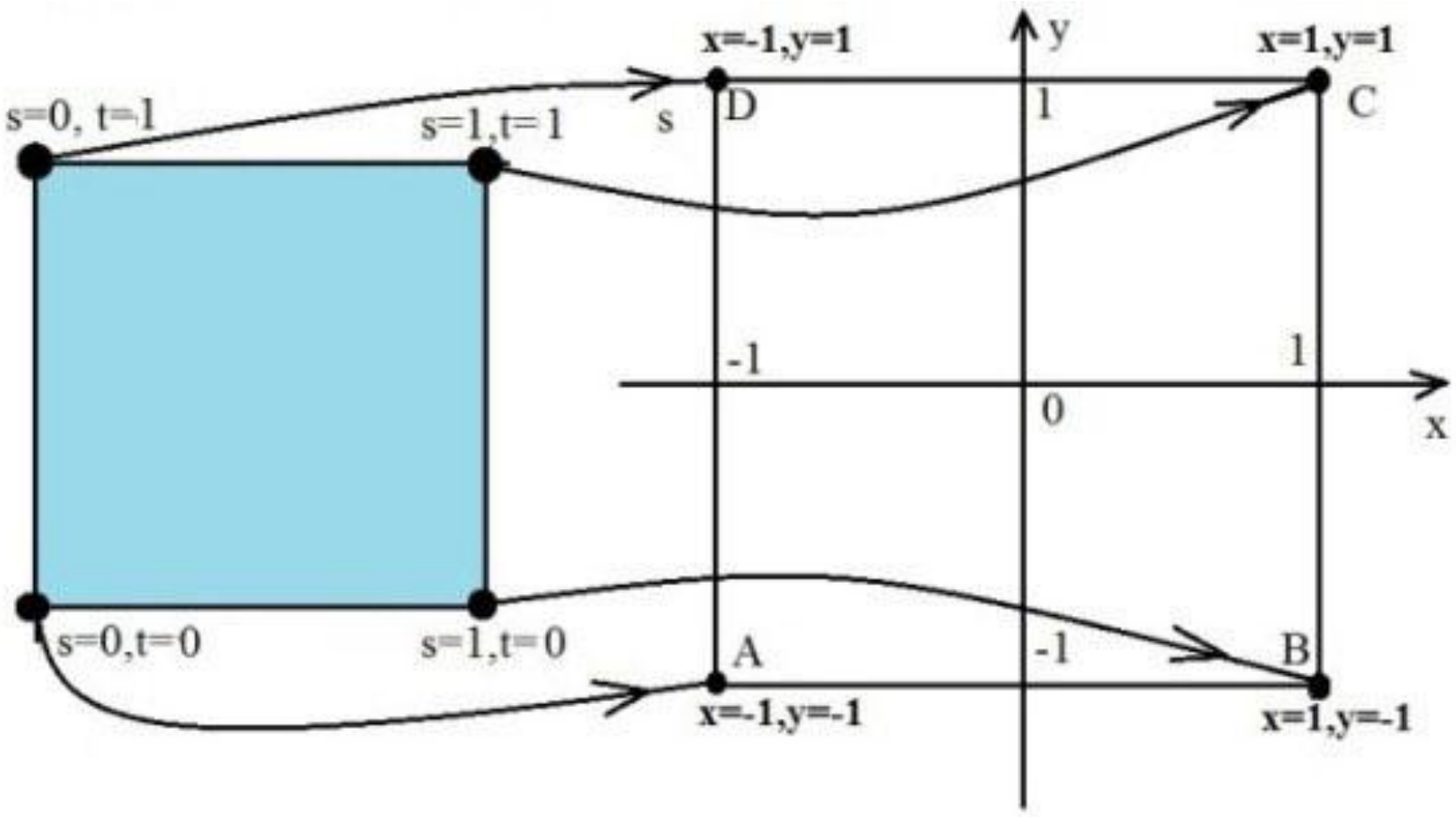
Компьютерная графика

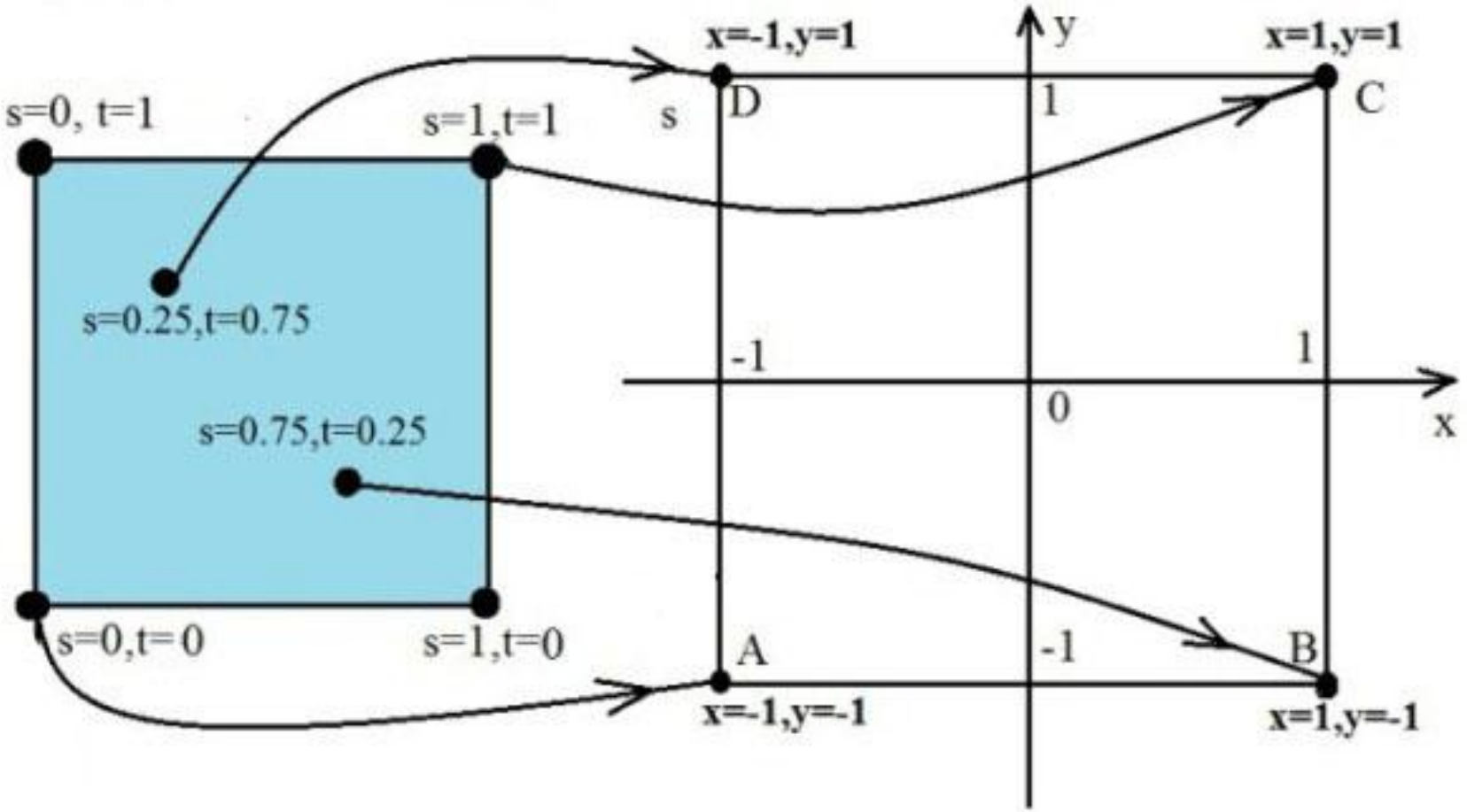
лекция 5

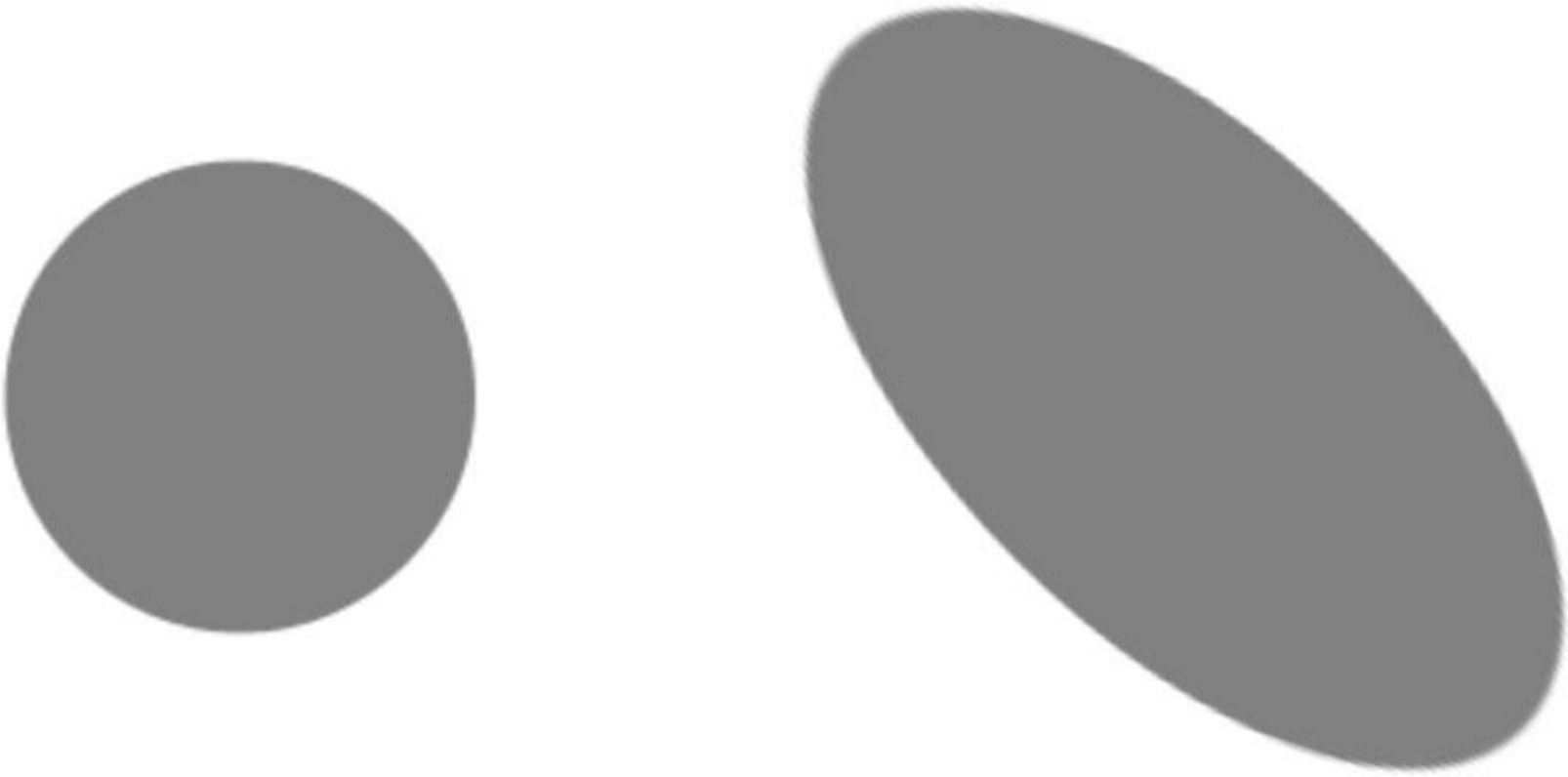
Наложение текстуры

Наложение текстуры



Наложение текстуры





пример наложения текстуры

Подключение модуля glaux.h:

#include “glaux.h”

Загрузка библиотеки glaux.lib:

#pragma comment (lib, "glaux.lib")

Объявление массива для хранения индексов текстур:

GLuint texture[2];

AUX_RGBImageRec *texture1; // задает указатель на структуру для хранения картинки.

Структура AUX_RGBImageRec определена в библиотеке glAux, и делает возможной загрузку картинки в память.

загрузка текстуры осуществляется следующей функцией:

texture1 = auxDIBImageLoad("dom2.bmp");

Для генерации свободных имен существует команда glGenTextures, формат которой выглядит так:

```
glGenTextures(1, &texture[0]);
```

Установка текущей текстуры производится командой:

```
glBindTexture(GL10.GL_TEXTURE_2D, texture[0] );
```

Установим параметры фильтрации текстуры - линейная фильтрация.

```
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);

glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
```

Загрузка текстуры:

```
glTexImage2D(GL_TEXTURE_2D, 0, 3, texture1->sizeX,
   texture1->sizeY, 0,  GL_RGB, GL_UNSIGNED_BYTE,
   texture1->data);
```

1 - Текстура будет двухмерной (GL_TEXTURE_2D).

2 - Ноль задает уровень детализации, это обычно ноль.

3 - число компонент цветовых данных, так как изображение сделано из трех цветовых компонент (красный, зеленый, синий).

4 - texture1- >sizeX - это ширина текстуры.

5 - texture1->sizeY - высота текстуры.

6 - Ноль - это бордюр. Он обычно остается нулем.

7 - GL_RGB сообщает OpenGL, что данные изображения представлены в порядке следования красных, зеленных и голубых компонент цвета.

8 - GL_UNSIGNED_BYTE означает, что данные из которых состоит изображение имеют размер байта и все числа без знака.

9 - texture1->data сообщает OpenGL, где брать сами данные. В этом случае указатель на данные в записи texture1

Функция загрузки текстуры

```
GLvoid LoadHouseTextures()
{
    AUX_RGBImageRec *texture1;
   texture1 = auxDIBImageLoad("dom2.bmp");
   glGenTextures(1, &texture[0]);
   glBindTexture(GL_TEXTURE_2D, texture[0]);

   glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FI
   LTER,GL_LINEAR);
   glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FIL
   TER,GL_LINEAR);

   glTexImage2D(GL_TEXTURE_2D, 0, 3, texture1->sizeX,
   texture1->sizeY, 0,  GL_RGB, GL_UNSIGNED_BYTE,
   texture1->data);
}
```

Текстурные координаты

void **glTexCoord[1 2 3 4][s i f d]** (type *coord*)

void **glTexCoord[1 2 3 4][s i f d]v** (type **coord*)

Вывод полигона с текстурой

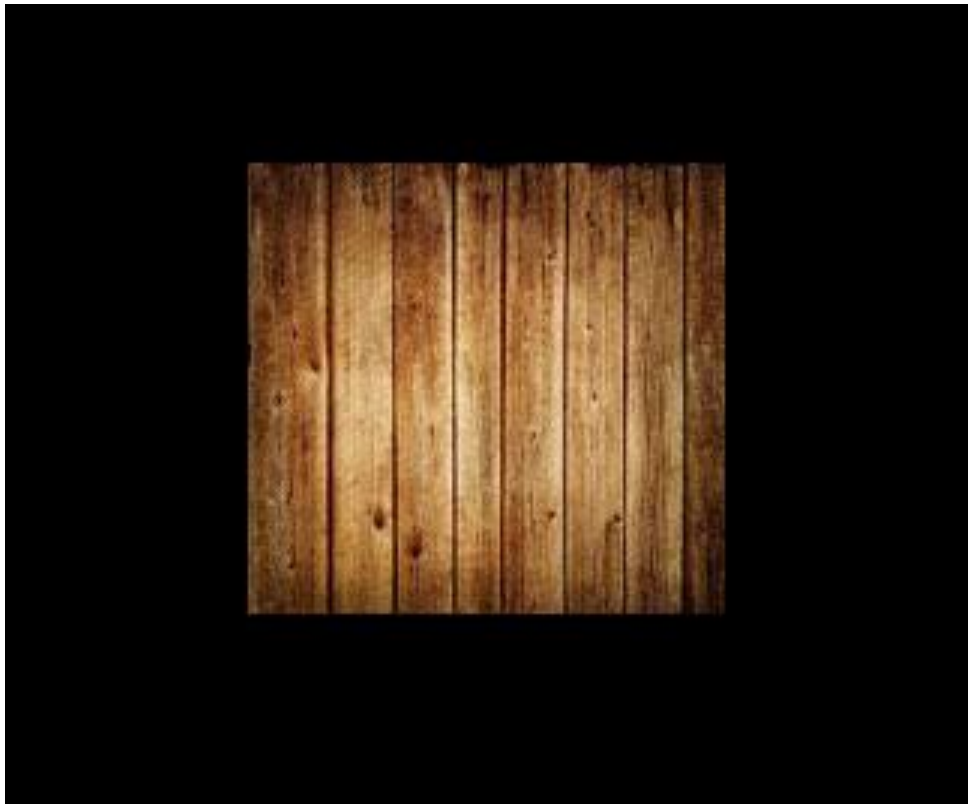
```
glBindTexture(GL_TEXTURE_2D, texture[0]);

glBegin(GL_QUADS);
        glTexCoord2f(0.0f, 0.0f);
  glVertex3f(x1, y1, z1);
  glTexCoord2f(1.0f, 0.0f);
  glVertex3f(x2, y2, z2);
  glTexCoord2f(1.0f, 1.0f);
  glVertex3f(x3, y3, z3);
  glTexCoord2f(0.0f, 1.0f);
  glVertex3f(x4, y4, z4);
 glEnd();
```

Вывод полигона с текстурой

```
glEnable (GL_TEXTURE_2D);
glColor3f(1.0f,1.0f,1.0f);
glBindTexture(GL_TEXTURE_2D, texture[1]);
  glBegin(GL_QUADS);
   glTexCoord2f(0, 0); glVertex3f(0, 0, 0);
 glTexCoord2f(0, 1); glVertex3f(0, 1, 0);
 glTexCoord2f(1, 1); glVertex3f(1, 1, 0);
 glTexCoord2f(1, 0); glVertex3f(1, 0, 0);
  glEnd();
glDisable(GL_TEXTURE_2D);
```

Пример наложения текстуры

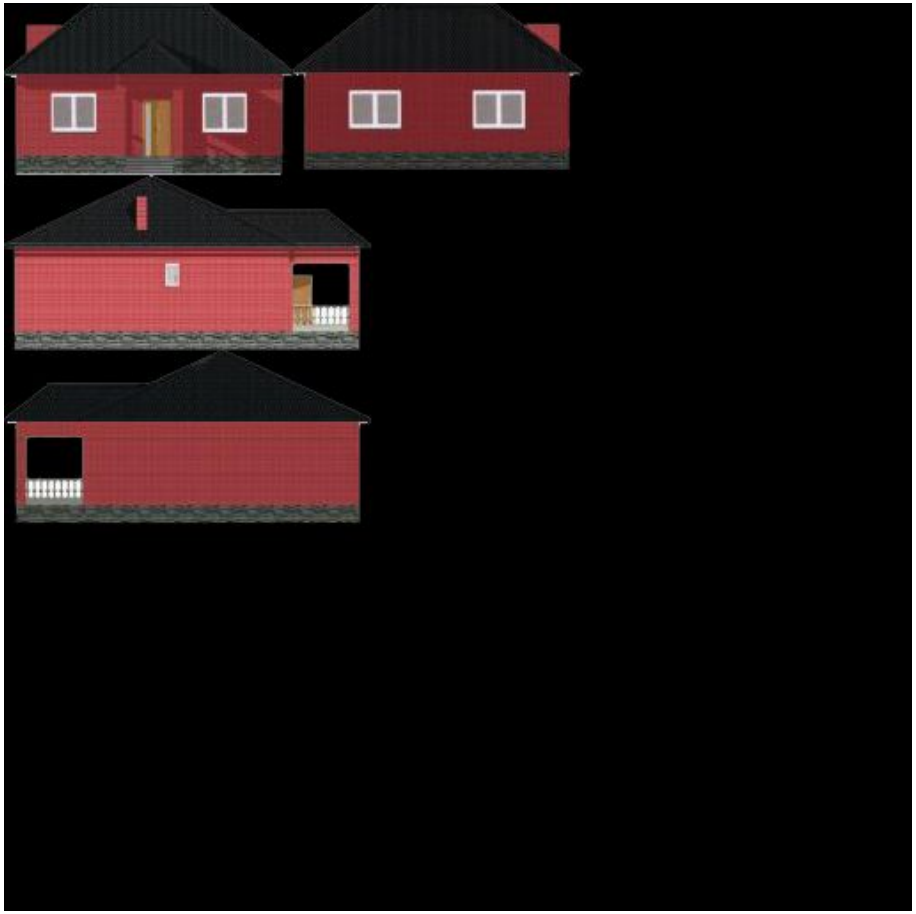


Вывод полигона с текстурой

```
glEnable (GL_TEXTURE_2D);
glColor3f(1.0f,1.0f,1.0f);
glBindTexture(GL_TEXTURE_2D, texture[1]);
  glBegin(GL_QUADS);
   glTexCoord2f(0, 0); glVertex3f(0, 0, 0);
 glTexCoord2f(1, 0); glVertex3f(0, 1, 0);
 glTexCoord2f(1, 1); glVertex3f(1, 1, 0);
 glTexCoord2f(0, 1); glVertex3f(1, 0, 0);
  glEnd();
glDisable(GL_TEXTURE_2D);
```

Пример наложения текстуры

Пример наложения текстуры



Размеры изображения должны быть кратны степени 2. В данном примере 512x512 пикселов.

Текстурные координаты

Вычисление нужных координат

Координаты точек в Photoshop

A(6, 96) B(6, 41) C(157, 41) D(157, 96)

E(1, 40) F(48, 1) G(117, 1) H(164, 40)

Пересчет текстурных координат

```
float Ax = 6/512.0, Ay = (512 - 96)/512.0,
Bx = 6/512.0, By = (512 - 41) /512.0,
Cx = 157/512.0, Cy = (512 - 41) /512.0,
Dx = 157/512.0, Dy = (512 - 96) /512.0,
Ex = 1/512.0, Ey = (512 - 40) /512.0,
Fx = 48/512.0, Fy = (512 - 1) /512.0,
Gx = 117/512.0, Gy = (512 - 1) /512.0,
Hx = 164/512.0, Hy = (512 - 40) /512.0;
```

Рисование дома

```
void DrawHome() {
    glEnable (GL_TEXTURE_2D);
    glColor3f(1.0f,1.0f,1.0f);
    glBindTexture(GL_TEXTURE_2D, texture[0]);

    DrawWall1();
    DrawWall2();
    DrawWall3();
    DrawWall4();
    DrawFloor();

    glDisable(GL_TEXTURE_2D);
}
```

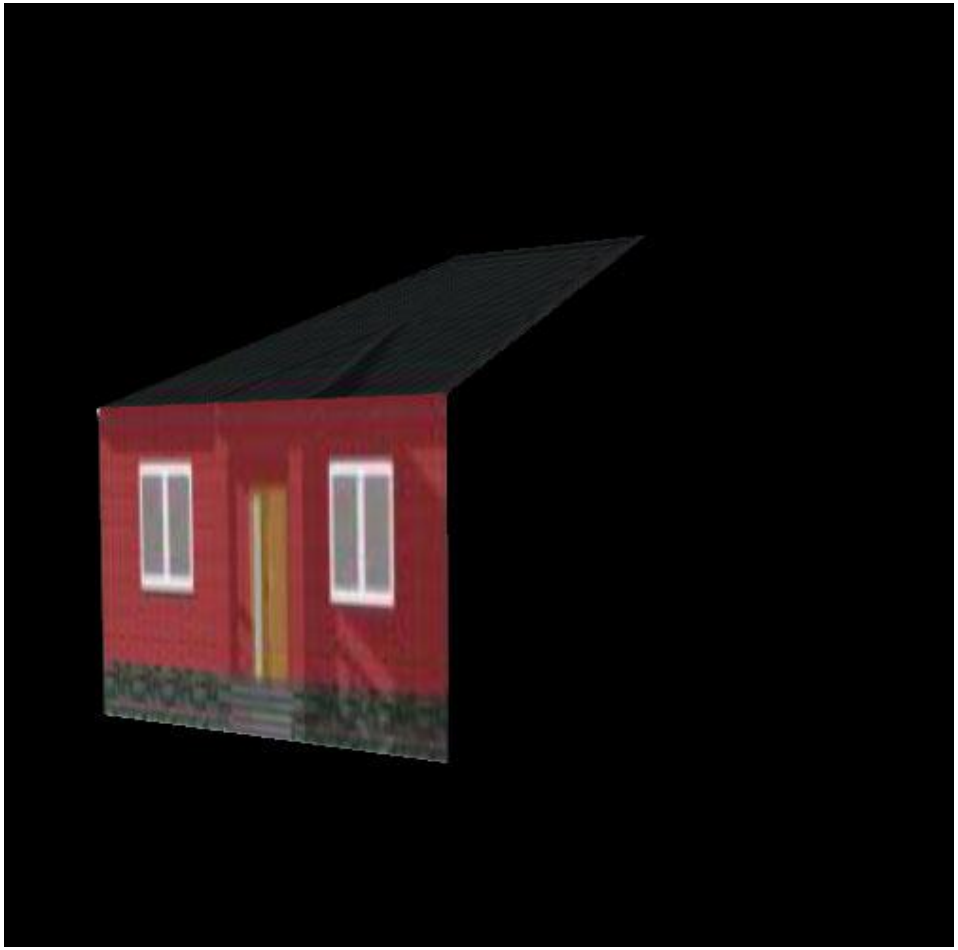
Рисование полигона стены дома

```
void DrawWall1()
{
    float Ax = 6/512.0, Ay = (512 - 96)/512.0,
    Bx = 6/512.0, By = (512 - 41) /512.0,
    Cx = 157/512.0, Cy = (512 - 41) /512.0,
    Dx = 157/512.0, Dy = (512 - 96) /512.0,
    Ex = 1/512.0, Ey = (512 - 40) /512.0,
    Fx = 48/512.0, Fy = (512 - 1) /512.0,
    Gx = 117/512.0, Gy = (512 - 1) /512.0,
    Hx = 164/512.0, Hy = (512 - 40) /512.0;
    float x=-1, y=0, z=1;
```

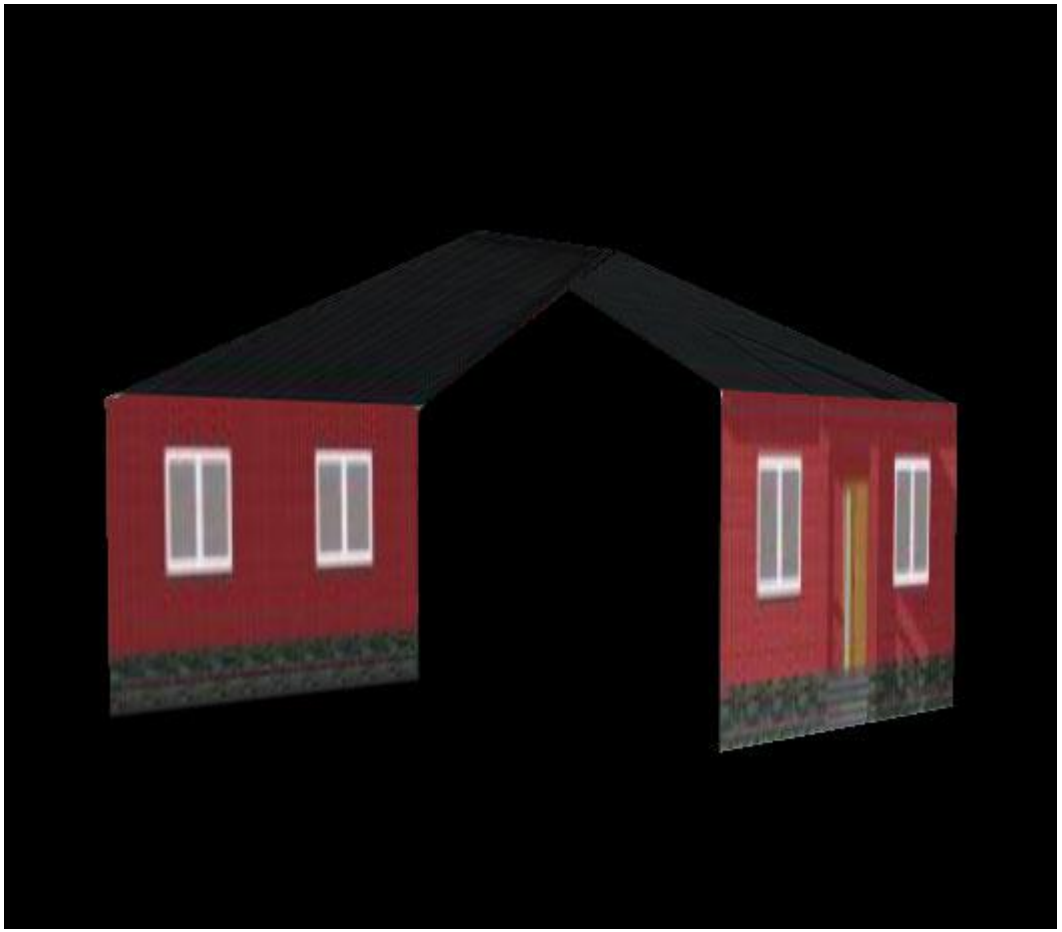
Рисование полигона стены дома

```
glBegin(GL_QUADS);
      glTexCoord2f(Ax, Ay);        glVertex3f(x, y, z);
       glTexCoord2f(Bx, By);       glVertex3f(x, y + 1, z);
       glTexCoord2f(Cx, Cy);       glVertex3f(x + 2, y + 1, z);
       glTexCoord2f(Dx, Dy);       glVertex3f(x + 2, y, z);
 glEnd();
 glBegin(GL_QUADS);
       glTexCoord2f(Ex, Ey);       glVertex3f(x, y + 1, z);
       glTexCoord2f(Fx, Fy);       glVertex3f(x + 0.5, y + 1.5, z-1);
       glTexCoord2f(Gx, Gy);       glVertex3f(x + 1.5, y + 1.5, z-1);
       glTexCoord2f(Hx, Hy);       glVertex3f(x + 2, y + 1, z);
 glEnd();
}
```

Пример наложения текстуры



Пример наложения текстуры



Пример наложения текстуры

Функции рисования текстурированного треугольника

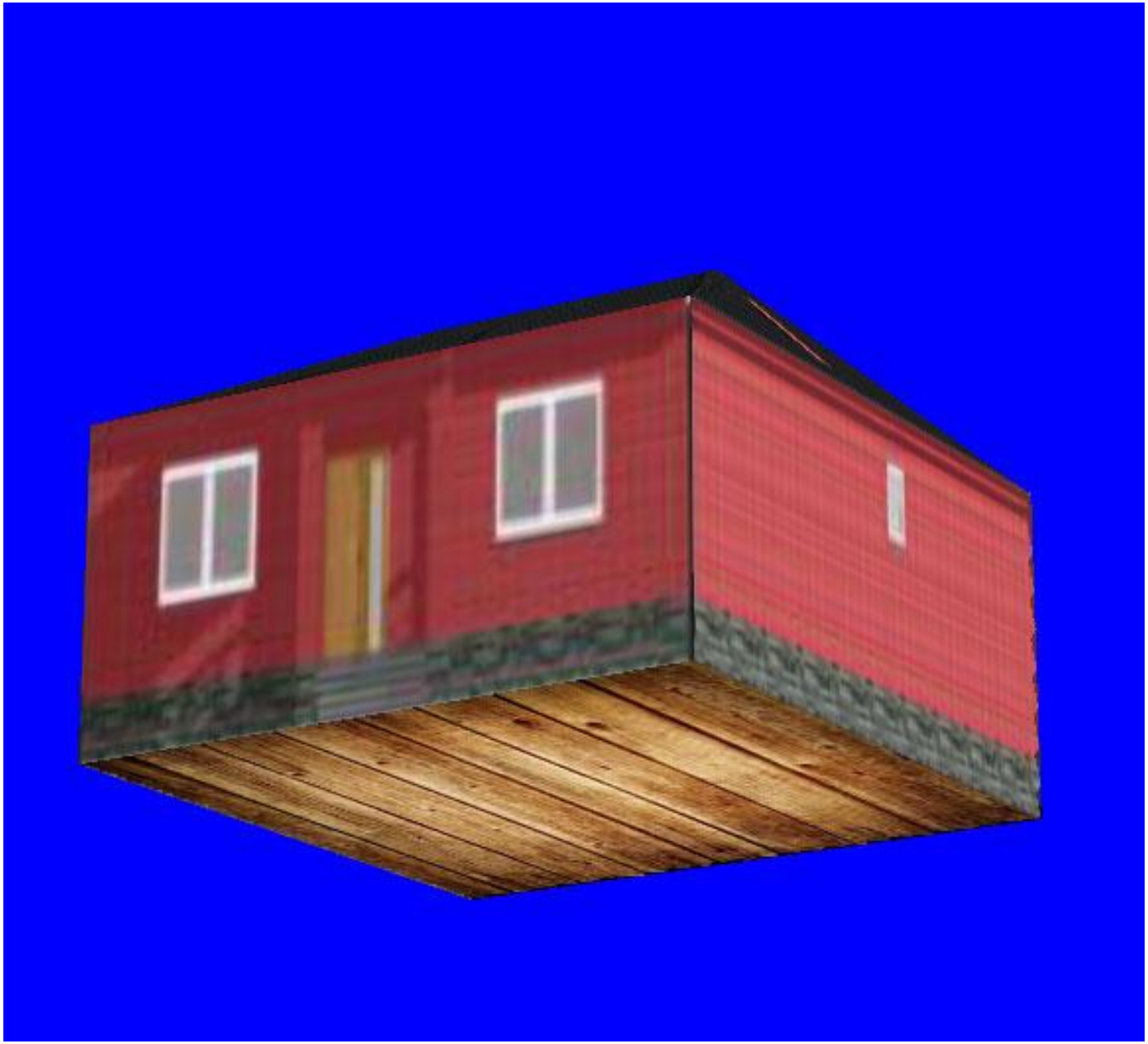
```
void DrawTriangleT(float x1, float y1, float z1, float x2,
   float y2, float z2, float x3, float y3, float z3,
      float tx1, float ty1, float tx2, float ty2, float tx3,
   float ty3, int tindex)
{
glEnable (GL_TEXTURE_2D);
glColor3f(1.0f,1.0f,1.0f);
glBindTexture(GL_TEXTURE_2D, texture[tindex]);
  glBegin(GL_TRIANGLES);
    glTexCoord2f(tx1, ty1);  glVertex3d(x1, y1, z1);
    glTexCoord2f(tx2, ty2);  glVertex3d(x2, y2, z2);
    glTexCoord2f(tx3, ty3);  glVertex3d(x3, y3, z3);
  glEnd();
glDisable(GL_TEXTURE_2D);
}
```

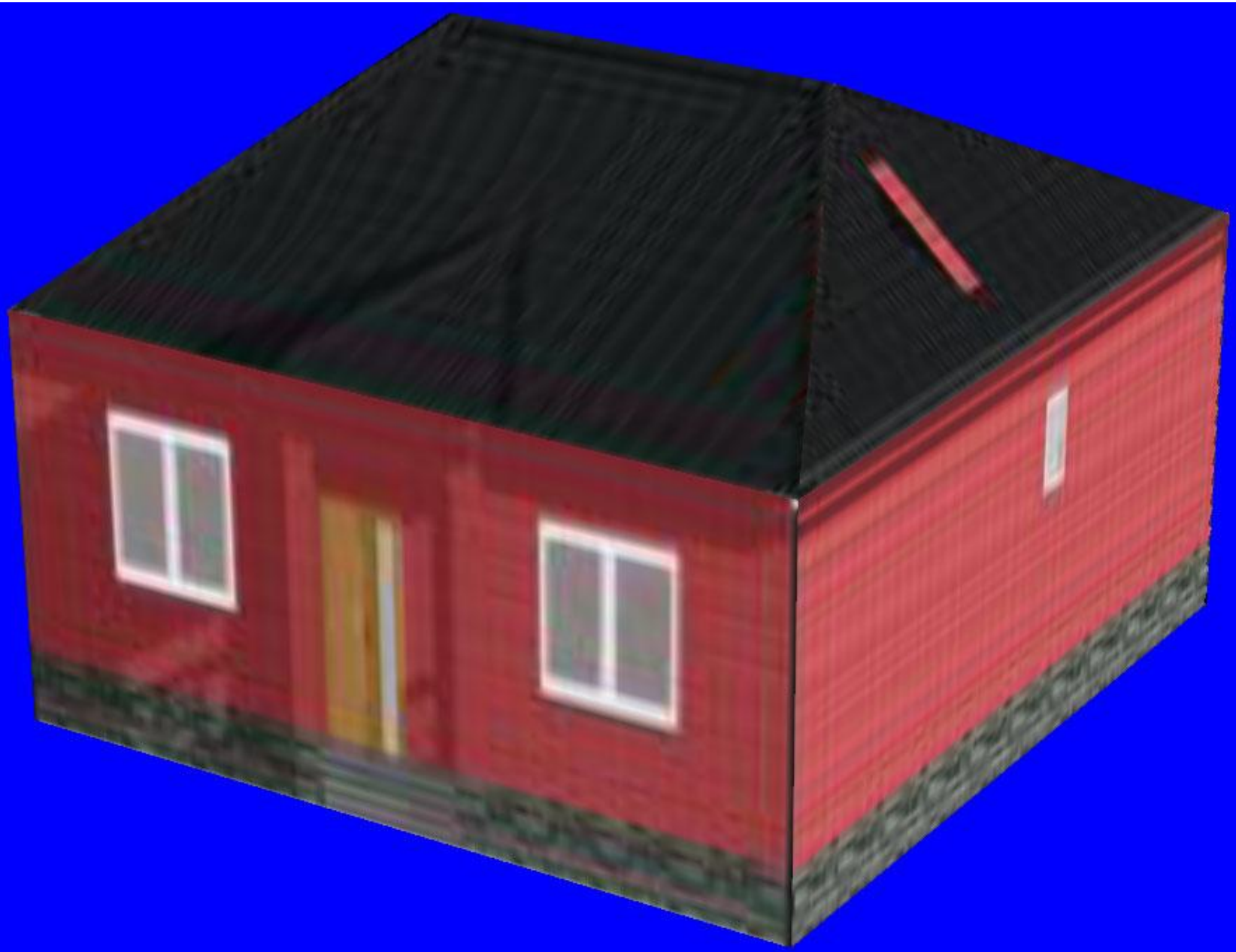
Функции рисования текстурированного четырехугольника

```
void DrawQuadT(float x1, float y1, float z1, float x2, float
   y2, float z2, float x3, float y3, float z3, float x4,
   float y4, float z4, float tx1, float ty1, float tx2, float
   ty2, float tx3, float ty3, float tx4, float ty4, int
   tindex)
{
glEnable (GL_TEXTURE_2D);
glColor3f(1.0f,1.0f,1.0f);
glBindTexture(GL_TEXTURE_2D, texture[tindex]);
  glBegin(GL_QUADS);
    glTexCoord2f(tx1, ty1);    glVertex3d(x1, y1, z1);
    glTexCoord2f(tx2, ty2);    glVertex3d(x2, y2, z2);
    glTexCoord2f(tx3, ty3);    glVertex3d(x3, y3, z3);
    glTexCoord2f(tx4, ty4);    glVertex3d(x4, y4, z4);
  glEnd();
glDisable(GL_TEXTURE_2D);
```

Функции рисования текстурированного четырехугольника

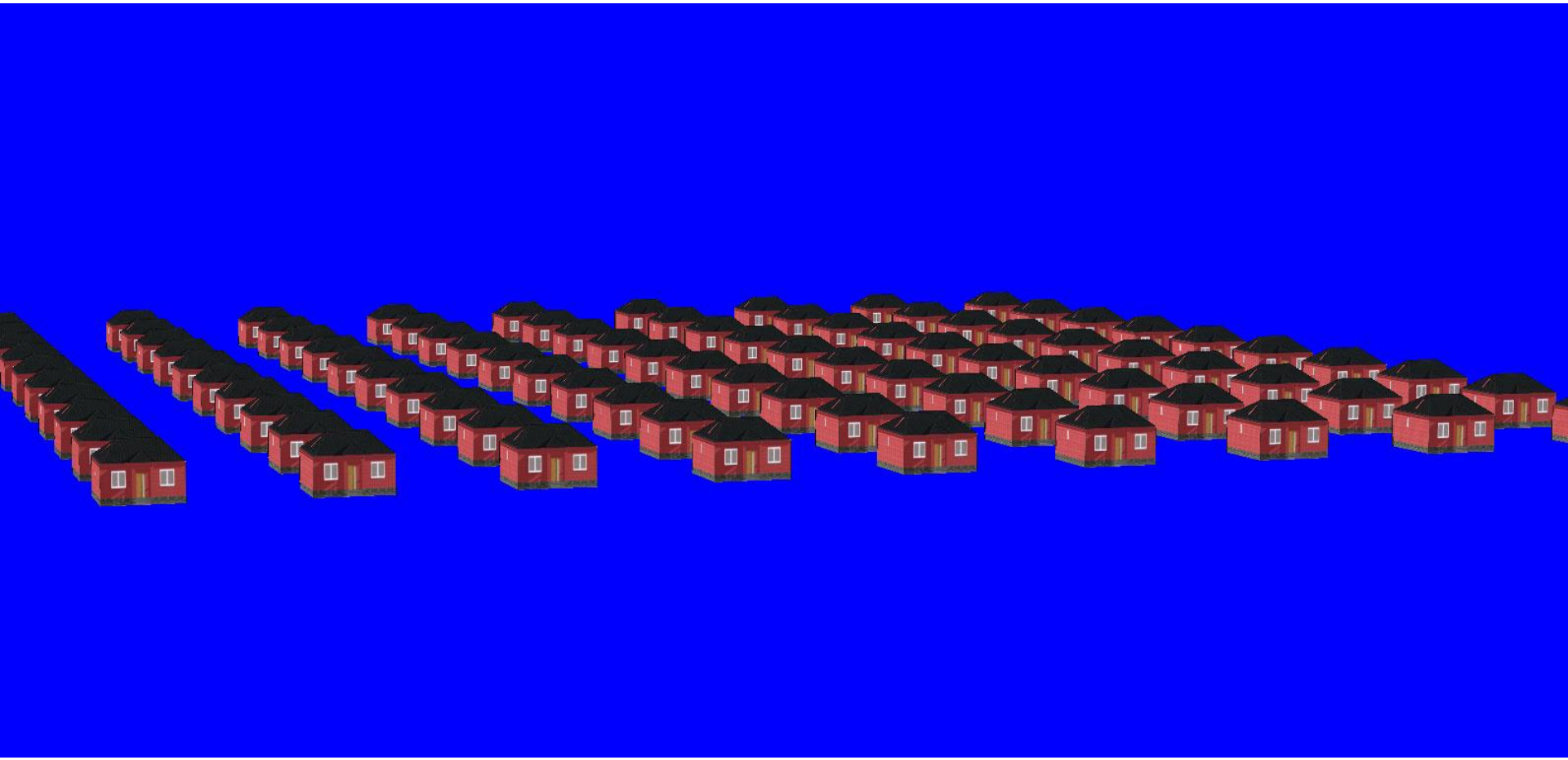
```
void DrawQuadT(vertex3 v1, vertex3 v2, vertex3 v3, vertex3 v4,
float tx1, float ty1, float tx2, float ty2,
float tx3, float ty3, float tx4, float ty4, GLuint texture, color3 color)
{
    glEnable(GL_TEXTURE_2D);
    glColor3f(color.r, color.g, color.b);
    glBindTexture(GL_TEXTURE_2D, texture);
    glBegin(GL_QUADS);
    glTexCoord2f(tx1, ty1);    glVertex3d(v1.x, v1.y, v1.z);
    glTexCoord2f(tx2, ty2);    glVertex3d(v2.x, v2.y, v2.z);
    glTexCoord2f(tx3, ty3);    glVertex3d(v3.x, v3.y, v3.z);
    glTexCoord2f(tx4, ty4);    glVertex3d(v4.x, v4.y, v4.z);
    glEnd();
    glDisable(GL_TEXTURE_2D);
}
```

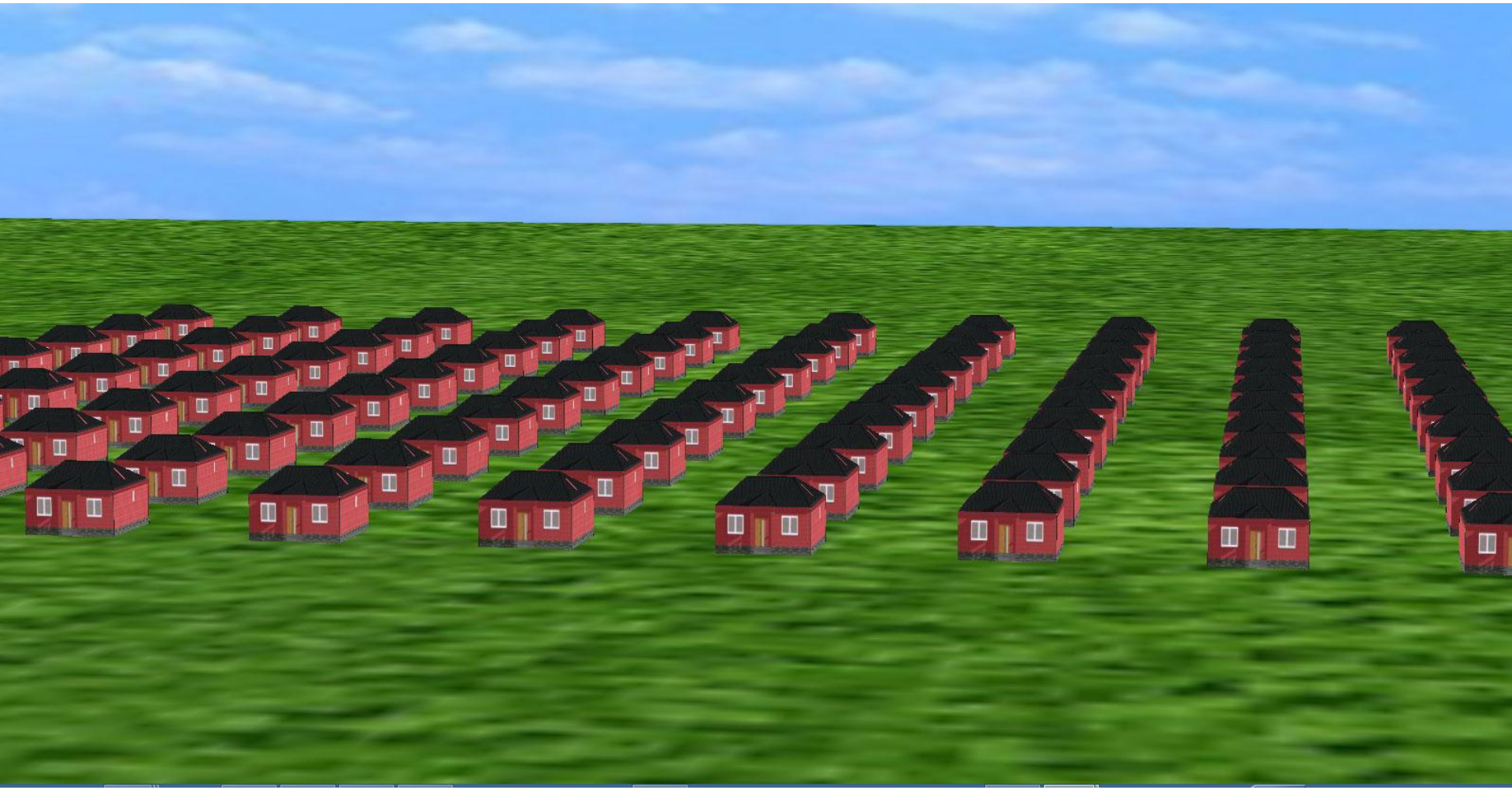




Рисование множества домов

```
void DrawSity()
{
    glScaled(0.1,0.1,0.1);
    for (int i=0; i<10; i++)
        for (int j=0; j<10; j++)
        {
            glPushMatrix();
            glTranslated(i*5, 0, j*5);
            DrawHome();
            glPopMatrix();
        }
}
```





Загрузка текстур без прозрачности

```
void LoadBMP(GLuint *texture, char *filename)
{    AUX_RGBImageRec *txt;
     txt = auxDIBImageLoad(filename);
     glGenTextures(1, texture);
     glBindTexture(GL_TEXTURE_2D, *texture);
     glTexParameteri(GL_TEXTURE_2D,
GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,
GL_TEXTURE_MIN_FILTER, GL_LINEAR);
     glTexImage2D(GL_TEXTURE_2D, 0, 3, txt->sizeX,
txt->sizeY, 0,
     GL_RGB, GL_UNSIGNED_BYTE, txt->data);
     delete txt;
}
```

Загрузка текстур без прозрачности

- Вызов функции :

- LoadBMP(&texture[0], "dom2.bmp"); //drawsity
- LoadBMP(&texture[1], "Sky.bmp");
- LoadBMP(&texture[2], "Grass.bmp");

Загрузка текстур с прозрачностью

```
void LoadTGA(GLuint *texture, char *filename)
// Loads A TGA File Into Memory
{
    GLubyte     TGAheader[12]={0,0,2,0,0,0,0,0,0,0,0,0};
// Uncompressed TGA Header
    GLubyte     TGAcompare[12];
// Used To Compare TGA Header
    GLubyte     header[6];
// First 6 Useful Bytes From The Header
    GLuint      bytesPerPixel;
// Holds Number Of Bytes Per Pixel Used In The TGA File
    GLuint      imageSize;
// Used To Store The Image Size When Setting Aside Ram
```

Загрузка текстур с прозрачностью

```
GLuint      temp;                       // Temporary Variable
GLuint      type=GL_RGBA;
// Set The Default GL Mode To RBGA (32 BPP)

GLubyte   *imageData;
// Image Data (Up To 32 Bits)
GLuintbpp;
// Image Color Depth In Bits Per Pixel.
GLuintwidth;                            // Image Width
GLuintheight;                           // Image height
```

Загрузка текстур с прозрачностью

```
FILE *file = fopen(filename, "rb");
// Open The TGA File
fread(TGAcompare,1,sizeof(TGAcompare),file);
fread(header,1,sizeof(header),file);
width  = header[1] * 256 + header[0];
// Determine The TGA Width
height = header[3] * 256 + header[2];
// Determine The TGA Height
bpp    = header[4];
// Grab The TGA's Bits Per Pixel (24 or 32)
bytesPerPixel    = bpp/8;
// Divide By 8 To Get The Bytes Per Pixel
imageSize        = width*height*bytesPerPixel; // Calculate
```

Memory

Загрузка текстур с прозрачностью

```
imageData=(GLubyte *)malloc(imageSize);
// Reserve Memory To Hold The TGA Data
fread(imageData, 1, imageSize, file);
for(GLuint i=0; i<int(imageSize); i+=bytesPerPixel)
// Loop Through The Image Data
{     // Swaps The 1st And 3rd Bytes ('R'ed and 'B'lue)
temp=imageData[i];
// Temporarily Store The Value At Image Data 'i'
imageData[i] = imageData[i + 2];
// Set The 1st Byte To The Value Of The 3rd Byte
imageData[i + 2] = temp;
// Set The 3rd Byte To The Value In 'temp' (1st Byte Value)
}
```

Загрузка текстур с прозрачностью

```
fclose (file);
// Close The File
glGenTextures(1, texture);
// Generate OpenGL texture IDs
glBindTexture(GL_TEXTURE_2D, *texture);
// Bind Our Texture
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
```

Загрузка текстур с прозрачностью

```
if (bpp==24)
// Was The TGA 24 Bits
{
        type=GL_RGB;
        // If So Set The 'type' To GL_RGB
}
glTexImage2D(GL_TEXTURE_2D, 0, type, width,
 height, 0, type, GL_UNSIGNED_BYTE, imageData);
}
```

Вызов функции : LoadTGA(&texture[0], "man.tga");

Пример текстуры с прозрачностью

Пример текстуры с прозрачностью

```
void InitMan()
{
        LoadTGA(&texture[0], "man.tga");
}
void DrawFrame(int t)
{
float w = 5;  float h = 6; int col = (t%6);  int row = (t/6);
float tx = 1/6.0*col,  ty = 1/5.0*(5-row);
DrawQuadT(-w, h, tx, ty+1/5.0,
                        w, h, tx+1/6.0, ty+1/5.0,
                        w, -h, tx+1/6.0, ty,
                        S-w, -h, tx, ty);
}
```

Пример текстуры с прозрачностью

```
void DrawMan()
{
 int t = GetTickCount()/50%30;
 DrawFrame(t);
}
```

Пример текстуры с прозрачностью

```
void DrawQuadT(float x1, float y1, float tx1, float ty1, float x2, float y2, float tx2, float ty2,  float x3, float y3, float tx3, float ty3,  float x4, float y4, float tx4, float ty4)
{    glEnable (GL_TEXTURE_2D);
      glBindTexture(GL_TEXTURE_2D, texture[0]);
      glEnable (GL_ALPHA_TEST);
      glAlphaFunc (GL_GREATER, 0.1);
      glEnable (GL_BLEND);
      glBegin(GL_QUADS);
            glTexCoord2f(tx1, ty1);  glVertex2f(x1, y1);
            glTexCoord2f(tx2, ty2);  glVertex2f(x2, y2);
            glTexCoord2f(tx3, ty3);  glVertex2f(x3, y3);
            glTexCoord2f(tx4, ty4);  glVertex2f(x4, y4);
      glEnd();
      glDisable(GL_TEXTURE_2D);    glDisable(GL_BLEND);
}
```

Примеры спрайт листов

Примеры спрайт листов

Примеры спрайт листов