**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Компьютерная графика»

**Лабораторная работа № 3**

Тема: Основы построения фотореалистичных изображений

Студент: Черных Сергей Дмитриевич

Группа: 08-305

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2022

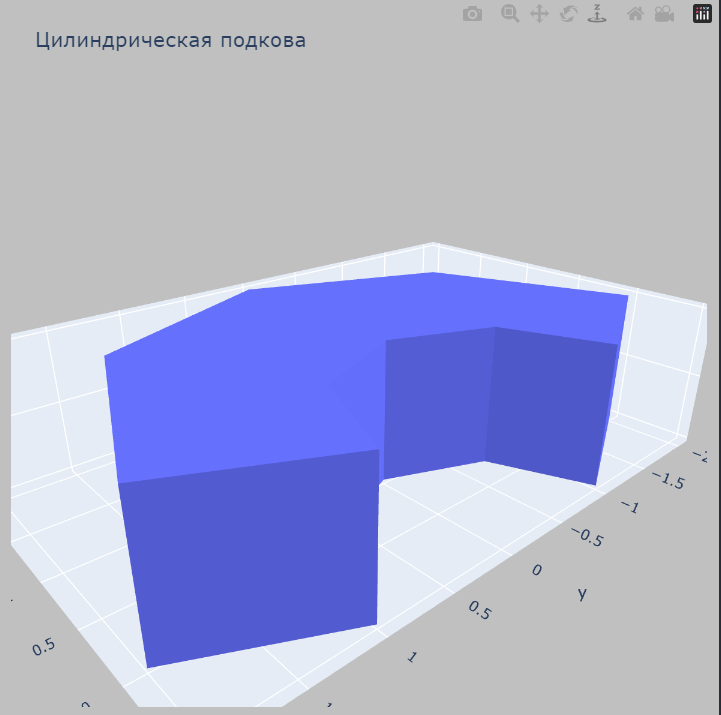
1. Постановка задачи

Разработать формат представления многогранника и процедуру его каркасной отрисовки в ортографической и изометрической проекциях. Обеспечить удаление невидимых линий и возможность пространственных поворотов и масштабирования многогранника. Обеспечить автоматическое центрирование и изменение размеров изображения при изменении размеров окна.

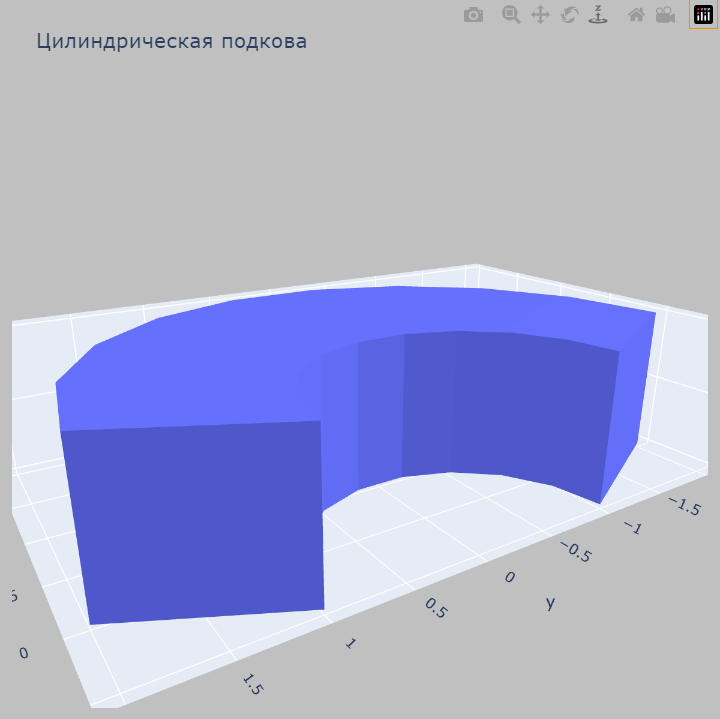
1. Описание программы

На вход поступает степень число построение точек для окружности n, передаётся в функцию get\_xyz() как nt = n (по умолчанию задан как 5), которое не меньше 3, высота и расстояние между нижними полуокружностями (ширина подковы) заданы по умолчанию как h = 1 и a = 0.05, радиусы окружностей внешней и внутренней, так же заданы по умолчанию как r\_outer = 2 и r\_inner = 1. Все параметры заданные по умолчанию, можно задать самостоятельно, передав в функцию соответствующие переменные. Для вычисления полигонов используется функция get\_ijk(), d в которую передаётся число nt = n (введённое ранее). Для отрисовки используется функция plot(), в которую передаются полученные из функций get\_xyz() и get\_ijk() массивы x, y, z, i, j, k соответственно.

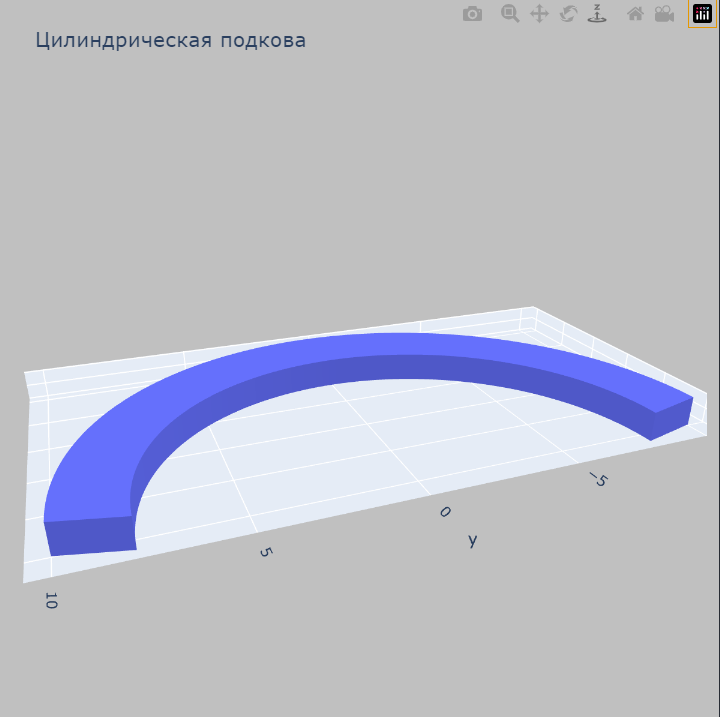
1. Набор тестов
2. NULL
3. N=10
4. N=100, r\_outter = 10, r\_inner = 8
5. n = 3, h=5, r\_outter = 6, r\_inner = 4, a=1,
6. Результаты выполнения тестов
7. NULL



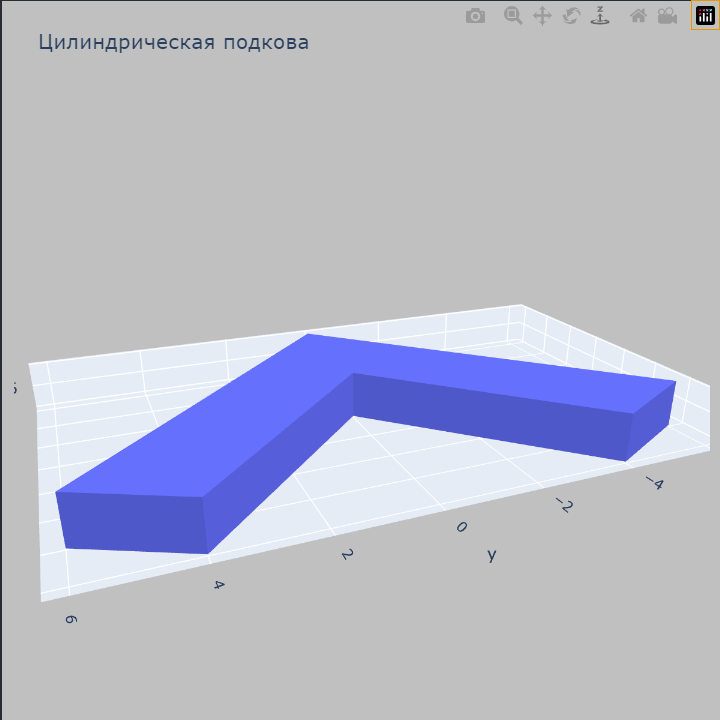
1. n=10



1. n=100, r\_outter = 10, r\_inner = 8



1. n = 3, h=5, r\_outter = 6, r\_inner = 4, a=1,



1. Листинг программы

# **Лабораторная работа №3**

### **Тема**: Основы построения фотореалистичных изображений.

**Задание**: Используя результаты Л.Р. №2, аппроксимировать заданное тело выпуклым многогранником. Точность аппроксимации задается пользователем. Обеспечить возможность вращения и масштабирования многогранника и удаление невидимых линий и поверхностей. Реализовать простую модель закраски для случая одного источника света. Параметры освещения и отражающие свойства материала задаются пользователем в диалоговом режиме.

Черных Сергей М8О-305Б-20

### **Вариант**: Цилиндрическая подкова

import numpy as np  
import plotly.graph\_objects as go

def get\_xyz(r\_outter=2, r\_inner=1, nt=5, a=00.5, h=1):  
 def cylinder(r, a=0, nt=5, h=1):  
 theta = np.linspace(-np.pi/2, np.pi/2, nt)  
 v = np.ones(nt) \* h  
 x = r\*np.cos(theta) - a  
 y = r\*np.sin(theta)  
 z = v  
 return x, y, z  
  
 x, y, z = cylinder(r\_outter, nt=nt, h=h)  
 x\_1, y\_1, z\_1 = cylinder(r\_outter, nt=nt, h=1)  
 z\_1 = np.zeros\_like(z\_1)  
 x = np.concatenate((x, x\_1), axis=0)  
 y = np.concatenate((y, y\_1), axis=0)  
 z = np.concatenate((z, z\_1), axis=0)  
  
 x\_2, y\_2, z\_2 = cylinder(r\_inner, nt=nt, a=a, h=h)  
 x\_3, y\_3, z\_3 = cylinder(r\_inner, nt=nt, a=a, h=h)  
 z\_3 = np.zeros\_like(z\_3)  
 x\_2 = np.concatenate((x\_2, x\_3), axis=0)  
 y\_2 = np.concatenate((y\_2, y\_3), axis=0)  
 z\_2 = np.concatenate((z\_2, z\_3), axis=0)  
  
 x = np.concatenate((x, x\_2), axis=0)  
 y = np.concatenate((y, y\_2), axis=0)  
 z = np.concatenate((z, z\_2), axis=0)  
  
 return x, y, z

def get\_ijk(nt=5):  
 i = np.array([])  
 j = np.array([])  
 k = np.array([])  
  
  
 for c in range(nt-1):  
 i = np.append(i, c)  
 j = np.append(j, c + 1)  
 k = np.append(k, c + nt)  
  
 for c in range(nt-1):  
 i = np.append(i, c + nt + 1)  
 j = np.append(j, c + nt)  
 k = np.append(k, c + 1)  
  
 for c in range(nt-1):  
 i = np.append(i, c + 2\*nt)  
 j = np.append(j, c + 2\*nt + 1)  
 k = np.append(k, c + 3\*nt)  
  
 for c in range(nt-1):  
 i = np.append(i, c + 3\*nt + 1)  
 j = np.append(j, c + 3\*nt)  
 k = np.append(k, c + 2\*nt + 1)  
  
 for c in range(nt-1):  
 i = np.append(i, 3\*nt + c)  
 j = np.append(j, c + 3\*nt + 1)  
 k = np.append(k, c + nt)  
  
 for c in range(nt-1):  
 i = np.append(i, c + nt + 1)  
 j = np.append(j, c + nt)  
 k = np.append(k, c + 3\*nt + 1)  
  
 for c in range(nt-1):  
 i = np.append(i, c + 2\*nt)  
 j = np.append(j, c + 2\*nt + 1)  
 k = np.append(k, c)  
  
 for c in range(nt-1):  
 i = np.append(i, c + 1)  
 j = np.append(j, c)  
 k = np.append(k, c + 2\*nt + 1)  
  
 i = np.append(i, 0)  
 j = np.append(j, 2\*nt)  
 k = np.append(k, nt)  
  
 i = np.append(i, 2\*nt)  
 j = np.append(j, 3\*nt)  
 k = np.append(k, nt)  
  
 i = np.append(i, nt - 1)  
 j = np.append(j, 4\*nt - 1)  
 k = np.append(k, 3\*nt - 1)  
  
 i = np.append(i, nt - 1)  
 j = np.append(j, 2\*nt - 1)  
 k = np.append(k, 4\*nt - 1)  
  
 return i, j, k

def plot(x, y, z, i, j, k):  
 fig = go.Figure(data=[  
 go.Mesh3d(  
 x=x,  
 y=y,  
 z=z,  
  
 i=i,  
 j=j,  
 k=k,  
 name='y',  
 showscale=False,  
 flatshading=True,  
 )  
 ])  
  
 fig.update\_layout(  
 autosize=False,  
 width=600,  
 height=600,

scene = dict( aspectmode='data'),  
 margin=dict(  
 l=10,  
 r=10,  
 b=10,  
 t=80,  
 pad=5  
 ),  
 paper\_bgcolor="silver",  
 title\_text = "Цилиндрическая подкова",  
 )  
 fig.show()

def check\_n(n):  
 if n < 3:  
 print("Input n value > 2")  
 else:  
 return n;

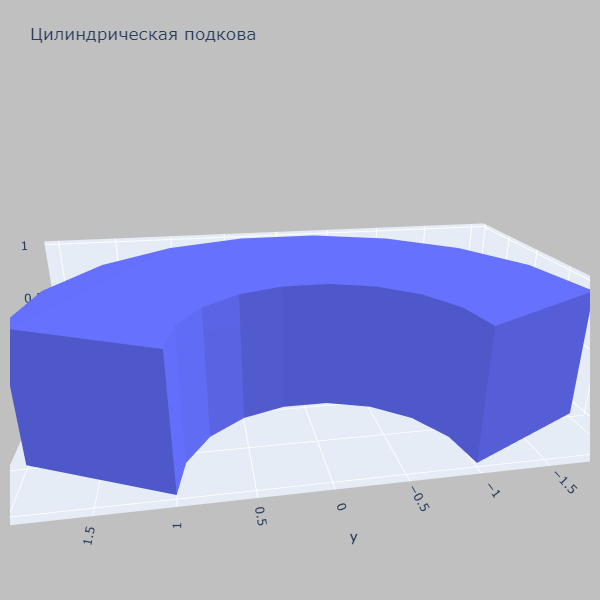
Введите n > 3

n = 10

check\_n(n)

10

x, y, z = get\_xyz(nt=n)  
i, j, k = get\_ijk(nt=n)  
  
plot(x, y, z, i, j, k)



ЛИТЕРАТУРА

1. Документация библиотеки Plotly <https://plotly.com/graphing-libraries/>
2. Документация библиотеки Plotly для анализа данных от МФТИ https://mipt-stats.gitlab.io/courses/ad\_fivt/plotly.html