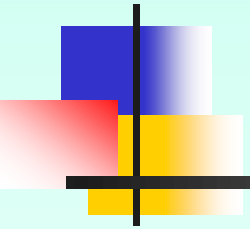


COMPUTER GRAPHICS

ЗАСОБИ ПРОГРАМУВАННЯ КОМП'ЮТЕРНОЇ ГРАФІКИ

Лек. 09 2021 ІПЗ-19

ЛЕКЦИЯ 11
ТЕКСТУРИРОВАНИЕ

Текстурирование

- **Определение**
- **Виды текстурирования**
- **Отображение текстуры**
- **Фильтрация текстуры**

ТЕКСТУРИРОВАНИЕ

Текстурирование (texture mapping) вид (метод) закраски, использующий изображения для моделирования отражающих свойств поверхности.

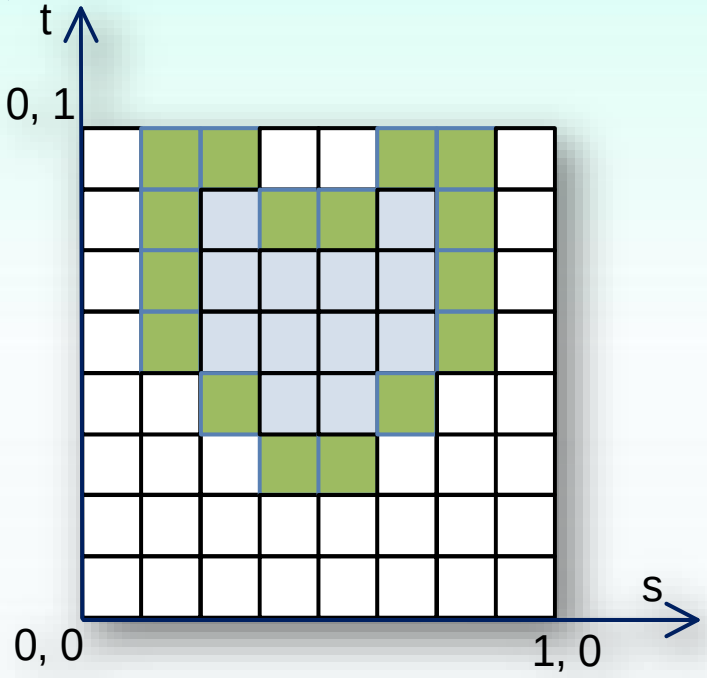
Текстура — растровое изображение, накладываемое на поверхность полигональной модели для придания ей цвета, окраски или иллюзии рельефа.

Образно -> нанесение рисунка на поверхности скульптурного изображения

ТЕКСТУРА

Текстура суть функция интенсивности **T(s,t) –** двумерный образец – шаблон текстуры. Переменные s, t – координаты текстуры, изменяются в диапазоне (0, 1)

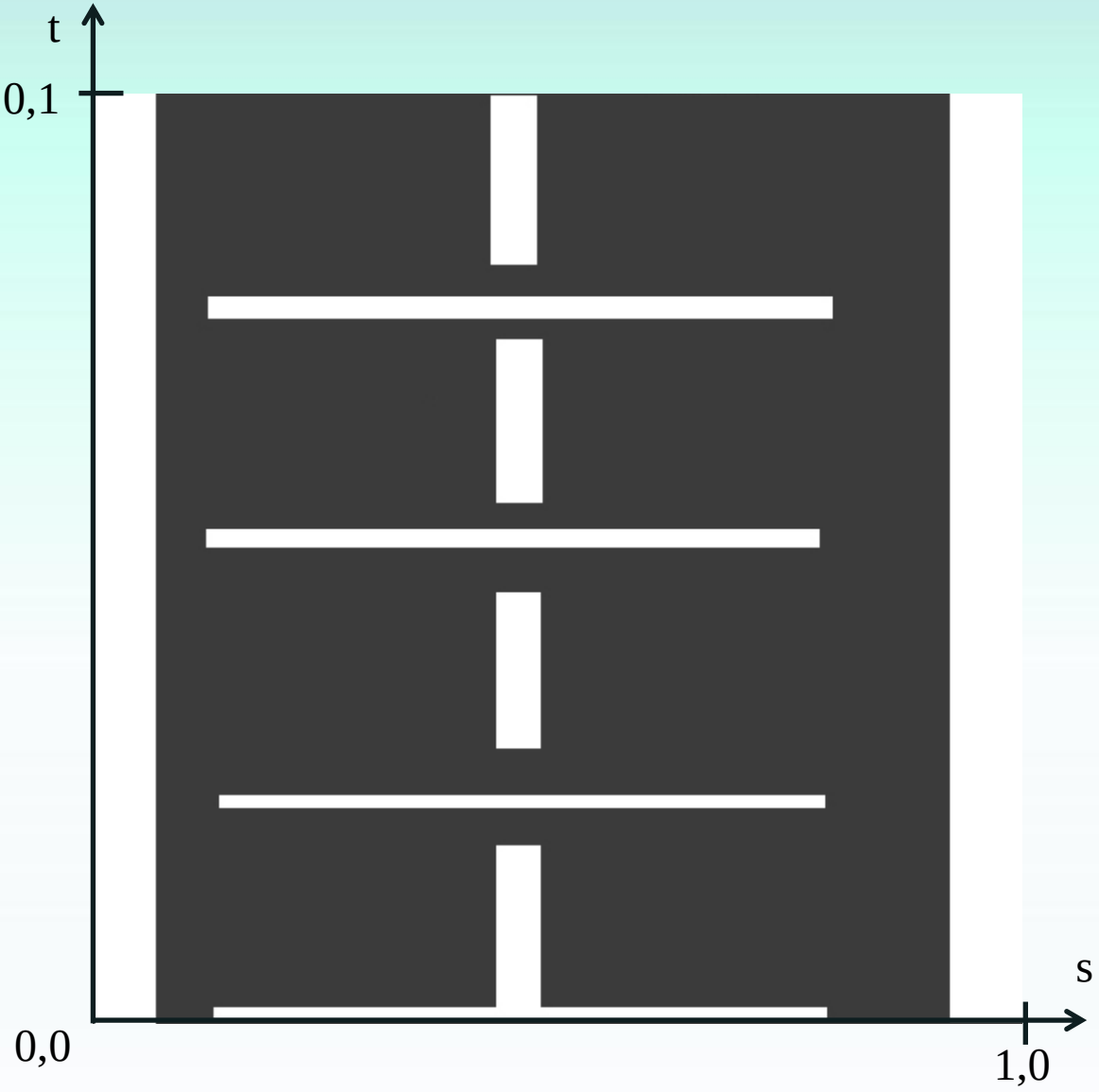
Текстура массив **текселей (texels) –** размерности n x m. Координаты s, t – легко пересчитываются к размерностям массива текселей



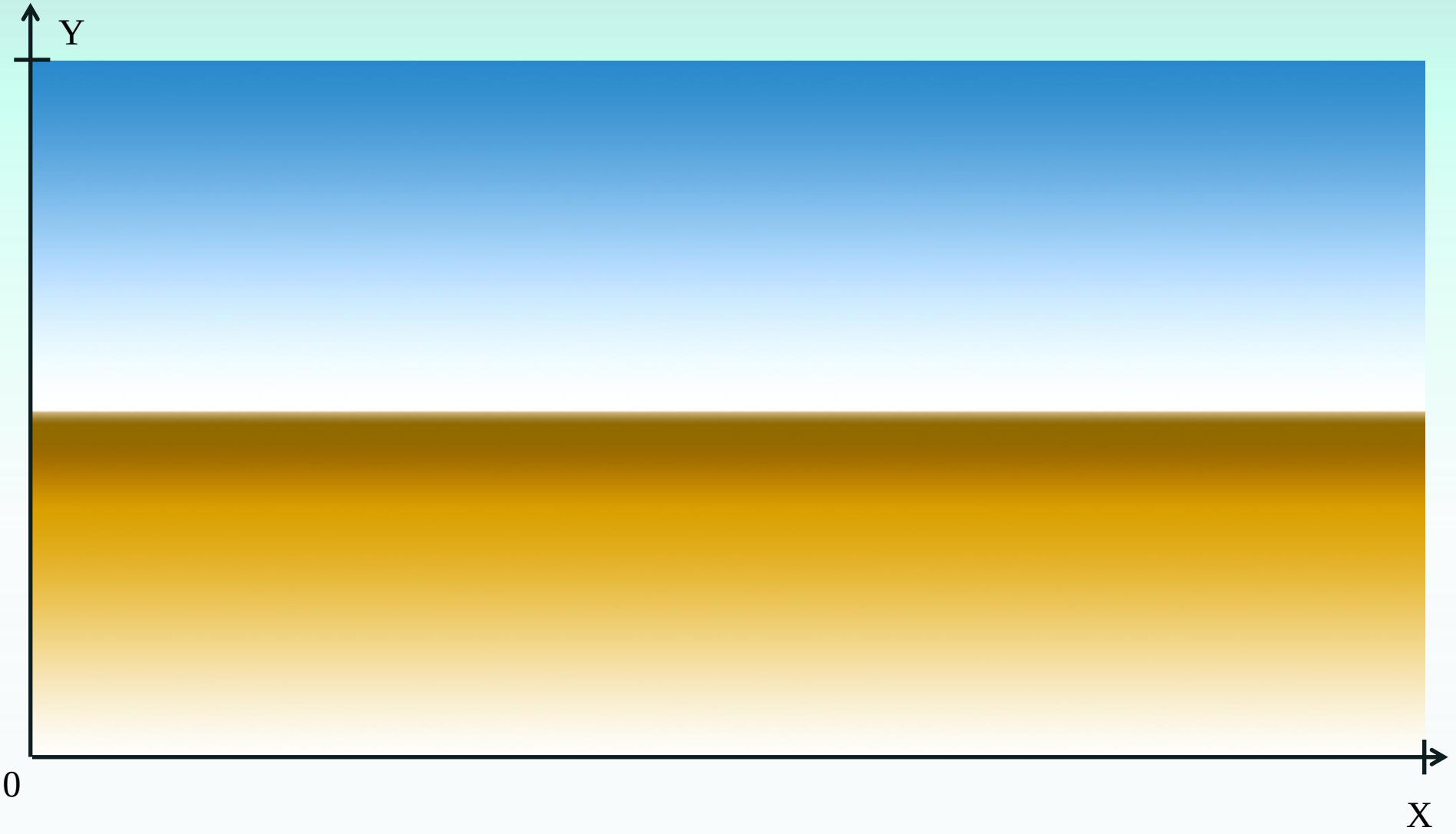
Тексель = пиксель текстуры

ТЕКСТУРА

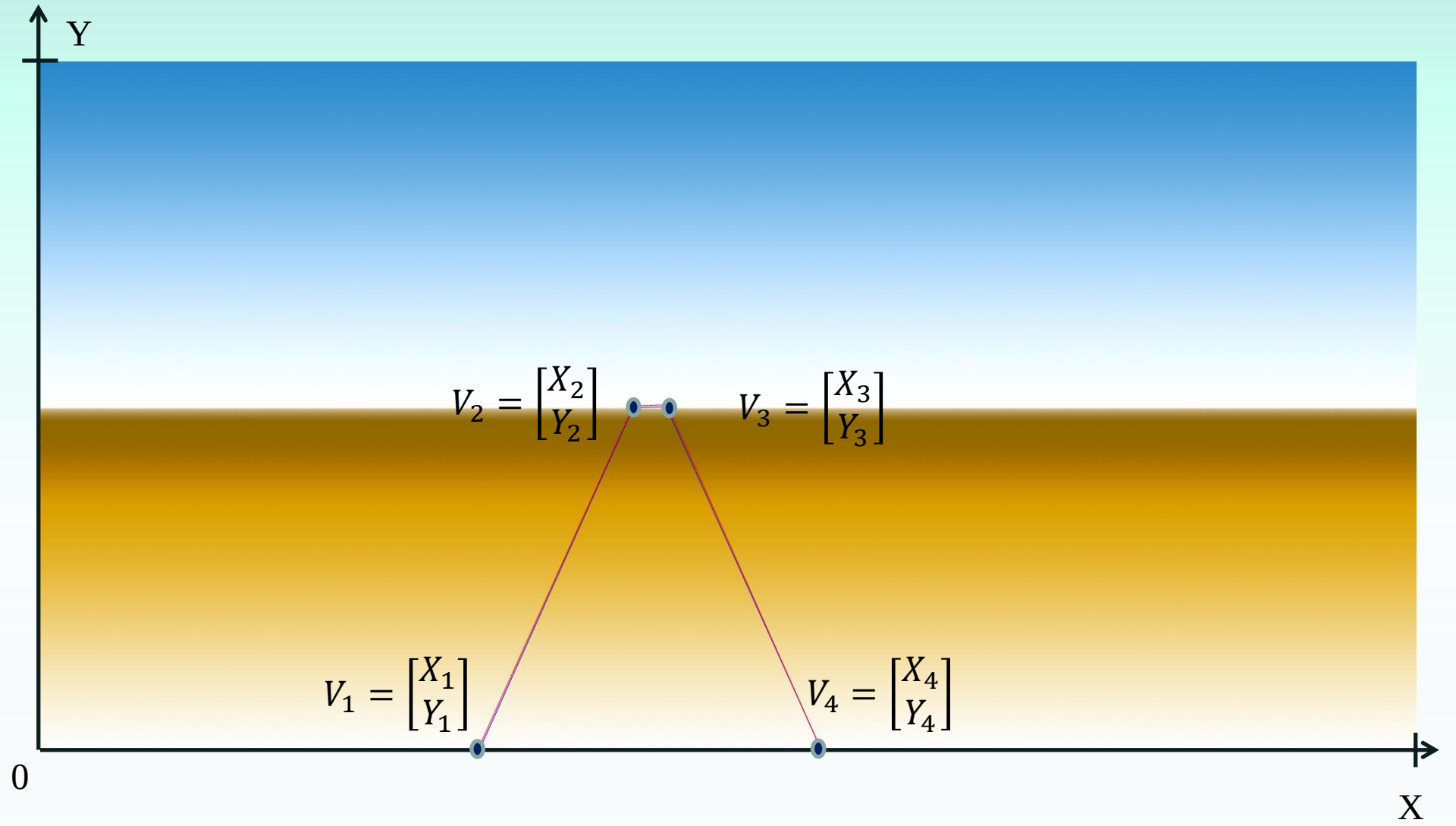
Например, текстура:



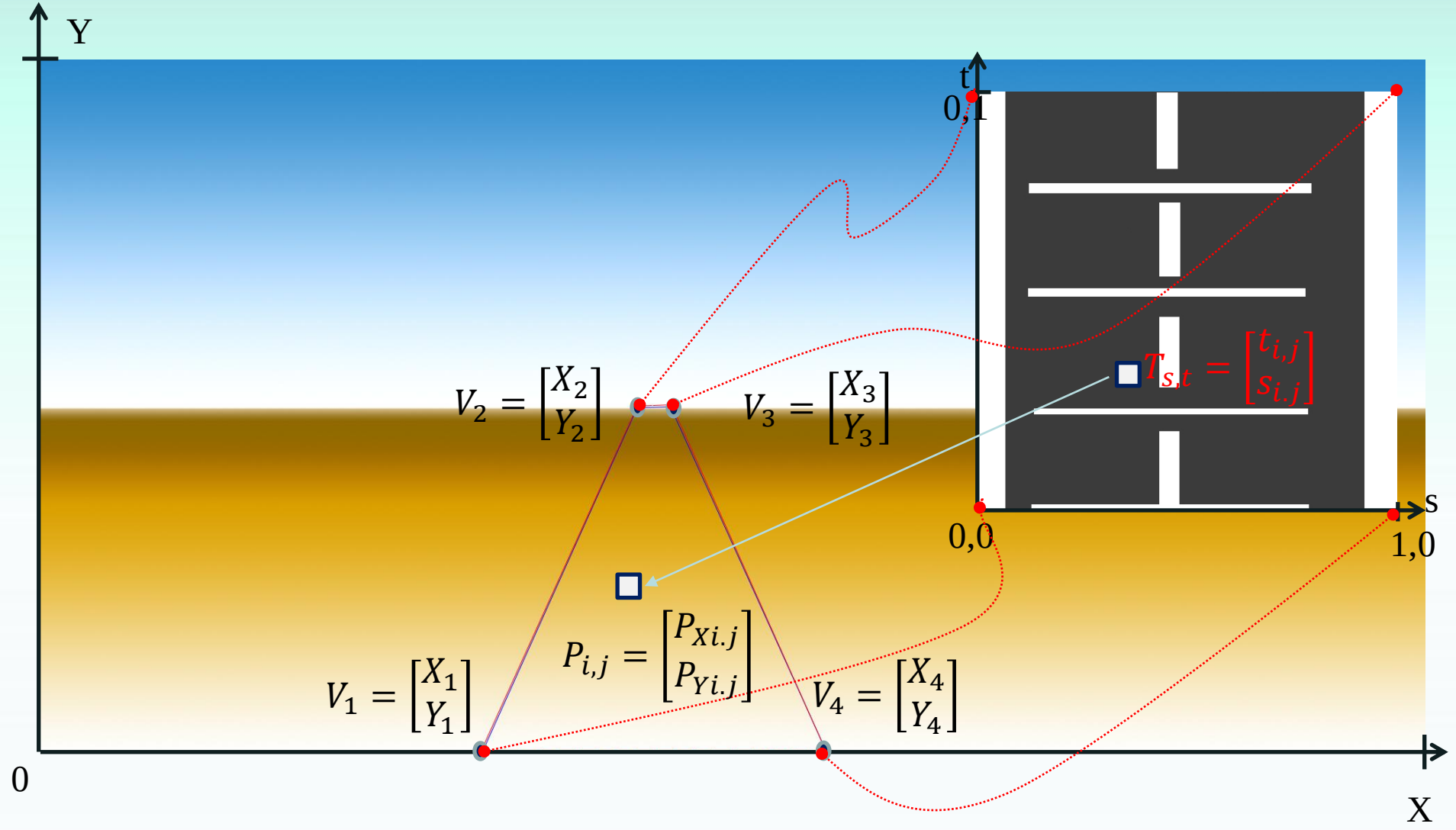
ТЕКСТУРИРОВАНИЕ



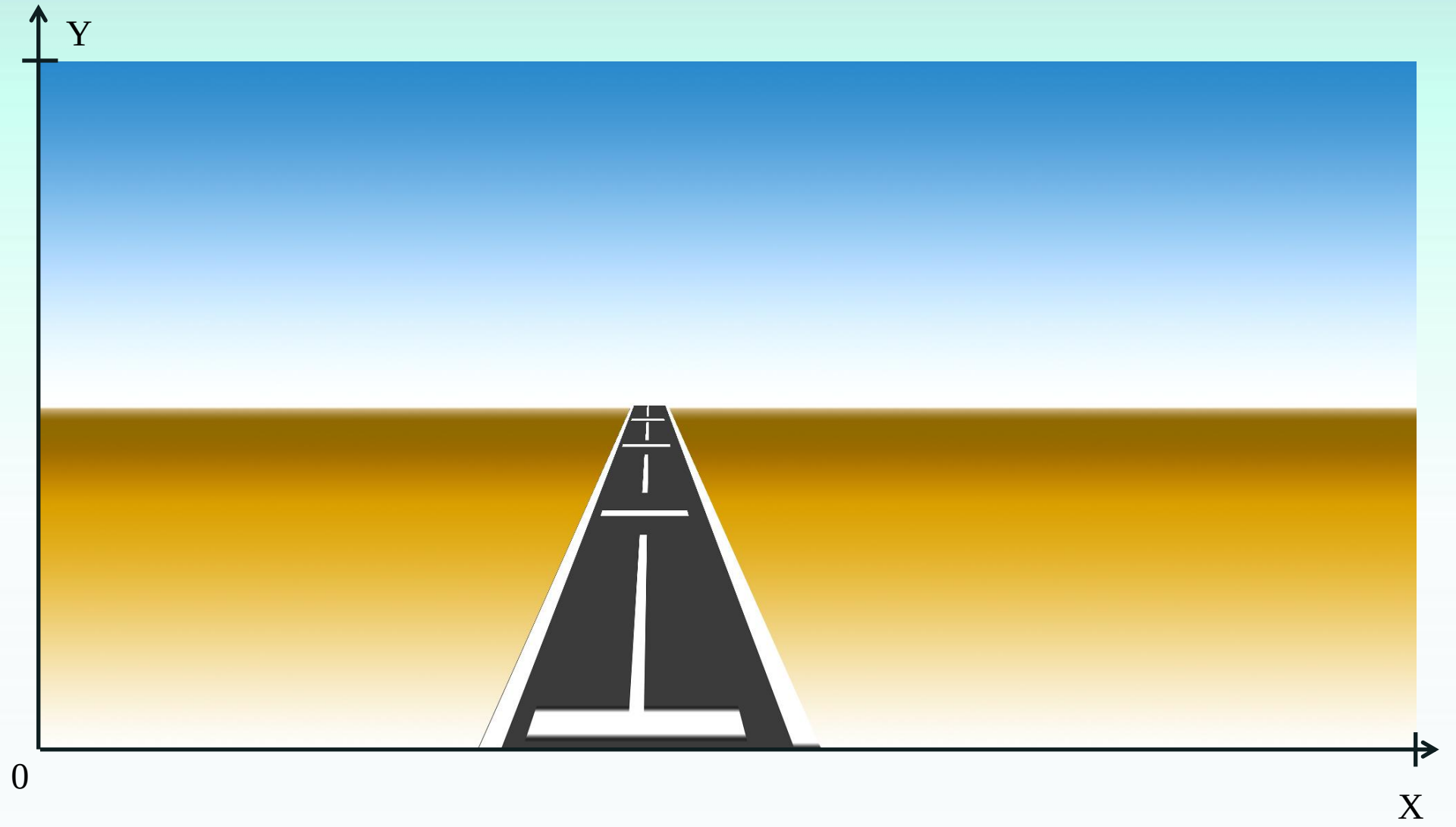
ТЕКСТУРИРОВАНИЕ



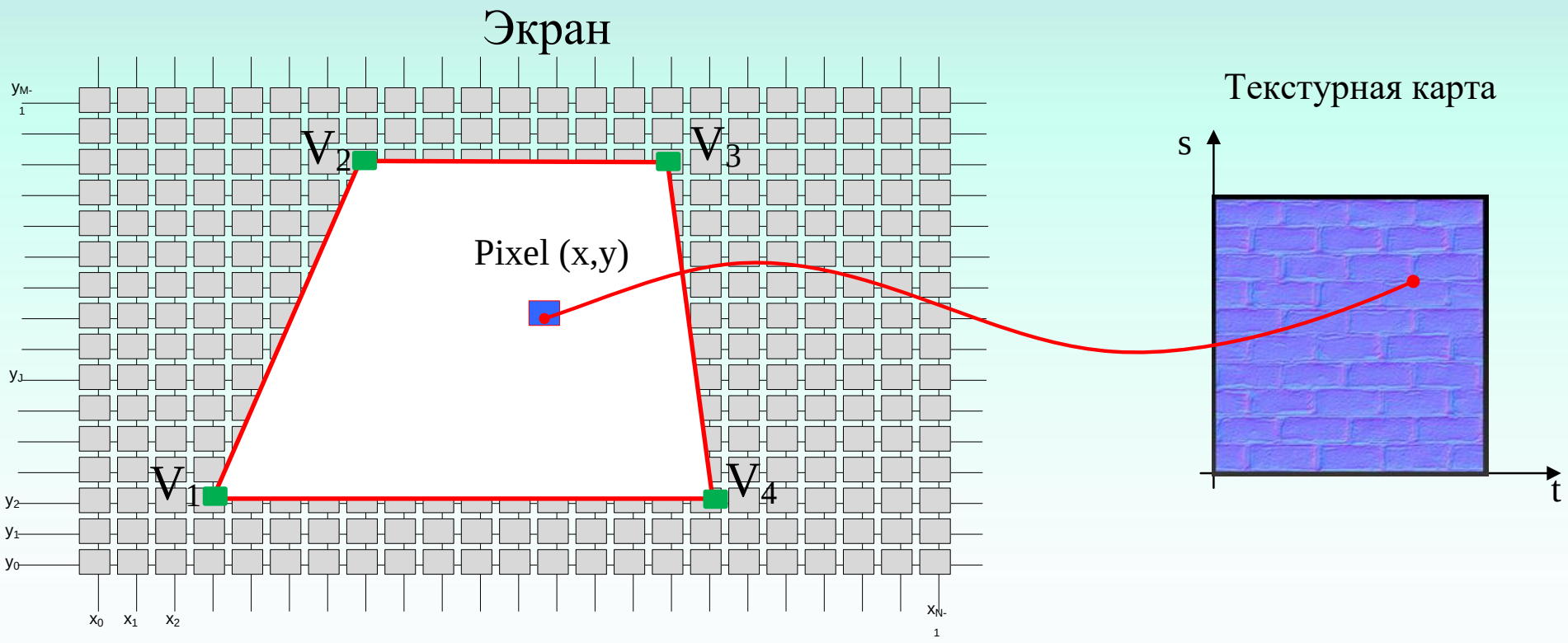
ТЕКСТУРИРОВАНИЕ



ТЕКСТУРИРОВАНИЕ



ТЕКСТУРИРОВАНИЕ в 2D



Используется параметрическое представление многогранника

ТЕКСТУРИРОВАНИЕ в 2D

Параметрическое представление многогранника

$$Y = (Y_2 - Y_1) * s + Y_1;$$
$$X_l = (X_2 - X_1) * s + X_1; \quad X_r = (X_3 - X_4) * s + X_4;$$
$$X = (X_r - X_l) * t + X_l;$$

Пиксел имеет координаты X_{pix}, Y_{pix}

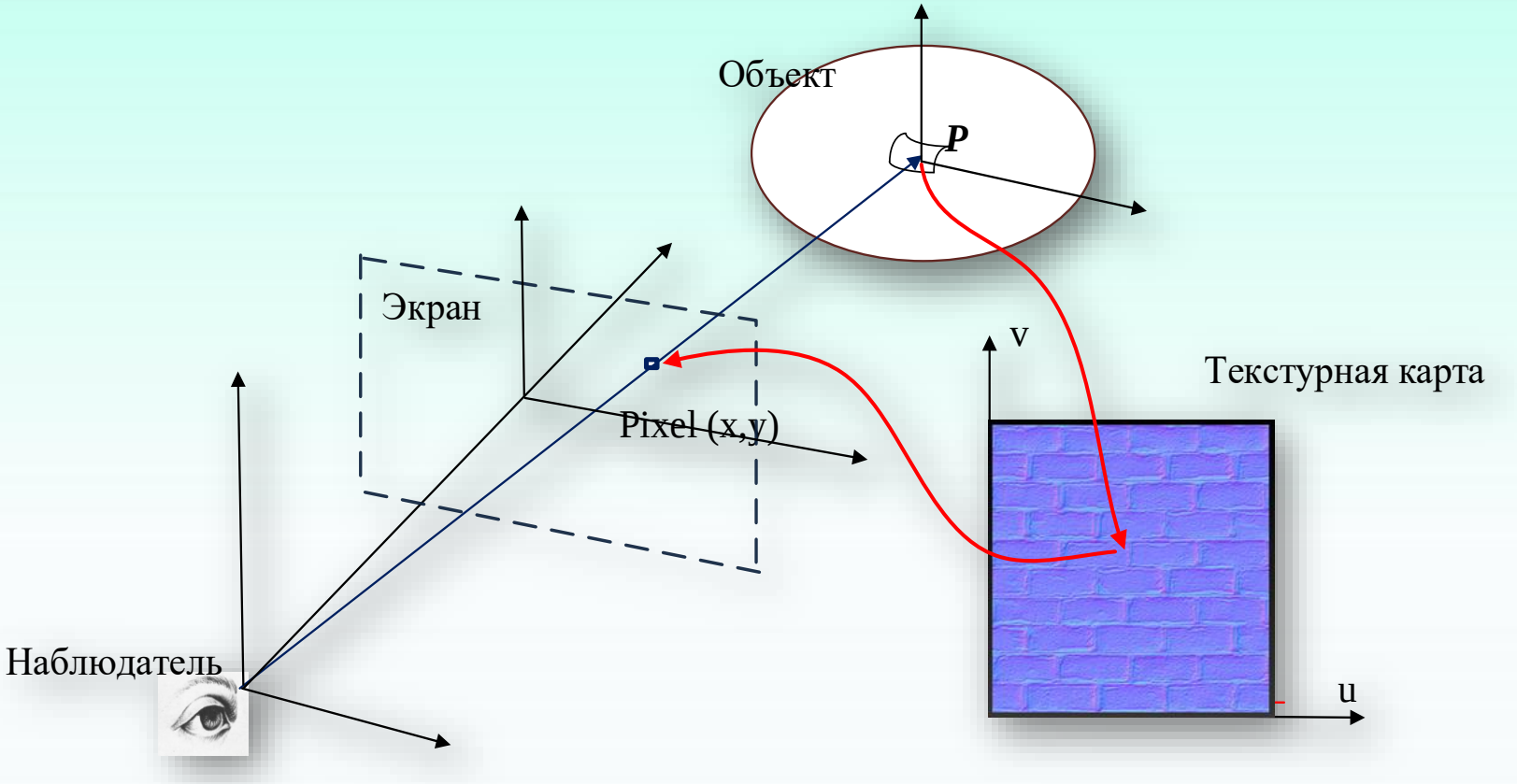
Тогда $s^* = (Y_{pix} - Y_1) / (Y_2 - Y_1)$

$$X_{pix} = (X_r(s^*) - X_l(s^*)) * t + X_l(s^*);$$
$$t^* = (X_{pix} - X_l(s^*)) / (X_r(s^*) - X_l(s^*))$$

Зная t^*, s^* *считаем координаты текстурной карты*

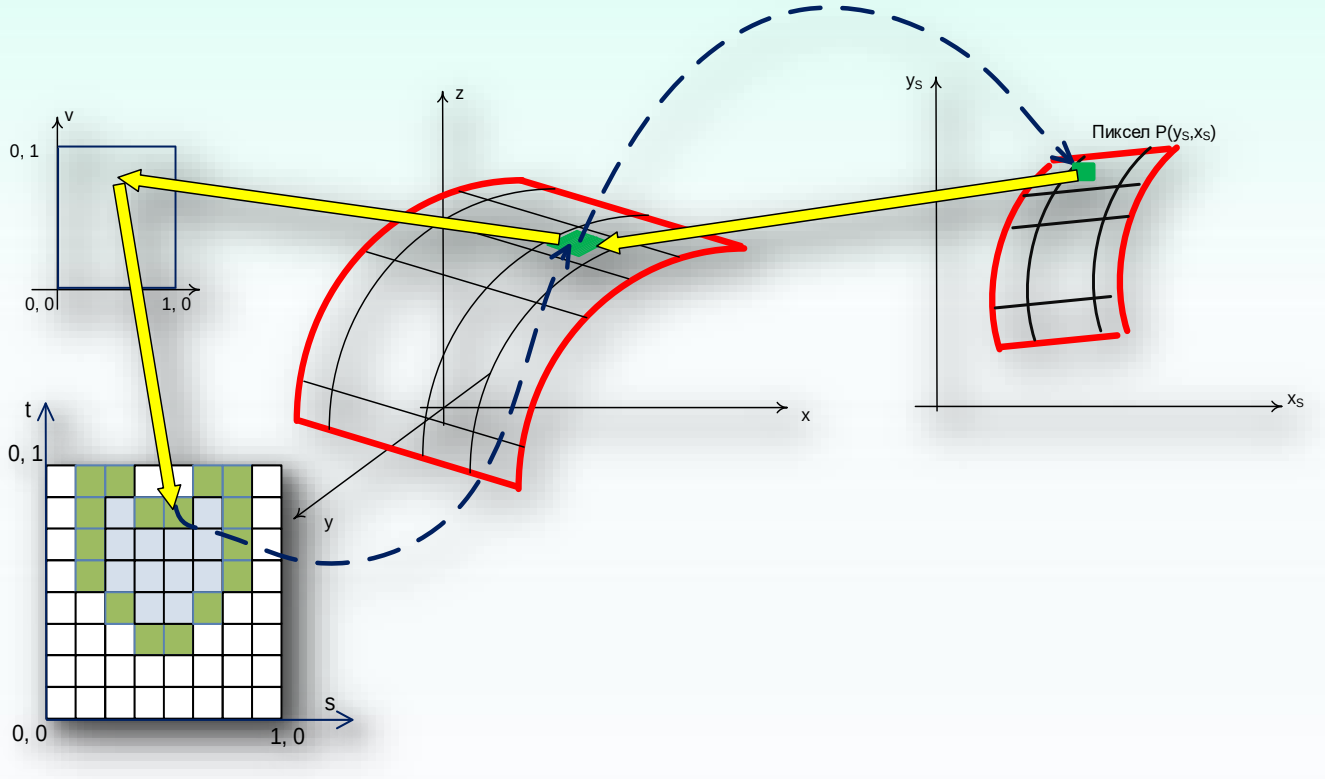
ТЕКСТУРИРОВАНИЕ в 3D
КЛАССИФИКАЦИЯ

ПРОЕКТИВНОЕ ТЕКСТУРИРОВАНИЕ (3D)



ИДЕЯ ПРОЕКТИВНЫХ МЕТОДОВ

Объект задан - *O*(x,y,z) в пространственных координатах **(x,y,z)** или параметрически ***O*(u,v)** – в виде функции параметров **(u,v)**
Экран задан в экранной системе координат **(x_S,y_S)**



ИДЕЯ ПРОЕКТИВНЫХ МЕТОДОВ

Пиксель с координатами центра (x_{Sp}, y_{Sp}).

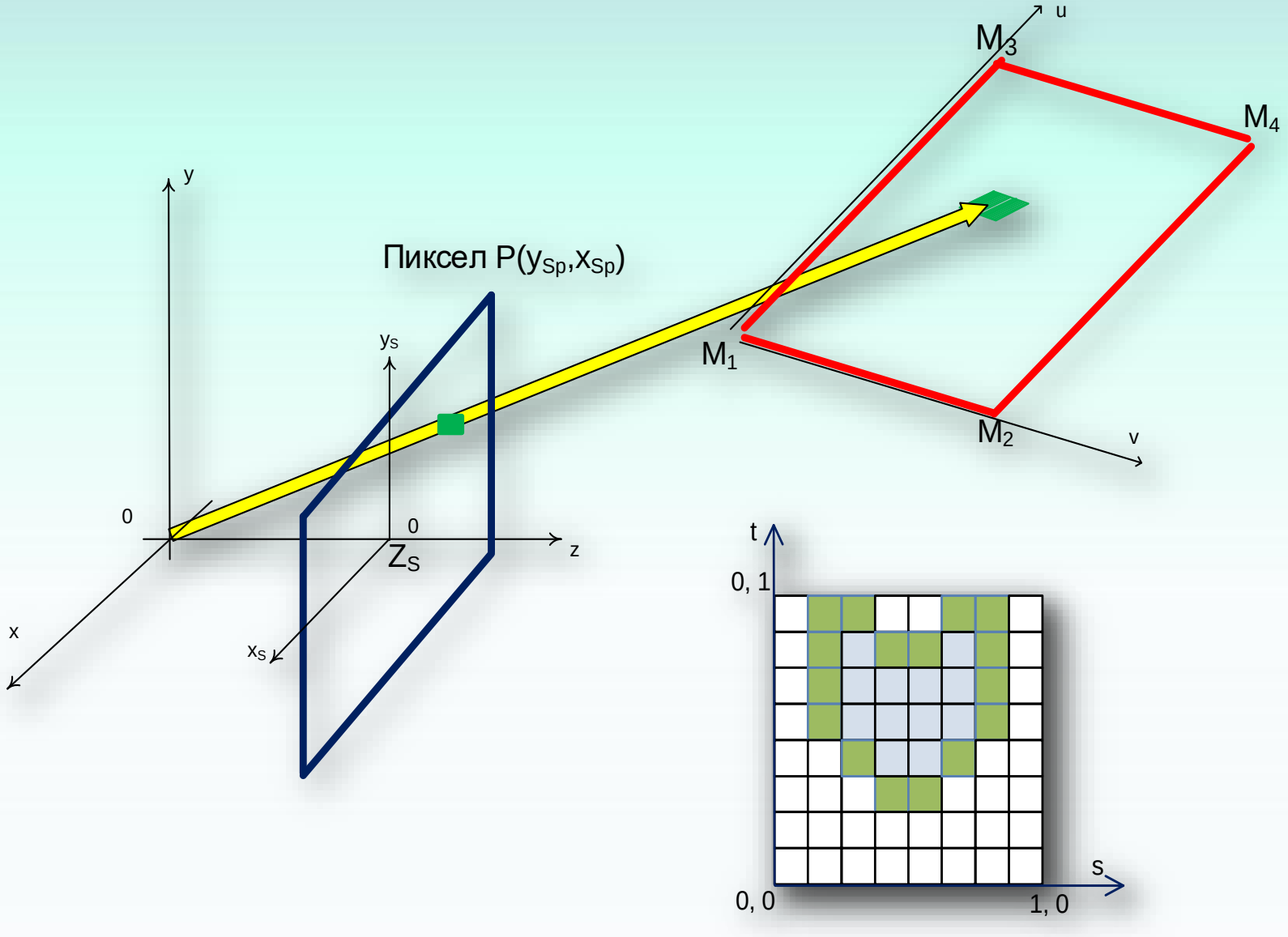
Отображаем на поверхность объекта – находим точку пересечения луча с поверхностью объекта $(x_{Sp}, y_{Sp}) \rightarrow (x_p, y_p, z_p)$.

Находим параметрические координаты точки пересечения $(x_p, y_p, z_p) \rightarrow (v_p, u_p)$.

Находим соответствующий тексель $(v_p, u_p) \rightarrow (t_p, s_p)$.

Закрашиваем пиксел $P(x_{Sp}, y_{Sp})$ цветом $T(t_p, s_p)$.

ОТОБРАЖЕНИЕ НА ПЛОСКОСТЬ



ОТОБРАЖЕНИЕ НА ПЛОСКОСТЬ

Плоскость полигона (правильный четырехугольник) задана его вершинами **M1, M2, M3** с координатами, соответственно:
(x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)

Параметрическое задание произвольной точки на плоскости «внутри» полигона :

$$x = (x_2 - x_1)v + (x_3 - x_1)u + x_1$$
$$y = (y_2 - y_1)v + (y_3 - y_1)u + y_1$$
$$z = (z_2 - z_1)v + (z_3 - z_1)u + z_1$$
$$0 \le v \le 1, 0 \le u \le 1$$

Параметрическое уравнение луча

$$x = (x_{Sp})q, \; y = (y_{Sp})q, \; z = (Z_S)q, \quad q > 1$$

ОТОБРАЖЕНИЕ НА ПЛОСКОСТЬ

Решая систему уравнений

$(x_2-x_1)v + (x_3-x_1)u + x_1 = (x_{Sp})q$

$(y_2-y_1)v + (y_3-y_1)u + y_1 = (y_{Sp})q$

$(z_2-z_1)v + (z_3-z_1)u + z_1 = (Z_S)q$

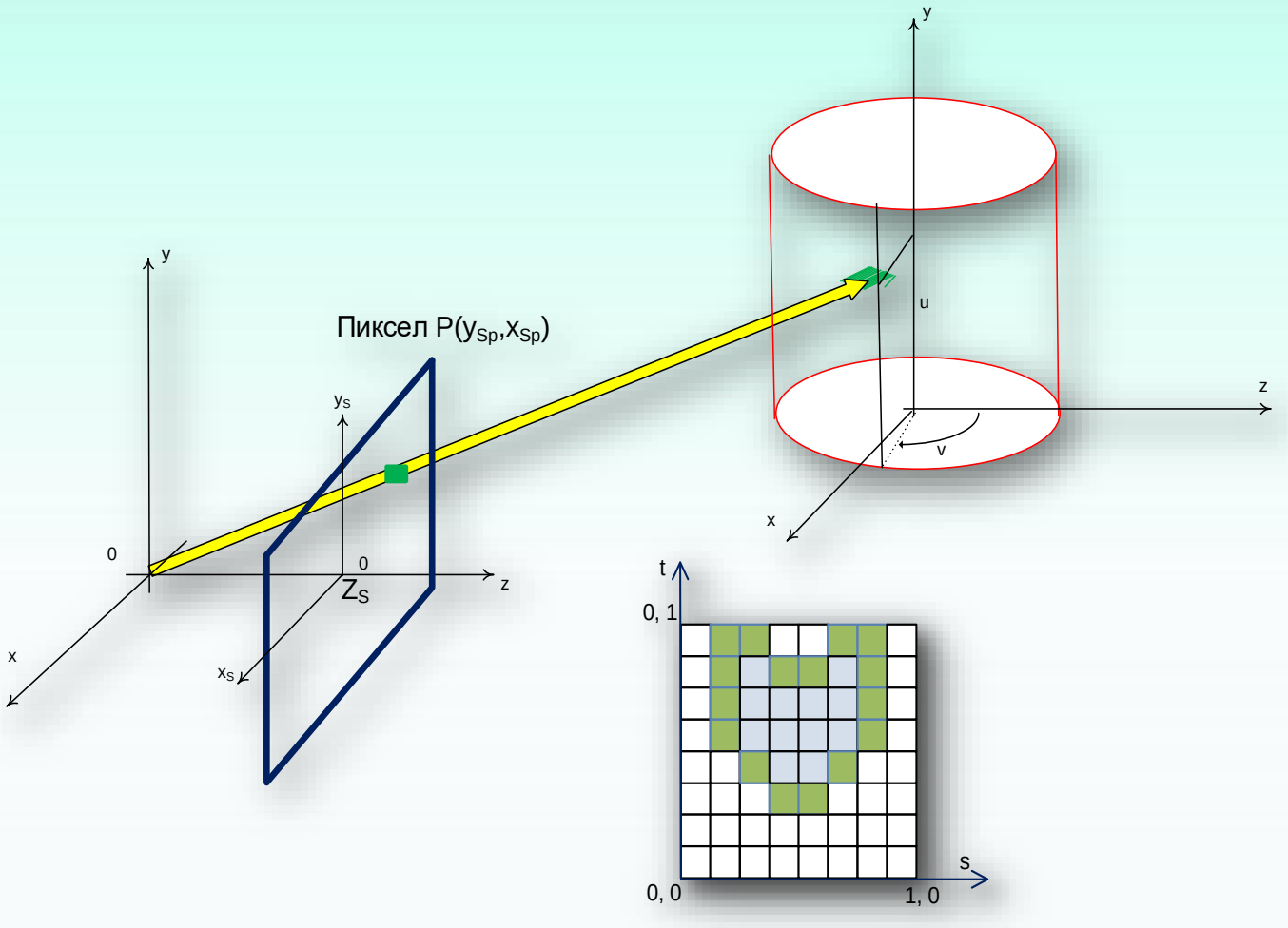
находим параметры точки пересечения q^*, v^*, u^*. Если они удовлетворяют $0 \le v* \le 1, 0 \le u* \le 1$ и $q^*>1$, то точка пересечения попадает в «полигон». Для нормализованной текстуры **(t=v, s=u)**

Цвет $P(x_{Sp}, y_{Sp}) = T(v^*, u^*)$

Аналогично с плоским треугольником

(см. барицентрический тест для определения v, u)

ОТОБРАЖЕНИЕ на ЦИЛИНДР



ОТОБРАЖЕНИЕ на ЦИЛИНДР

Каноническое параметрическое задание цилиндрической поверхности

$x = \sin(2\pi v)$

$y = H u$

$z = \cos(2\pi v)$

Зная координаты точки пересечения луча с цилиндром x^*, y^*, z^*, находим

$v^* = 1/2\pi \arcsin z^*$

$u^* = 1/H\, y^*$

Цвет $P(x_{Sp}, y_{Sp}) = T(v^*, u^*)$

ОТОБРАЖЕНИЕ на СФЕРУ

Каноническое параметрическое задание сферической поверхности

$$x = R\ cos\ (2\pi\ u)\ sin\ (2\pi\ v)$$
$$y = R\ sin\ (\pi\ u\ )$$
$$z = R\ cos\ (2\pi\ u)\ \ cos\ (2\pi\ v)$$

Зная координаты точки пересечения луча с сферой x^*, y^*, z^*, находим

$$u^* = 1/\pi\ arcsin\ (y^*/R)$$
$$v^* = 1/2\pi\ arcsin\ (x^*/(R\ cos\ (\pi\ u^*\ )))$$

Цвет $P(x_{Sp}, y_{Sp}) = T(v^*, u^*)$

ОТОБРАЖЕНИЕ на ТОР

Каноническое параметрическое задание тора поверхности

x = ? ? ?

y = ? ? ?

z = ? ? ?

Найти самостоятельно !!!

Зная координаты точки пересечения луча с тором **x*, y*, z*** , находим

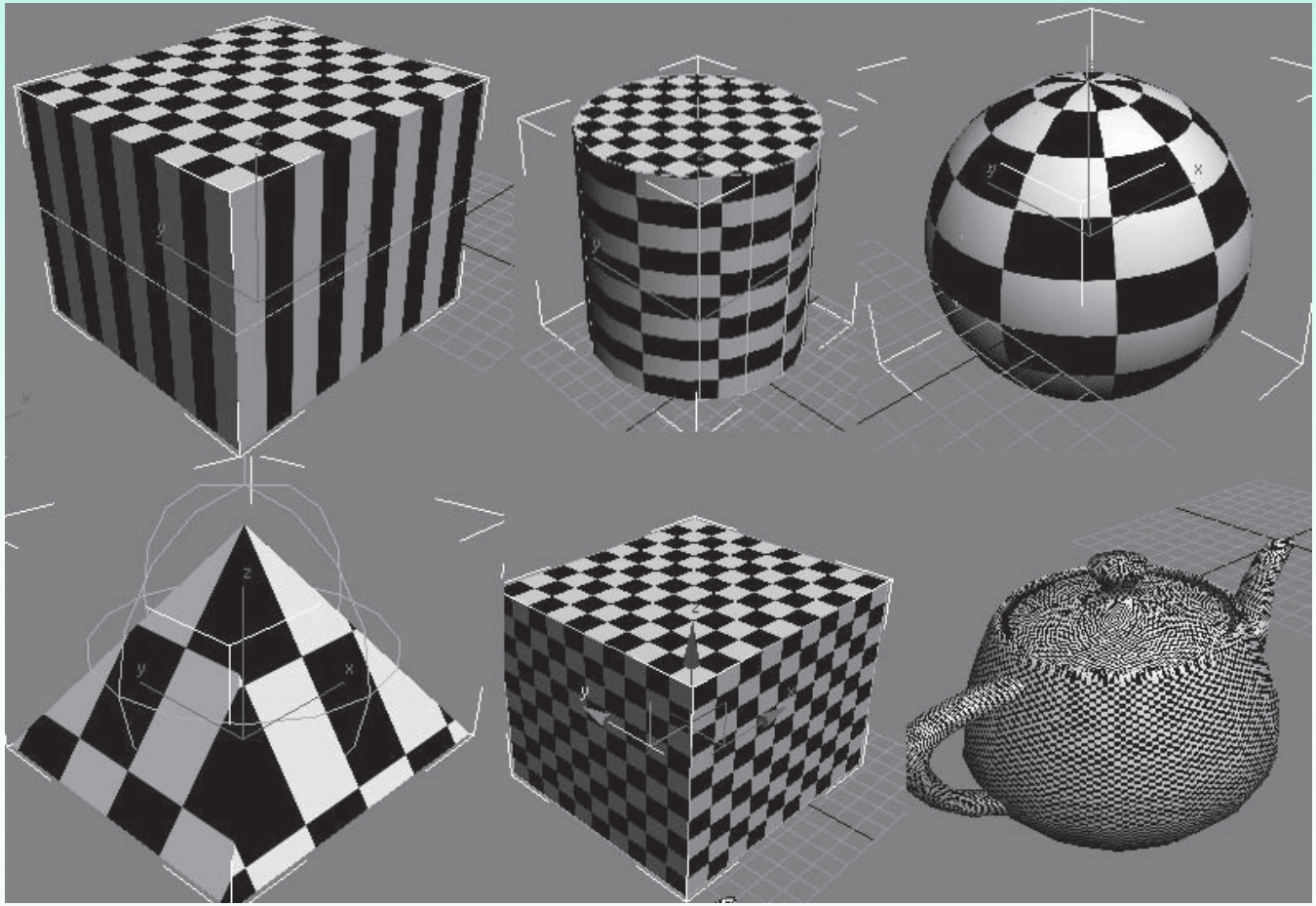
v*= ? ? ?

u* = ? ? ?

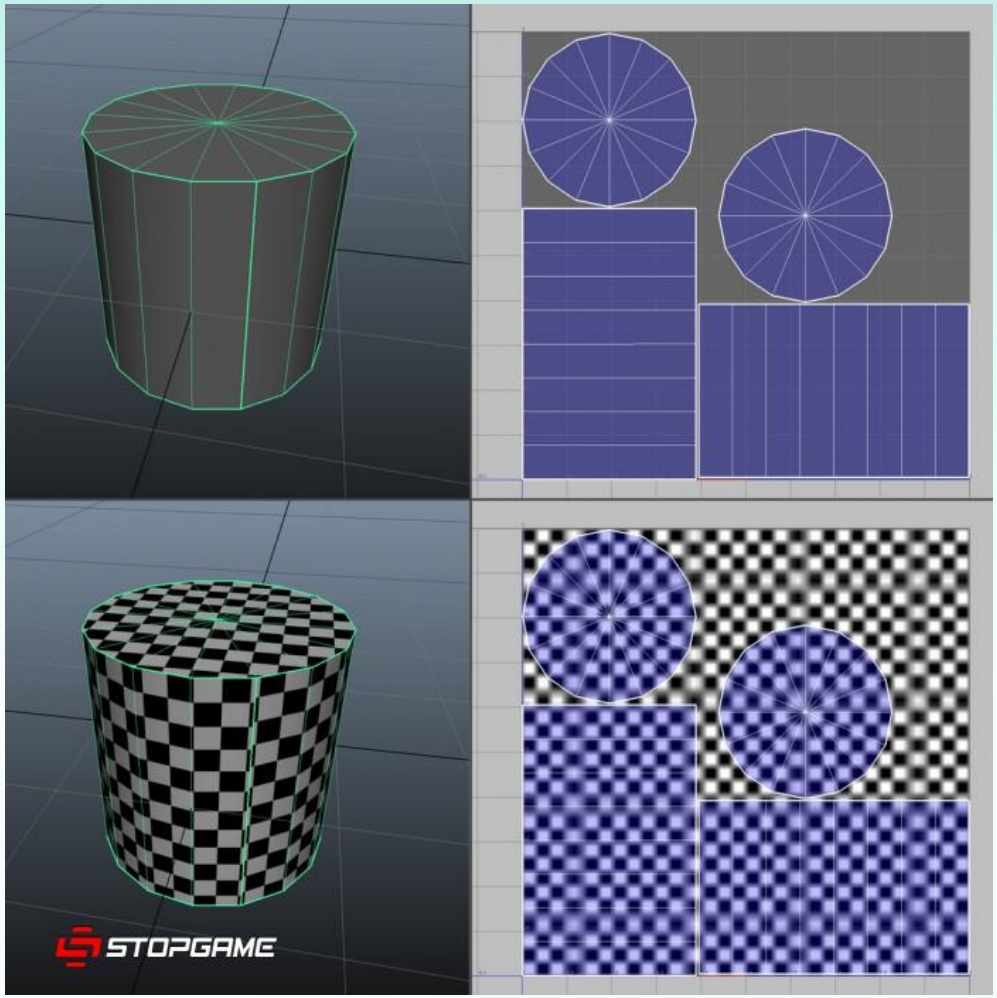
Найти самостоятельно !!!

Цвет $\mathbf{P(x_{Sp}, y_{Sp}) = T(v^*, u^*)}$

ПРОЕКТИВНЫЙ ПОДХОД

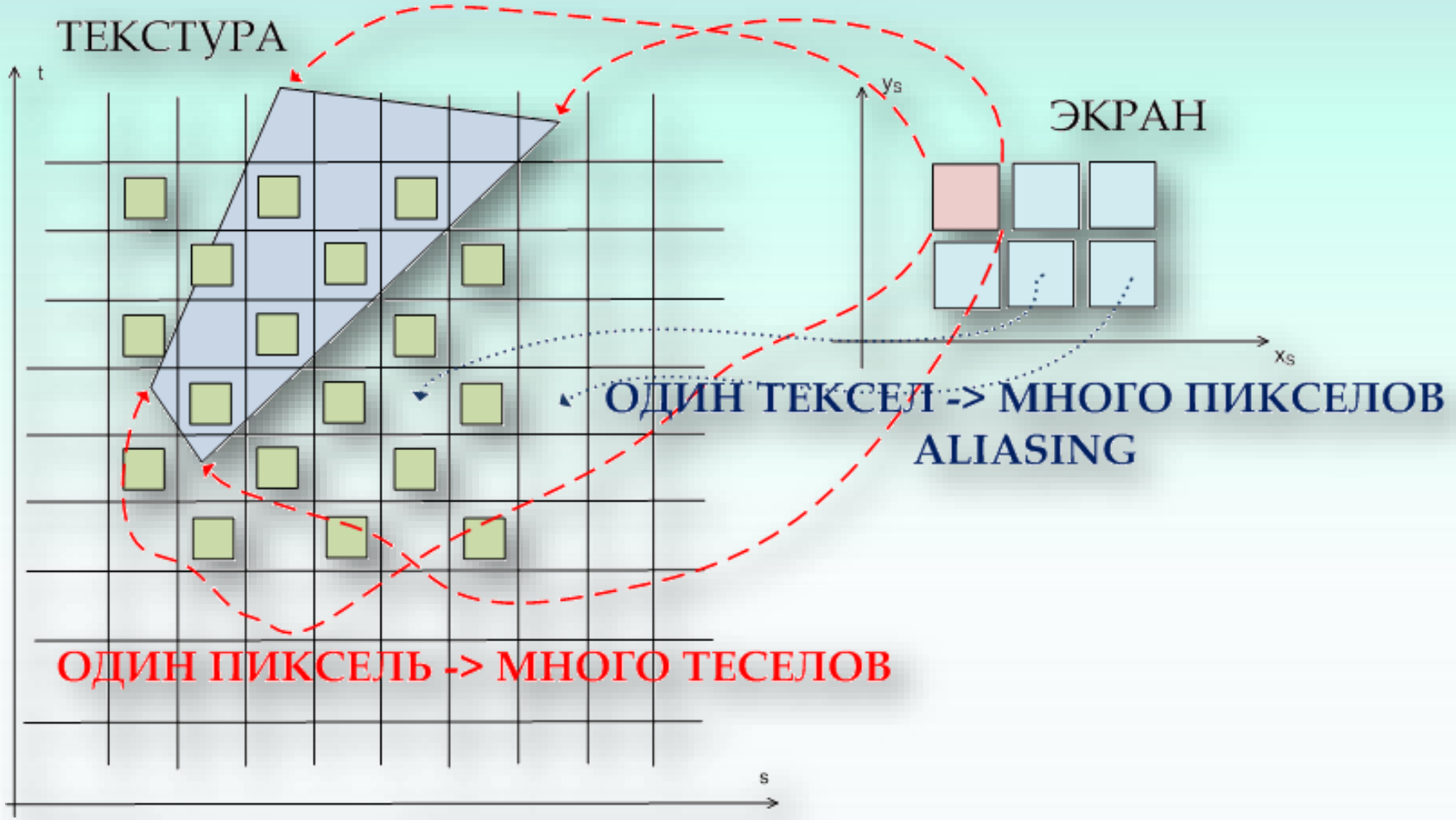


ПРОЕКТИВНЫЙ ПОДХОД



UV - развертки («увихи»)

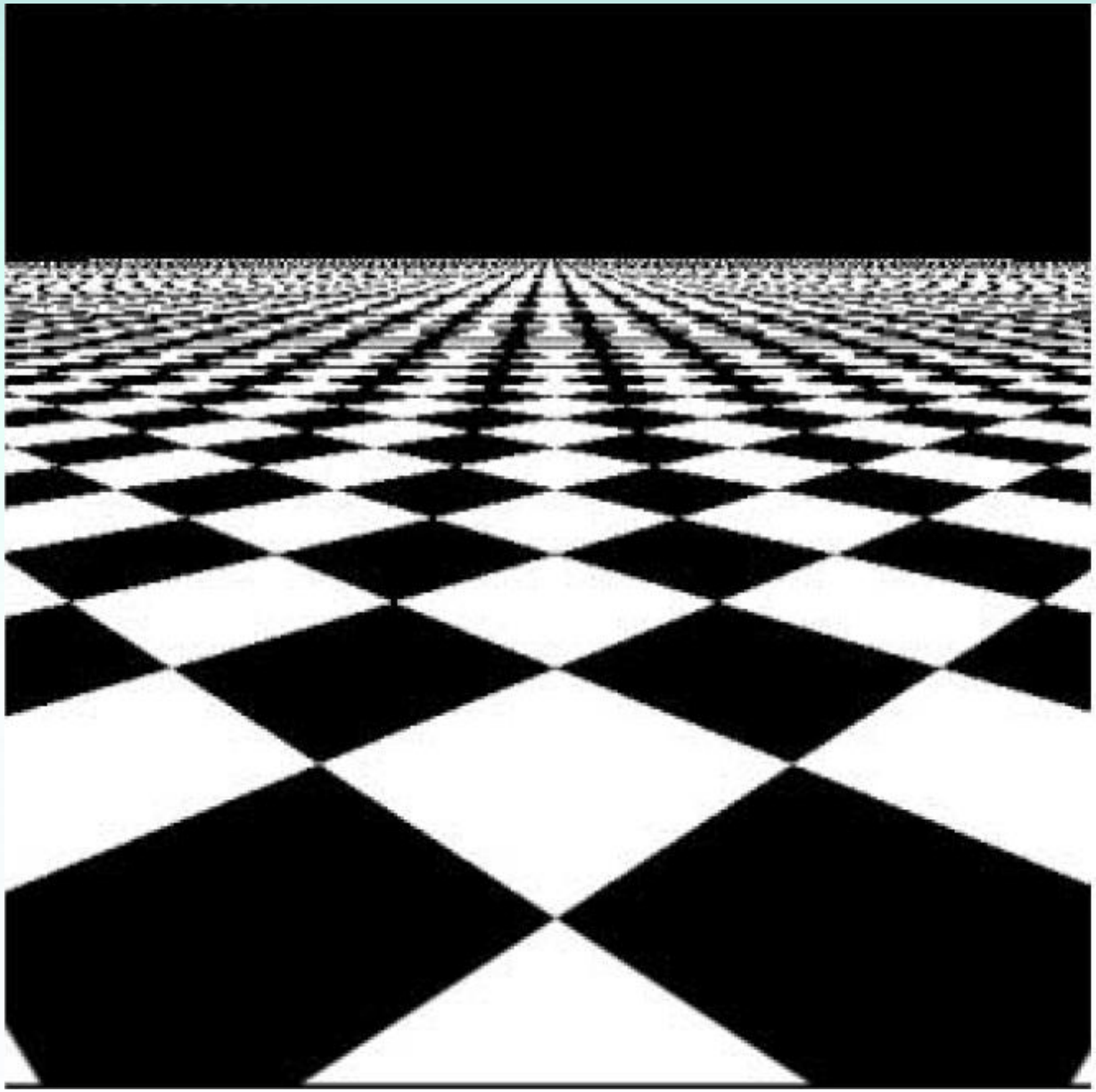
АРТЕФАКТЫ + ФИЛЬТРАЦИЯ



А. Текстура маленькая / объект близко – как выбрать ЦВЕТ пикселя ?

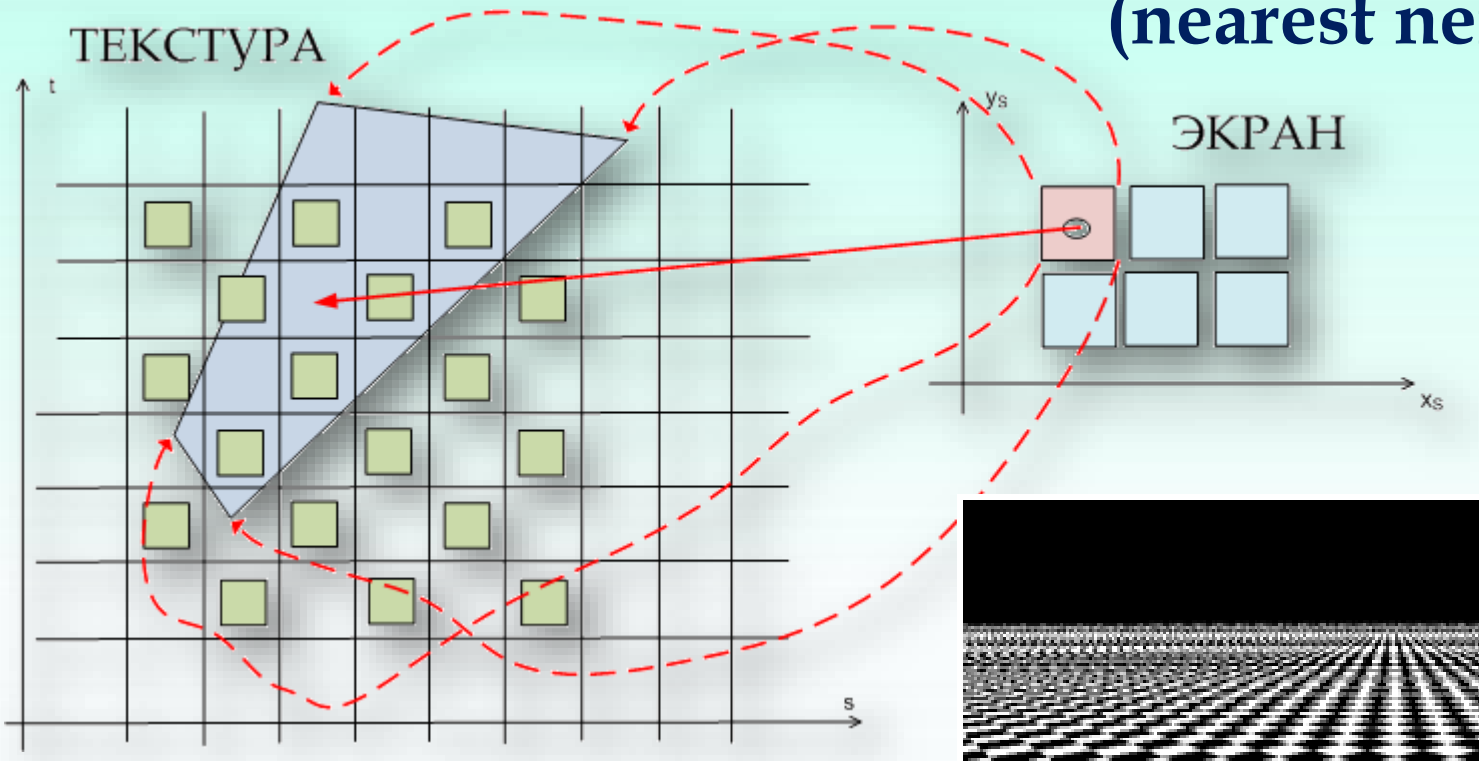
Б. Разрешение текстуры слишком высокое (текстура большая, объект далеко) – случайный цвет пикселя ?

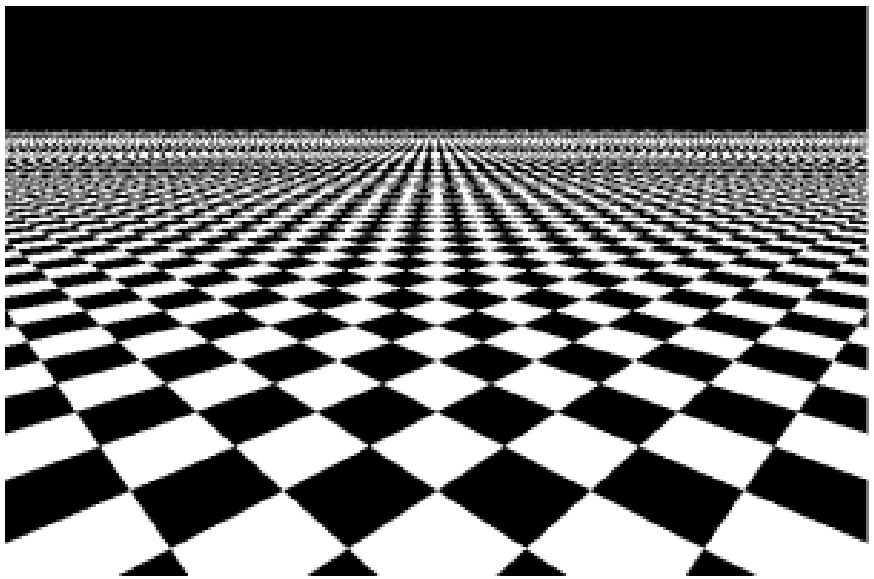
АРТЕФАКТЫ + ФИЛЬТРАЦИЯ



АРТЕФАКТЫ + ФИЛЬТРАЦИЯ

Ближайший (центральный) тексель (nearest neighbor)

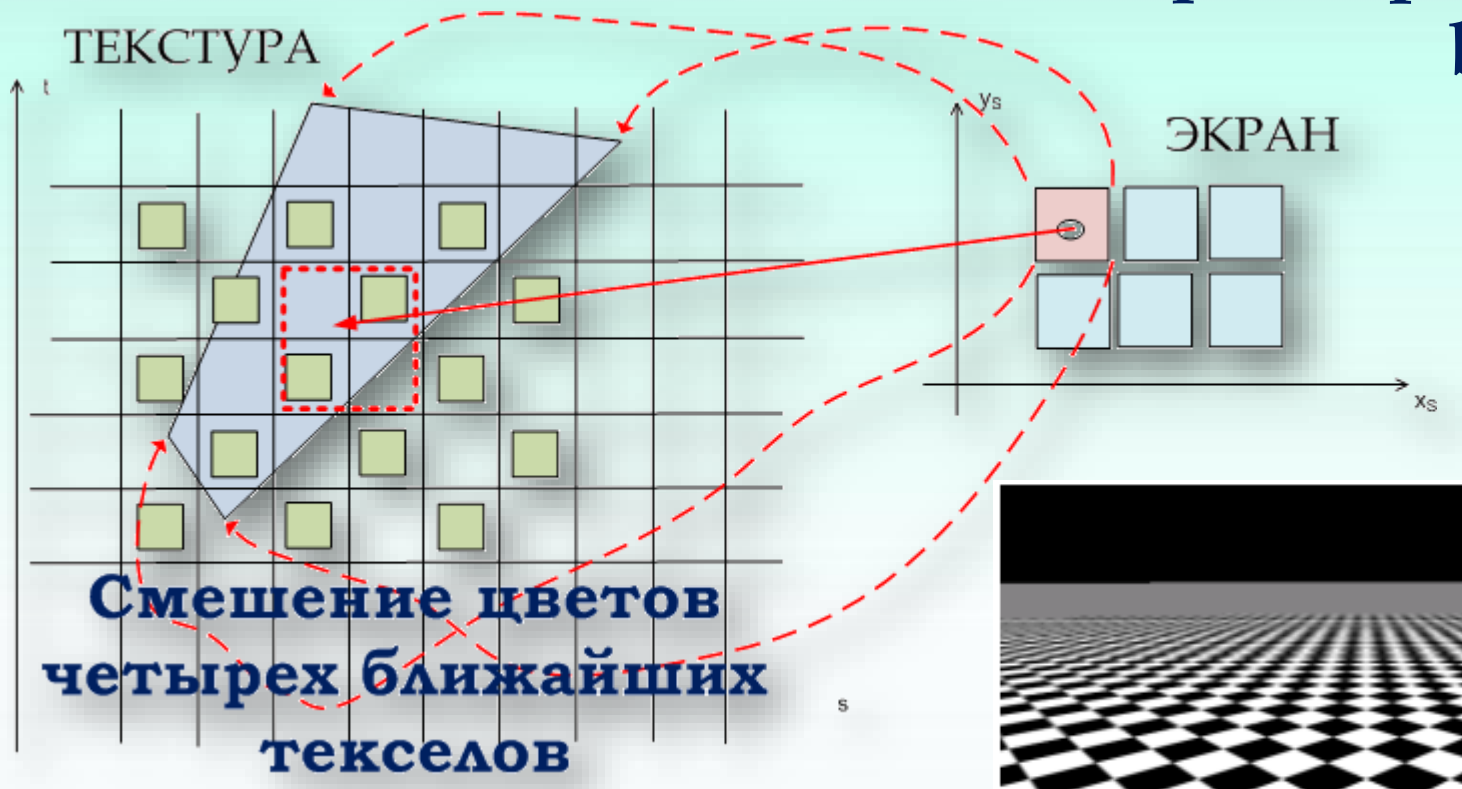


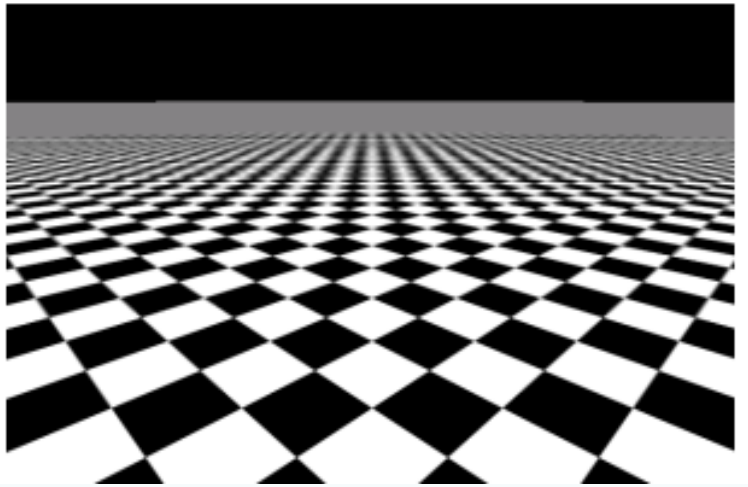


Быстро
Низкое качество

АРТЕФАКТЫ + ФИЛЬТРАЦИЯ

ФИЛЬТРАЦИЯ Билинейная фильтрация bilinear





Быстро

Достаточно качественно

Плохо – когда смотрим на плоскость под углом

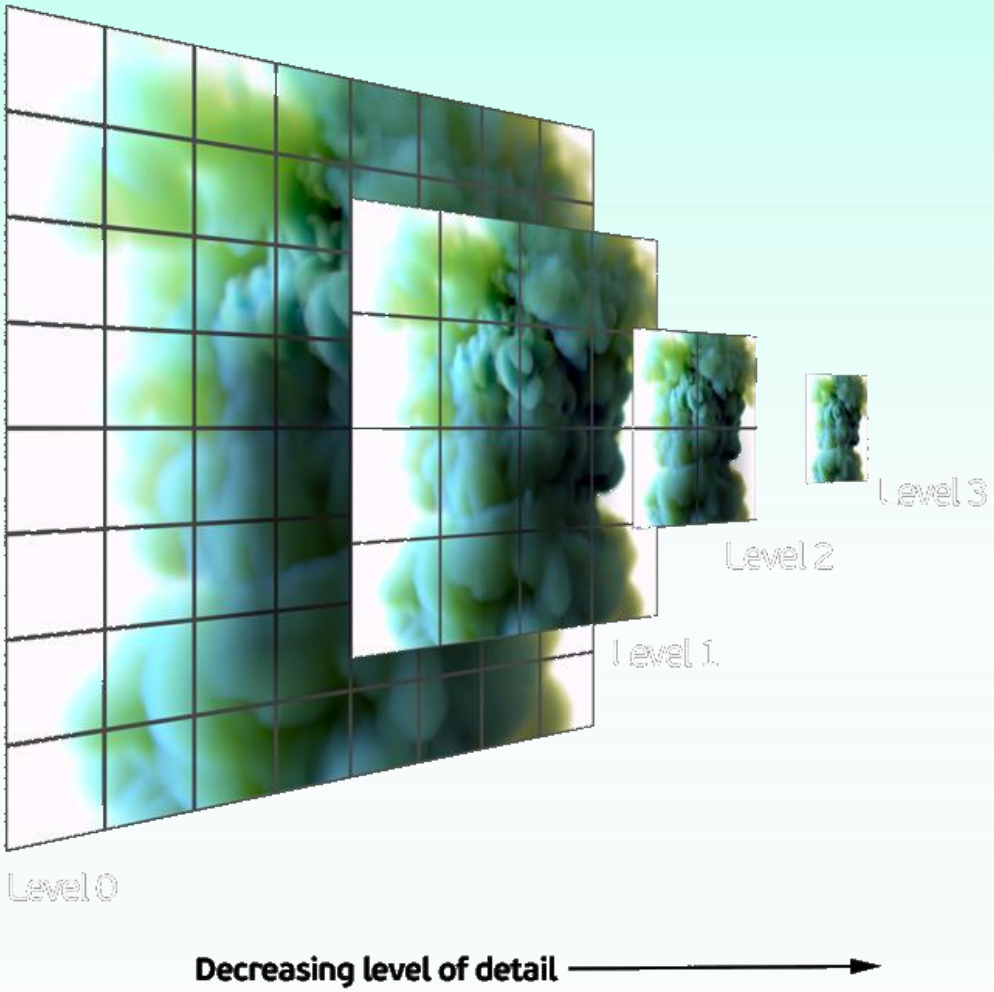
MIP mapping + LOD

MIP (multum in parvo-много в малом) – использование нескольких копий одной текстуры с разной детализацией

ИДЕЯ: Создаётся **MIP-пирамида** — последовательность текстур с разрешением от максимального до 1×1. Например: 1024×1024, 512×512 , 256×256, 128×128, …, 2×2, 1×1. Каждая из этих текстур называется MIP-уровнем (MIP level) или уровнем детализации (level of detail, **LOD**).

На всех этих текстурах находится **одно и то же** изображение.

MIP mapping + LOD



MIP mapping + LOD

1. **Nearest**: округляем LOD до ближайшего уровня, а (s, t) до ближайшего тексела
2. **Linear:** уровни детализации [LOD] и [LOD+1], на каждом берем цвет точки (s, t) округляя до ближайшего тексела, цвет пиксела определяем линейной интерполяцией цветов [LOD] и [LOD+1] по вещественному значению уровня LOD
3. **Trilinear:** аналогичен Linear, но значения цвета на обоих уровнях, рассчитываются по методу Bilinear.

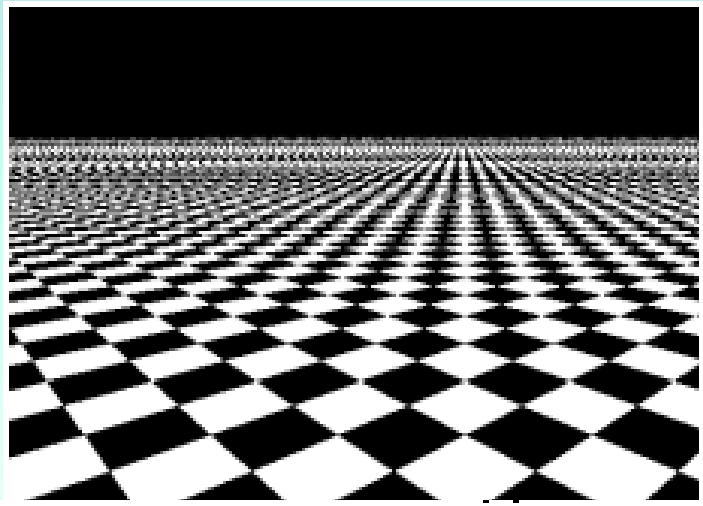
АРТЕФАКТЫ + ФИЛЬТРАЦИЯ

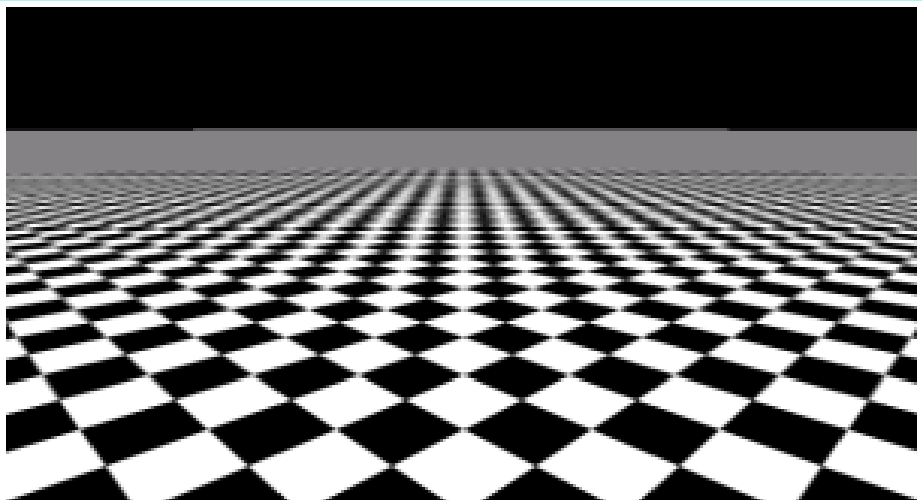
Анизотропная фильтрация (anisotropic)

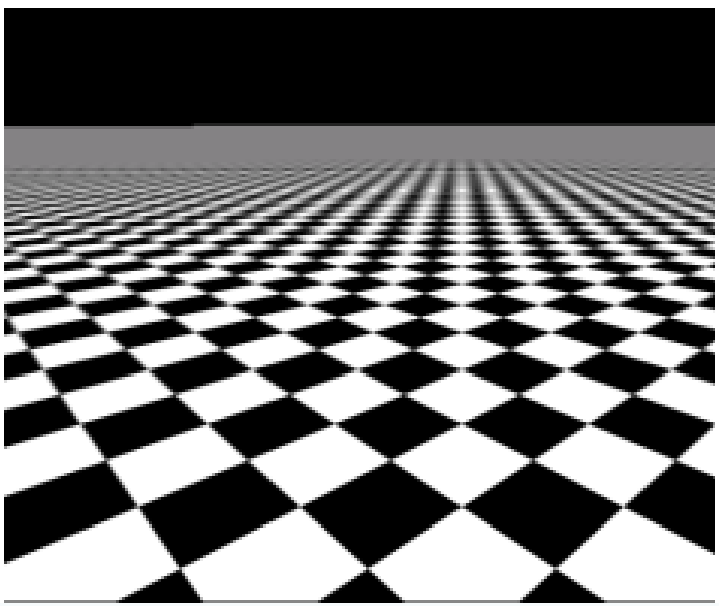
При наложении текстуры на плоскость учитывается ее положение в пространстве.

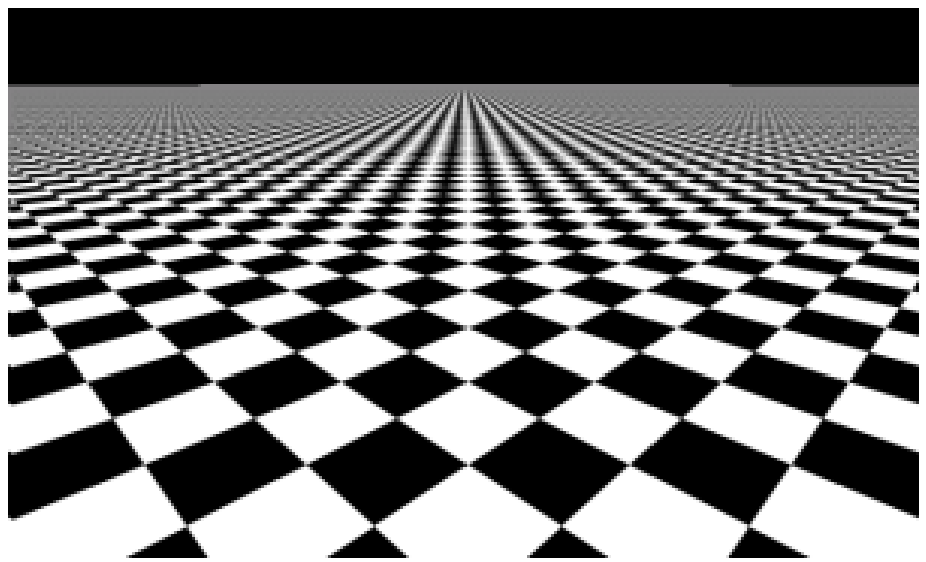
Для определения цвета пикселя используется выборка из 16 или 32 текселя.

СРАВНЕНИЕ ФИЛЬТРАЦИЙ









Ближайший сосед **Билинейная**

Трилинейная **Анизотропная**

ПРОЦЕДУРНОЕ ТЕКСТУРИРОВАНИЕ

Модель Фонга:

$$I = I_a K_a + \sum_{i=1}^{n} \frac{I_i}{{D_i}^2 + k} [K_d \cos({\theta'}_i) + K_m cos^s(\alpha_i)]$$

Изменяя «попиксельно» любой параметр можно добиться некоторого псевдотекстурирования закрашиваемой поверхности

ПРОЦЕДУРНОЕ ТЕКСТУРИРОВАНИЕ

Displacement Maps – текстура содержит смещение отображаемой поверхности по высоте

Normal Maps – текстура содержит «приращение» нормали

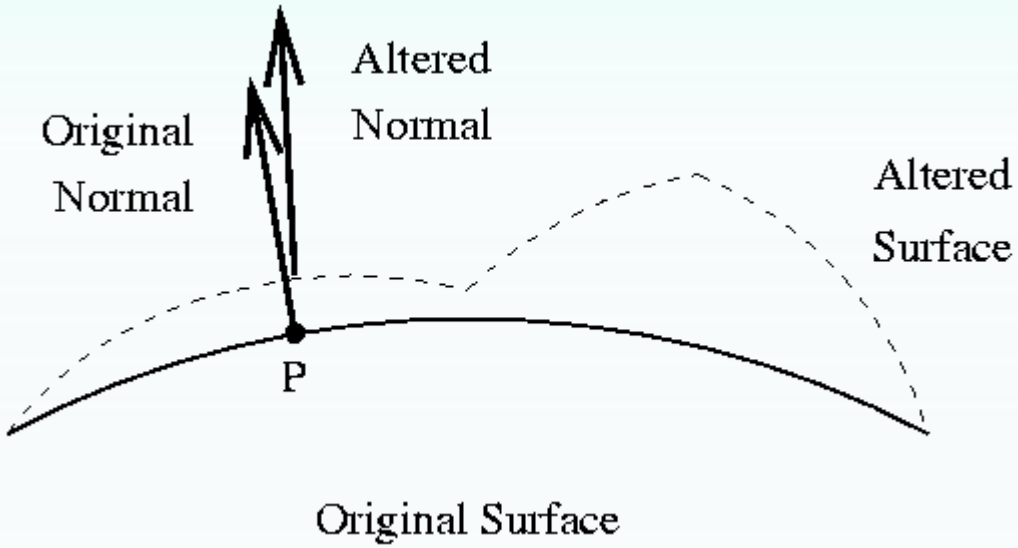
Bump Maps (рельефное текстурирование) - текстура содержит смещение пикселя по высоте и изменение нормали

NORMAL MAPPING

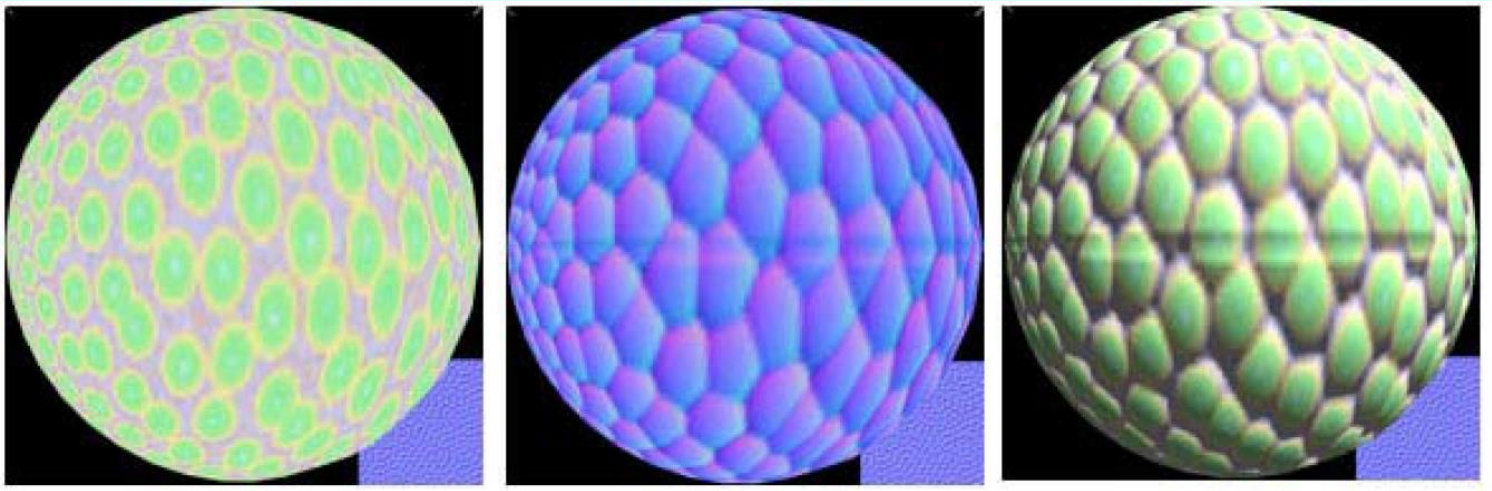
«Деформируем» вектор нормали

$$\theta' = \arccos(< V', N >)$$

Текстура содержит вектор «отклонение нормали» (r,g,b текстры трактуется как x,y,z отклонения нормали)



BUMP MAPPING



Вопросы для экзамена

ТЕМА: ТЕКСТУРИРОВАНИЕ

1. Суть проективного текстурирования.
2. Проектирование на плоскость.
3. Проектирование на цилиндр.
4. Проектирование на сферу.
5. Артефакты при мэппировании. Фильтрация.
6. Mip mapping. Трилинейная фильтрация.
7. Процедурное текстурирование. Normal mapping

Литература:
http://www.ray-tracing.ru/

END # 09