

# **Zaawansowane systemy programowania grafiki. Przezroczystość oraz sprajty punktowe**

Aleksander Denisiuk  
Uniwersytet Warmińsko-Mazurski  
Olsztyn, ul. Słoneczna 54  
[denisjuk@matman.uwm.edu.pl](mailto:denisjuk@matman.uwm.edu.pl)

18 maja 2021

# ***Przezroczystość oraz sprajty punktowe***

Przezroczystość

Sprajty punktowe

Najnowsza wersja tego dokumentu dostępna jest pod adresem

<http://wmii.uwm.edu.pl/~denisjuk/uwm>

## Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcian
- ❖ Funkcje OpenGL

Sprajty punktowe

# Przezroczystość

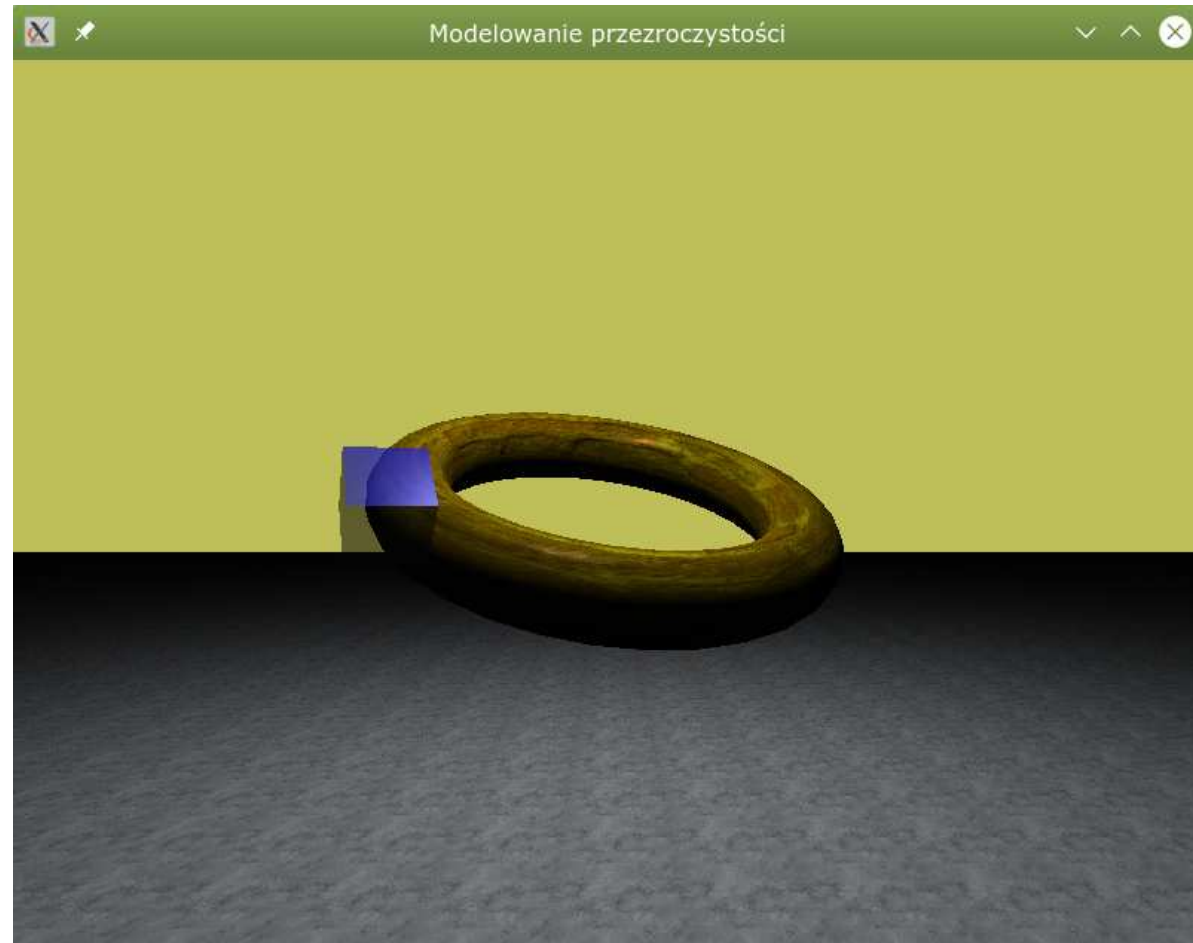
# Scena

## Przezroczystość

### ❖ Scena

- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcian
- ❖ Funkcje OpenGL

## Sprajty punktowe



# Czynności

Przezroczystość

❖ Scena

❖ Shadery

❖ Programy

❖ Programy

❖ Sześcian

❖ Funkcje OpenGL

Sprajty punktowe

- Oteksturowany torus
- Przezroczysty sześcian bez tekstury
- Każdy obiekt ma swój program cieniujący

# Shadery

Przezroczystość

❖ Scena

❖ Shadery

❖ Programy

❖ Programy

❖ Sześcian

❖ Funkcje OpenGL

Sprajty punktowe

- Jak światło punktowe, tylko bez tekstury

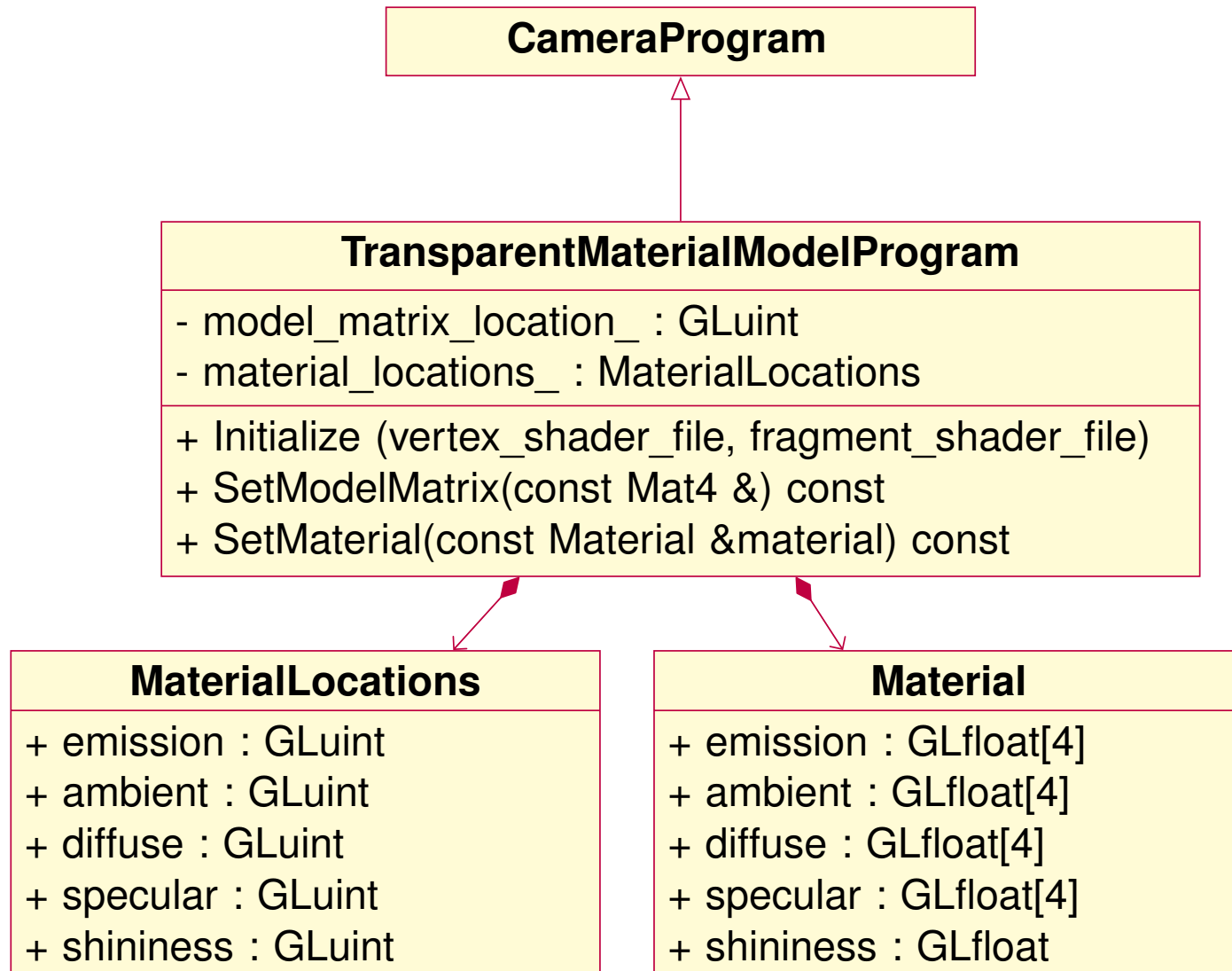
```
struct Vertex {  
    vec3    normal;  
    vec3    light_dir;  
    vec3    view_dir;  
    float   dist;  
} frag_vertex;
```

# TransparentMaterialModelProgram

## Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcian
- ❖ Funkcje OpenGL

## Sprajty punktowe

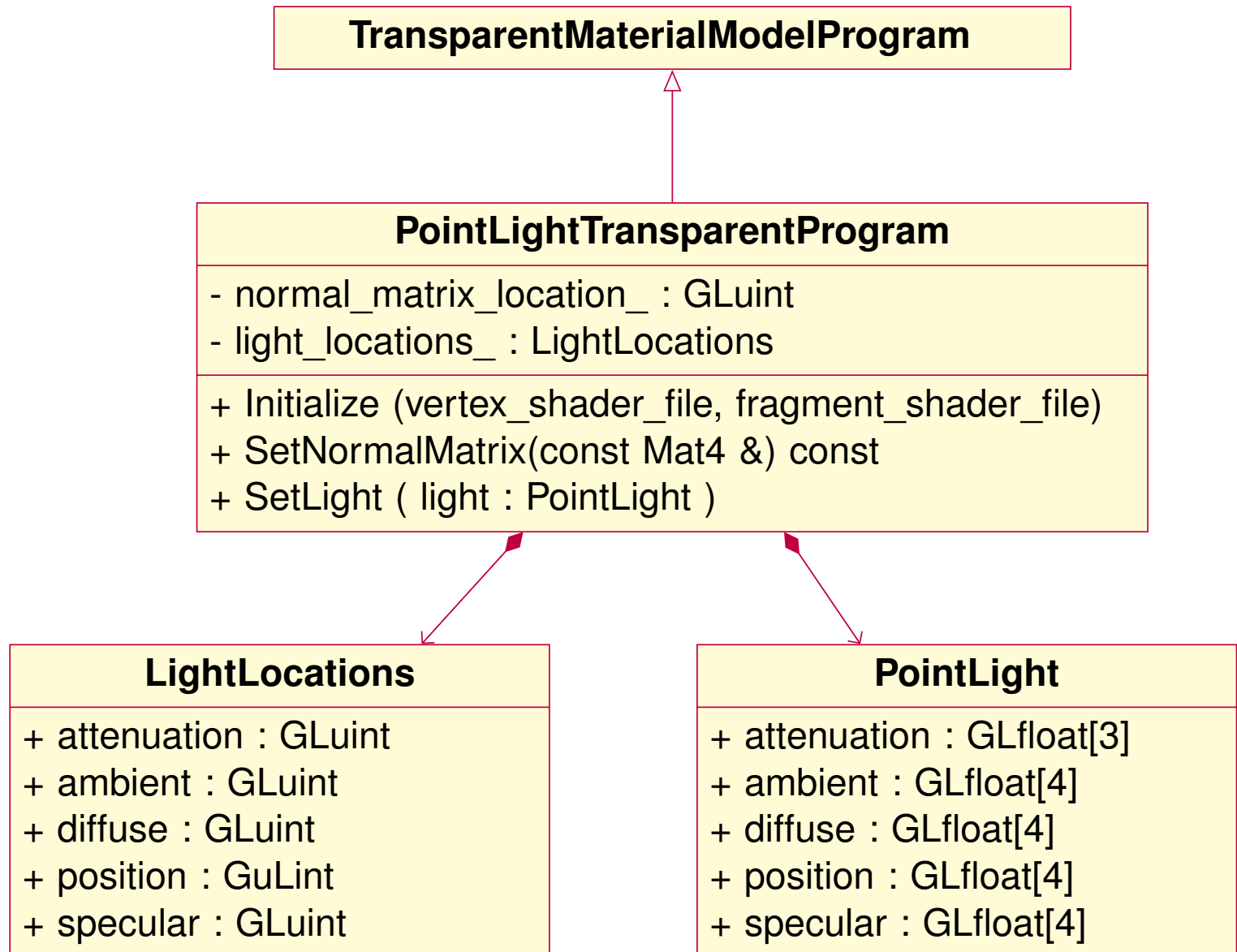


# PointLightTransparentProgram

## Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcian
- ❖ Funkcje OpenGL

## Sprajty punktowe





# Wierzchołki sześcianu

Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcián
- ❖ Funkcje OpenGL

Sprajty punktowe

```
typedef struct MaterialVertex{  
    float position[4];  
    float normal[3];  
} MaterialVertex;
```

# ***Materiał sześcianu***

Przezroczystość

❖ Scena

❖ Shadery

❖ Programy

❖ Programy

❖ **Sześcian**

❖ Funkcje OpenGL

Sprajty punktowe

```
Material RedMaterial={  
    {0.2f, 0.2f, 0.2f, 0.25f}, //Ambient  
    {0.9f, 0.1f, 0.1f, 0.25f}, //Diffuse  
    {0.5f, 0.5f, 0.5f, 0.25f}, //Specular  
    {0.0f, 0.0f, 0.0f, 0.25f}, //Emission  
    20.0f //Shininess  
};
```

# Wyświetlenie sześcianu

## Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcián
- ❖ Funkcje OpenGL

## Sprajty punktowe

```
.....  
glEnable (GL_BLEND) ;  
glBlendFunc (GL_SRC_ALPHA,  
             GL_ONE_MINUS_SRC_ALPHA) ;  
glDepthMask (0) ;  
  
glDrawElements (GL_TRIANGLES, 36,  
               GL_UNSIGNED_INT, (GLvoid*) 0) ;  
  
glDisable (GL_BLEND) ;  
glDepthMask (1) ;  
  
.....
```

- Przezroczyste — jako ostatnie

# Mieszanie kolorów

## Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcián
- ❖ Funkcje OpenGL

## Sprajty punktowe

```
void glBlendFunc (GLenum sfactor,  
                  GLenum dfactor);
```

- `sfactor` — współczynnik przy źródłowym kolorze
- `dfactor` — współczynnik przy kolorze docelowym

$$\text{Kolor} = C_s \cdot \text{sfactor} + C_d \cdot \text{dfactor}$$

- przykładowe współczynniki:

- ◆ `GL_ZERO, GL_ONE`
- ◆ `GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR`
- ◆ `GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR`
- ◆ `GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA`
- ◆ `GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA`
- ◆ `GL_CONSTANT_COLOR,`  
`GL_ONE_MINUS_CONSTANT_COLOR`
- ◆ `GL_CONSTANT_ALPHA,`  
`GL_ONE_MINUS_CONSTANT_ALPHA`

# *Pisanie do buforu głębokości*

Przezroczystość

- ❖ Scena
- ❖ Shadery
- ❖ Programy
- ❖ Programy
- ❖ Sześcian
- ❖ Funkcje OpenGL

Sprajty punktowe

```
void glDepthMask (GLboolean flag);
```

Przezroczystość

**Sprajty punktowe**

- ❖ Scena
- ❖ Shadery
- ❖ Punkty
- ❖ Tekstura
- ❖ Program
- ❖ Window
- ❖ Tryb  
pełnoekranowy

# Sprajty punktowe

# Scena

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

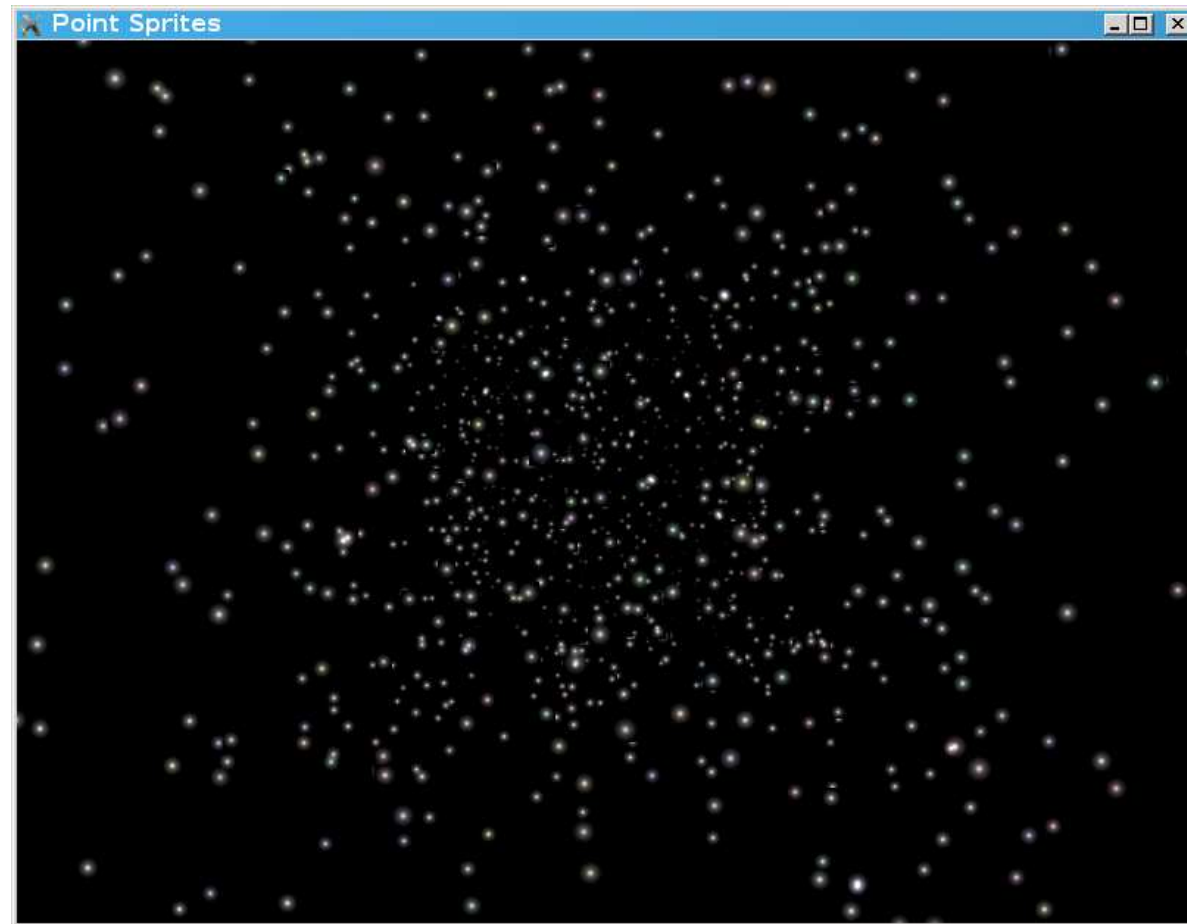
❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy



# Czynności

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

- Punkty jako piksele  $n \times n$
- Tekstura na punktach
- Program cieniujący
- Animacja



# Shader wierzchołków. Parametry i zmienne

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
#version 430 core
```

```
layout (location = 0) in vec4 in_position;
```

```
layout (location = 1) in vec4 in_color;
```

```
uniform float time;
```

```
uniform mat4 projection_matrix;
```

```
flat out vec4 star_color;
```

# Shader wierzchołków, *main*

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
void main(void) {  
    vec4 vertex = in_position;  
  
    vertex.z += time;  
    vertex.z = fract(vertex.z);  
  
    float size = (20.0 * vertex.z * vertex.z);  
  
    star_color  
        = smoothstep(1.0, 7.0, size) * in_color;  
  
    vertex.z = (999.9 * vertex.z) - 1000.0;  
    gl_Position = projection_matrix * vertex;  
    gl_PointSize = size;  
}
```

# Shader fragmentów

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
#version 430 core
```

```
layout (location = 0) out vec4 color;
```

```
uniform sampler2D texture_unit;
```

```
flat in vec4 star_color;
```

```
void main(void) {  
    color = star_color  
        * texture(texture_unit, gl_PointCoord);  
}
```

# Wierzchołki dla punktów

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
typedef struct ColorVertex{  
    float Position[4];  
    float Color[4];  
} StarVertex;
```

- Już było

# Nowy model

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
class ArrayModel {
public:
    ~ArrayModel();

protected:
    GLuint vao_;
    GLuint vertex_buffer_;
};

ArrayModel::~~ArrayModel() {
    glDisableVertexAttribArray(1);
    glDisableVertexAttribArray(0);
    glBindBuffer(GL_ARRAY_BUFFER, 0);
    glDeleteBuffers(1, &vertex_buffer_);
    glBindVertexArray(0);
    glDeleteVertexArrays(1, &vao_);
}
```

- Może dodać `glDisableVertexAttribArray(2), 3, 4?`

# Klasa Stars

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

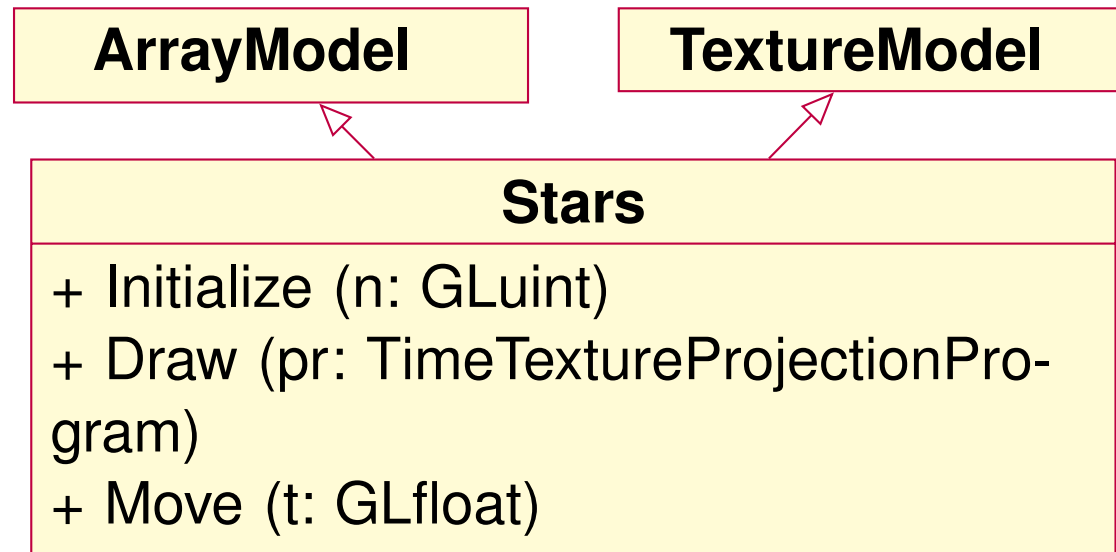
❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy



```
void Stars::Move(GLfloat t) {
    time_ = t;
}
```

# Stars: inicjalizacja

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
void Stars::Initialize(GLuint n) {  
    n_ = n;  
    time_ = 0.0f;  
    srand(time(NULL));  
    glGenVertexArrays(1, &vao_);  
    glBindVertexArray(vao_);  
  
    glGenBuffers(1, &vertex_buffer_);  
    glBindBuffer(GL_ARRAY_BUFFER, vertex_buffer_);  
}
```

# Mapowanie pamięci GPU na CPU

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
glBufferData(GL_ARRAY_BUFFER,
             n_ * sizeof(ColorVertex),
             NULL,
             GL_STATIC_DRAW);
ColorVertex * star
= (ColorVertex*) glMapBufferRange(
    GL_ARRAY_BUFFER, 0,
    n_ * sizeof(ColorVertex),
    GL_MAP_WRITE_BIT
    | GL_MAP_INVALIDATE_BUFFER_BIT);
```



# Generowanie wierzchołków

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
for (unsigned int i = 0; i < n_; i++) {
    star[i].position[0]
        = (random_float() * 2.0f - 1.0f) * 100.0f;
    star[i].position[1]
        = (random_float() * 2.0f - 1.0f) * 100.0f;
    star[i].position[2] = random_float();
    star[i].position[3] = 1.0f;
    star[i].color[0] = 0.8f+random_float()*0.2f;
    star[i].color[1] = 0.8f+random_float()*0.2f;
    star[i].color[2] = 0.8f+random_float()*0.2f;
    star[i].color[3] = 1.0f;
}
glUnmapBuffer(GL_ARRAY_BUFFER);
```

# Argumenty shadera wierzchołków

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
glVertexAttribPointer(0, 4, GL_FLOAT,  
                      GL_FALSE, sizeof(ColorVertex), NULL);  
glVertexAttribPointer(1, 4, GL_FLOAT,  
                      GL_FALSE, sizeof(ColorVertex),  
                      (void *) sizeof(star[0].position));  
glEnableVertexAttribArray(0);  
glEnableVertexAttribArray(1);
```

## ● gdzie

```
float random_float() {  
    return ((float)rand() / RAND_MAX);  
}
```

# Wyświetlenie

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

```
void Stars::Draw(  
    const TimeTextureProjectionProgram &pr) {  
    glUseProgram(pr);  
    pr.SetTime(time_);  
    glBindVertexArray(vao_);  
    glActiveTexture(texture_unit_);  
    glBindTexture(GL_TEXTURE_2D, texture_);  
    glEnable(GL_BLEND);  
    glBlendFunc(GL_ONE, GL_ONE);  
    glEnable(GL_PROGRAM_POINT_SIZE);  
    glDrawArrays(GL_POINTS, 0, n_);  
    glDisable(GL_BLEND);  
    glDisable(GL_PROGRAM_POINT_SIZE);  
    glBindVertexArray(0);  
    glUseProgram(0);  
}
```

# *Tekstura dla gwiazd*

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

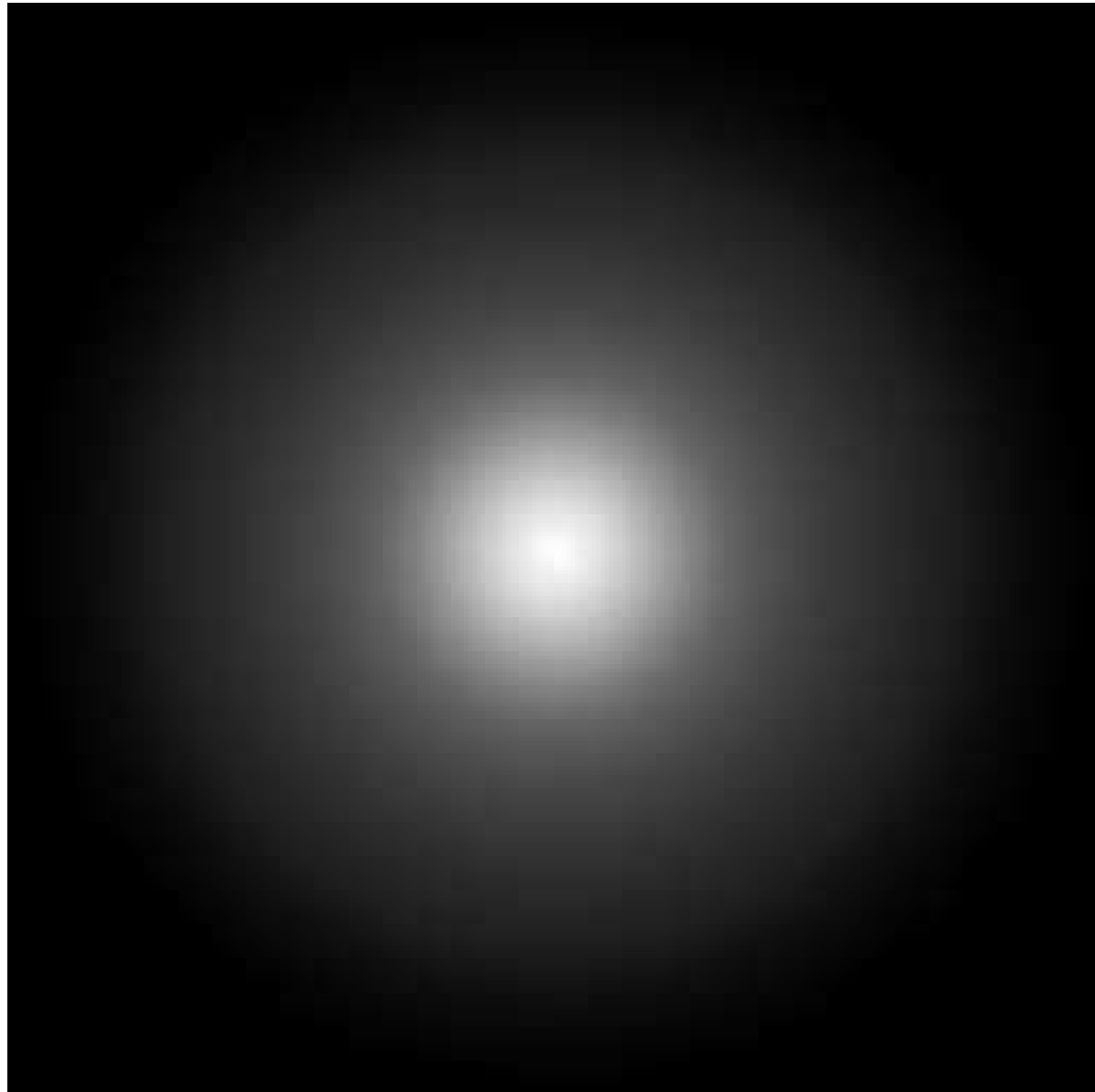
❖ Punkty

❖ **Tekstura**

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy



# Program

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

**BaseProgram**



**ProjectionProgram**

- projection\_matrix\_location\_ : GLuint

+ Initialize (vertex\_shader\_file, fragment\_shader\_file);

+ SetProjectionMatrix (Mat4)

# GLint GetUniformLocationOrDie(const char \*);

# *TextureProjectionProgram*

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

**ProjectionProgram**



**TextureProjectionProgram**

- texture\_unit\_location\_ : GLuint

+ Initialize (vertex\_shader\_file, fragment\_shader\_file);

+ SetTextureUnit (GLfloat)

# *TimeTextureProjectionProgram*

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ Window

❖ Tryb  
pełnoekranowy

**TextureProjectionProgram**



**TimeTextureProjectionProgram**

- time\_location\_ : GLuint

+ Initialize (vertex\_shader\_file, fragment\_shader\_file);

+ SetTime (GLfloat)

# Window

Przezroczystość

Sprajty punktowe

❖ Scena

❖ Shadery

❖ Punkty

❖ Tekstura

❖ Program

❖ **Window**

❖ Tryb

pełnoekranowy

```
void Window::Run(void) {  
    while (!glfwWindowShouldClose(window_)) {  
        glClear(GL_COLOR_BUFFER_BIT  
                | GL_DEPTH_BUFFER_BIT);  
  
        stars_.Move(0.1f * static_cast<float>(  
            clock()) / CLOCKS_PER_SEC);  
        stars_.Draw(program_);  
  
        glfwSwapBuffers(window_);  
        glfwPollEvents();  
    }  
}
```



# Tryb pełnoekranowy

Przezroczystość

Sprawy punktowe

- ❖ Scena
- ❖ Shadery
- ❖ Punkty
- ❖ Tekstura
- ❖ Program
- ❖ Window
- ❖ Tryb pełnoekranowy

```
glfwSetWindowMonitor(window_,  
                      glfwGetPrimaryMonitor(),  
                      0, 0, width_, height_,  
                      GLFW_DONT_CARE);
```

## ● Wyjście:

```
glfwSetWindowMonitor(window_,  
                      nullptr,  
                      0, 0, width_, height_,  
                      GLFW_DONT_CARE);
```