# Московский авиационный институт (национальный исследовательский университет)

## Факультет информационных технологий и прикладной математики

## Кафедра вычислительной математики и программирования


**Курсовая работа по курсу «Компьютерная графика»**


Студент: И. П. Моисеенков
Преподаватель: Г. С. Филиппов
Группа: М8О-308Б-19

Дата: 08.12.2021
Оценка:
Подпись:

**Москва, 2021**

# Курсовая работа

# Каркасная визуализация порции поверхности.

**Задача:** Составить и отладить программу, обеспечивающую каркасную визуализацию порции поверхности заданного типа. Исходные данные готовятся самостоятельно и вводятся из файла или в панели ввода данных. Должна быть обеспечена возможность тестирования программы на различных наборах исходных данных. Программа должна обеспечивать выполнение аффинных преобразований для заданной порции поверхности, а также возможность управлять количеством изображаемых параметрических линий. Для визуализации параметрических линий поверхности разрешается использовать только функции отрисовки отрезков в экранных координатах.

**Вариант 11:** Кинематическая поверхность. Образующая – астроида, направляющая – кривая Безье 3D 2-й степени

# 1 Описание

Кинематическая поверхность — поверхность, полученная перемещением образующей по заданной траектории. Образующая может иметь вид как геометрического примитива: линии, дуги, окружности, — так и сложной кривой или сплайна. Производящая (образующая) кинематической поверхности перемещается в пространстве по определенной траектории Она может в процессе движения сохранять свою форму (иметь неизменный вид), а также непрерывно ее изменять.

Астроиду (образующую) можно задать параметрическим способом:

$$x = Rcos^3t$$

$$y = Rsin^3t$$

Для задания кривой Безье (направляющей) нужно знать три точки, по которым и будет строиться кривая. Эти точки запрашиваются у пользователя при запуске программы. Затем по этим точкам строится кривая Безье второй степени по следующей формуле:

$$B(t) = (1-t)^2 * P_0 + 2(1-t)t * P_1 + t^2 * P_2, \quad 0 \leq t \leq 1$$

Для построения кинматической поверхности я воспользовался библиотекой matplotlib для языка python. Сначала необходимо построить кривую Безье. Для этого вычисляются координаты 20 точек, лежащих на этой кривой. Получаем массив размерности (20, 3).

Теперь необходимо построить астроиду вокруг каждой из посчитанных точек. Будем вычислять координаты астроиды для 30 значений параметра t. Таким образом получим массив размерности (20, 30, 3) - для каждой из 20 точек кривой Безье строим 30 точек астроиды в трехмерном пространстве. Затем остается построить поверхность по посчитанному набору точек. Такая операция неоднократно выполнялась мной за время выполнения лабораторных работ.

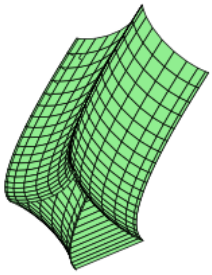
# 2 Исходный код

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d.art3d import Poly3DCollection


def find_bezier_curve(p0, p1, p2, steps):
    """
    Finds the coordinates of 3D quadratic Bzier curve
    p0, p1, p2 - points
    steps - number of steps
    returns an array of curve's coordinates
    """
    bezier = []
    for i in range(steps):
        t = i / steps

        # B(t) = (1-t)^2 * p0 + 2(1-t)t * p1 + t^2 * p2, 0 <= t <= 1
        coef0 = (1-t) ** 2
        coef1 = 2 * (1-t) * t
        coef2 = t ** 2
        bezier.append(coef0 * p0 + coef1 * p1 + coef2 * p2)

    return np.array(bezier)


# main points
p0 = np.array([-1, 4, 8])
p1 = np.array([-5, 1, -5])
p2 = np.array([0, -1, -2])

bezier = find_bezier_curve(p0, p1, p2, 20) # (bezier_steps, 3)

# counting astroid coordinates along the Bzier curve
astroid_approximation = 30
astroid_radius = 5
astroid = []
for bezier_point in bezier:
```

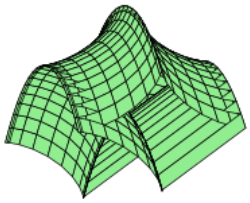
```
    cur_astroid_points = []
    for i in range(astroid_approximation):
        t = i / astroid_approximation
        cur_astroid_points.append([
            astroid_radius * np.cos(5 * t)**3 + bezier_point[0],
            2 * bezier_point[1],
            astroid_radius * np.sin(5 * t)**3 + bezier_point[2]
        ])

    astroid.append(np.array(cur_astroid_points))
astroid = np.array(astroid) # (bezier_steps, astroid_approximation, 3)

sides = [] # sides of figure
for i in range(astroid.shape[0] - 1):
    for j in range(astroid.shape[1]):
        sides.append([
            astroid[i][j],
            astroid[(i + 1) % len(astroid)][j],
            astroid[(i + 1) % len(astroid)][(j + 1) % len(astroid[i])],
            astroid[i][(j + 1) % len(astroid[i])]
        ])

# plotting figure
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
plt.axis('off')

ax.set_xlim([-10, 10])
ax.set_ylim([-10, 10])
ax.set_zlim([-10, 10])

ax.add_collection3d(Poly3DCollection(sides, edgecolors='black', facecolor='lightgreen'
    , linewidths=0.5))
plt.show()
```

# 3 Результат работы

Кинематическая поверхность.
Образующая - астроида, направляющая - кривая Безье 2 степени



Кинематическая поверхность.
Образующая - астроида, направляющая - кривая Безье 2 степени



# 4 Выводы

Выполнив курсовую работу, я узнал, что такое кинематическая поверхность и как она строится. Для закрепления знаний я построил кинематическую поверхность, образующая которой - астроида, а направляющая - кривая Безье 2 степени.

Также я еще раз повторил материал, связанный с кривыми Безье и еще раз потренировался в их построении.