

Введение в компьютерную графику

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

E-mail: CGSG@yandex.ru

COMPUTER VISION

распознавание образов

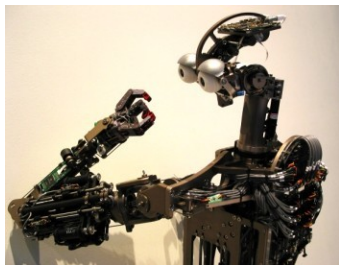


IMAGE PROCESSING

обработка изображений



COMPUTER GRAPHICS

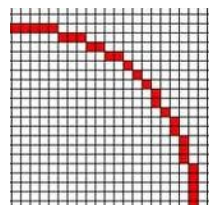
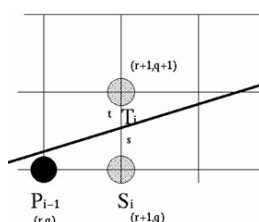
машинная графика



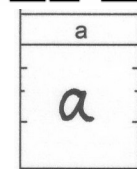
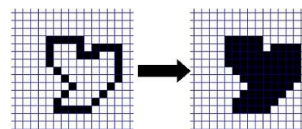
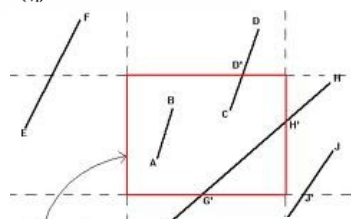
Введение



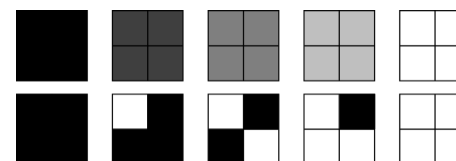
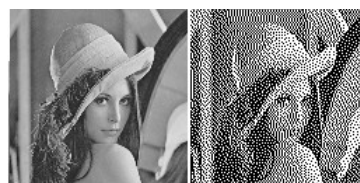
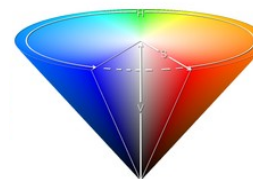
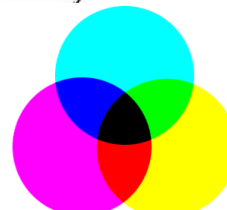
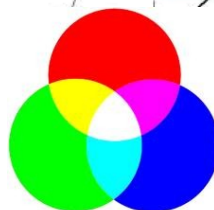
Растровая графика



Script



Цвет



План курса

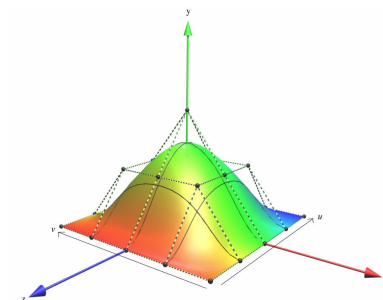
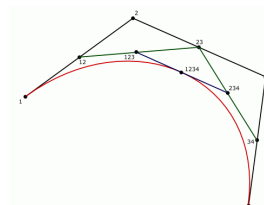
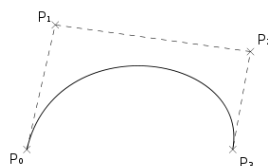
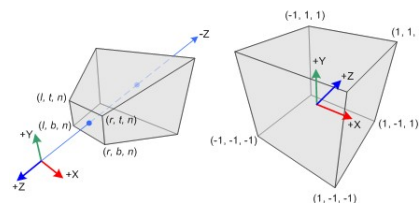
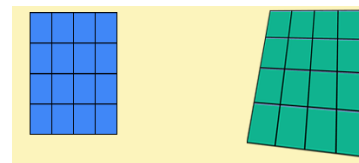
Математика для КГ

$$trans(\Delta x, \Delta y, \Delta z) = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

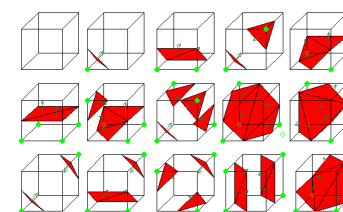
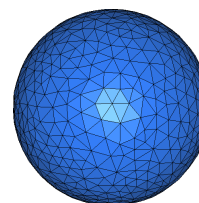
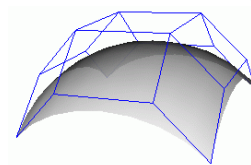
$$rot(x, \theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$rot(y, \theta) = \begin{bmatrix} \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

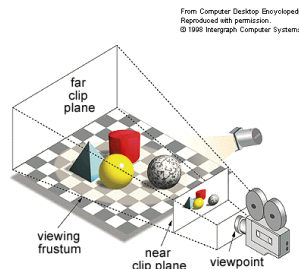
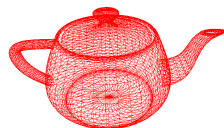
$$rot(z, \theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Кривые и поверхности

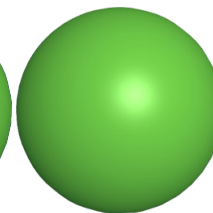
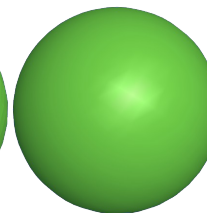
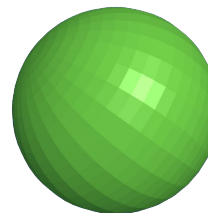
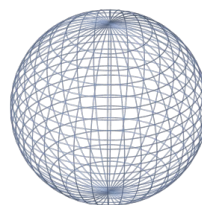


Визуализация

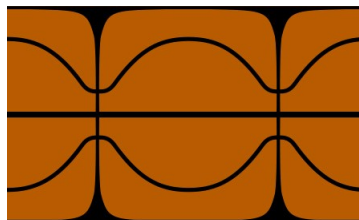


Удаление невидимых линий и поверхностей

Освещение и тонирование



Текстурирование



План курса



Алгоритм трассировки лучей

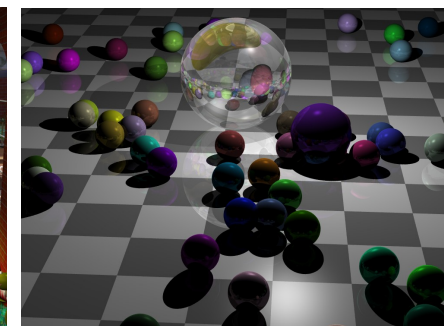
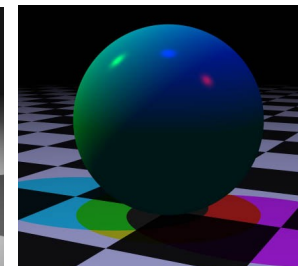
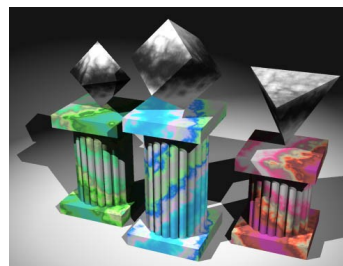
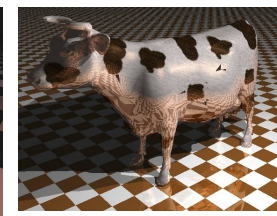
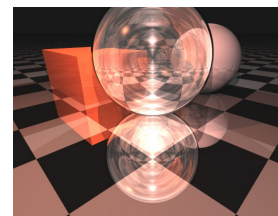
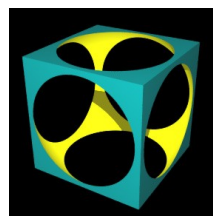
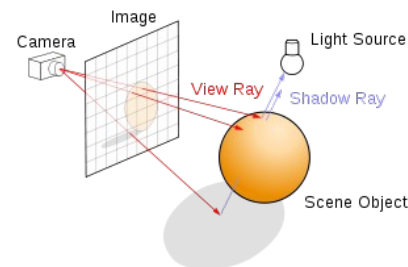
Базовый алгоритм

Модели освещения

Методы оптимизации

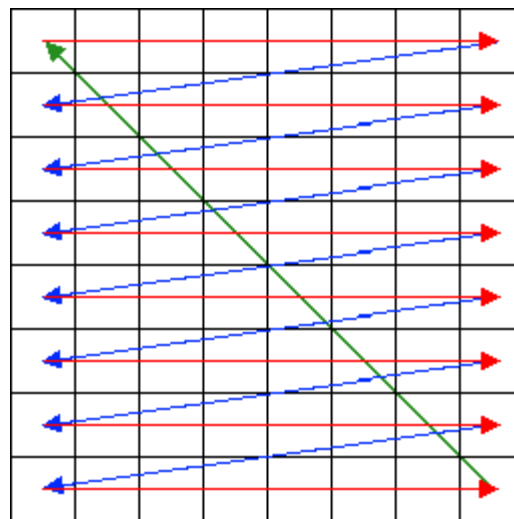
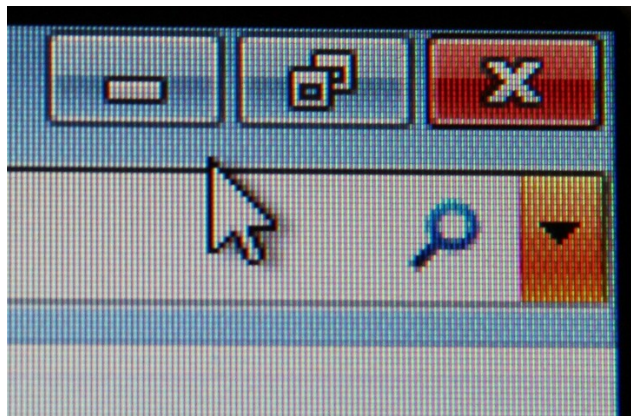
Шум

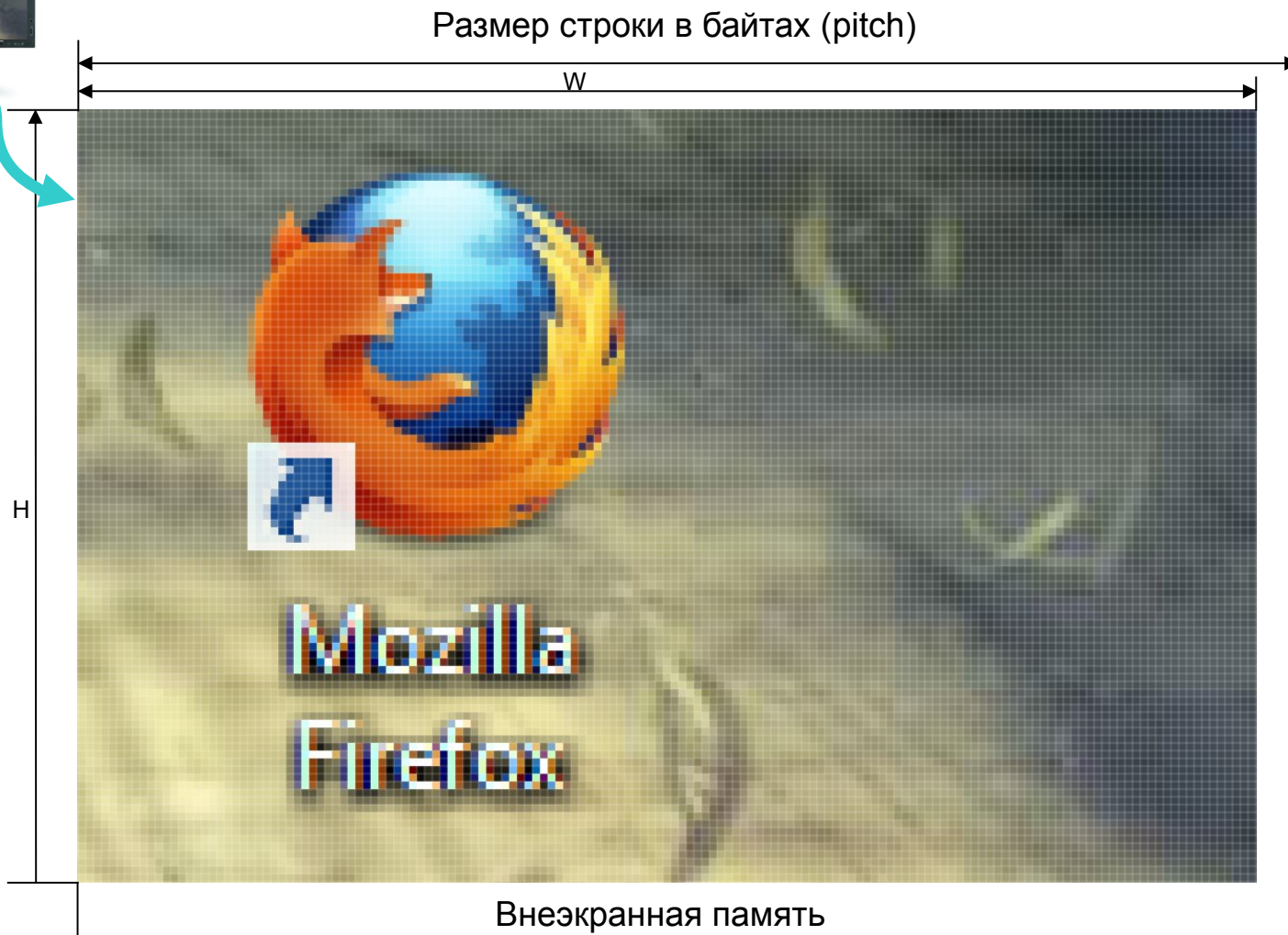
Распределенная трассировка лучей





Vector vs. Raster



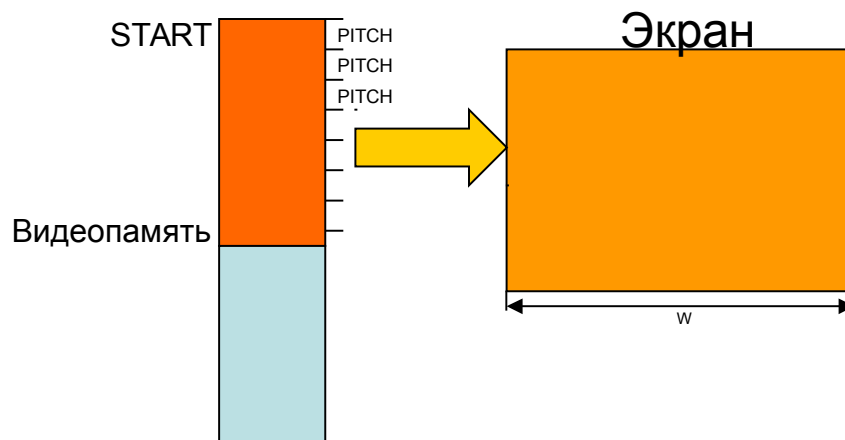


Адресация точек:

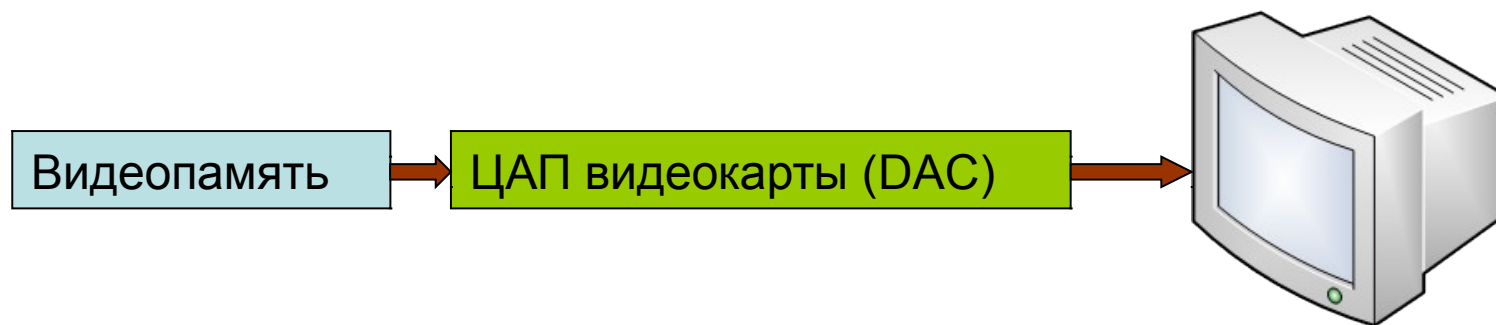
Стартовый адрес + Y * Pitch + X

```
#define START ?????
#define PITCH ?????

void DrawPixel( int X, int Y, unsigned long ColorRGB )
{
    *(unsigned long *) (START + Y * PITCH + X) = ColorRGB;
}
```



Цвет: палитра



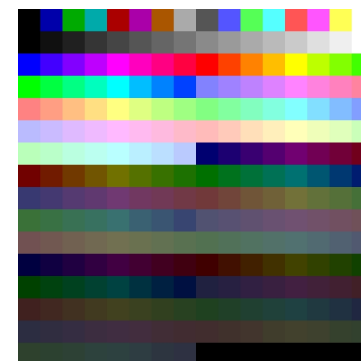
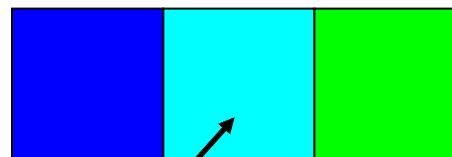
Номера цветов в видеопамяти

1	3	2
---	---	---

Палитра: таблица соответствия цветов

	Red	Green	Blue	
0	0	0	0	
1	0	0	255	
2	0	255	0	
3	0	255	255	
...	
255	127	111	55	

Точки (пиксели) на экране

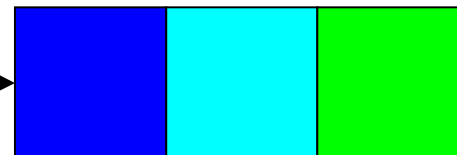


Стандартная палитра VGA

Триады RGB цветов в видеопамяти

0,0,255	0,255,255	0,255,0
---------	-----------	---------

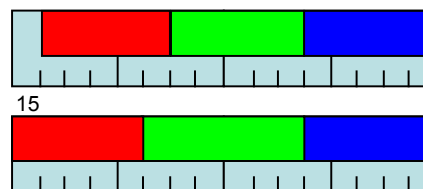
Точки (пиксели) на экране

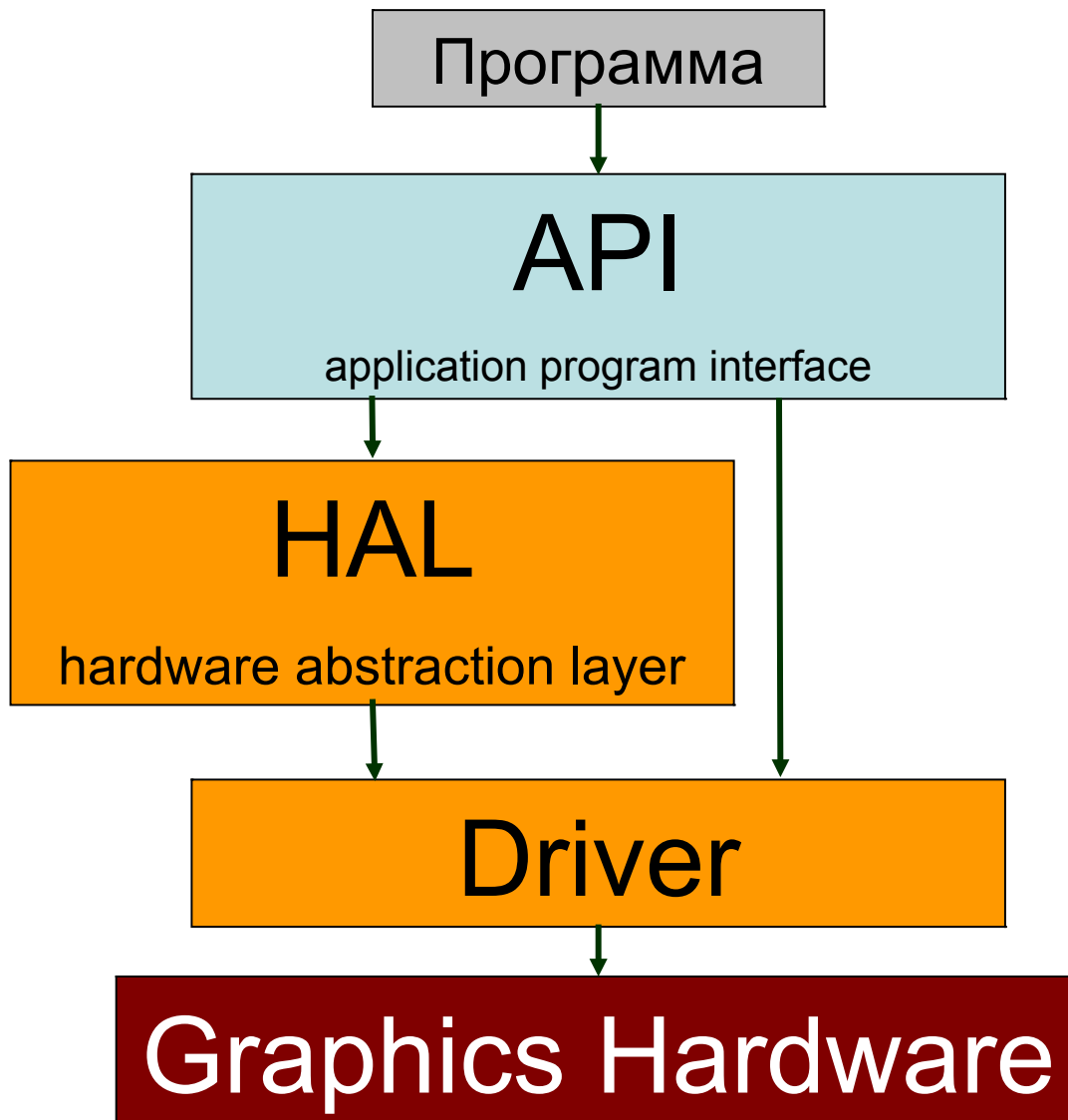


TrueColor: 24/32 бита (8r 8g 8b)



HiColor: 15/16 бит (5r 5g 5b / 5r 6g 5b)





- Функции рисования линейных объектов (и точек)
 - `Line(x1,y1,x2,y2)` `MoveTo(x,y)` `LineTo(x,y)` `SetPixel(x,y,c)`
- Функции рисования площадных объектов
 - `Rectangle(x1,y1,x2,y2)` `Circle(x,y,r)` `Polygon(points,n)`
- Функции вывода текста
 - `DrawText(x,y,string)`
- Функции задания атрибутов рисования
 - цвета и формы «перьев» для линейных объектов, шаблоны «кистей» для площадных, параметры вывода шрифтов (текста), код логической операции при выводе (**REPLACE**, **OR**, **AND**, **XOR**);
 - управление областями отсечения
- Функции задания преобразований системы координат
- Функции управления буферизацией
 - создание, копирование, уничтожение буферов изображений (bitmaps)


```
#define VGA256_MODE 0x13
#define TEXT_MODE 3

void SetMode( int mode )
{
    __asm {
        mov  AX, mode
        int  0x10
    }
}

void PutPixel( int x, int y, int color )
{
    *(unsigned char *) (0xA0000000UL + y * 320U + x) = color;
}

int main( void )
{
    /* Инициализация */
    SetMode(VGA256_MODE);

    /* Отрисовка зеленой точки */
    PutPixel(10, 10, 2);

    sleep(5);
    /* Выход из графического режима */
    SetMode(TEXT_MODE);
    return 0;
}
```

```
#include <vga.h>

int main( void )
{
    /* Инициализация */
    vga_init();
    vga_setmode(5);

    /* Отрисовка зеленой точки */
    vga_setcolor(2);
    vga_drawpixel(10, 10);

    /* Отрисовка красного отрезка прямой */
    vga_setcolor(4);
    vga_drawline(20, 10, 150, 100);

    sleep(5);
    /* Выход из графического режима */
    vga_setmode(0);
    return 0;
}
```

```
#include <windows.h>

. . .
LRESULT WinFunc( HWND hWnd, UINT Msg,
                  WPARAM wParam, LPARAM lParam )
{
    HDC hDC;
    HPEN hPen, hOldPen;

    switch (Msg)
    {
        . . .
        /* Создание контекста */
        hDC = GetDC(hWnd);

        /* Отрисовка зеленой точки */
        SetPixel(hDC, 10, 10, RGB(0, 255, 0));

        /* Создание красного пера */
        hPen = CreatePen(PS_SOLID, 1, RGB(255, 0, 0));
        hOldPen = SelectObject(hDC, hPen);

        /* Отрисовка отрезка прямой */
        MoveToEx(hDC, 20, 10, NULL);
        LineTo(hDC, 150, 100);

        /* Удаление пера */
        SelectObject(hDC, hOldPen);
        DeleteObject(hPen);

        /* Удаление контекста */
        ReleaseDC(hWnd, hDC);
        break;
    }
    . . .
    return DefWindowProc(hWnd, Msg, wParam, lParam);
}
```

```
#include <glut.h>

void Display( void )
{
    /* Очистка кадра */
    glClear(GL_COLOR_BUFFER_BIT);

    /* Отрисовка зеленой точки */
    glBegin(GL_POINTS);
        glColor3d(0, 1, 0);
        glVertex2d(0.1, 0.1);
    glEnd();

    /* Отрисовка красного отрезка прямой */
    glBegin(GL_LINES);
        glColor3d(1, 0, 0);
        glVertex2d(0.2, 0.1);
        glVertex2d(0.8, 0.5);
    glEnd();

    /* Окончание кадра */
    glFinish();
}

int main( void )
{
    glutInitDisplayMode(GLUT_RGB);
    glutInitWindowPosition(0, 0);
    glutInitWindowSize(500, 500);
    glutCreateWindow("Sample");

    glutDisplayFunc(Display);

    glutMainLoop();
    return 0;
}
```

• Упражнение

- Необходимо написать программу-заготовку с использованием любого удобного графического интерфейса (библиотеки или непосредственного рисования на «низком» уровне) для последующих работ по реализации алгоритмов компьютерной графики. Главное требование – наличие функции рисования отдельного «пикселя» с «экранной» точностью (каждая физическая точка экрана задается цветом отдельного «пикселя»)