Санкт-Петербургский Государственный Электротехнический Университет "ЛЭТИ"

кафедра физики

Задание №1 по дисциплине

"Физические основы информационных технологий"

Название: Искривление луча в оптическом канале

|  |  |
| --- | --- |
| Фамилия И.О.: | Попандопуло А. Г. |
| группа: | 1303 |
| Преподаватель: | Альтмарк А.М. |
| Итоговый балл: |  |
|  |  |

Крайний срок сдачи: 22.10.23

.

Санкт-Петербург 2023

**Условие задания**

Найти длину траектории светового луча *S* в прямолинейном дисперсионном оптоволоконном канале, Рис.1, c показателем преломления n1. Оптоволокно окружено средой с показателем преломления n2. Функцию распределения показателя преломления n1(y, ω) можно представить как:

,

где y – поперечная координата, ω – циклическая частота светового луча.

Функцию , функцию Zf(y), описывающую координату z выходного торца волновода, начальный угол ввода луча α в волновод, координату ввода луча в волновод *y*=y0, радиус канала R можно взять в файле FOIT\_IDZ1.xlsx. Все геометрические размеры даются в безразмерных координатах.

Необходимо построить график траектории луча, а также записать ответ *S* в текстовый файл IDZ1\IDZ1.txt. Помимо текстового файла IDZ1.txt в папке IDZ1 должен находиться Word-файл (Pdf-файл) с отчетом, а также файл с кодом (Python, Mathcad, Mathematica). Для лучшего понимания отчетности смотрите папку “Пример организации яндекс-папки студентов”.

Пример содержания файла IDZ1.txt:

4.53258

n2

Y

R

Zf(y)

y0

n1

Z

α

0

n2

-R

n2

Рисунок 1

**Вариант 9:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| R | n2 | f1(y) | Zf(y) | ω \* 10^14, рад/с | y0 | α , град |
| 0.8 | 1 | 1.4 + 0.12\*Cos[3\*y] | 18 + 3\*Sin[17.951958020513104\*y] | 3.1 | 0.4 | 40 |

**Основные теоретические положения**

Закон преломления света (закон Снеллиуса):

Падающий и преломлённый лучи и перпендикуляр, проведенный к границе раздела двух сред в точке падения луча, лежат в одной плоскости;

Отношение синуса угла падения к синусу угла преломления есть величина постоянная для двух сред, равная относительному показателю преломления:

Явление полного внутреннего отражения:

Рассмотрим луч света, который переходит из среды с большим показателем преломления в вещество с меньшим абсолютным показателем преломления (например, из воды в воздух).

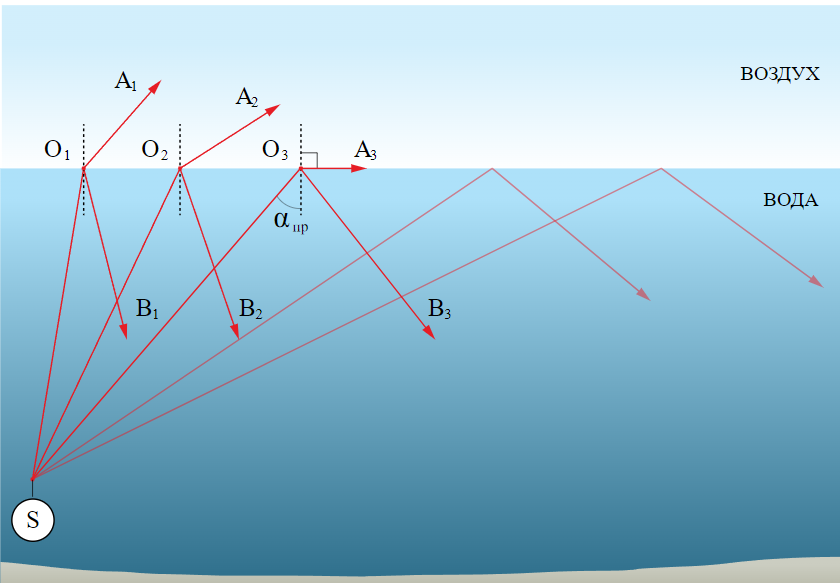


Рисунок 2 – схема полного внутреннего отражения

В этом случае угол преломления луча больше, чем угол падения. Если увеличивать угол падения, то при некотором угле угол преломления станет равным 90 градусам. При дальнейшем увеличении угла падения луч полностью отражается от границы раздела и не проходит в другую среду. Это явление называется явлением полного внутреннего отражения.

Закон преломления света для :

Явление полного внутреннего отражения наблюдается только при переходе светового луча из среды с большим абсолютным показателем преломления в среду с меньшим абсолютным показателем преломления вещества, а также при угле падения большем или равном углу

Явление полного внутреннего отражения используется в волоконной оптике – для передачи световых сигналов на большие расстояния.

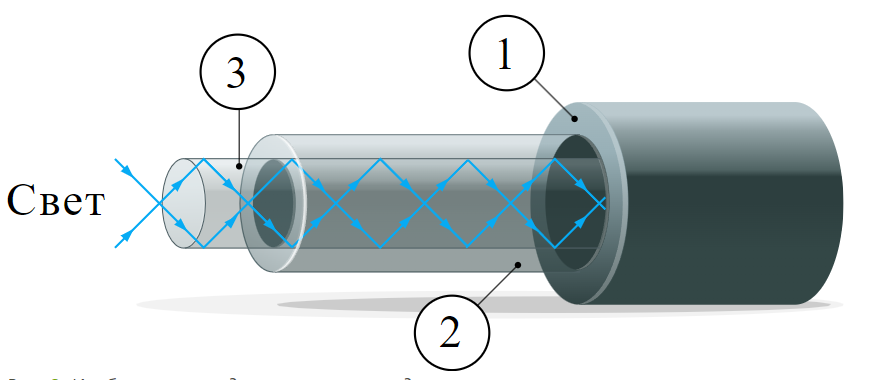
1 – защитная оболочка

Рисунок 3 – изображение хода луча в световоде

2 – оболочка (с меньшим показателем преломления)

3 – сердцевина (с большим показателем преломления)

**Вывод:**

В ходе выполнения индивидуального домашнего задания на практике были изучены закон преломления света и явление полного внутреннего отражения, реализована программа осуществляющая моделирование и нахождение длины траектории луча.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Файл **IDZ1.py**

import numpy as np

import matplotlib.pyplot as plt

def f1(y):

return 1.4 + 0.12 \* np.cos(3 \* y)

def n1(y):

global omega

return f1(y) \* (1 - ((0.35 \* 10 \*\* 14) / omega) \*\* 2)

def z\_f(y):

return 18 + 3 \* np.sin(17.951958020513104 \* y)

def get\_new\_angle(curr\_angle, n1, n2):

return np.arcsin(np.clip((np.sin(curr\_angle) \* n1) / n2, -1, 1))

def model(r, n\_outer, y0, z0, alpha\_0):

global step

alpha\_0 = np.deg2rad(alpha\_0)

s = 0

course = 1

arr\_y = [y0]

arr\_z = [z0]

beta = get\_new\_angle(alpha\_0, n\_outer, n1(y0))

arr\_y += [y0 + step \* np.sin(beta)] # first y coord

arr\_z += [z0 + step \* np.cos(beta)] # first z coord

n\_1 = n1(arr\_y[-2]) # refractive coeff of first area take from prev coord

n\_2 = n1(arr\_y[-1]) # # refractive coeff of second area take from last coord

alpha = np.pi / 2 - beta # new first angle

beta = get\_new\_angle(alpha, n\_1, n\_2)

s += step # accum length

while arr\_z[-1] < z\_f(arr\_y[-1]): # while haven't reached yet

if abs(np.sin(beta)) >= 1:

course \*= -1

beta = alpha

arr\_y += [arr\_y[-1] + step \* np.cos(beta) \* course]

arr\_z += [arr\_z[-1] + step \* np.sin(beta)]

n\_1 = n1(arr\_y[-2])

n\_2 = n1(arr\_y[-1]) if abs(arr\_y[-1]) <= r else n\_outer # possible to exceed radius

alpha = beta

beta = get\_new\_angle(alpha, n\_1, n\_2)

s += step # accum length

return arr\_y, arr\_z, s

def plot\_model(y\_values, z\_values, r):

global step

y = np.arange(-r, r, step)

plt.plot(z\_values, y\_values, 'r')

plt.plot(z\_f(y), y, 'c')

plt.axhline(r, color='b')

plt.axhline(-r, color='b')

plt.xlabel('Ось Z')

plt.ylabel('Ось Y')

plt.show()

step = 0.00001

radius = 0.8

n\_out = 1

omega = 3.1 \* 10 \*\* 14

y\_start = 0.4

z\_start = 0

alpha\_start = 40

y\_points, z\_points, trajectory\_length = model(radius, n\_out, y\_start, z\_start, alpha\_start)

print("Длина траектории =", trajectory\_length)

plot\_model(y\_points, z\_points, radius)