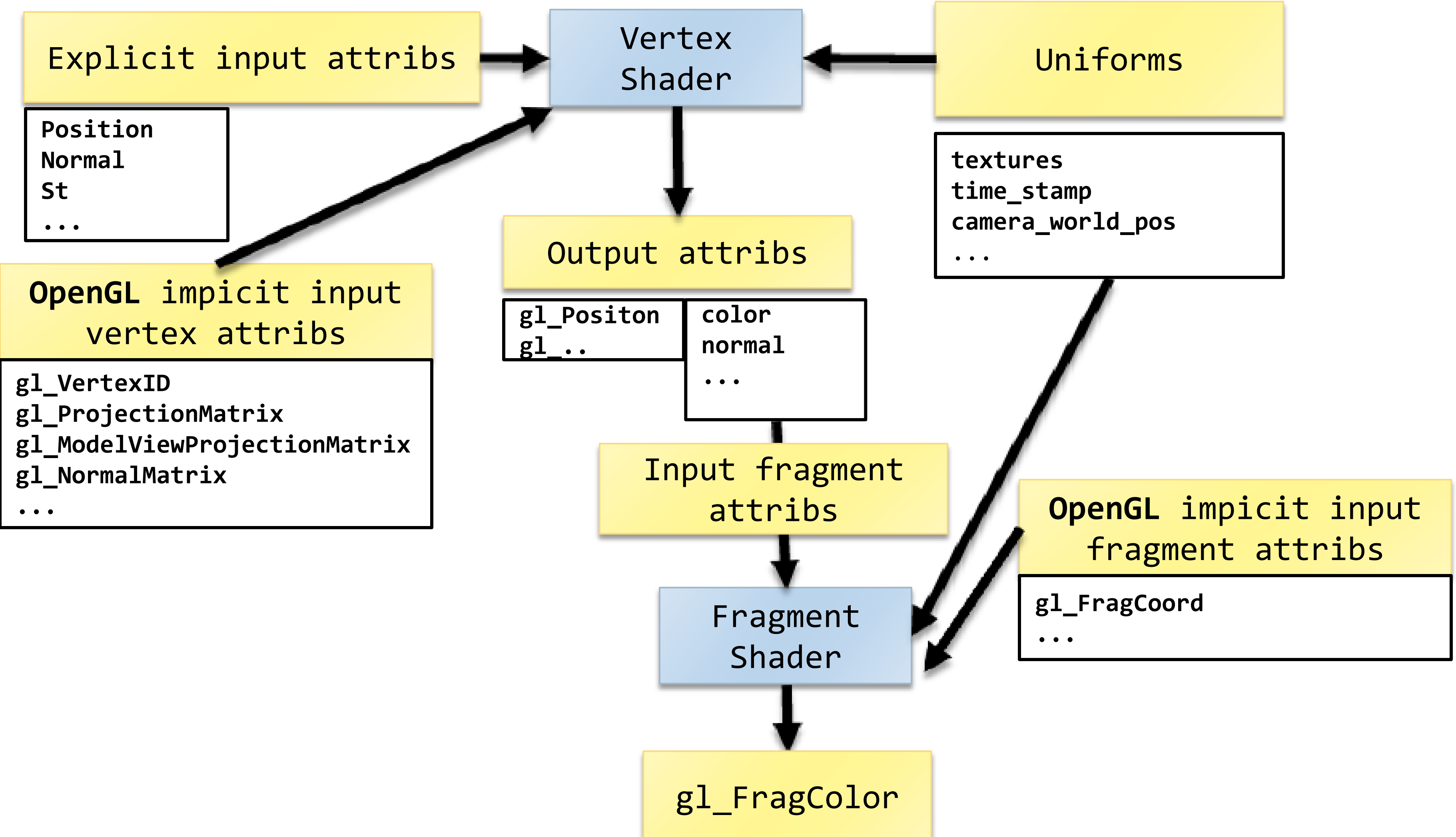
$$c_r = c_s a_s + c_d(1 - a_s)$$

Additive blend

$$c_r = c_s a_s + c_d$$

Blend режим (слева) и additive режим *справа*

Vertex->fragment shader



Отрисовка в OpenGL

```
GLuint vbo;
void * vx_data;

glGenBuffersARB(1, &vbo);
glBindBufferARB(GL_ARRAY_BUFFER_ARB, vbo);
glBufferDataARB(GL_ARRAY_BUFFER_ARB, data_size_in_bytes, vx_data, GL_STATIC_DRAW_ARB);


glBindBufferARB(GL_ARRAY_BUFFER_ARB, vbo);
glBindBufferARB(GL_ELEMENT_ARRAY_BUFFER_ARB, vbo_idx);

glEnableVertexAttribArray(0);
glVertexAttribPointer(0, 2, GL_FLOAT, false, stide, offset_0); // xy
glEnableVertexAttribArray(1);
glVertexAttribPointer(1, 3, GL_FLOAT, false, stide, offset_1); // color

glDrawArrays(GL_QUADS, 0, 24);
glDrawElements(GL_QUADS, 24, GL_UNSIGNED_INT, 0);
```

Вопросы