**федеральное государственное автономное образовательное учреждение высшего образования**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет информационных технологий**

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № 10**

**Дисциплина:** Физическое моделирование компьютерных игр\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Тема:** Моделирование оптических систем. Часть 2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Выполнил(а): студент(ка) группы \_221-3710\_\_**

\_\_\_\_\_\_Пельт Сергей Александрович\_\_\_\_\_

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва2025**

**Моделирование оптических систем. Часть 2**

В данной лабораторной работе реализовано моделирование распространения светового луча через оптические среды с различными показателями преломления, а также его отражение от зеркал. При прохождении границы между двумя материалами учитывается закон преломления Снеллиуса, а также реализуется обработка полного внутреннего отражения.

Используемые формулы:

* Отражение луча: **Vector3.Reflect(I, N)**
* Отношение показателей преломления: **n = n1 / n2**
* Угол падения: **cos1 = -dot(I, N)**
* **sin²(θ2) = n² \* (1 - cos1²)**
* Если **sin²(θ2) > 1** — происходит полное внутреннее отражение
* Вектор преломлённого луча:

T = n \* I + (n \* cos1 - cos2) \* N, где cos2 = sqrt(1 - sin²(θ2))

Также построен путь луча (rayPath) и отображён с помощью LineRenderer.

**Листинг кода:**

void SimulateRay()

{

rayPath.Clear();

rayPath.Add(transform.position + Vector3.back \* 2);

Vector3 curDir = Quaternion.Euler(0, 0, angle) \* Vector3.right;

Vector3 curPos = transform.position;

float curRefractiveIndex = 1f;

int curReflections = 0;

GameObject prevMat = new();

Destroy(prevMat);

while (true)

{

RaycastHit hit;

if (Physics.Raycast(curPos, curDir, out hit))

{

curReflections += 1;

prevMat.SetActive(true);

rayPath.Add(hit.point + Vector3.back \* 2);

curPos = hit.point;

Vector3 normal = hit.normal;

if (hit.transform.CompareTag("Target"))

{

targetReached = true;

break;

}

else if (hit.transform.CompareTag("Mirror") && (curReflections < maxReflections || maxReflections == 0))

{

// Отражение от зеркала

curDir = Vector3.Reflect(curDir, normal);

}

else if (curReflections < maxReflections || maxReflections == 0)

{

OpticMaterial material = hit.transform.GetComponent<OpticMaterial>();

float n1 = curRefractiveIndex; // Текущий показатель преломления среды

float n2 = material.refractiveIndex; // Показатель преломления следующей среды

// Преломление по закону Снеллиуса

curDir = RefractRay(curDir, normal, n1, n2);

curRefractiveIndex = n2;

}

else

{

break;

}

prevMat = hit.transform.gameObject;

prevMat.SetActive(false);

targetReached = false;

}

else

{

prevMat.SetActive(true);

rayPath.Add(curPos + curDir \* 100f + Vector3.back \* 2);

break;

}

}

UpdateLineRenderer();

}

Vector3 RefractRay(Vector3 direction, Vector3 normal, float n1, float n2)

{

direction.Normalize();

normal.Normalize();

float n = n1 / n2; // Отношение показателей преломления: n = n1 / n2

float cos1 = -Vector3.Dot(direction, normal); // cos(θ1) между направлением луча и нормалью

// sin²(θ2) = n² \* (1 - cos1²)

float sin2sqr = n \* n \* (1.0f - cos1 \* cos1);

if (sin2sqr > 1.0f)

{

// Полное внутреннее отражение

Debug.Log("Reflect");

return Vector3.Reflect(direction, normal);

}

float cos2 = Mathf.Sqrt(1.0f - sin2sqr); // cos(θ2) = √(1 - sin²(θ2))

// Вектор преломлённого луча: T = n \* I + (n \* cos1 - cos2) \* N

return n \* direction + (n \* cos1 - cos2) \* normal;

}

Ссылка на билд: <https://github.com/Sergey-Pelt/Physics>