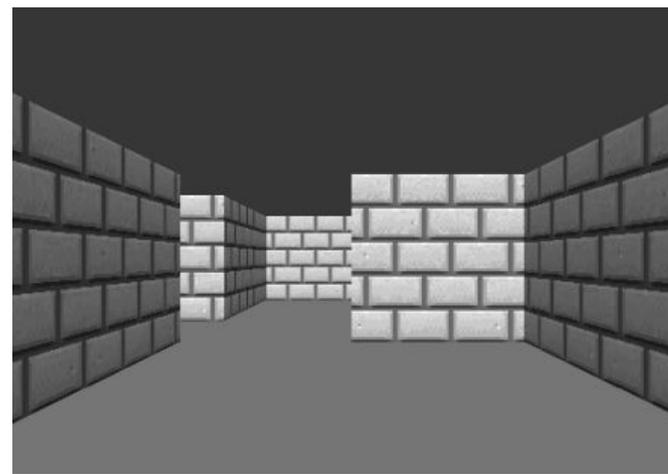
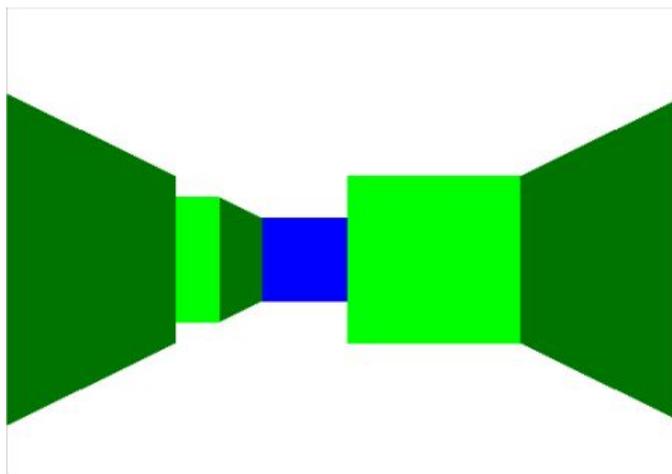
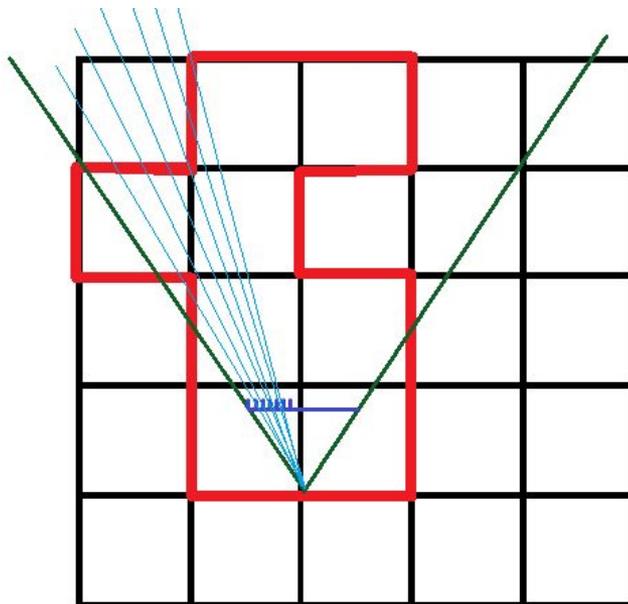
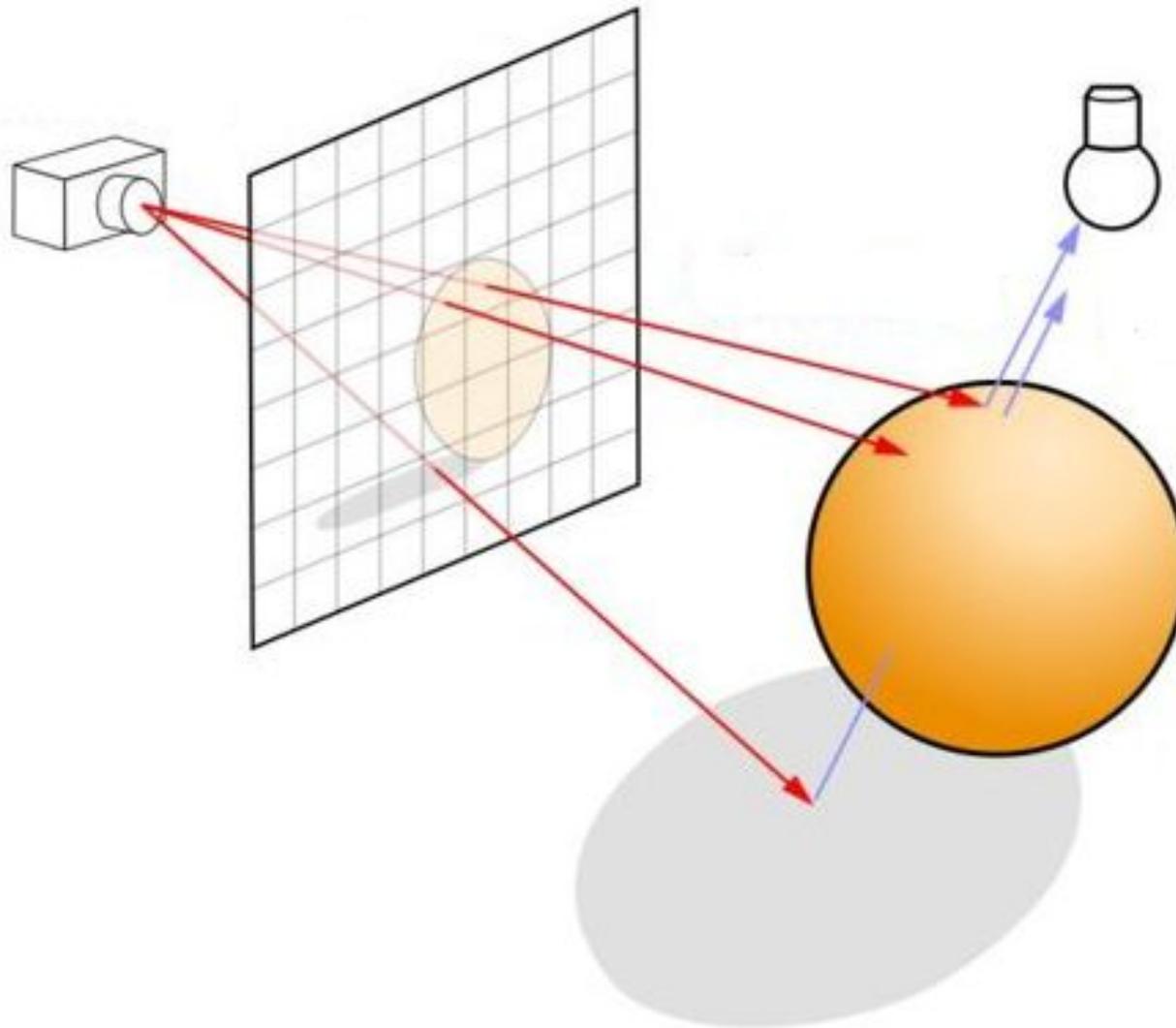


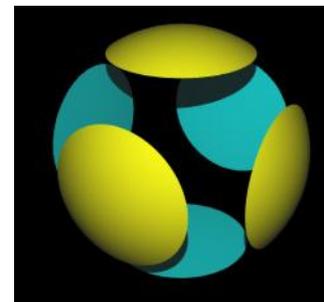
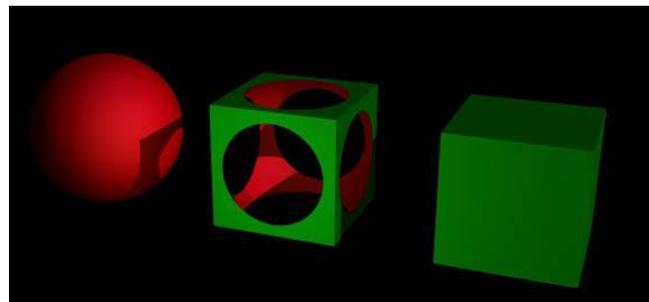
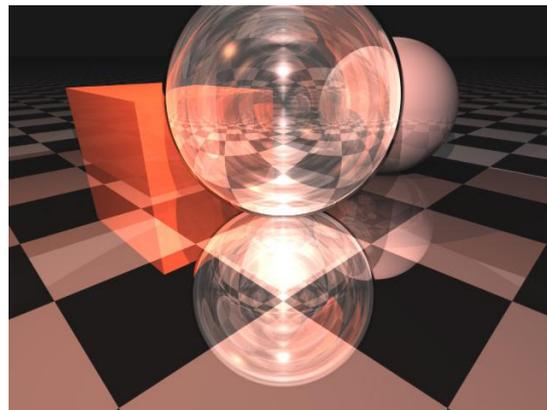
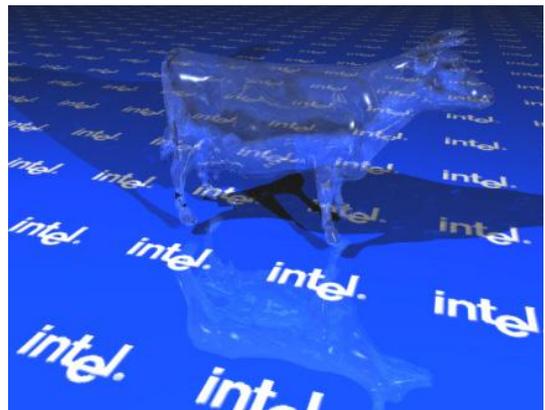
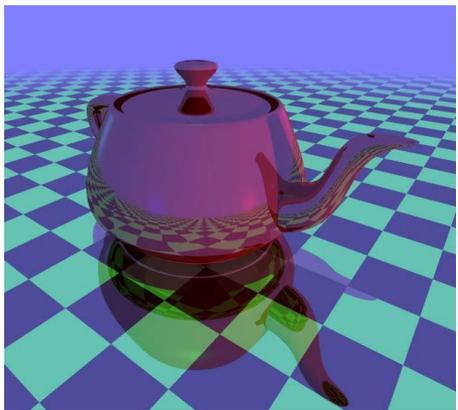
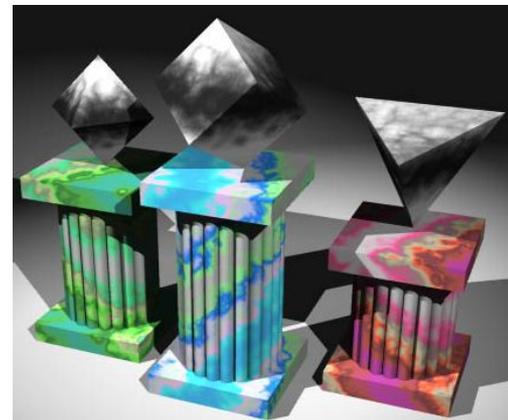
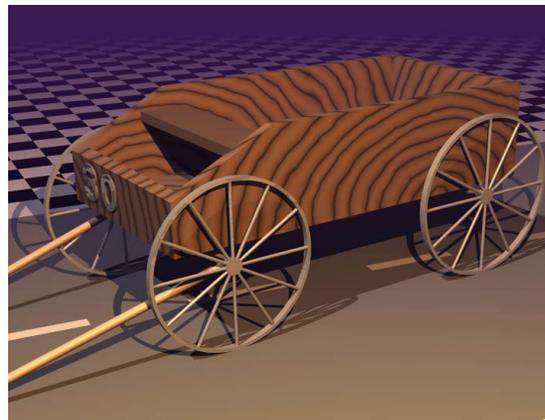
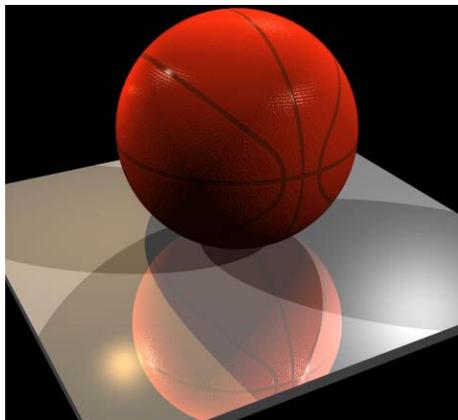
Введение в алгоритм трассировки лучей

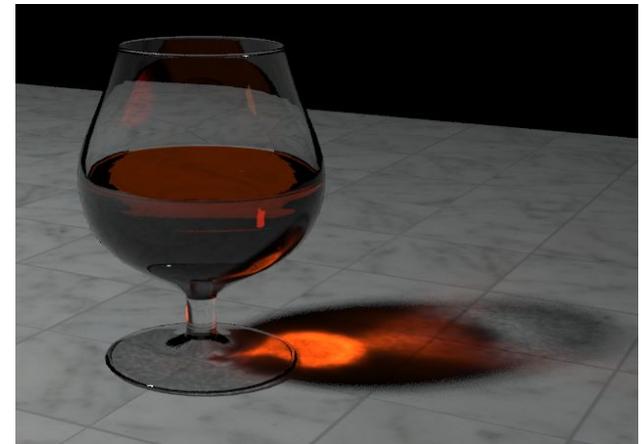
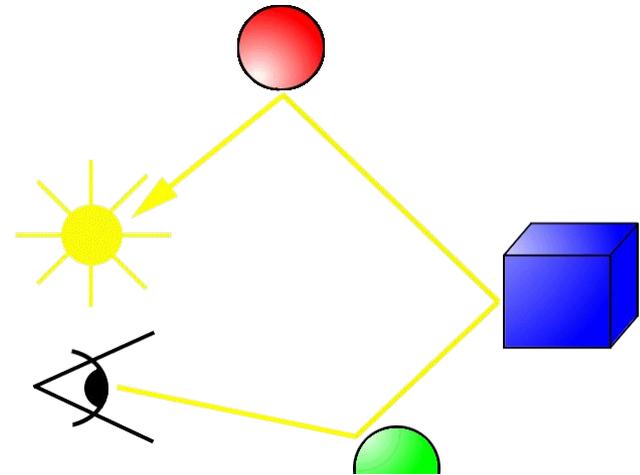
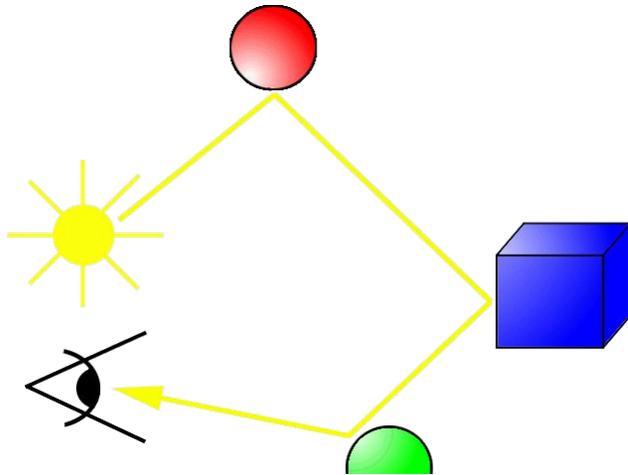
URL: <http://www.school30.spb.ru/cgsg/cgc/>

E-mail: CGSG@yandex.ru



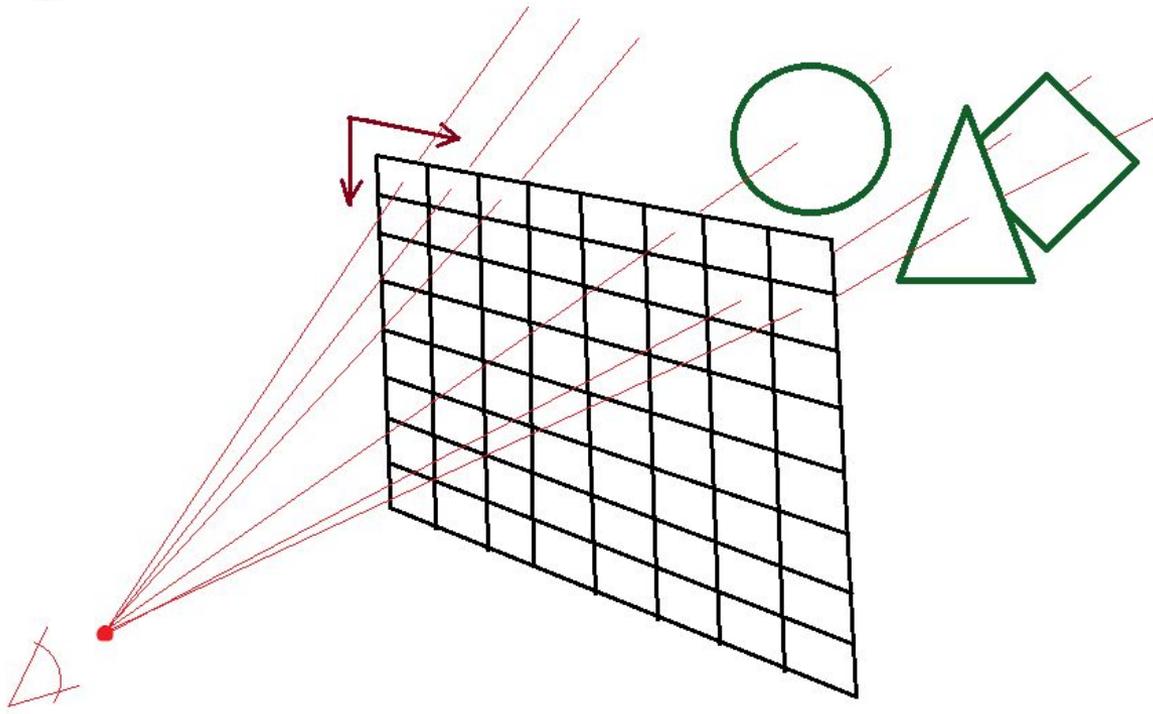
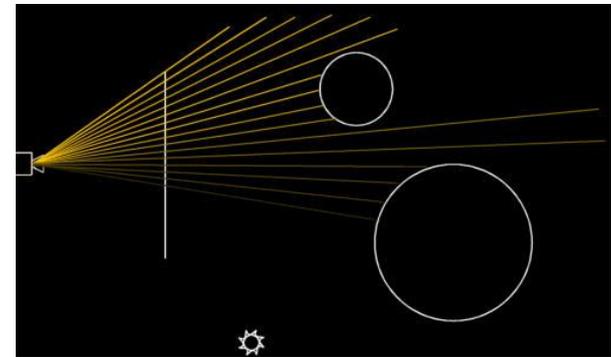






Луч: $\vec{P} = \vec{O} + \vec{D} \cdot t$

$$\begin{cases} x = O_x + D_x \cdot t \\ y = O_y + D_y \cdot t \\ z = O_z + D_z \cdot t \end{cases}$$



$$\vec{A} = \vec{N} \cdot Dist,$$

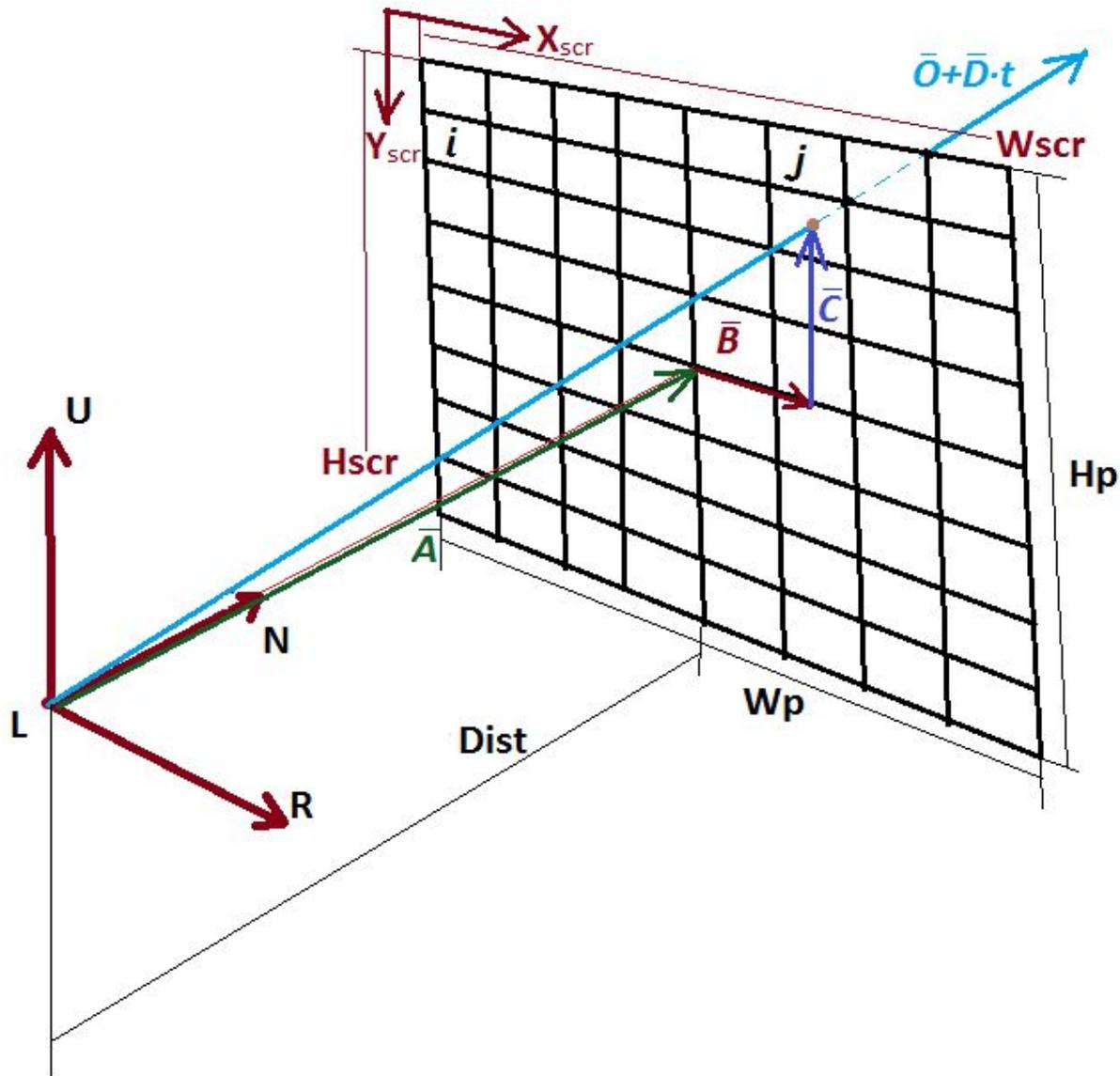
$$\vec{B} = \vec{R} \cdot \frac{(j + 0.5 - Wscr / 2) \cdot Wp}{Wscr}$$

$$\vec{C} = \vec{U} \cdot \frac{-(i + 0.5) + Hscr / 2}{Hscr} \cdot Hp$$

$$\vec{X} = \vec{A} + \vec{B} + \vec{C}$$

$$\vec{O} = \vec{L} + \vec{X}$$

$$\vec{D} = Normalize(\vec{X})$$



- Объекты вида:

$$F(x, y, z) = 0$$

- Подставляем уравнение луча:

$$F(Ox + Dx \cdot t, Oy + Dy \cdot t, Oz + Dz \cdot t) = 0$$

- И решаем относительно t
($t > 0$)

```

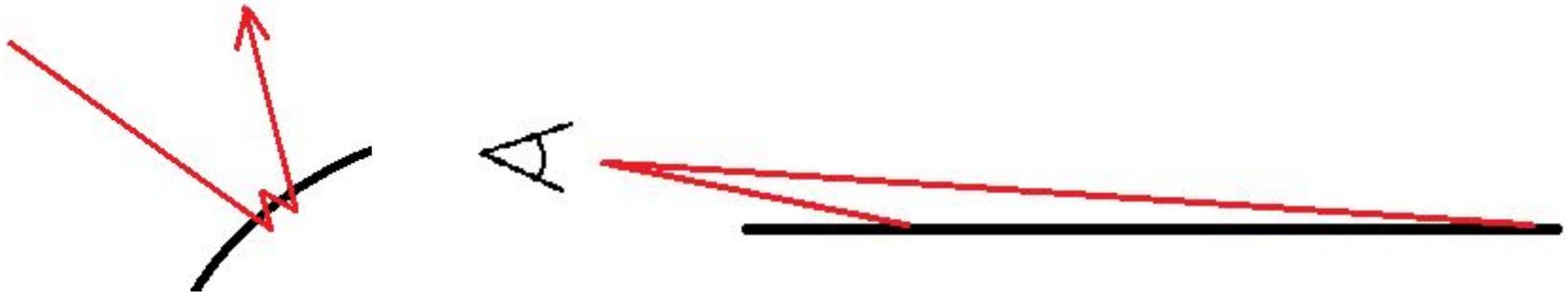
if (value == 0)
...
if (value > 0)
...
    
```

```

Threshold = 0.000001;
if (ABS(value) < Threshold)
...
if (value > Threshold)
...
    
```

~~$$Ray = \vec{P}_{\text{пересечения}} + \vec{R}_{\text{отраженный}} \cdot t$$~~

$$Ray = (\vec{P}_{\text{пересечения}} + \vec{R}_{\text{отраженный}} \cdot Threshold) + \vec{R}_{\text{отраженный}} \cdot t$$



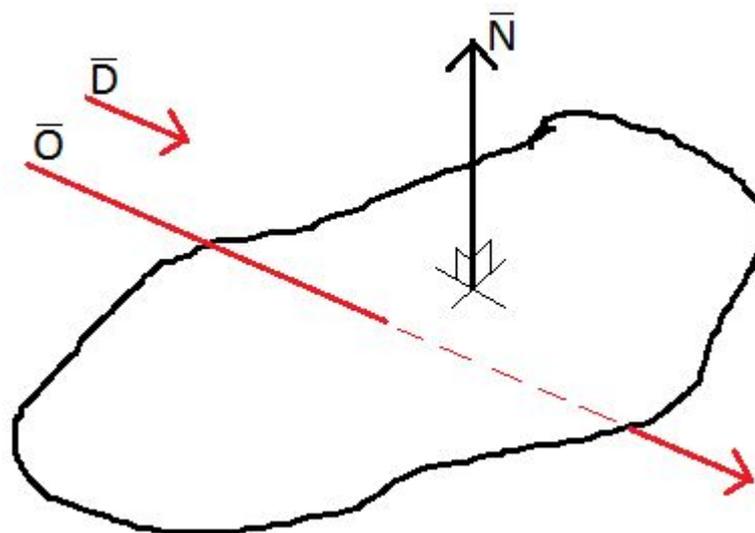
$$\vec{N} = (A, B, C)$$

$$\vec{P} = (x, y, z)$$

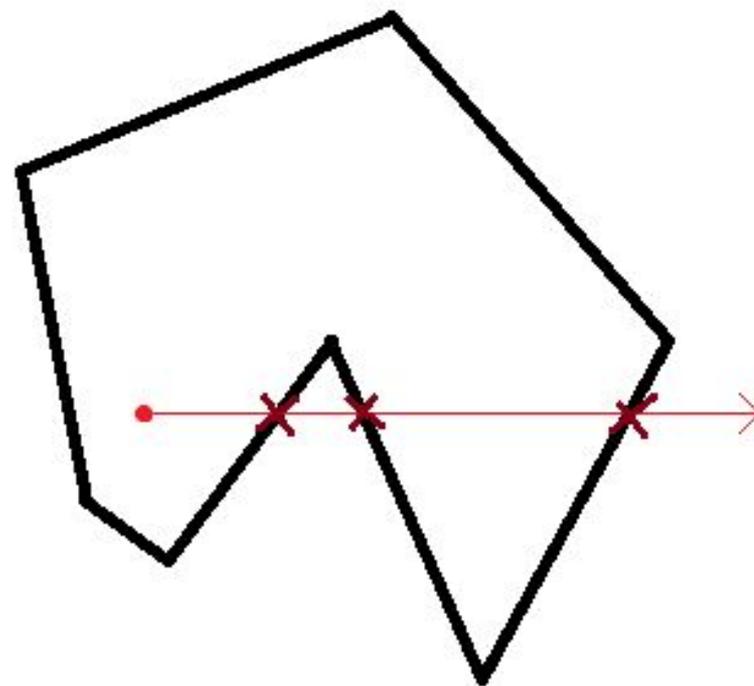
$$F(x, y, z) = A \cdot x + B \cdot y + C \cdot z + D = 0$$

$$A \cdot (Ox + Dx \cdot t) + B \cdot (Oy + Dy \cdot t) + C \cdot (Oz + Dz \cdot t) + D = 0$$

$$t = \frac{-(A \cdot Ox + B \cdot Oy + C \cdot Oz + D)}{A \cdot Dx + B \cdot Dy + C \cdot Dz}, \quad t = \frac{-(\vec{N} \cdot \vec{P} + D)}{\vec{N} \cdot \vec{D}}$$



1. Пересечение с плоскостью
2. Проверка принадлежности точки пересечения многоугольнику



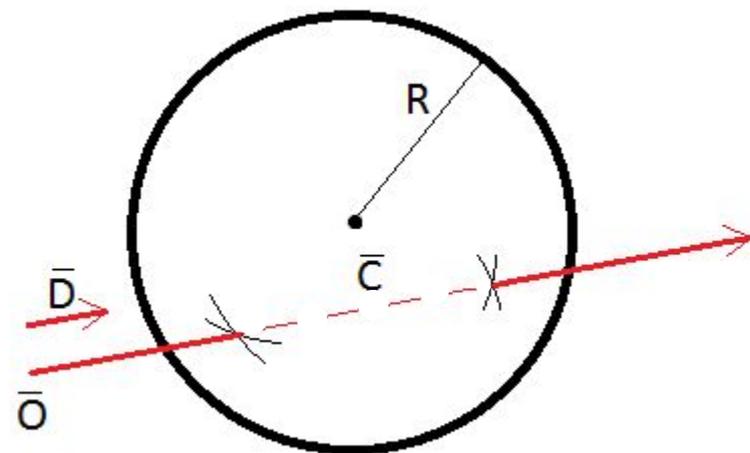
$$F(x, y, z) = (x - C_x)^2 + (y - C_y)^2 + (z - C_z)^2 - R^2 = 0$$

$$(\vec{P} - \vec{C})^2 - R^2 = 0$$

$$(\vec{O} + \vec{D} \cdot t - \vec{C})^2 - R^2 = 0$$

$$\vec{D}^2 \cdot t^2 + 2 \cdot (\vec{D} \cdot (\vec{O} - \vec{C})) \cdot t + (\vec{O} - \vec{C})^2 - R^2 = 0$$

$$t_{0,1} = -\vec{D} \cdot (\vec{O} - \vec{C}) \pm \sqrt{(\vec{D} \cdot (\vec{O} - \vec{C}))^2 - (\vec{O} - \vec{C})^2 + R^2}$$



$$\vec{a} = (\vec{C} - \vec{O})$$

$$OC^2 = \vec{a} \cdot \vec{a}$$

$$OK = \vec{a} \cdot \vec{D}$$

$$OK^2 = (\vec{a} \cdot \vec{D})^2$$

$$h^2 = R^2 - (OC^2 - OK^2)$$

// Луч стартует внутри сферы

if ($OC^2 < R^2$)

{

t = $OK + \text{sqrt}(h^2)$;

return TRUE;

}

// Луч оставляет центр сферы "позади"

if ($OK < 0$)

return FALSE;

// Луч проходит мимо сферы

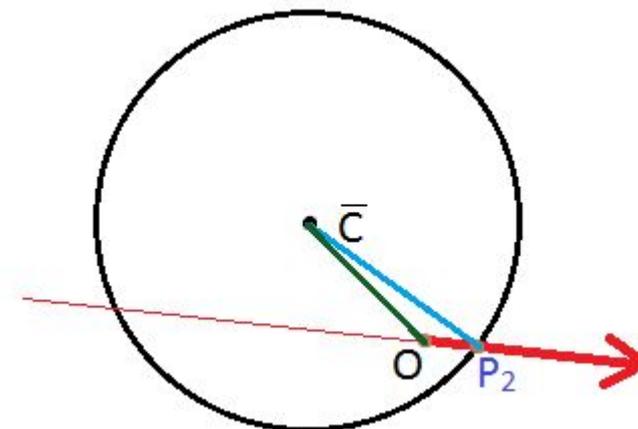
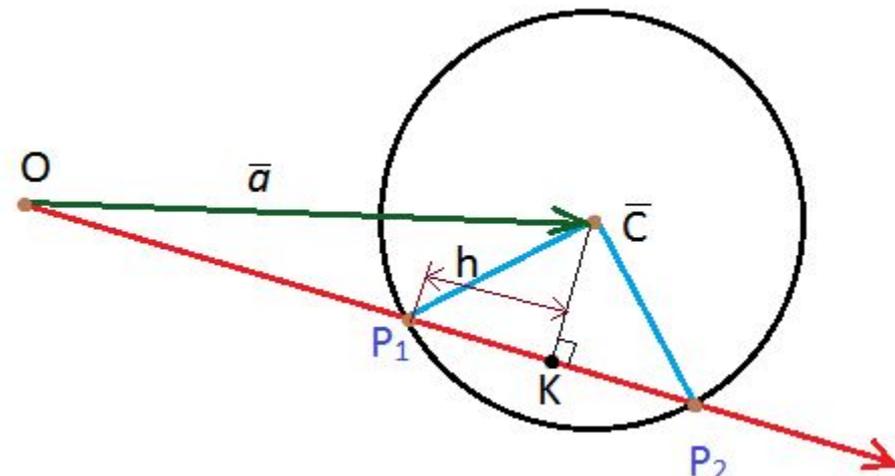
if ($h^2 < 0$)

return FALSE;

// Луч стартует извне сферы

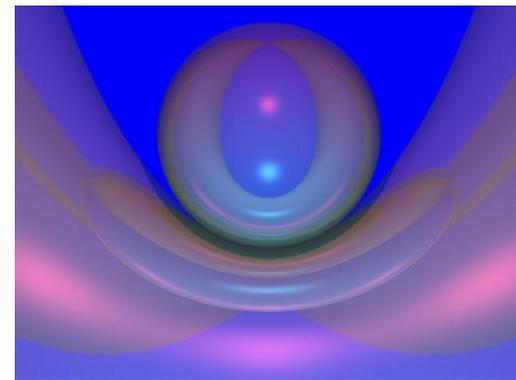
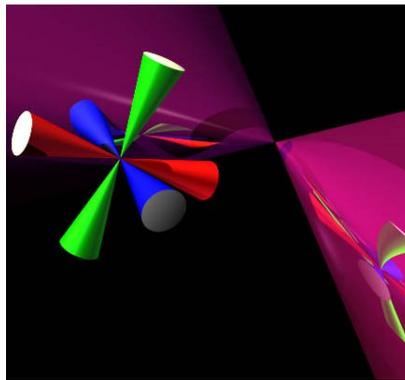
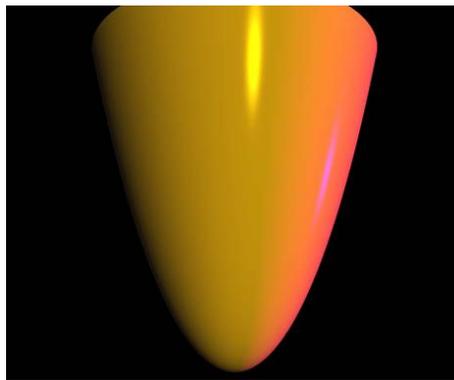
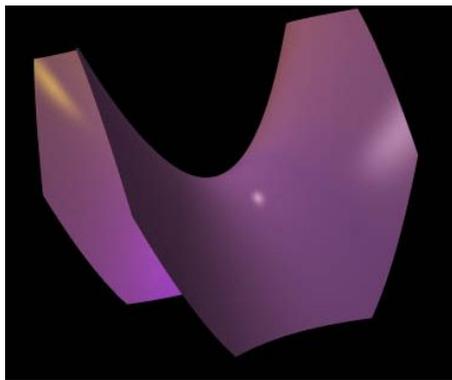
t = $OK - \text{sqrt}(h^2)$;

return TRUE;



$$(x \quad y \quad z \quad 1) \cdot \begin{pmatrix} A & B & C & D \\ B & E & F & G \\ C & F & H & I \\ D & G & I & J \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = 0$$

$$F(x, y, z) = A \cdot x^2 + 2 \cdot B \cdot x \cdot y + 2 \cdot C \cdot x \cdot z + 2 \cdot D \cdot x + \\ E \cdot y^2 + 2 \cdot F \cdot y \cdot z + 2 \cdot G \cdot y + \\ H \cdot z^2 + 2 \cdot I \cdot z + J = 0$$



$$a = A \cdot Dx^2 + 2 \cdot B \cdot Dx \cdot Dy + 2 \cdot C \cdot Dx \cdot Dz + \\ E \cdot Dy^2 + 2 \cdot F \cdot Dy \cdot Dz + \\ H \cdot Dz^2$$

$$b = 2 \cdot (A \cdot Ox \cdot Dx + B \cdot (Ox \cdot Dy + Dx \cdot Oy) + C \cdot (Ox \cdot Dz + x \cdot Oz) + \\ D \cdot Dx + E \cdot Oy \cdot Dy + F \cdot (Oy \cdot Dz + Dy \cdot Oz) + G \cdot Dy + \\ H \cdot Oz \cdot Dz + I \cdot Dz)$$

$$c = A \cdot Ox^2 + 2 \cdot B \cdot Ox \cdot Oy + 2 \cdot C \cdot Ox \cdot Oz + 2 \cdot D \cdot Ox + \\ E \cdot Oy^2 + 2 \cdot F \cdot Oy \cdot Oz + 2 \cdot G \cdot Oy + \\ H \cdot Oz^2 + 2 \cdot I \cdot Oz + J$$

$$t_{0,1} = \frac{-b \pm \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$$

- определяется:

$$B_1 = (X_1, Y_1, Z_1), B_2 = (X_2, Y_2, Z_2)$$

- алгоритм

если $Dx = 0$ то

если $Ox < X_1$ или $Ox > X_2$ то пересечений нет

иначе

вычисляем пересечения:

$$t_0 = (X_1 - Ox) / Dx, t_1 = (X_2 - Ox) / Dx$$

если $t_0 > t_1$ то $\text{Swap}(t_0, t_1)$

если $t_0 > t_{near}$ то установить t_{near} в t_0

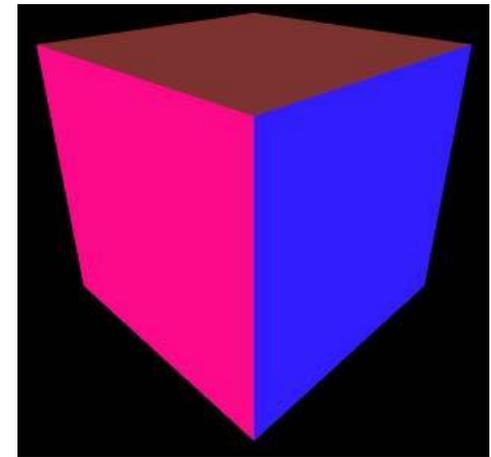
если $t_1 < t_{far}$ то установить t_{far} в t_1

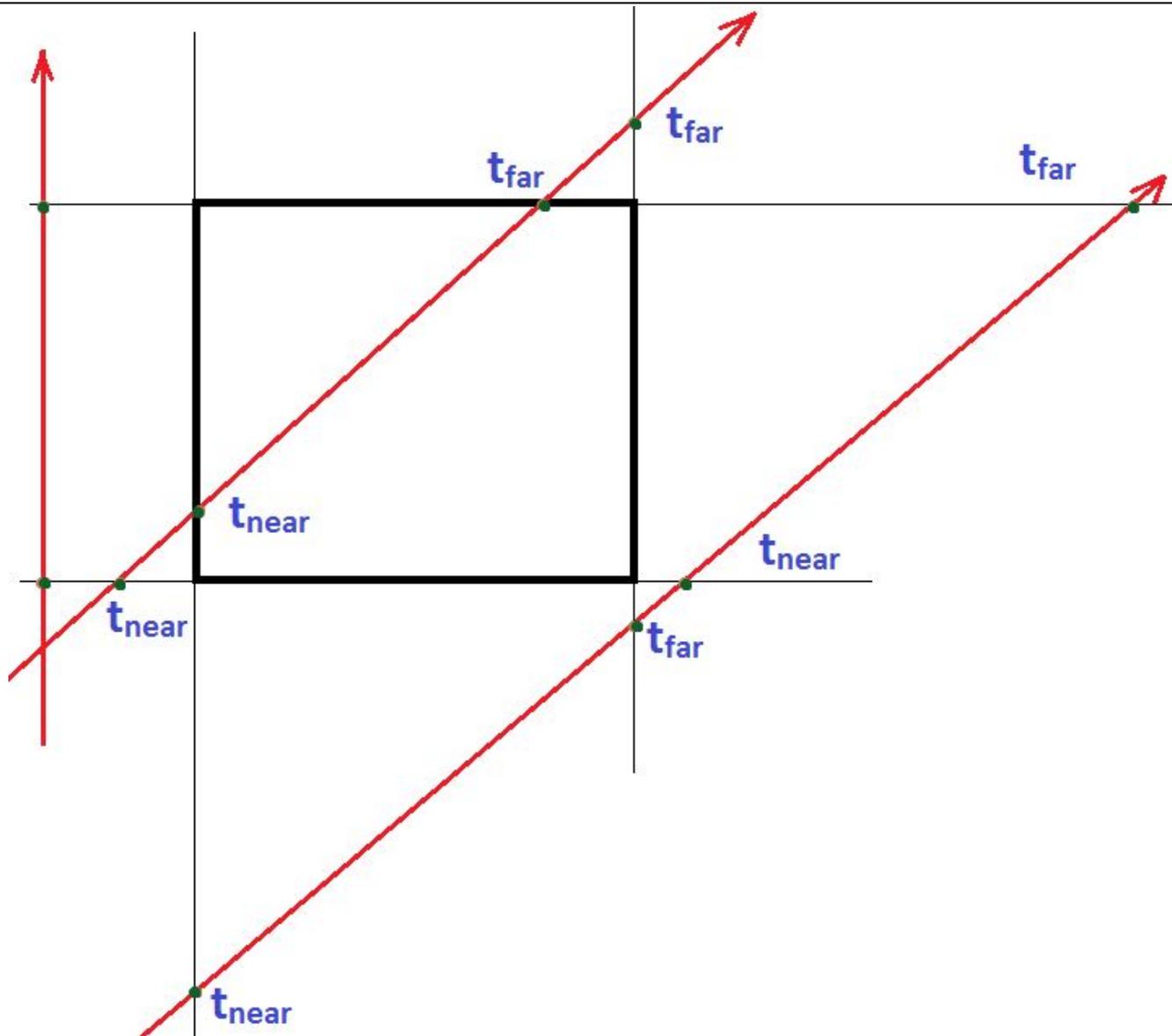
если $t_{near} > t_{far}$ то луч проходит мимо

если $t_{far} < 0$ то параллелепипед "сзади" луча

переход к следующей оси

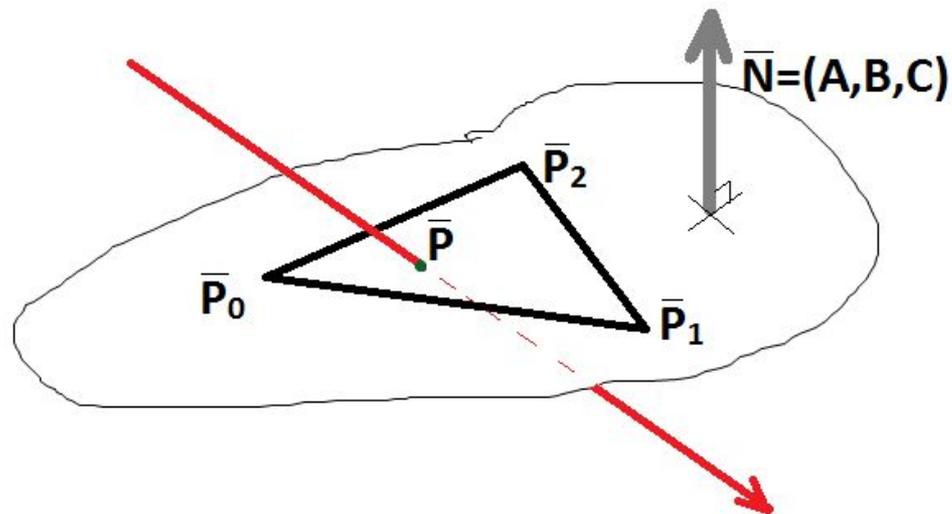
ответ t_{near}





$$\vec{N} = (\vec{P}_1 - \vec{P}_0) \times (\vec{P}_2 - \vec{P}_0)$$

$$D = \vec{N} \cdot \vec{P}_0$$



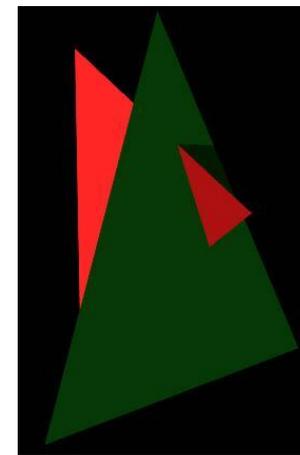
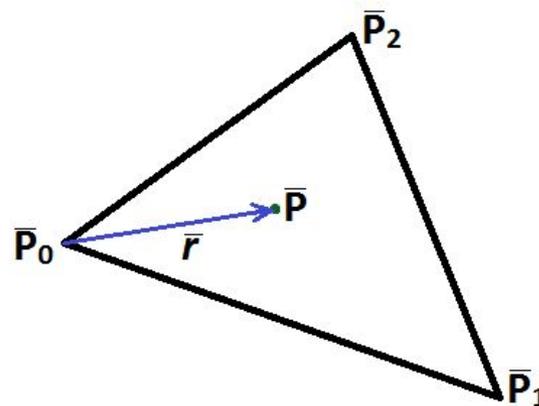
$$\vec{r} = \vec{P} - \vec{P}_0$$

$$\vec{s}_1 = \vec{P}_1 - \vec{P}_0$$

$$\vec{s}_2 = \vec{P}_2 - \vec{P}_0$$

$$\vec{r} = \vec{s}_1 \cdot u + \vec{s}_2 \cdot v$$

если $u \geq 0$ и $v \geq 0$ и $(u + v) \leq 1$ то $P \in \Delta$

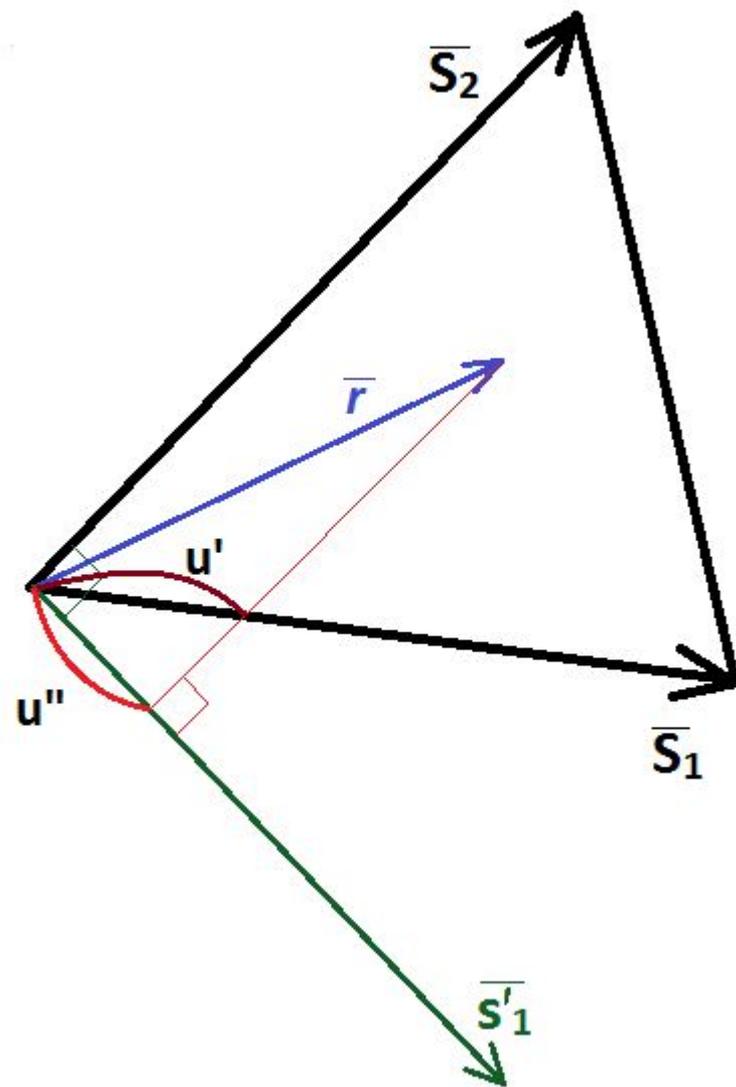


$$u = \vec{P} \cdot \vec{U}_v + U_s$$

$$v = \vec{P} \cdot \vec{V}_v + V_s$$

$$\begin{aligned} \vec{P} &= \vec{P}_0 + (\vec{P}_1 - \vec{P}_0) \cdot u + (\vec{P}_2 - \vec{P}_0) \cdot v = \\ &= \vec{P}_0 \cdot w + \vec{P}_1 \cdot u + \vec{P}_2 \cdot v, \quad w = 1 - u - v \end{aligned}$$

u, v, w – барицентрические координаты



- Луч:

$$\vec{P} = \vec{O} + \vec{D} \cdot t$$

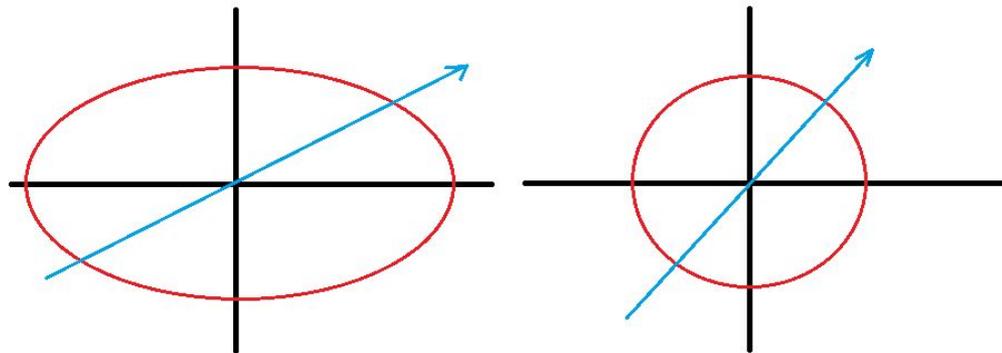
- Точки объекта подвергаются преобразованию M

- Луч преобразуется:

$$\vec{P}' = \vec{O}' + \vec{D}' \cdot t$$

$$\vec{O}' = \vec{O} \cdot \mathbf{M}^{-1}$$

$$\vec{D}' = \text{Normalize}(\vec{D} \cdot \mathbf{M}_{3 \times 3}^{-1})$$



- Ищем пересечение (t)

- Найденное t сокращаем на длину вектора

$$\vec{D} \cdot \mathbf{M}_{3 \times 3}^{-1}$$

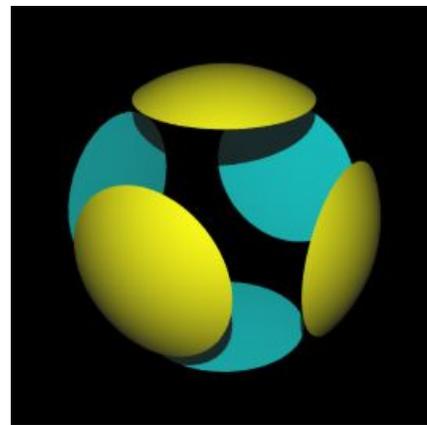
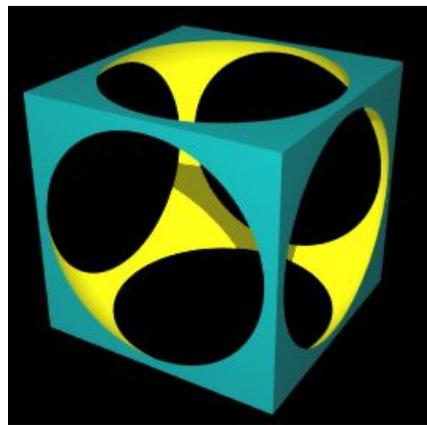
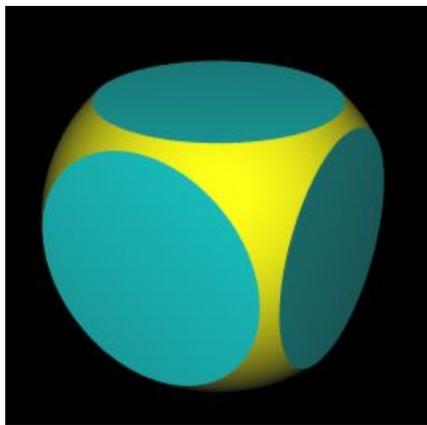
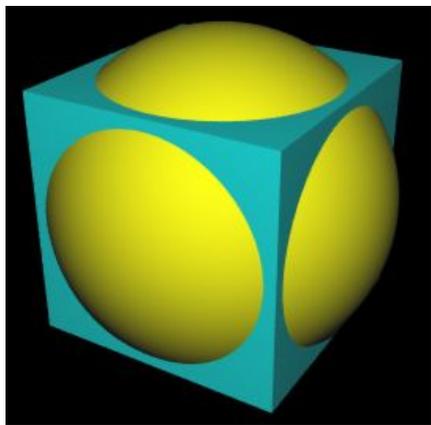
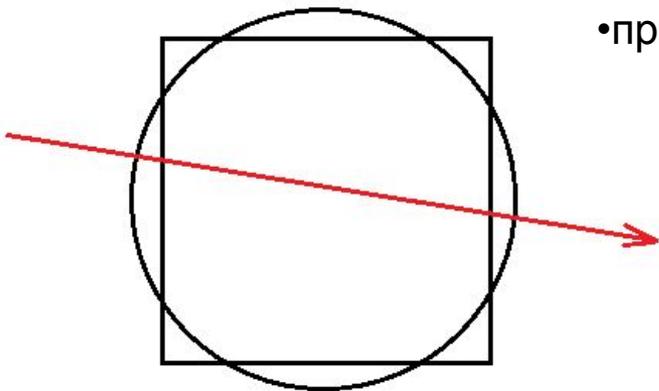


Необходим поиск ВСЕХ пересечений объекта с лучом

• пример:

▪ пересечение:

- остаются все пересечения, принадлежащие внутренности второго объекта, из получившегося списка берем ближайшее



- Практические задания (до 27.11.2011)
 - Реализовать простейший алгоритм трассировки луча для сфер. Решить задачу удаления невидимых поверхностей. Сферы представлять разными цветами. Результат записывать в файл с растровым изображением.