УДК 004.92

Создания эффекта движения звезд на языке шейдера

Черкашин Александр Михайлович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс создания эффект движения звезд. В процессе работы использовался язык шейдера GLSL. В результате был выведен эффект движения звезд.
Ключевые слова: шейдер, OpenGL Shading Language, GLSL, эффект.

Creating the effect of moving stars in the shader language

Cherkashin Alexander Mihailovich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the process of creating the effect of moving stars. In the process of work, the GLSL shader language was used. As a result, the effect of the movement of stars was derived.
Keywords: shader, OpenGL Shading Language, GLSL, effect.

1 Введение

1.1 Актуальность исследования

Данная статья описывает возможность написания на языке шейдера создание эффекта звезд.

1.2 Цель исследования

Целью работы является создание эффекта звезд при помощи написания программы на языке шейдера.

1.3 Обзор исследований

В работе М. Стийн предлагает программу на языке шейдера в Unreal Engine, демонстрирующую динамический цикл смены дня и ночи, модели движения солнца, луны и звездного неба. Программа может быть использована для создания игр с хорошим соотношением производительности и точности [1]. Й. Юднич представляет приложение Kosmos которое демонстрирует интерактивную визуализацию вымышленной трехмерной планетарной системы со звездой [2]. В работе Дел Н. А. Гроссо, А. Сирота описывают библиотеку с открытым исходным кодом под названием Ratcave, которая выполняя роль 3D движка использует OpenGL [3].

2. Рабочий процесс

В данной работе использовались два изображения (рис 2.1 и 2.2) для демонстрации программы шейдера, с использованием программы glslViewer [4].

Программа написана на языке OpenGL Shading Language (GLSL).

Листинг 2.1. Исходный код программы для создания эффекта движения звезд.

```
1  #version 330
2  #ifdef GL_ES
3  precision mediump float;
4  #endif
5  #define MAXARR 8
6  uniform vec2 u_resolution;
7  uniform float u_time;
8  vec3 hsv2rgb(vec3 c) {
9          vec4 K = vec4(1.0, 2.0 / 3.0, 1.0 / 3.0, 3.0);
10         vec3 p = abs(fract(c.xxx + K.xyz) * 6.0 - K.www);
11         return c.z * mix(K.xxx, clamp(p - K.xxx, 0.0, 1.0), c.y);
12 }
13 void main (void) {
14         vec3 col = vec3(0.0);
15         vec2 uv = (gl_FragCoord.xy / u_resolution.xy * 2.0) - 1.0;
16         int imax = MAXARR;
17         vec3 p[MAXARR];
18         float cola[MAXARR];
19         float v[MAXARR];
20         float i_p = 0.0;
21         for (float s = 0.0; s < 2000.0; s++) {
22                 for (int i = 0; i < imax; i++) {
23                         i_p = float(i) / float(imax);
24                         p[i] = abs(p[i]) / v[i] - 0.8;
25                         if (int(s) % 10 == 0) {
26                                 p[i] = vec3(0.01 * float(i+1), 0.2 * float(i+1),
   0.2*fract(0.01 * ceil(0.05 * s + u_time * 20.0)) );
27                                 cola[i] += v[i] / 20000.0 * float(i+1);
28                                 p[i].xy += s * uv / 20000.0 * float(i+1);
29                         }
30                         v[i] = dot(p[i], p[i]);
31                 }
32         }
33         for (int i = 0; i < imax; i++) {
34                 i_p = float(i) / float(imax);
35                 col += hsv2rgb(vec3(i_p, 0.8, cola[i]));
36         }
```

```
37          gl_FragColor = vec4(col, 1.0);
38  }
```

В строке 1 (листинг 2.1) выбрана версия шейдера 3.30.

В строках 5 — 7:

u_resolution - разрешение экрана,

u_time — время в секундах,

MAXARR — максимальное количество итераций для создания разных звезд.

В строках 8 — 12:

Функция hsv2rgb преобразует цветовую модель от HSV до RGB, аргументы:

hsv.x — цветовой тон (hue).

hsv.y — насыщенность (saturation).

hsv.z — значение цвета, яркость (value) [5].

В строках 14 — 32:

Переменная uv — экранное пространство от -1 до 1. col — основной цвет. imax — максимальное число итераций для создания разных слоев звезд. P — координаты звезд, cola — яркость звезд, i — шаг итерации для слоя звезд, s — шаг итерации для глубины третей оси координаты, i_p — i / imax.

В строке 26 задаем координаты различных звезд, используем движение по глубине.

В строке 27 рисуем звезды разной яркости.

В строке 28 задаем расстояние между звездами.

В строке 33 — 36 складываем яркость, и задаем цветовой тон в зависимости номера i слоя звезд.

glslViewer main.frag -w 800 -h 600

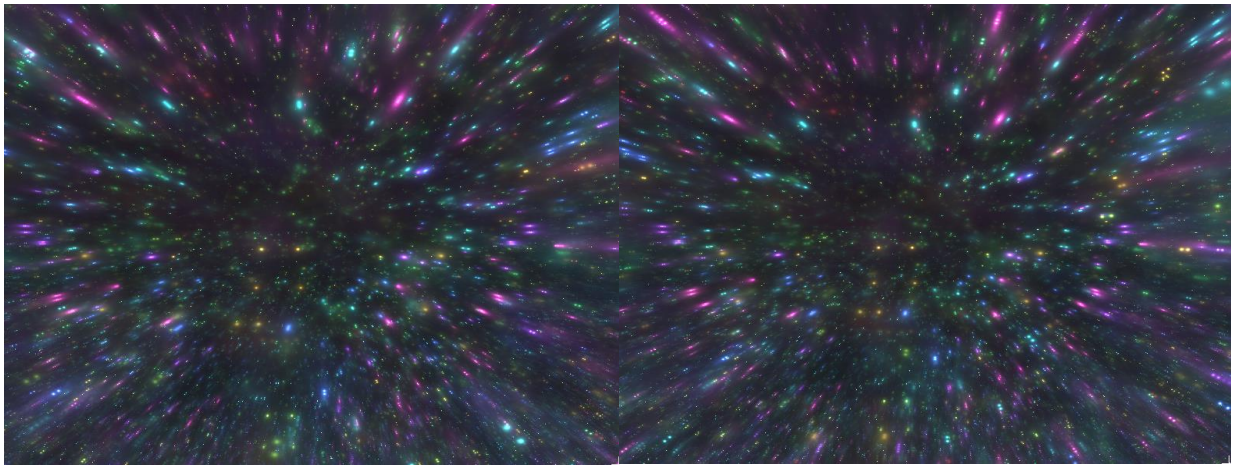

Рисунок 2.1. Результаты работы программы

3 Выводы

В результате работы программа была написана на языке шейдера для получения эффекта звезд.

Библиографический список

1. Stijn M. Dynamic Day and Night Cycle in a Binary Star System. Graduation work 2018-19. URL: http://stijnmaris.com/wp-content/uploads/2019/03/StijnMaris_ResearchPaper_DynamicDayNightCycle.pdf
2. Judnich J. Kosmos: a virtual 3-D universe. Theses/Dissertations. Santa Clara, 2013.
3. Del Grosso N. A., Sirota A. Ratcave: A 3D graphics python package for cognitive psychology experiments //Behavior research methods. 2019. Т. 51. №. 5. С. 2085-2093.
4. patriciogonzalezvivo/glslViewer: Console-based GLSL Sandbox for 2D/3D shaders shaders. // GitHub URL: https://github.com/patriciogonzalezvivo/glslViewer (дата обращения: 2022-09-04).
5. hughsk/glsl-hsv2rgb: Fast GLSL conversion from HSV color to RGB // GitHub URL: https://github.com/hughsk/glsl-hsv2rgb (дата обращения: 2022-09-04).