

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Забайкальский государственный университет»  
(ФГБОУ ВО «ЗабГУ»)

Энергетический факультет  
Кафедра информатики, вычислительной техники и прикладной математики

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

Тема: «Разработка графического движка на основе обратной трассировки лучей»

Выполнил:  
студент гр. ВМК-16 \_\_\_\_\_  \_\_\_\_\_ Мархандаев Сергей Викторович

Руководитель ВКР:  
к.т.н., доцент кафедры ИВТ и ПМ \_\_\_\_\_ Макаров Дмитрий Андреевич

Чита  
2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Забайкальский государственный университет»  
(ФГБОУ ВО «ЗабГУ»)

Энергетический факультет  
Кафедра информатики, вычислительной техники и прикладной математики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
к выпускной квалификационной работе

по направлению подготовки 09.03.01 Информатика и вычислительная техника  
профиль Вычислительные машины, комплексы, системы и сети

на тему: «Разработка графического движка на основе обратной трассировки лучей»

Выполнил студент группы ВМК-16 Мархандаев Сергей Викторович

Консультанты:

а) Экономическая часть: Ткач ст. преподаватель кафедры ЭиБУ,  
Т.И. Кашурникова

б) Безопасность и экологичность проекта: ВР доцент кафедры ВХЭиПБ, к.т.н.,  
И.А. Бондарь

в) Специальная часть: Макаров доцент кафедры ИВТ и ПМ, к.т.н.,  
Д.А. Макаров

Нормоконтроль: Шевелёва старший преподаватель, Е.Б. Шевелёва

Руководитель работы: доцент кафедры ИВТ и ПМ, к.т.н., Макаров Дмитрий Андреевич

Допускаю к защите:

Зав. кафедрой ИВТ и ПМ Валова О.В. Валова

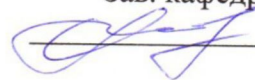
15 июня 2020 г.

Чита  
2020

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Забайкальский государственный университет»  
(ФГБОУ ВО «ЗабГУ»)

Энергетический факультет  
Кафедра информатики, вычислительной техники и прикладной математики

УТВЕРЖДАЮ  
Зав. кафедрой ИВТ и ПМ

 О.В. Валова  
11 мая 2020 г.

ЗАДАНИЕ  
на выпускную квалификационную работу

Студенту Мархандаеву Сергею Викторовичу  
по направлению 09.03.01 Информатика и вычислительная техника  
профиль Вычислительные машины, комплексы, системы и сети

1 Тема выпускной квалификационной работы «Разработка графического движка на основе обратной трассировки лучей»

Утверждена приказом (распоряжением) ректора по университету от 20.12.2019 № 1003-с

2 Срок подачи студентом законченной работы 15 июня 2020 г.

3 Исходные данные к работе:

- документация по игровому движку Unity;
- документация по использованным дополнительным модулям;
- спецификация языка C#.

4 Перечень подлежащих разработке в выпускной квалификационной работе вопросов:

- а) специальная часть;
- б) экономическая часть;
- в) безопасность и экологичность проекта.

5 Перечень графического материала (если имеется):-

6 Консультанты по выпускной квалификационной работе (с указанием, относящихся к ним разделов):

а) экономическая часть: Кашурникова Тина Иннокентьевна Тес

б) безопасность и экологичность проекта: Бондарь Ирина Алексеевна ИР

в) нормоконтроль: Шевелёва Екатерина Борисовна ШЕ

г) специальная часть: Макаров Дмитрий Андреевич МА

Дата выдачи задания 11 мая 2020 г.

Руководитель ВКР Д.А Макаров Д.А Макаров

(подпись, расшифровка подписи)

Задание принял к исполнению

11 мая 2020 г.

Подпись студента Сергей / Мархандаев Сергей Викторович /  
(имя, отчество, фамилия)

# Календарный план

№	Наименование раздела выпускной квалификационной работы	Неделя							
		1	2	3	4	5	6	7	8
1	Теоретическая часть	■							
2	Специальная часть		■	■	■				
3	Экономическая часть					■			
4	Безопасность и экологичность проекта						■		
5	Защита выпускной квалификационной работы							■	■



(подпись, расшифровка подписи)

/ Д.А Макаров / 15 июня 2020 г.

## РЕФЕРАТ

Пояснительная записка – 48 с., 12 табл., 7 источников.

C++, ОБРАТНАЯ ТРАССИРОВКА ЛУЧЕЙ, QT CREATOR 4.7.2,  
МОДЕЛИРОВАНИЕ ПОВЕДЕНИЯ СВЕТА

В данной работе представлена реализация графического движка при помощи метода обратной трассировки лучей средствами среды программирования Qt Creator 4.7.2.



## СОДЕРЖАНИЕ

1 Постановка и анализ задачи.....	10
2 Анализ данных.....	11
3 Программная реализация.....	12
4 Документирование.....	24
4.1 Техническое задание.....	24
4.2 Руководство пользователя.....	25
5 Экономическая часть.....	24
5.1 Основные положения экономической части.....	26
5.1 Трёхуровневый анализ продукта.....	27
5.2 Определение трудоёмкости разработки программы.....	29
5.3 Определение стоимости разработки продукта.....	34
5.4 Безопасность и экологичность проекта.....	37
6 Общие положения охраны труда.....	37
6.1 Требования к ПЭВМ.....	37
6.2 Требования к помещениям для эксплуатации ПЭВМ.....	40
6.3 Требования к шуму и вибрации в помещениях с эксплуатируемым ЭВМ.....	41
6.4 Требования к освещению помещений и рабочих мест с ПЭВМ.....	41
6.5 Требования к организации и оборудованию рабочих мест.....	43
6.6 Требования к организации медицинского обслуживания.....	45
6.7 Требования электробезопасности.....	45
6.8 Меры оказания первой медицинской помощи при поражении электрическим током .....	48
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ .....	49

## ВВЕДЕНИЕ

ООО «ВЕКТОР» – занимается выполнением работ подготовительного периода при прокладке трубопроводов, строительстве ЛЭП, а также расчисткой просек.

В этой нише предприятие работает с 2016 года.

В связи с возможной сменой или добавлением новой специализации предприятию необходимо начать новую линию по разработке программного обеспечения.

В качестве первого шага в данном направлении руководство предприятия ООО «ВЕКТОР» приняло решение собственными силами реализовать графический движок, что было поручено сотруднику ООО «ВЕКТОР» Мархандаеву С. В.



## **1 Постановка и анализ задачи**

Графический движок будет предназначен для просмотра 3D-сцен. Должны присутствовать возможность как работы в реальном времени, так и процесс создания улучшенного изображения.

Исходя из современных требований к подобному программному обеспечению, реализация данного экземпляра графического движка должна быть выполнена с опорой на возможности графического процессора видеокарты.

Исходя из возможностей предприятия и личных способностей ответственного за данную задачу сотрудника предприятия, предпочтительным методом визуализации сцены является обратная трассировка лучей как самый доступный на начальном этапе понимания и наиболее простой в реализации силами одного сотрудника.

## **2 Анализ данных**

Для графического движка данными является содержимое визуализируемой 3D-сцены, включающее в себя:

- взаимное расположение объектов относительно сцены и друг друга;
- размеры объектов;
- материалы, из которых состоят объекты, которые представляются характеристиками поверхности объекта;
- источники света, их цвет, сила и положение в пространстве сцены;
- цвет фона сцены.

Выходные данные – двумерный массив пикселей, при этом для каждого пикселя произведена процедура обратной трассировки лучей. Этот массив может выводиться на экран компьютера, либо записан в файл как изображение определенного формата.

### 3 Программная реализация

Графический движок написан на языке C++ в редакторе Qt Creator

Файл widget.h

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>
#include <QImage>
#include <QMatrix4x4>

#define FLOAT_INF std::numeric_limits<float>::max() struct

Light_source
{
    QVector3D position;
    float power;

    Light_source(const QVector3D &position, const float &power):
        position(position), power(power){}

    Light_source(): position(QVector3D(0, 0, 0)), power(0) {}
};

struct Sphere
{
    QVector3D center;
    float radius; QColor
    color;

    Sphere(const QVector3D &center, const float &radius, const QColor
    &color):
        center(center), radius(radius), color(color) {}

    Sphere():
        center(QVector3D(0, 0, 0)), radius(0), color(QColor(0, 0, 0)) {}

    bool intersection(const QVector3D &origin, const QVector3D
    &ray_direction,
                    QVector3D &intersection) const
    {
        // Vector between the ray's origin and the sphere's center QVector3D
        from_origin_to_center = center - origin;

        float dp = QVector3D::dotProduct(from_origin_to_center,
        ray_direction);

        // Center of the sphere is behind the ray's origin if (dp < 0)
        {
            intersection.setX(FLOAT_INF);
```

```

        intersection.setY(FLOAT_INF);
        intersection.setZ(FLOAT_INF);
        return false;
// Scalar projection of from_origin_to_center on ray_direction float sp = dp /
ray_direction.length();

// Vector projection of from_origin_to_center on ray_direction QVector3D vp = origin
+ sp * ray_direction;

float distance = vp.distanceToPoint(center); // Distance between ray and center

// The ray misses the sphere if
(distance > radius)
{
    intersection.setX(FLOAT_INF);
    intersection.setY(FLOAT_INF);
    intersection.setZ(FLOAT_INF);
    return false;
}

intersection = origin + ray_direction *
                (sp - sqrtf(radius * radius - distance * distance));

    return true;
}
};

```

```

class Widget : public QWidget
{
    Q_OBJECT
    QImage image;
    QMatrix4x4 cameraToWorldMatrix;

    QVector3D cameraPosition; float
    degreesX;
    float degreesY; float
    degreesZ;

    QVector3D axisX;
    QVector3D axisZ;

    QSize imageSize;

    double fov;
    float aspect_ratio;

    QVector<Sphere> spheres;
    QVector<Light_source> lights;

    QColor scene_color;

```

```

public:
    Widget(QWidget *parent = nullptr, int width = 480, int height = 270);
    ~Widget();

    void setImageSize(int width, int height); void
    addSphere(int imageX, int imageY); void
    deleteSphere(int imageX, int imageY);
    void resizeSphere(int imageX, int imageY, const QPoint &angle);

protected:
    virtual void paintEvent(QPaintEvent*); virtual void
    keyPressEvent(QKeyEvent *event);

    virtual void mouseReleaseEvent(QMouseEvent *event); virtual
    void wheelEvent(QWheelEvent *event);

private:
    void render(QImage &image);
    void transform(QVector3D &origin, QVector3D &pixel_position);
    bool raySceneIntersection(const QVector3D &rayOrigin, const QVector3D
    &rayDirection,
    &closest_intersection);
    int &sphere_number, QVector3D
    QColor trace(const QVector3D &origin, const QVector3D &direction);
    };

#endif // WIDGET_H

```

Файл widget.cpp

```

#include "widget.h"
#include <QPainter>
#include <QKeyEvent>
#include <QtMath>
#include <QVector3D>
#include <QVector4D>
#include <QMatrix4x4>
#include <QLayout>
#include <QLabel>
#include <QLineEdit>

QColor light(const QVector<Sphere> &spheres, const QVector<Light_source>
&lights,
             const int &sphere_number, const QVector3D &hit)
{
    QColor color = spheres[sphere_number].color; float
    point_light = 0.0f;

```

```

    for (int i = 0; i < lights.size(); ++i)
    {
        bool no_shadow = true;

        QVector3D dir_to_light_source = (lights[i].position -
hit).normalized();

        for(int j = 0; j < spheres.size(); ++j)
        {
            if(j != sphere_number)
            {
                QVector3D between_light_and_sphere;
                if(spheres[j].intersection(hit,
                    (lights[i].position -
hit).normalized(),

{ } } }

no_shadow = false;

between_light_and_sphere))

    if(no_shadow)
    {
        QVector3D sphere_normal = (hit -
spheres[sphere_number].center).normalized(); point_light
        += lights[i].power *
        qMax(0.0f, QVector3D::dotProduct(sphere_normal,
dir_to_light_source));
    }
}

// Set manually
float ambient_light = 0.3f;

// Manually set coefficients for ambient and source parts of lighting float coef_ambient
= 1.5f;
float coef_point = .5f;

float red = static_cast<float>(color.redF()) * point_light * coef_point
+ static_cast<float>(color.redF()) * ambient_light *
coef_ambient;
float green = static_cast<float>(color.greenF()) * point_light * coef_point
+ static_cast<float>(color.greenF()) * ambient_light * coef_ambient;
float blue = static_cast<float>(color.blueF()) * point_light * coef_point
+ static_cast<float>(color.blueF()) * ambient_light * coef_ambient;

// Trim values that exceed [0-1] range color.setRedF(static_cast<double>(qMin(qMax(0.0f,
red), 1.0f))); color.setGreenF(static_cast<double>(qMin(qMax(0.0f, green), 1.0f)));
color.setBlueF(static_cast<double>(qMin(qMax(0.0f, blue), 1.0f)));

return color;

```

```

}

bool Widget::raySceneIntersection(const QVector3D &rayOrigin,
                                const QVector3D &rayDirection, int
                                &sphere_number, QVector3D
                                &closest_intersection)
{
    bool no_intersection = true;

    sphere_number = -1;

    closest_intersection.setX(FLOAT_INF);
    closest_intersection.setY(FLOAT_INF);
    closest_intersection.setZ(FLOAT_INF);

    for(int i = 0; i < spheres.size(); ++i)
    {
        QVector3D intersection_point;
        spheres[i].intersection(rayOrigin, rayDirection, intersection_point);

        if ((closest_intersection - rayOrigin).length() > (intersection_point
- rayOrigin).length())
        {
            no_intersection = false; closest_intersection =
            intersection_point; sphere_number = i;
        }
    }
    if (no_intersection)

        {}
    else
        {} }
return false;
return true;

QColor Widget::trace(const QVector3D &origin, const QVector3D &direction)
{
    int sphere_number;
    QVector3D intersection_point;

    if(raySceneIntersection(origin, direction, sphere_number,
intersection_point))
    {

return light(spheres, lights, sphere_number, intersection_point);
return scene_color;

void image_to_scene(const QImage &image, const int &x, const int &y,
                    const float &aspect_ratio, const double &fov, float &fx,
                    float &fy)
{
    // Coordinates in range [0,1]

```



```

    fx = (x + 0.5f) / static_cast<float>(image.width()); fy = (y + 0.5f) /
    static_cast<float>(image.height());

    // Now in range [-1,1] and mathematic fx = 2 *
    fx - 1;
    fy = 1 - 2 * fy;

    // Add aspect ratio fx *=
    aspect_ratio;

    float fov_tan = static_cast<float>(tan(qDegreesToRadians(fov) / 2));

    // Add fov
    fx *= fov_tan; fy
    *= fov_tan;
}

void transformV3d(QVector3D &v3d, const QMatrix4x4 &m4x4)
{
    QVector4D v4d = QVector4D(v3d.x(), v3d.y(), v3d.z(), 1); v4d = v4d *

    m4x4;

    v3d = v4d.toVector3D();
}

void Widget::transform(QVector3D &origin, QVector3D &pixel_position)
{
    //float radX = qDegreesToRadians(degreesX);
    //float radY = qDegreesToRadians(degreesY); float radZ =
    qDegreesToRadians(degreesZ); //float cosX = cosf(radX);
    //float cosY = cosf(radY); float
    cosZ = cosf(radZ);
    //float sinX = sinf(radX);
    //float sinY = sinf(radY); float
    sinZ = sinf(radZ);

    cameraToWorldMatrix = QMatrix4x4(cosZ, 0, -sinZ, 0,
                                     -sinZ, cosZ, 0, 0,
                                     0, 0, 1, 0,
                                     0, 0, 0, 1);

    transformV3d(axisX, cameraToWorldMatrix);
    QVector3D axisY = QVector3D(0, 1, 0);
    transformV3d(axisY, cameraToWorldMatrix);
    transformV3d(axisZ, cameraToWorldMatrix);

    cameraToWorldMatrix.rotate(degreesX, axisX);
    cameraToWorldMatrix.rotate(degreesY, axisY);

    transformV3d(origin, cameraToWorldMatrix); transformV3d(pixel_position,
    cameraToWorldMatrix);

    origin += cameraPosition; pixel_position

```

```

    += cameraPosition;
}

void Widget::render(QImage &image)
{
    int width = imageSize.width(); int height
    = imageSize.height(); fov = 60; //
    degrees

    aspect_ratio = width / static_cast<float>(height);

    image = QImage(width, height, QImage::Format_RGB32); for (int
    i = 0; i < width; ++i)
        for (int j = 0; j < height; ++j)
        {
            float x, y;

            image_to_scene(image, i, j, aspect_ratio, fov, x, y); QVector3D origin(0, 0,
            0);
            QVector3D pixel_position(x, y, 1);

            transform(origin, pixel_position); QVector3D
            direction = (pixel_position -
                        origin).normalized();

            image.setPixelColor(i, j, trace(origin, direction));
        }
}

Widget::Widget(QWidget *parent, int width, int height)
    : QWidget(parent)
{
    cameraToWorldMatrix = QMatrix4x4(1, 0, 0, 0,
                                      0, 1, 0, 0,
                                      0, 0, 1, 0,
                                      0, 0, 0, 1);
    cameraPosition = QVector3D(0, 0, 0);

    degreesX = 0;
    degreesY = 0;
    degreesZ = 0;

    axisX = QVector3D(1, 0, 0);
    axisZ = QVector3D(0, 0, 1); imageSize =

    QSize(width, height);

    Sphere sphere1(QVector3D(0, 0, 200), 50, QColor(255, 128, 128));
    Sphere sphere2(QVector3D(60, 10, 140), 10, QColor(255, 192, 128));
    Sphere sphere3(QVector3D(0, 0, 300), 70, QColor(0, 192, 128));

    spheres.push_back(sphere1);

```

```

spheres.push_back(sphere2);
spheres.push_back(sphere3);

Light_source light1(QVector3D(150, 15, 100), 2.5f);
Light_source light2(QVector3D(0, 0, 500), 1.5f);
Light_source light3(QVector3D(-100, -50, -40), 1.5f);

lights.push_back(light1);
lights.push_back(light2);
lights.push_back(light3);

scene_color = QColor(192, 192, 192);

render(image);
resize(static_cast<int>(this->height() * aspect_ratio),
        this->height());
setFixedSize(this->size());
}

/*virtual*/ void Widget::paintEvent(QPaintEvent*)
{
    QPainter p(this);
    p.drawImage(0, 0, image.scaled(this->width(),this->height(),
Qt::KeepAspectRatio));
}

/*virtual*/ void Widget::keyPressEvent(QKeyEvent *event)
{
    switch(event->key())
    {
        case Qt::Key_Up: if(degreesX
< 90)
            degreesX += 10;
            render(this->image);
            this->update();
            break;

        case Qt::Key_Down:
            if(degreesX > -90)
                degreesX -= 10;
            render(this->image);
            this->update();
            break;

        case Qt::Key_Left: degreesY
+= 10;
            render(this->image);

            this->update(); break;

        case Qt::Key_Right:
            degreesY -= 10;
            render(this->image);

```

```

        this->update();
        break;

        case Qt::Key_W:
            cameraPosition = cameraPosition + 10 * QVector3D(axisZ.x(), 0,
axisZ.z()).normalized();
            render(this->image);
            this->update();
            break;

        case Qt::Key_S:
            cameraPosition = cameraPosition - 10 * QVector3D(axisZ.x(), 0,
axisZ.z()).normalized();
            render(this->image);
            this->update();
            break;

        case Qt::Key_D:
            cameraPosition = cameraPosition + QVector3D(10, 0, 0) *
cameraToWorldMatrix;
            render(this->image);
            this->update();
            break;

        case Qt::Key_A:
            cameraPosition = cameraPosition - QVector3D(10, 0, 0) * cameraToWorldMatrix;
            render(this->image);
            this->update();
            break;

        case Qt::Key_Space:
            cameraPosition = cameraPosition + QVector3D(0, 10, 0);
            render(this->image);
            this->update(); break;

        case Qt::Key_Shift:
            cameraPosition = cameraPosition - QVector3D(0, 10, 0);
            render(this->image);
            this->update(); break;

        default:
            QWidget::keyPressEvent(event);
    }
}

void Widget::addSphere(int imageX, int imageY)
{
    Sphere newSphere;
    newSphere.radius = 15;
    newSphere.color = QColor(255, 255, 255);

    float x, y;
    image_to_scene(this->image, imageX, imageY, this->aspect_ratio, this-

```

```

    QVector3D origin = QVector3D(0, 0, 0); QVector3D
    pixel_position = QVector3D(x, y, 1); transform(origin,
    pixel_position);

    QVector3D direction = (pixel_position - origin).normalized();

    float distance = 200; // if no spheres ahead in that direction int sphere_number;
    QVector3D intersection_point;
    if (raySceneIntersection(origin, direction, sphere_number,
    intersection_point))
    {
        distance = (spheres[sphere_number].center - origin).length() - (newSphere.radius
        + spheres[sphere_number].radius);
    }

    newSphere.center = origin + direction * distance;

    spheres.push_back(newSphere);
}
void Widget::deleteSphere(int imageX, int imageY)
{
    float x, y;
    image_to_scene(this->image, imageX, imageY, this->aspect_ratio, this-
    >fov, x, y);

    QVector3D origin = QVector3D(0, 0, 0); QVector3D
    pixel_position = QVector3D(x, y, 1); transform(origin,
    pixel_position);

    QVector3D direction = (pixel_position - origin).normalized(); int

    sphere_number;
    QVector3D intersection_point;
    if (raySceneIntersection(origin, direction, sphere_number,
    intersection_point))
    {
        spheres.remove(sphere_number);
    }
}
/*virtual*/ void Widget::mouseReleaseEvent(QMouseEvent *event)
{
    int widgetX = event->x(); int
    widgetY = event->y();

    int imageX = static_cast<int>(static_cast<float>(widgetX) /
    this->width() * imageSize.width()); int
    imageY = static_cast<int>(static_cast<float>(widgetY) /
    this->height() * imageSize.height());

    switch(event->button())
    {
        case Qt::LeftButton:

```

```

        addSphere(imageX, imageY);
        this->render(this->image); this->update();
        break;
    case Qt::RightButton:

        deleteSphere(imageX, imageY);
        this->render(this->image); this->update();
        break;
    default:
        QWidget::mouseReleaseEvent(event);
    }
}

void Widget::resizeSphere(int imageX, int imageY, const QPoint &angle)
{
    float x, y;
    image_to_scene(this->image, imageX, imageY, this->aspect_ratio, this->fov, x, y);

    QVector3D origin = QVector3D(0, 0, 0); QVector3D
    pixel_position = QVector3D(x, y, 1); transform(origin,
    pixel_position);

    QVector3D direction = (pixel_position - origin).normalized(); int
    sphere_number;
    QVector3D intersection_point;
    if (raySceneIntersection(origin, direction, sphere_number,
    intersection_point))
    {
        spheres[sphere_number].radius += angle.y() / 12;
        if(spheres[sphere_number].radius <= 0)
            spheres.remove(sphere_number);
    }
}

/*virtual*/ void Widget::wheelEvent(QWheelEvent *event)
{
    QPoint angle = event->angleDelta(); if(!angle.isNull())
    {
        int widgetX = event->x(); int
        widgetY = event->y();

        int imageX = static_cast<int>(static_cast<float>(widgetX) /
        this->width() * imageSize.width()); int
        imageY = static_cast<int>(static_cast<float>(widgetY) /
        this->height() * imageSize.height());

        resizeSphere(imageX, imageY, angle);
    }
    QWidget::wheelEvent(event);
}

```

```
Widget::~Widget()
{
}
void Widget::setImageSize(int width, int height)
{
    imageSize.setWidth(width); imageSize.setHeight(height);
    fov, x, y);
```



## **4 Документирование**

### **4.1 Техническое задание**

#### **4.1.1 Введение**

Программа – графический движок на основе обратной трассировки лучей реализовывает систему освещения, приближенную к реалистичной.

#### **4.1.2 Назначение разработки**

Программа позволяет создавать и отрисовывать сцены с приближенным к реалистичному освещением, перемещать по ним камеру.

#### **4.1.3 Требования к функциональным характеристикам**

В движке должны быть реализованы: отражение света, прозрачность, затенение, светимость, направленное и фоновое освещение объектов. Следующие параметры должны иметь возможность настройки: глубина отражений, плотность лучей, поле зрения, цвет пространства. Программа предполагает одно действующее лицо – пользователя. Ему доступны следующие действия: создание и изменение объектов сцены, настройка параметров отображения, запуск отрисовки кадра.

#### **4.1.4 Требования к входным и выходным данным**

Входными данными в программе являются игровые настройки пользователя (разрешение экрана, качество графики и т.д.), а также непосредственное управление во время игрового процесса с помощью клавиатуры.

Выходными данными являются графическая интерпретация сцены на

мониторе пользователя и звук, сопровождающий его.

#### 4.1.5 Требования к надежности

В случае ошибки или неправильной работы программы пользователь должен иметь возможность узнать об этом. Также должна быть информация о том, что вызвало ошибку, и на каком этапе выполнения программы она произошла.

#### 4.1.6 Требования к составу и параметрам технических средств

Минимальные системные требования: для работы программы требуется компьютер с операционной системой.

Оптимальные системные требования: Процессор Intel ® Core (TM) i5-2400 3.30 Ghz.

#### 4.1.7 Требование к информационной и программной совместимости

Программа должна функционировать на ОС семейства Windows 7 и выше.

#### 4.1.8 Требование к программной документации

Программная документация должна включать в себя руководство пользователя.

### 4.2 Руководство пользователя

Управление камерой – клавиши Стрелка вверх, Стрелка вниз, Стрелка вправо, Стрелка влево

Перемещение – клавиши W, A, S, D, Shift, Пробел Создать сферу – левая кнопка мыши

Удалить сферу – правая кнопка мыши Изменить радиус сферы – колесико мыши.

## **5 Экономическая часть**

### **5.1 Основные положения экономической части**

На сегодняшний день быстрые темпы развития информационных технологий внесли много новшеств в экономическую сферу деятельности предприятий. Информационные технологии в экономике – средство виртуальной экономики.

В настоящее время обмен наибольшим количеством информации наблюдается в сферах торговли, промышленности, финансово-банковской сфере, маркетинге, сфере оказания услуг. Информация является одним из основных факторов, определяющих развитие технологий. Современные информационные технологии являются средством снижения трудоёмкости работы пользователя, то есть средством автоматизации процесса управления информацией.

Любой программный продукт, разрабатываемый для предприятия, затем внедряется исключительно для того, чтобы ускорить или упростить выполнение каких-либо задач. Программные продукты можно условно разделить на два вида:

1. Программа, сокращающая работу над рутинными или регулярными задачами. Такой вид программ ускоряет выполнение несложных операций, занимающих достаточно много рабочего времени у персонала.

2. Программа, упрощающая работу со сложной системой или механизмом, для работы с которым напрямую, ввиду специфики работы, требуется узкоспециализированный сотрудник.

Программное обеспечение, которое выполняет сразу обе вышеперечисленных задачи, можно считать полноценным рабочим инструментом.

Подобное программное обеспечение ценится на рынке больше всего так позволяет унифицировано решать как комплексные, так и повседневные простые задачи.

Разрабатываемая программа предназначена для просмотра интерактивной 3D-сцены.

Показатели экономической эффективности от внедрения программы определяются всеми полученными позитивными экономическими результатами и как результат: рост прибыли или уменьшение расходов предприятия.

Стоимость программного продукта может рассматриваться с двух точек зрения:

- с точки зрения формирования объектов нематериальных активов как исключительного права;
- с точки зрения формирования затрат на производство, например, общехозяйственные расходы.

В данном разделе будут рассмотрены расчёты трудоёмкости, расчёты экономической эффективности и сценарии ценообразования.

Целью экономической части ВКР является расчет эффекта от внедрения программного продукта. В результате расчета находится себестоимость программного продукта, а также эффективности его использования.

Стоимость программного обеспечения рассматривается с двух сторон:

- формирование нематериальных активов для использования на предприятии;
- то же самое для коммерческого использования и продвижения программы на рынке.

В данном разделе будут проведены расчеты трудоемкости, калькуляция сметы затрат, оценка конкурентоспособности и сценарии ценообразования создания программного продукта.

## **5.2 Трёхуровневый анализ продукта**

Основным, с точки зрения маркетинга, для продукта являются его потребительские свойства, то есть способность удовлетворить потребности

владельца им. Из этого следует то, что рассматривать продукт необходимо с точки зрения конечного пользователя.

Проект существует для удовлетворения потребностей потребителя в развлечении и отдыхе.

Качественный проект должен предоставлять пользователю удобное управление и понятный интерфейс, обширное количество задач, который пользователь сможет решить при в процессе пользования программой.

Одной из составляющих трёхуровневого анализа является сущность продукта. Она определяется тем, какие потребности удовлетворяет продукт.

Таблица 1 - Трёхуровневый анализ разработанного продукта

Уровень	Описание
Сущность товара	Видеоигра, способной удовлетворить потребности пользователя в развлечении и отдыхе.
Фактический товар	Интерактивная 3D-сцена с возможностью перемещения по ней и изменения содержащихся в ней объектов.
Добавленный товар	Техническая поддержка пользователей, а также длительная поддержка продукта и исправление ошибок.

Фактический продукт является второй составляющей трёхуровневого анализа. Фактический продукт представляет собой совокупность свойств продукта, определяющих его форму.

Третьей составляющей является дополнительный продукт.

Он включает в себя все то, что производитель может предложить помимо основного товара.

Дополнительные предложения позволяют увеличить ценность продукта для покупателя и выгодно выделить его среди продуктов конкурентов.

Основываясь на этом можно провести трёхуровневый анализ программного продукта.

### 5.3 Определение трудоёмкости разработки программы

Трудоемкость в любой предметной области определяется на общепринятых нормах в заданной сфере деятельности. В области создания программных решений трудоемкость определяется нормами принятыми в области компьютерных информационных технологий. Такими нормами являются нормы времени, которые служат для определения нормы труда разработчиков программного обеспечения, обоснования трудности разработки программного обеспечения, а также численности разработчиков. Эти нормы включают в себя задачи статистики, комплексы задач управления, а также задачи, которые связаны с расчетами трудоемкости.

В ходе расчёта трудоёмкости выделяется пять стадий выполнения работ. Техническое задание. Это стадия, на которой заказчиком формируются основные требования к автоматизированной системе.

После формирования требований производится обоснование возможности решения задачи, производится разработка ведущей концепции и согласование сроков разработки:

1. Эскизный проект. На данной стадии происходит дальнейшая проработка технического задания, по итогам которой разрабатывается математическая модель и создается алгоритм разработки автоматизированной системы.

2. Технический проект. На этой стадии происходит разработка программной документации и выбор наиболее эффективных средств реализации.

3. Рабочий проект. Наиболее трудоёмкая стадия, во время которой выполняется программная реализация, её тестирование и отладка. Результатом по завершении этой стадии служит готовая автоматизированная система, сопровождаемая рабочей документацией и руководством пользователя.

4. Стадия внедрения включает себя проверку правильности работы



автоматизированной системы на практике в процессе подготовки требуемой документации, а после автоматизированная система может эксплуатироваться пользователями.

Для расчёта трудозатрат программиста при разработке данного программного продукта следует определить степень новизны.

Таблица 2 – Затраты времени при выполнении работ на стадии

«Техническое задание»

Подсистемы	Степень новизны			
	А	Б	В	Г
Перспективное планирование развития и размещения отрасли, управление проектированием и капитальным строительством.	26	57	29	35
Управление материально техническим снабжением, управление сбытом продукции.	65	80	54	29
Бухгалтерский учет, управление финансовой деятельностью	28	72	52	35
Управление организацией труда и заработной платой, управление кадрами.	63	46	35	16

У разрабатываемых комплексов задач существует четыре степени новизны:

– степень А – это разработка комплекса задач, включающих применение совершенно новых методов разработки, а также осуществление научно-исследовательских работ;

– степень Б – это разработка принципиально новых проектных решений, задач, методов и систем, не имеющих аналогов;

– степень В это разработка проекта с использованием уже существующих аналогичных проектных решений с условием их изменения;

степень Г – привязка готовых проектных решений к собственному проекту.

Разрабатываемая программа соответствует степени новизны В.

На стадии «Техническое задание» расход времени на выполнение задачи

обозначается за T1. В свою очередь T1 имеет значение равное 16 (чел./дн.), которое определяется в соответствии с таблицей 2 [3].

На стадии «Эскизный проект» расход времени на выполнение задачи обозначается за T2. В свою очередь T2 имеет значение равное 24 (чел./дн.), которое определяется в соответствии с таблицей 3 [3].

Таблица 3 – Затраты времени при выполнении работ на стадии «Эскизный проект»

Подсистемы	Степень новизны			
	А	Б	В	Г
Перспективное планирование развития и размещения отрасли, управление проектированием и капитальным строительством.	181	167	87	35
Управление материально – техническим снабжением, управление сбытом продукции.	115	79	53	55
Бухгалтерский учет, управление финансовой деятельностью	121	78	74	51
Управление организацией труда и заработной платой, управление кадрами.	144	112	49	57

Для остальных этапов («Технический проект», «Рабочий проект» и «Внедрение») предварительные нормы затрат времени определяются по формуле (1):

$$T_{3,4,5} = r_1 \Phi_1^{r_2} \Phi_2^{r_3} \text{ (чел./дн. )}, \quad (1)$$

где  $r_1$ ,  $r_2$ ,  $r_3$  – коэффициенты, приведенные в таблицах 4 – 6 [2];

$\Phi_1$  – количество макетов входной информации;

$\Phi_2$  – количество разновидностей форм выходной информации.

Коэффициенты  $r_1$ ,  $r_2$ ,  $r_3$  принимают следующие значения в соответствии с таблицей 7.

Так как разработчик комплекса участвовал в подготовке информационного обеспечения, на стадиях «Технический проект» и «Рабочий проект» коэффициенты  $r_1$ ,  $r_2$ ,  $r_3$  были увеличены в 1,1 раз (повышающий

коэффициент).

Таблица 4 – Коэффициенты для определения нормы времени при выполнении работ на стадии «Технический проект»

Подсистемы	Разработчик	Коэффициенты		
		А	Б	В
Перспективное планирование развития и размещения отрасли, управление проектированием и капитальным строительством.	ПЗ	31,04	0,44	0,34
	ПО	9,34	4,56	0,17
Управление материально – техническим снабжением, управление сбытом продукции.	ПЗ	21,99	0,46	0,35
	ПО	8,33	0,48	0,16
Бухгалтерский учет, управление финансовой деятельностью	ПЗ	18,01	0,56	0,4
	ПО	10,19	0,59	0,19
Управление организацией труда и заработной платой, управление кадрами.	ПЗ	15,9	0,45	0,34
	ПО	10,74	0,53	0,7

Таблица 5 – Коэффициенты для определения нормы времени при выполнении работ на стадии «Рабочий проект»

Подсистемы	Разработчик	Коэффициенты		
		А	Б	В
Перспективное планирование развития и размещения отрасли, управление проектированием и капитальным строительством.	ПЗ	8,11	0,47	0,49
	ПО	56,05	0,44	0,42
Управление материально– техническим снабжением, управление сбытом продукции.	ПЗ	19,32	0,46	0,48
	ПО	31,81	0,45	0,43
Бухгалтерский учет, управление финансовой Деятельностью	ПЗ	10,1	0,54	0,52
	ПО	32,99	0,55	0,49
Управление организацией труда и заработной платой, управление кадрами.	ПЗ	1,1	0,47	0,51
	ПО	49,78	0,42	0,41

Для разрабатываемого проекта  $\Phi_1 = 3$ , а  $\Phi_2 = 1$ .

Ниже представлен расчет норм временных затрат  $T_3$ ,  $T_4$ ,  $T_5$ .

$$T_3 = 10,26 \cdot 30,52 \cdot 10,17 = 18,17 \text{ (чел./дн.)};$$

$$T_4 = 37,19 \cdot 30,49 \cdot 10,47 = 63,71 \text{ (чел./дн.)};$$

$$T_5 = 8 \cdot 30,46 \cdot 10,51 = 13,26 \text{ (чел./дн.)}.$$

Тогда общие трудозатраты будут равны:

$$T_{\Sigma} = T_1 + T_2 + T_3 + T_4 + T_5 = 39 + 74 + 18,17 + 63,71 + 13,26 = 208,14 \text{ (чел./дн.)}.$$

Таблица 6 – Коэффициенты для определения нормы времени при выполнении работ на стадии «Внедрение»

Подсистемы	Разработчик	Коэффициенты		
		А	Б	В
Перспективное планирование развития и размещения отрасли, управление проектированием и капитальным строительством.	ПЗ	9,10	0,44	0,44
	ПО	10,89	0,38	0,48
Управлениематериально–техническим снабжением, управление сбытом продукции.	ПЗ	8,74	0,49	0,45
	ПО	8	0,46	0,51
Бухгалтерский учет, управление финансовой деятельностью	ПЗ	9,16	0,43	0,43
	ПО	7,12	0,43	0,43
Управление организацией труда изарботной платой, управление кадрами.	ПЗ	9,1	0,44	0,44
	ПО	10,91	0,38	0,48

Таблица 7 – Значения коэффициентов  $r_1$ ,  $r_2$ ,  $r_3$

Стадия разработки	$r_1$	$r_2$	$r_3$
Технический проект (Т3)	13,24	0,54	0,18
Рабочий проект (Т4)	32,11	0,46	0,48
Внедрение (Т5)	7,9	0,47	0,49

Численность исполнителей, необходимая для выполнения работ по созданию программы вычисляется по формуле (2):

$$Ч = \frac{T_{вр}}{\Phi_{пл}}, \quad (2)$$

где  $\Phi_{пл}$  – фонд рабочего времени, планируемый на одного разработчика;

$T_{вр}$  – временные трудозатраты на реализацию программы, (чел./дн.).

Платформа разрабатывается в течении девяти месяцев с сентября по

май, что составляет 182 рабочих дня, соответственно рекомендуемая численность разработчиков проекта составляет:

$$Ч = 208,14 / 182 = 1,14 \approx 1 \text{ (чел.)}.$$

#### 5.4 Определение стоимости разработки продукта

Себестоимость – это стоимостная оценка используемых в процессе производства материальных, трудовых и других затрат, а также затрат на реализацию продукции. Стоимость программного продукта определена на основе укрупненного метода учёта затрат по материальным затратам, расходам на оплату труда, отчислений на социальное страхование, амортизации основных фондов и прочих расходов. Стоимость рассчитывается по формуле (3):

$$C_{по} = МЗ + ФЗП + A_0 + П, \quad (3)$$

где  $МЗ$  – материальные затраты;  
 $ФЗП$  – фонд заработной платы разработчика программного продукта;  
 $A_0$  – амортизационные отчисления;  
 $П$  – прочие расходы.

Количество рабочих часов, затраченных на разработку данной программы, при учёте того, что проект разрабатывался в течение 182 рабочих дней, а длительность рабочего дня составляет восемь часов, получим общее количество рабочих часов:

$$КЧ = T_{\Sigma ок} \cdot 8 = 182 \cdot 8 = 1456 \text{ (ч.)}.$$

Материальные затраты включают в себя расходы на электроэнергию.

Определение затрат на электроэнергию ведётся из расчёта того, что потребление энергии компьютером составляет 0,5 кВт/ч. Стоимость 1 кВт/ч. равна 4,2 (р.), тогда затраты на электроэнергию составят:

$$1456 \cdot 0,5 \cdot 4,2 = 3057,6 \text{ (р.)}.$$

Заработная плата определяется на основе трудового договора между

работником и работодателем. Должностной оклад заместителя директора по хозяйственной части согласно штатному расписанию ООО «ВЕКТОР» составляет 13000 (р.).

По закону Забайкальского края от 26.09.2008 № 39-33К «О районном коэффициенте и процентной надбавке к заработной плате работников бюджетных организаций» установлен районный коэффициент в размере 20 %, а также предусмотрена надбавка 30 % за стаж работы в районах Крайнего Севера и приравненных к ним местностях, а также в остальных районах Севера.

Таким образом, ежемесячная заработная плата составляет:

$$13000 \cdot 1,5 = 19500 \text{ (р.)}.$$

Учитывая, что разработка программно-аппаратного комплекса ведётся в течение девяти месяцев, фонд заработной платы будет равен:

$$\text{ФЗП} = 19500 \cdot 9 = 175500 \text{ (р.)}.$$

Порядок уплаты страховых взносов во внебюджетные фонды определяется статьёй 425 НК РФ от 03.07.2016 №243-ФЗ «О страховых взносах в Пенсионный фонд Российской Федерации, Фонд социального страхования Российской Федерации, Федеральный фонд обязательного медицинского страхования».

Начиная с 1 января 2019 г., взносы на социальное страхование составляют 30 % от общей заработной платы. Законом установлены тарифы страховых взносов, описанных ниже:

Пенсионный фонд Российской Федерации – 22,00 %;

Фонд социального страхования Российской Федерации – 2,9 %;

Федеральный фонд обязательного медицинского страхования – 5,1 %.

Таким образом, отчисления составили:

$$175500 \cdot 0,3 = 52650 \text{ (р.)}.$$

Основное средство, подлежащее амортизации при разработке программной системы – это компьютер. Первоначальная стоимость одного компьютера 28000 (р.).

Вычислительная техника входит в амортизационную группу 2.

По установленным нормам, постановление Правительства Российской Федерации от 18.11.2006 г. № 697, время службы компьютера берётся от двух до трёх лет включительно.

Установим норму службы компьютера равную трём годам (3 года = 36 мес.).

В общем порядке  $N_a$  – норма амортизации определяется так:

$$N_a = 1 / 36 \cdot 100 \% = 2,78 \%$$

Следовательно, амортизация за девять месяцев составит:

$$A(9 \text{ мес.}) = 28000 \cdot 2,78 \% / 100 \% \cdot 9 = 7005,6 (\text{p.})$$

Величина прочих расходов составляет 15 % от основной заработной платы. В прочие расходы входит амортизация нематериальных активов, а также накладные расходы:

$$П = 19500 \cdot 0,15 \cdot 9 = 26325 (\text{p.}).$$

Таким образом, общая стоимость затрат на создание программы составляет:

$$\text{СПО} = 3057,6 + 175500 + 52650 + 7005 + 26325 = 264548,2 (\text{p.}).$$

Программный продукт является объектом нематериальных активов, и его первоначальная стоимость оценивается по совокупным затратам на создание, то есть равна 264548,2 (p.)

Рассмотрим три возможных сценария ценообразования в таблице 8.

Основываясь на данных, представленных в таблице 2, цена за одну копию программы в пессимистическом сценарии слишком высока. Оптимистический же и рационалистический сценарии предполагают вполне подходящую стоимость для целевой аудитории. Примем оптимистический сценарий. В таком случае, окончательная цена будет равна 1653,42 (p.).



Таблица 8 – Сценарии ценообразования

Сценарий	Описание	Цена за одну
Пессимистический	Спрос – низкий. Ожидаемое количество реализованных штук равно 1. Рентабельность равна 0 %	264548,2 (р.)
Оптимистический	Спрос высокий. Количество реализованных штук равно 200. Рентабельность равна 25 %	$264548,2 \cdot 1,25 / 200 = 1653,42$ (р.)
Рационалистический	Спрос–средний. Количество реализованных штук равно 80. Рентабельность равна 5 %	$264548,2 \cdot 1,05 / 80 = 3472,2$ (р.)

## 5.5 Оценка экономической эффективности

Экономическая эффективность – это результат, который можно получить, соизмерив показатели доходности производства по отношению к общим затратам и использованным ресурсам. Если первый показатель выше по сравнению со второй составляющей, значит, целей достигнуто, все потребности удовлетворены. Если ситуация наоборот, значит, экономического эффекта не наблюдается и предприятие несет убытки.

Т.к. в данной работе проект создавался непосредственно для продажи на рынке, то основным показателем экономической эффективности будет являться доход, полученный с продажи копий проекта, по отношению к затратам на его создание.

Затраты на создание продукта составляют 264548,2 (р.) в свою очередь, при оптимистично сценарии ценообразования выручка с продажи продукта составит  $1653,42 \cdot 200 = 330684$  (р.)

Отсюда экономическая эффективность:

$$E = 330684 / 264548,2 = 1,25$$

Следует учитывать, что невысокая эффективность обусловлена лишь

большими начальными затратами на компьютерное оборудование, которое в дальнейшем будет использоваться для множества других проектов.

Благодаря произведённым расчётам ясно видно, что данный проект является рентабельным и экономически обоснованным, а отпускная цена продукта, является приемлемой для потребителя, обеспечит конкурентоспособность продукции.

## **6 Безопасность и экологичность проекта**

### **6.1 Общие положения охраны труда**

Гигиенические требования к персональным электронно-вычислительным машинам (ПЭВМ) и организации работы с ними контролируются санитарно-эпидемиологическими правилами и нормами СанПиН 2.2.2/2.4.1340-03 (далее – санитарные правила) [1]. Настоящие санитарные правила разработаны в соответствии с Федеральным законом «О санитарно-эпидемиологическом благополучии населения» от 30.03.1999 г. № 52-ФЗ и Положении о государственном санитарно-эпидемиологическом нормировании, утвержденным постановлением Правительства Российской Федерации от 24.07.2000 г. №554.

Санитарные правила действуют на всей территории Российской Федерации и устанавливают требования к ПЭВМ и условиям труда. Требования санитарных правил направлены на предотвращение неблагоприятного влияния вредных факторов производства и трудового процесса при работе с ПЭВМ на здоровье человека.

Требования санитарных правил распространяются на: организацию и условия работы с ПЭВМ, персональные и портативные ЭВМ, устройства отображения информации, периферийные устройства и игровые комплексы на базе ПЭВМ.

### **6.2 Требования к ПЭВМ**

Эксплуатация персонально электронно-вычислительных машин (в дальнейшем ПЭВМ) обязана происходить в соответствии с текущими санитарными требованиями. Каждый их тип подлежит обязательной санитарно-эпидемиологической экспертизе. Экспертиза проводится в специализированных испытательных лабораториях, которые являются

аккредитованными в установленном порядке.

Список контролируемых гигиенических параметров опасных и вредных факторов представлен в таблице 9..

Уровни допустимых значений звукового давления, создаваемого ПЭВМ представлены в таблице 10. Измерение данных уровней звукового давления производится на расстоянии 50 см. от поверхности оборудования.

Таблица 9 – Контролируемые гигиенические параметры

№	Вид продукции	Код ОКП	Контролируемые гигиенические параметры
1	Персональные и портативные ЭВМ	40 1300, 40 1350, 40 1370	Уровни электромагнитных полей (ЭМП), акустического шума, концентрации вредных веществ в воздухе, визуальные показатели ВДТ, мягкое рентгеновское излучение
2	Периферийные устройства	40 3000	Уровни ЭМП, акустического шума, концентрация вредных веществ в воздухе
3	Устройства отображения информации	40 3200	Уровни ЭМП, визуальные показатели, концентрация вредных веществ в воздухе, мягкое рентгеновское излучение
4	Игровые автоматы на базе ПЭВМ	96 8575	Уровни ЭМП, акустического шума, концентрация вредных веществ в воздухе, визуальные показатели ВДТ, мягкое рентгеновское излучение

Таблица 10 – Уровни допустимых значений звукового давления

Уровни звукового давления в октановых полосах со среднегеометрическими частотами									Уровни звука в дБА
31,5 Гц	63 Гц	125 Гц	250 Гц	500 Гц	1000 Гц	2000 Гц	4000 Гц	8000 Гц	
86 дБ	71 дБ	61 дБ	54 дБ	49 дБ	45 дБ	42 дБ	40 дБ	38 дБ	50

Временные допустимые уровни (ВДУ) электромагнитных полей (ЭМП), создаваемые ПЭВМ, не должны превышать примерных значений, указанных в таблице 11.

Таблица 11 – Допустимые уровни ЭМП

Наименование параметров		ВДУ
Напряженность электрического поля	в диапазоне частот с 5 Гц до 2 кГц	25 В/м
	в диапазоне частот с 2 кГц до 400 кГц	2,5 В/м
Плотность магнитного потока	в диапазоне частот с 5 Гц до 2 кГц	250 нТл
	в диапазоне частот с 2 кГц до 400 кГц	25 нТл
Электростатический потенциал экрана видеомонитора		500 В

Допустимые параметры устройств отображения информации указаны в таблице 12. Для дисплеев на электронно-лучевых трубках частота обновления экрана не должна быть ниже 75 Гц при любом разрешении. Для остальных типов дисплеев частота обновления экрана должна быть не менее 60 Гц.

Таблица 12 – Допустимые параметры устройств отображения информации

№	Параметр	Допустимые значения
1	Яркость белого поля	Не менее 35 кд/м <sup>2</sup>
2	Неравномерность яркости рабочего поля	Не более $\pm 20 \%$
3	Контрастность (для монохромного режима)	Не менее 3:1
4	Временная нестабильность изображения	Не должна фиксироваться
5	Пространственная нестабильность изображения	Не более $2 \cdot 10^{-4} \text{ л}$ , где L – проектное расстояние наблюдения, мм

Концентрация вредных веществ в воздушном пространстве, производимых во время эксплуатации ПЭВМ, не должна превышать предельно допустимой концентрации (далее ПДК) рассчитанных для атмосферного воздуха. Мощность мягкого рентгеновского излучения в любой точке на расстоянии 0,05 м от экрана ПЭВМ и его корпуса не должна превышать 1 мкЗв/ч.

Корпус ПЭВМ должен быть окрашен в мягких и спокойных тонах с диффузным рассеиванием света. ПЭВМ и периферийные устройства

должны иметь матовую поверхность с отсутствием деталей способных создавать блики. При этом коэффициент отражения поверхности должен варьироваться от 0,4 до 0,6 условных единиц.

### **6.3 Требования к помещениям для эксплуатации ПЭВМ**

Эксплуатация ПЭВМ разрешена в помещениях с естественным и искусственным освещением. В случае отсутствия естественного освещения, использование ПЭВМ возможна только при соответствующем обосновании и наличии положительного заключения санитарно-эпидемиологической экспертизы, которое выдается в установленном порядке.

Площадь одного рабочего места пользователя ПЭВМ, в состав которого входит жидкокристаллический или плазменный дисплей, должна быть не менее 4,5 м<sup>2</sup>.

Для внутренней отделки интерьера помещения применяются диффузно отражающие материалы с коэффициентом отражения для потолка от 0,7 до 0,8; для стен от 0,5 до 0,6; для пола от 0,3 до 0,5. Полимерные материалы используются при наличии санитарно-эпидемиологического заключения.

Уровни естественного и искусственного освещения должны соответствовать действующим требованиям нормативной документации. Эксплуатация ПЭВМ наиболее рекомендована в помещениях с расположением окон на северной или северо-восточной стороне. Оконные проемы при этом должны быть оборудованы регулируемыми жалюзи или занавесами, которые позволяют полностью закрыть оконные проемы.

Помещения с размещенными в них рабочими местами ПЭВМ, должны быть оборудованы защитным заземлением, которое удовлетворяет техническим требованиям по эксплуатации. Нежелательна схема размещения рабочих мест, если вблизи присутствуют силовые кабели, высоковольтные трансформаторы или технологическое оборудование, которые способны

создать помехи в работе ПЭВМ.

#### **6.4 Требования к шуму и вибрации в помещениях с эксплуатируемым ЭВМ**

Предельно допустимый уровень шума на рабочих местах должен быть ниже установленного уровня в соответствии с действующими санитарно-эпидемиологическими нормативами.

В образовательных и культурно-развлекательных учреждениях для детей и подростков, а также в производственных помещениях, где расположены и эксплуатируются ПЭВМ, уровень шума и вибрации должен быть ниже предельно допустимых значений, установленных в соответствии с существующими санитарно-эпидемиологическими нормативами.

Оборудование, имеющее высокий уровень шума (например, печатающее устройство), которое превышает установленные нормативы, должно размещаться вне помещения с рабочим местом ПЭВМ.

#### **6.5 Требования к освещению помещений и рабочих мест с ПЭВМ**

На рабочих местах столы следует располагать таким образом, чтобы дисплеи ПЭВМ были ориентированы боковой стороной к световым проёмам, чтобы естественный свет преимущественно падал с левой стороны от рабочего места.

Искусственное освещение в помещениях эксплуатации ПЭВМ должно осуществляться системой общего равномерного освещения. В производственных и административно-общественных помещениях, при преимущественной работе с документами, следует применять системы комбинированного освещения. В этом устанавливаются дополнительно светильники местного освещения для зоны работы с документами.

Искусственное освещение должно обеспечивать требуемую освещённость на рабочих местах не ниже нормируемых значений: экран – не более 300 лк; клавиатура и рабочий стол от 300 до 500 лк.

Яркость светильников общего освещения в зоне углов излучения от 50 до 90 градусов с вертикалью в продольной и поперечной плоскостях должна составлять не более 200 кд/м<sup>2</sup>, защитный угол светильников должен быть не менее 40 градусов.

Неравномерное распределение яркости в поле зрения пользователя ПЭВМ следует свести к минимуму. В областях между рабочими поверхностями соотношение яркости должно быть меньше, чем три к одному, а между рабочими поверхностями, стенами и оборудованием ниже, чем десять к одному.

Для искусственного освещения следует применять преимущественно люминесцентные лампы белого света и компактные люминесцентные лампы. При устройстве отражённого освещения в производственных и административно-общественных помещениях допускается применение металлогалогенных ламп. В светильниках местного освещения допускается применение ламп накаливания, в том числе галогенные.

Для освещения помещений с ПЭВМ следует применять светильники с зеркальными параболическими решетками, укомплектованными электронными пускорегулирующими аппаратами (ЭПРА). Допускается использование многоламповых светильников с ЭПРА, состоящих из равного числа опережающих и отстающих ветвей. Применение светильников без рассеивателей и экранирующих решеток не допускается. При отсутствии светильников с ЭПРА лампы многоламповых светильников или рядом расположенные светильники общего освещения следует включать на разные фазы трёхфазной сети.

Общее освещение при использовании люминесцентных светильников следует выполнять в виде сплошных или прерывистых линий светильников,



расположенных сбоку от рабочих мест, параллельно линии зрения пользователя при рядом расположении дисплеев.

При периметральном расположении компьютеров линии светильников должны располагаться локализовано над рабочим столом ближе к его переднему краю, обращённому к оператору.

Коэффициент запаса для осветительных установок общего освещения должен приниматься равным 1,4 условных единиц.

Коэффициент пульсации не должен превышать 5 %.

Для обеспечения нормируемых значений освещённости в помещениях для использования ПЭВМ следует проводить чистку стёкол, оконных рам и светильников не реже двух раз в год и проводить своевременную замену перегоревших ламп

## **6.6 Требования к организации и оборудованию рабочих мест**

При размещении рабочих мест с ПЭВМ расстояние между рабочими столами с мониторами, должно быть не менее 2,0 м, а расстояние между боковыми поверхностями мониторов – не менее 1,2 м.

Рабочие места с ПЭВМ в помещениях с источниками вредных производственных факторов должны размещаться в изолированных кабинах с организованным воздухообменом.

Экран монитора должен находиться от глаз пользователя на расстоянии от 600 до 700 мм.

Конструкция рабочего стола должна обеспечивать оптимальное размещение на рабочей поверхности используемого оборудования, поддержание рациональной рабочей позы при эксплуатации ПЭВМ, а также позволять изменять позу с целью снижения статического напряжения мышц шейно-плечевой области и спины для предупреждения развития утомления. Поверхность рабочего стола должна иметь коэффициент отражения от 0,5

до 0,7. Тип рабочего кресла следует выбирать с учетом роста пользователя, характера и продолжительности работы с ПЭВМ. Рабочее кресло должно быть оборудовано подъёмно-поворотным механизмом, позволяющее производить регулировку по высоте и углам наклона сиденья и спинки, а также расстояния спинки от переднего края сиденья. При этом настройка каждого параметра должна быть независимой, легко осуществляемой и иметь надежную фиксацию.

Поверхность сиденья, спинки и других элементов кресла должна быть полумягкой, с нескользящим, слабо электризующимся и воздухопроницаемым покрытием, обеспечивающим легкую очистку от загрязнений

Поверхность элементов рабочего стула (сиденье, спинка, подлокотники) должна быть полумягкой, с нескользящим, слабо электризующимся и воздухопроницаемым покрытием, обеспечивающим легкую чистку от различных загрязнений. Так же конструкция стула должна обеспечивать: ширину и глубину сиденья не менее 400 мм, поверхность сиденья с закругленным передним краем, стационарные или съемные подлокотники. И обладать возможностью регулировки и надежной фиксации: высоты поверхности сиденья в пределах от 400 до 550 мм и углов наклона вперед до 15 градусов и назад до 5 градусов, высоту опорной поверхности спинки от 280 до 320 мм, ширину - не менее 380 мм и радиус кривизны горизонтальной плоскости – 400 мм, расстояния спинки от переднего края сиденья в пределах от 260 до 400 мм, подлокотников по высоте над сиденьем в пределах от 200 до 260 мм и внутреннего расстояния между подлокотниками в пределах от 350 до 500 мм. Высота рабочего стола для взрослых пользователей должна регулироваться в пределах от 680 до 800 мм. При отсутствии возможности регулировки высоты рабочей поверхности стола, высота должна составлять 725 мм.

Также рабочее место (рабочий стол), должен быть оснащен пространством для ног, высотой не менее 600 мм, шириной – не менее 500 мм, глубиной на уровне колен – не менее 450 мм и на уровне вытянутых ног не менее 650 мм.

Обязательно должна проводиться влажная ежедневная уборка помещений и систематическое проветривание после каждого часа работы с ПЭВМ.

### **6.7 Требования к организации медицинского обслуживания**

Для лиц, профессионально связанных с эксплуатацией ПЭВМ и, проводящих более 50% своего рабочего времени за работой с ПЭВМ, должны пройти обязательное предварительное при поступлении на работу и периодические медицинские осмотры в установленном порядке. К непосредственной работе с ПЭВМ допускаются лица, не имеющие медицинских противопоказаний.

Женщины со времени установления беременности и в период кормления грудью не допускаются к работам связанным с использованием ПЭВМ и переводятся на работы, не требующие использования ПЭВМ [6].

Медицинское освидетельствование студентов высших учебных заведений проводится в порядке и в сроке, установленные соответственно Минздравмедпромом России, Госкомсанэпиднадзором России, Госкомвузом России и Минобразования России.

### **6.8 Требования электробезопасности**

Для защиты от поражения электрическим током все токоведущие части и узлы ПЭВМ должны быть надежно защищены от случайных прикосновений кожухами, по правилам устройства электроустановок (ПУЭ), корпус устройства должен быть заземлен. Заземление выполняется при помощи изолированного медного проводника, с сечением 1,5 мм<sup>2</sup>, который, в свою очередь, при помощи сварки присоединяется к общей шине заземления, с сечением 48 мм<sup>2</sup>. Питание ПЭВМ должно осуществляться от силового щита,

через автоматически срабатывающий предохранитель, срабатывающий при коротком замыкании. Эксплуатация устройства должна производиться лицами, получившими допуск, имеющими квалификационную группу и допуск по электробезопасности не ниже третьего. Работа по устранению неисправностей и наладка должна производиться персоналом с квалификационной группой по технике безопасности не ниже третьей и только после снятия напряжения питания с устройства.

### **6.9 Меры оказания первой медицинской помощи при поражении электрическим током**

Если пострадавший соприкасается с токоведущими частями, необходимо быстро освободить его от действия электрического тока. Прикасаться к человеку, находящемуся под напряжением, опасно для жизни. Поэтому нужно быстро отключить ту часть установки, которой касается пострадавший. Для освобождения пострадавшего от провода следует воспользоваться сухой одеждой, доской или каким-либо другим предметом, не проводящему электрический ток или взяться за его одежду (если она сухая), избегая при этом прикосновения к металлическим предметам и открытым частям тела.

Далее необходимо:

- уложить пострадавшего на спину на твердую поверхность;
- проверить наличие у пострадавшего дыхания (определить по подъему грудной клетки, запотеванию зеркала и пр.);
- проверить наличие пульса на лучевой стороне у запястья или на сонной артерии на переднебоковой поверхности шеи;
- выяснить состояние зрачка, широкий зрачок указывает на резкое ухудшение кровоснабжения мозга;
- вызов врача по телефону 03 во всех случаях обязателен.

## ЗАКЛЮЧЕНИЕ

При помощи Qt Creator 4.7.2 создано и скомпилировано приложение, чьими выходными данными является изображение заданной в коде программы 3D-сцены, полученное при помощи метода обратной трассировки лучей, перемещение по ней камеры и модификация находящихся в ней объектов.

Дальнейшее направление для работы – усовершенствование алгоритма отрисовки, возможность отображать новые виды объектов, создание программных интерфейсов, переработка и улучшение кода программы, обособление функций программы в качестве отдельной библиотеки.

## СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

- 1 Шлее, М. Qt 5.10. Профессиональное программирование на C++. — СПб.: БХВ-Петербург, 2018. — 1072 с.: ил. — (в подлиннике).
- 2 Боресков А. В. Программирование компьютерной графики. Современный OpenGL. – М.: ДМК Пресс, 2019. – 372 с.: ил.
- 3 [www.scratchapixel.com](http://www.scratchapixel.com)
- 4 Андреев Г.И. Практикум по оценке интеллектуальной собственности: учебное пособие / Г.И. Андреев, В.В. Витчинка. – Москва: Финансы и статистика, 2003. – 176 с.
5. Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. Санитарно-эпидемиологические правила и нормативы СанПиН 2.2.2/2.4.1340-03.
6. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки. Санитарно-эпидемиологические правила и нормативы СанПиН 2.2.4/2.1.8.562-96.
7. Гигиенические требования к условиям труда женщин. Санитарно-эпидемиологические правила и нормативы по СанПиН 2.2.0.555-96.

## Отзыв

о работе выпускника Забайкальского государственного университета

Маржадова Сергей Викторович  
(фамилия, имя, отчество)

по выполнению выпускной квалификационной работы по направлению подготовки \_\_\_\_\_  
09.03.01 Информатика и вычислительная техника

Тема Разработка графического движка на основе обратной трассировки лучей

Объем работы:

количество листов пояснительной записки 48 стр

количество листов чертежей \_\_\_\_\_

количество схем \_\_\_\_\_

Заключение о степени соответствия выполненной работы заданию работа не в полной мере соответствует заданию

Проявленная выпускником самостоятельность при выполнении работы студент выполнил работу самостоятельно

Плановость и дисциплинированность в работе работа выполнена с отклонением от календарного плана

Умение пользоваться литературным, справочным, нормативным материалом студент умеет пользоваться справочным и нормативным материалом

Умение представить результаты выпускной квалификационной работы (в виде плакатов, мультимедийной презентации и др.) мультимедийная презентация имеется

Индивидуальные способности выпускника знает современные среды программирования

Положительные стороны работы \_\_\_\_\_

Недостатки работы 1. Мало приложена специальная часть пояснительной записки к ВКР, 2. Не в полной мере реализованы замечания

Характеристика общенаучной специальной подготовки выпускника общенаучная и специальная подготовка выпускника отвечает требованиям направления подготовки 09.03.01 Информатика и вычислительная техника

Заключение и предполагаемая оценка удовлетворительно




Руководитель \_\_\_\_\_

Дата «26» 06 2020 г.



УТВЕРЖДАЮ

Заведующий кафедрой ИВТ и ПМ

 О.В. Валова  
(подпись, И.О.Ф.)

« 26 » 06 2020 г.

### Заключение

о результатах проверки выпускной квалификационной работы в системе  
«Антиплагиат»

В выпускной квалификационной работе обучающегося

Мархандаев Сергей Викторович

(фамилия, имя, отчество)

Энергетического факультета группы ВМК-16

Разработка графического движка на основе обратной трассировки лучей

(название выпускной квалификационной работы)

доля оригинального текста составляет 60,40 процентов.

Расширенный отчет об источниках, с которыми были обнаружены совпадения  
фрагментов текста работы, находится на кафедре ИВТ и ПМ.

Менеджер кафедры

  
(подпись)

Р.С. Долгих  
(И.О.Ф.)

Дата «26» 06 2020 г.