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.: Ягодаров М. А.

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, с показателем преломления n_1. Оптоволокно окружено средой с показателем преломления n_2. Функцию распределения показателя преломления $n_1(y, \omega)$ можно представить как:

$$n_1(y,\omega) = f_1(y)\left(1 - \left(\frac{(0.35 * 10^{14})}{\omega}\right)^2\right),$$

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

Функцию $f_1(y)$, 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 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

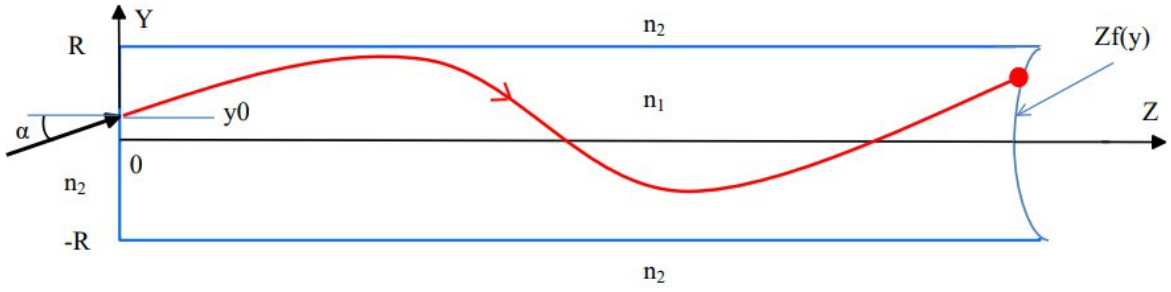

Рисунок.1

Исходные данные

Вар	R	n2	f1(y)	Zf(y)	w * 10^14	y0	α, град
17	1.2	1	1.4 - 0.18*y^4	12 + 2*Sin [17.951958020513104*y]	3.2	0.4	42

Основные теоретические положения

Если световые волны достигают границы раздела двух сред и проникают в другую среду, то направление их распространения также изменяется — происходит преломление света. Преломление света — это изменение направления распространения световой волны при переходе из одной прозрачной среды в другую.

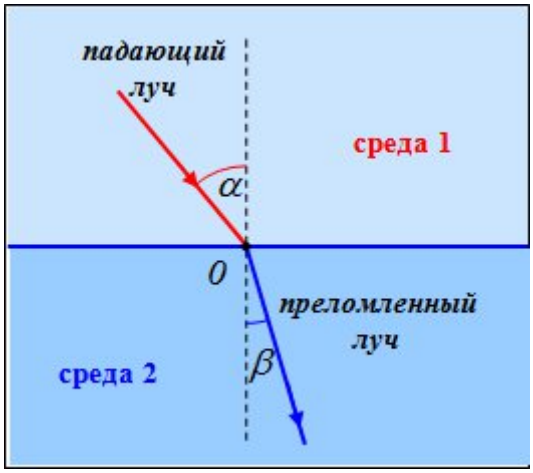

Рисунок 2 – преломление света

Закон преломления гласит:

Падающий луч, перпендикуляр к границе раздела сред в точке падения и преломленный луч лежат в одной плоскости, причем отношение синуса угла падения к синусу угла преломления постоянно для данной пары сред и равно показателю преломления второй среды относительно первой

$$\frac{sin\alpha}{sin\beta} = \frac{n_2}{n_1}$$

Здесь n_2показатель преломления среды, в которой распространяется преломленная волна, n_1 показатель преломления среды, в которой распространяется падающая волна.

ПРИЛОЖЕНИЕ А

ПРОГРАММА MAIN.PY

```
import matplotlib.pyplot as plt
import numpy as np



radius: float = 1.2
n2: int = 1
omega: float = 3.2
y0: float = 0.4
alpha: float = np.deg2rad(42)



def f1(y: float) -> float:
    return 1.4 - 0.18 * y ** 4



def zf(y):
    return 12 + 2 * np.sin(17.951958020513104 * y)



def calculate_n1(y: float) -> float:
    return f1(y) * (1 - (0.35 / omega) ** 2)



def calculate_sin0() -> (float, int):
    n1_0 = calculate_n1(y0)
    sin_beta = np.sin(np.pi / 2 - np.arcsin(np.sin(np.abs(alpha))
* n2 / n1_0))
    return sin_beta, np.sign(alpha)



steps_cnt: int = 10000
```

```
delta: float = 2 * radius / steps_cnt

y: float = y0
z: float = 0
n: float = calculate_n1(y)
sin, sign = calculate_sin0()
length: float = 0



def next_step() -> None:
    global y, z, n, sin, sign, length

    length += delta
    y += sign * np.sqrt(1 - sin ** 2) * delta
    z += np.abs(sin) * delta

    new_n: float = calculate_n1(y) if radius >= np.abs(y) else n2
    new_sin: float = sin * n / new_n

    if np.abs(new_sin) >= 1:
        sign *= -1
    else:
        n = new_n
        sin = new_sin



y_points: list[float] = []
z_points: list[float] = []
end_y_points: list[float] = []
end_z_points: list[float] = []



def process() -> None:
    global y_points, z_points, end_y_points, end_z_points

    while z < zf(y):
```

```
            y_points.append(y)
            z_points.append(z)
            next_step()

        end_y_points = np.arange(-radius, radius, delta)
        end_z_points = zf(end_y_points)



    if __name__ == '__main__':
        process()

        print(length)

        bottom_line_end = end_z_points[0]
        top_line_end = end_z_points[-1]

        wave_color = 'blue'
        lines_color = 'green'
        end_color = 'orange'

        plt.plot(z_points, y_points, wave_color,
                 end_z_points, end_y_points, end_color,
                 [0, bottom_line_end], [-radius, -radius],
lines_color,
                 [0, top_line_end], [radius, radius], lines_color)
        plt.show()
```