

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**  
**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Отчет по лабораторной работе № 3.6**  
**по дисциплине «Технологии распознавания образов»**

Выполнил студент группы ПИЖ-б-о-21-1

Трушева В. О. .«\_\_»\_\_\_\_\_2023г.

Подпись студента\_\_\_\_\_

Работа защищена « \_\_ »\_\_\_\_\_20\_\_г.

Проверила Воронкин Р.А. \_\_\_\_\_  
(подпись)

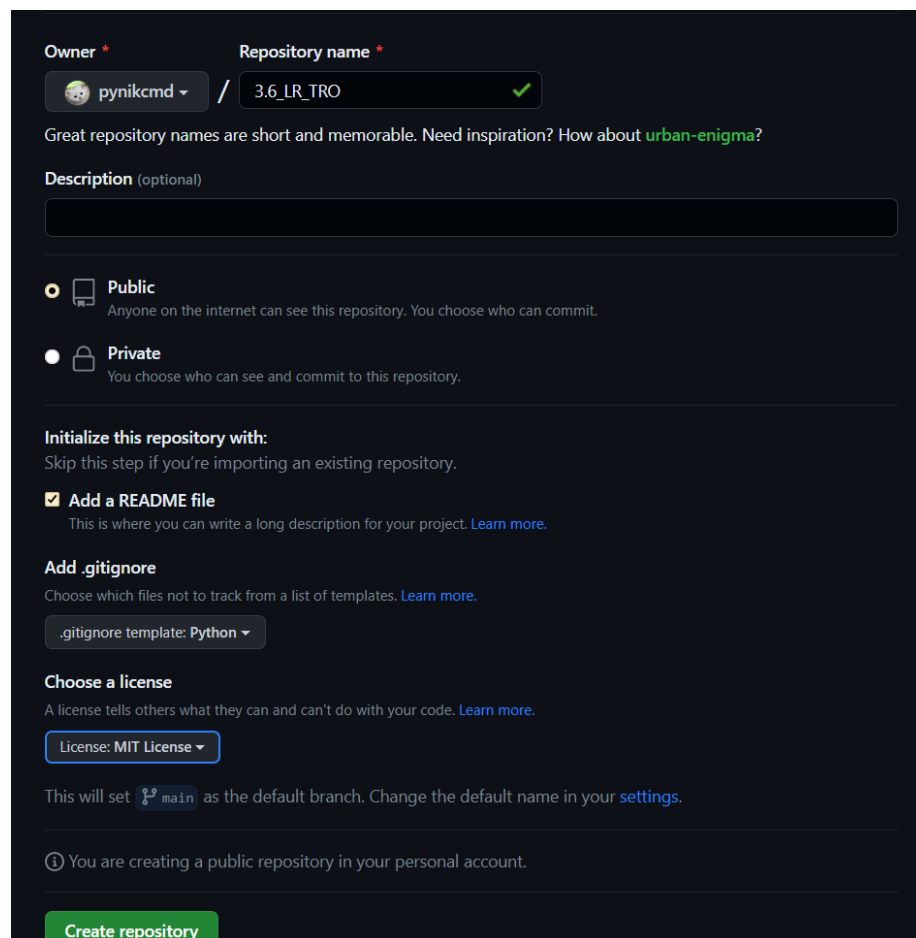
Ставрополь 2023

Цель работы: исследовать базовые возможности визуализации данных в трехмерном пространстве средствами библиотеки matplotlib языка программирования Python.

## Методика и порядок выполнения работы

1. Изучить теоретический материал работы.

2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования (выбор языка программирования будет доступен после установки флажка Add .gitignore).



The screenshot shows the GitHub 'Create repository' form. At the top, the 'Owner' is 'pynikcmd' and the 'Repository name' is '3.6\_LR\_TRO', which is marked with a green checkmark. Below this, there is a text input field for the 'Description (optional)'. The 'Visibility' section shows 'Public' selected with a radio button, and 'Private' is unselected. Under 'Initialize this repository with:', the checkbox for 'Add a README file' is checked. The 'Add .gitignore' section shows the 'Python' template selected. The 'Choose a license' section shows the 'MIT License' selected. At the bottom, there is a green 'Create repository' button. A note at the bottom states: 'You are creating a public repository in your personal account.'

Рисунок 1 – Создание репозитория

3. Выполните клонирование созданного репозитория на рабочий компьютер.

```
D:\fgit>git clone https://github.com/pynikcmd/3.6_LR_TR0.git
Cloning into '3.6_LR_TR0'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 2 – Клонирование репозитория

4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
D:\fgit\3.6_LR_TR0>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/fgit/3.6_LR_TR0/.git/hooks]
```

Рисунок 3 – Модель git-flow

5. Дополните файл .gitignore необходимыми правилами для выбранного языка программирования, интерактивной оболочки Jupyter notebook и интегрированной среды разработки.

6. Проработать примеры лабораторной работы в отдельном ноутбуке.

```
Jupyter Primers Last Checkpoint: 8 часов назад (autosaved)
File Edit View Insert Cell Kernel Help Kernel starting, please wait... Saving every 120 sec. Trusted Python 3 (pykernel)
In [3]: import matplotlib.pyplot as plt
import matplotlib inline
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
```

### Линейный график

```
In [4]: x = np.linspace(-np.pi, np.pi, 50)
y = x
z = np.cos(x)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot(x, y, z, label='parametric curve')

Out[4]: <mpl_toolkits.mplot3d.art3d.Line3D at 0x20b53284f50>
```

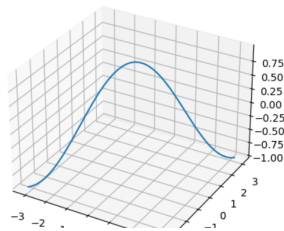


Рисунок 4 – Примеры лабораторной работы

### Точечный график

```
In [5]: np.random.seed(123)
x = np.random.randint(-5, 5, 40)
y = np.random.randint(0, 10, 40)
z = np.random.randint(-5, 5, 40)
s = np.random.randint(10, 100, 40)
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z, s=s)

Out[5]: <mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x20b554c2790>
```

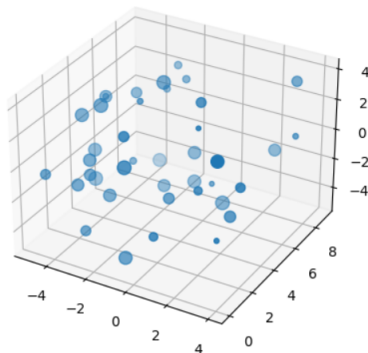


Рисунок 5 – Примеры лабораторной работы

### Каркасная поверхность

```
In [6]: u, v = np.mgrid[0:2*np.pi:20j, 0:np.pi:10j]
x = np.cos(u)*np.sin(v)
y = np.sin(u)*np.sin(v)
z = np.cos(v)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_wireframe(x, y, z)

Out[6]: <mpl_toolkits.mplot3d.art3d.Line3DCollection at 0x20b55321290>
```

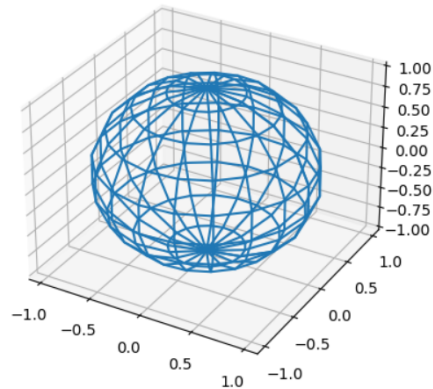


Рисунок 6 – Примеры лабораторной работы

### Поверхность

```
In [7]: u, v = np.mgrid[0:2*np.pi:20j, 0:np.pi:10j]
x = np.cos(u)*np.sin(v)
y = np.sin(u)*np.sin(v)
z = np.cos(v)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(x, y, z, cmap='inferno')

Out[7]: <mpl_toolkits.mplot3d.art3d.Poly3DCollection at 0x20b553177d0>
```

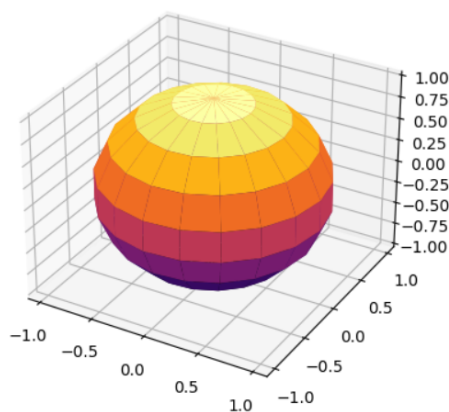


Рисунок 7 – Примеры лабораторной работы

7. Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.) требующей построения трехмерного графика, условие которой предварительно необходимо согласовать с преподавателем.

Условие:

Изобразим график функции Розенброка. Данная функция определяется следующим образом:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2$$

Эта функция используется в оптимизации для тестирования эффективности алгоритмов. Минимальное значение этой функции равно 0 и достигается в точке (1, 1).

Приведенная программа ниже генерирует трехмерный график функции Розенброка, который показывает, как значение функции меняется в зависимости от значений аргументов  $x$  и  $y$ . Цвет и высота поверхности на графике соответствуют значению функции в каждой точке  $(x, y)$ .

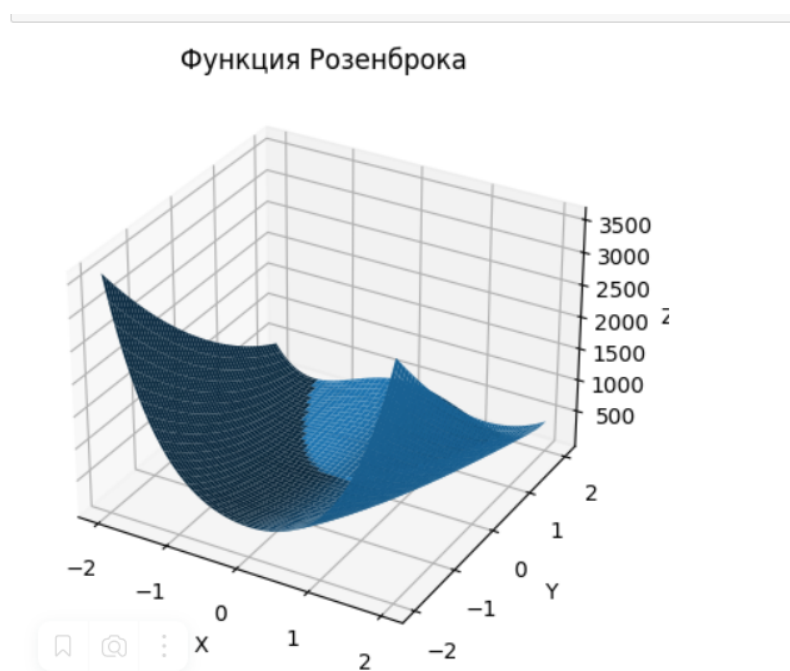


Рисунок 8 – Результат работы программы

8. Зафиксируйте сделанные изменения в репозитории.

9. Выполните слияние ветки для разработки с веткой main (master).

10. Отправьте сделанные изменения на сервер GitHub.

## Контрольные вопросы

1. Как выполнить построение линейного 3D-графика с помощью matplotlib?

Для построения линейного графика используется функция `plot()`.

2. Как выполнить построение точечного 3D-графика с помощью matplotlib?

Для построения точечного графика используется функция `scatter()`.

3. Как выполнить построение каркасной поверхности с помощью matplotlib?

Для построения каркасной поверхности используется функция `plot_wireframe()`.

4. Как выполнить построение трехмерной поверхности с помощью matplotlib?

Для построения поверхности используйте функцию `plot_surface()`.