**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Функциональное программирование»**

**на тему**

**«Анализ космических данных с использованием параллельных вычислений»**

**Вариант: 1**

**Студент гр. 23Б15-пу**

**Сурин И.С.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 2](#_Toc185849457)
2. [Описание задачи 2](#_Toc185849458)
3. [Теоретическая часть 2](#_Toc185849459)
4. [Основные шаги работы программы 3](#_Toc185849460)
5. [Описание программы 4](#_Toc185849461)
6. [Рекомендации для пользователя 5](#_Toc185849462)
7. [Рекомендации для программиста 6](#_Toc185849463)
8. [Исходный код программы 6](#_Toc185849464)
9. [Контрольный пример 6](#_Toc185849465)
10. [Вывод 8](#_Toc185849466)
11. [Источники 8](#_Toc185849467)

# **Цель работы**

Создать программное обеспечение для обработки изображений, полученных с космических телескопов, с целью анализа и классификации астрофизических объектов. Особое внимание уделяется использованию параллельных вычислений для ускорения обработки больших объемов данных и повышения общей производительности.

# **Описание задачи**

Программа должна выполнять следующие функции:

1. Принимать изображения в форматах .jpg, .jpeg, .png.
2. Обрабатывать изображения следующим образом:
   1. Преобразование изображения в оттенки серого.
   2. Выделение контуров объектов с использованием алгоритма Канни.
   3. Классификация объектов на основе их яркости и размера.
3. Отображать обнаруженные объекты на изображении с помощью цветных контуров.
4. Сохранять результаты обработки в XML-файле, содержащем информацию о найденных объектах.
5. Обеспечивать наличие графического интерфейса, позволяющего выбирать папки для загрузки и сохранения данных, а также запускать процесс обработки.

# **Теоретическая часть**

Обработка изображений

1. Для анализа изображений используется библиотека OpenCV.
2. Контуры объектов выделяются с помощью алгоритма Канни, который используется для извлечения границ объектов в изображении.
3. Площадь и яркость объектов рассчитываются путем анализа пикселей, попавших в выделенные контуры.

Классификация объектов

1. Классификация объектов выполняется на основе двух основных параметров: площади и средней яркости:
   1. Звезда: объекты с малой площадью и низкой яркостью.
   2. Галактика: объекты средней площади с умеренной яркостью.
   3. Сверхновая: объекты с большой площадью и высокой яркостью.

Параллельная обработка

1. Изображения разделяются на четыре части (квадранта), которые обрабатываются параллельно, что позволяет ускорить обработку.
2. Для реализации параллельных вычислений используется библиотека ProcessPoolExecutor, которая распределяет задачи по всем доступным ядрам процессора.
3. Параллельное выполнение задач значительно повышает производительность программы, особенно при работе с большими изображениями.

Создание графического интерфейса

1. Для создания графического интерфейса используется библиотека Tkinter.
2. Интерфейс позволяет пользователю выбрать папки для загрузки изображений и сохранения результатов, а также запустить процесс обработки изображений.

# **Основные шаги работы программы**

1. Обработка изображения
   1. Загружается изображение в оттенках серого.
   2. Изображение делится на четыре секции, каждая из которых обрабатывается независимо.
2. Классификация объектов
   1. Для каждого объекта вычисляются параметры, такие как площадь и яркость. На основе этих параметров каждый объект классифицируется как звезда, галактика или сверхновая.
3. Параллельная обработка
   1. Каждый квадрант изображения обрабатывается в отдельном процессе, что позволяет эффективно использовать многозадачность и ускорить обработку.
   2. Результаты из всех процессов объединяются в единый список статистики.
4. Сохранение результатов
   1. Изображения сохраняются с наложенными цветными контурами, которые визуально выделяют объекты на изображении.
   2. Информация о найденных объектах сохраняется в XML-файле, что позволяет дальше использовать данные для анализа или отчетности.

# **Описание программы**

Данная программа реализована на языке Python 3.12.7 и использует несколько популярных библиотек, таких как cv2 для работы с изображениями, numpy для выполнения математических операций, pandas для обработки и анализа данных, а также tkinter для создания графического интерфейса пользователя. Основная цель программы — обработка астрономических изображений, выделение объектов на изображениях и сохранение результатов анализа в формате CSV.

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Функционал | Входные данные | Возвращаемое значение |
| classify\_object() | Классифицирует объект на основе яркости и размера (звезда, галактика, суперновая или неизвестно). | brightness — яркость объекта (тип float), size — площадь объекта (тип float). | str |
| process\_image\_section() | Обрабатывает часть изображения: ищет контуры, вычисляет параметры объектов (центр, радиус, яркость, площадь) и классифицирует их. | image\_section — часть изображения (тип np.ndarray), offset\_x — смещение по оси X (тип int), offset\_y — смещение по оси Y (тип int). | list |
| draw\_circles() | Рисует окружности вокруг объектов на изображении на основе их классификации. | full\_image — изображение, на которое нужно наложить окружности (тип np.ndarray), object\_stats — список статистики объектов (тип list). | None |
| process\_image() | Загружает изображение, делит его на квадранты, обрабатывает каждую часть параллельно, объединяет результаты и сохраняет изображение с окружностями. | image\_path — путь к изображению (тип str), output\_folder — путь к папке для сохранения результатов (тип str). | list |
| process\_image\_quadrant() | Обрабатывает один квадрант изображения, вызывая process\_image\_section. | quadrants — кортеж с изображением (часть изображения) и смещениями по осям X и Y (тип tuple). | list |
| parallel\_process\_images() | Параллельно обрабатывает список изображений, собирая статистику объектов в единый DataFrame. | image\_paths — список путей к изображениям (тип list[str]), output\_folder — путь к папке для сохранения результатов (тип str), num\_workers — количество рабочих процессов (тип int). | pd.Dataframe |

# **Рекомендации для пользователя**

Запуск программы: Для начала работы откройте файл gui.py, чтобы запустить графический интерфейс.

Выбор папок: Укажите папку, содержащую изображения для обработки. Выберите папку, куда будут сохранены обработанные изображения и XML-файл с результатами.

Запуск обработки: Нажмите на кнопку "Запустить обработку", чтобы начать анализ.Дождитесь завершения обработки и появления сообщения об успешном завершении.

Просмотр результатов:Все обработанные изображения будут сохранены с наложенными цветными контурами объектов. Результаты анализа объектов, включая их координаты, яркость и классификацию, будут сохранены в XML-файле.

# **Рекомендации для программиста**

Необходимые библиотеки. Для корректной работы программы необходимо установить следующие библиотеки Python:

1. OpenCV: для выполнения обработки изображений.
2. NumPy: для работы с массивами данных и числовыми вычислениями.
3. tkinter: для создания графического интерфейса. Обычно эта библиотека уже входит в стандартный набор Python, но в случае минимальной установки, убедитесь в ее наличии.

Структура проекта: Убедитесь, что все изображения находятся в одной папке и имеют поддерживаемые форматы (например, .jpg, .png). Проверьте, что у вас есть права на запись в папку, куда будут сохраняться обработанные изображения и XML-файл.

# **Исходный код программы**

<https://github.com/Ignatio27/Func_prog_corse_2_spbu.git>

# **Контрольный пример**

1. Запуск программы: Для запуска программы используйте файл gui**.py.** Программа запустит графический интерфейс (Рис. 1).

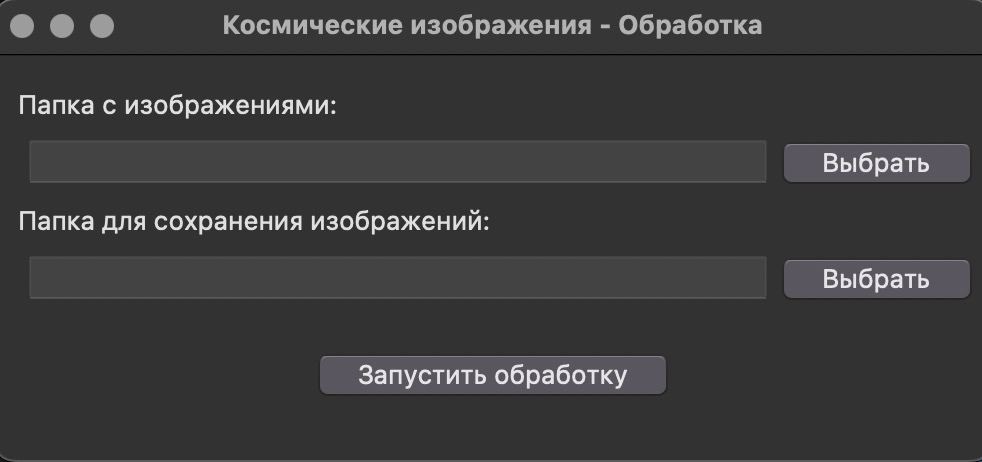


Рис 1. Графический интерфейс

2. Директория с изображениями: При нажатии на кнопку: “Выбрать папку” откроется проводник в котором будет необходимо выбрать директорию, из которой мы будем выбирать изображения для анализа.

3. Папка для сохранения: При нажатии на кнопку: “Выбрать папку” откроется проводник, в котором будет необходимо выбрать файл для сохранения получившейся после обработки изображений информации.

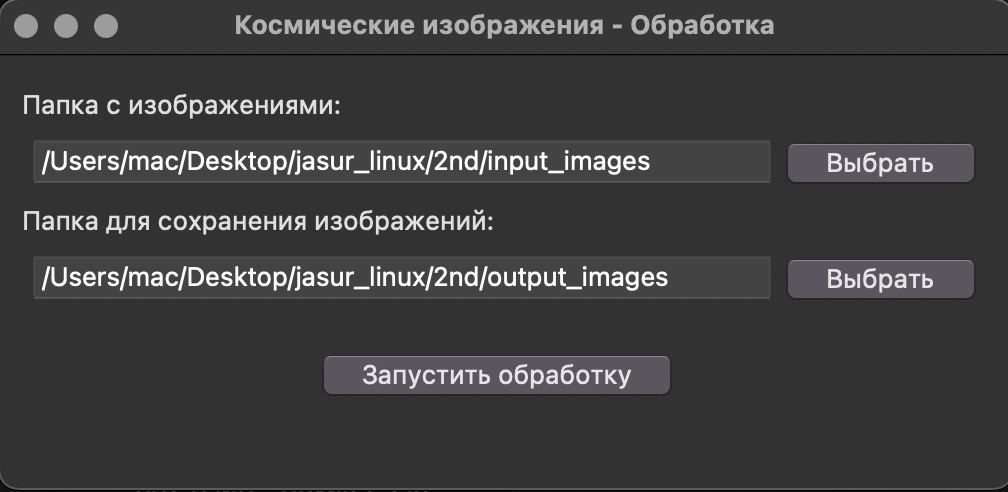


Рис 2 выбор папок

4. Обработка: При нажатии на кнопку: “Запустить обработку” запуститься процесс обработки картинок из выбранной директории, после окончания процесса будут сохранены изображения с обведёнными на них объектами.

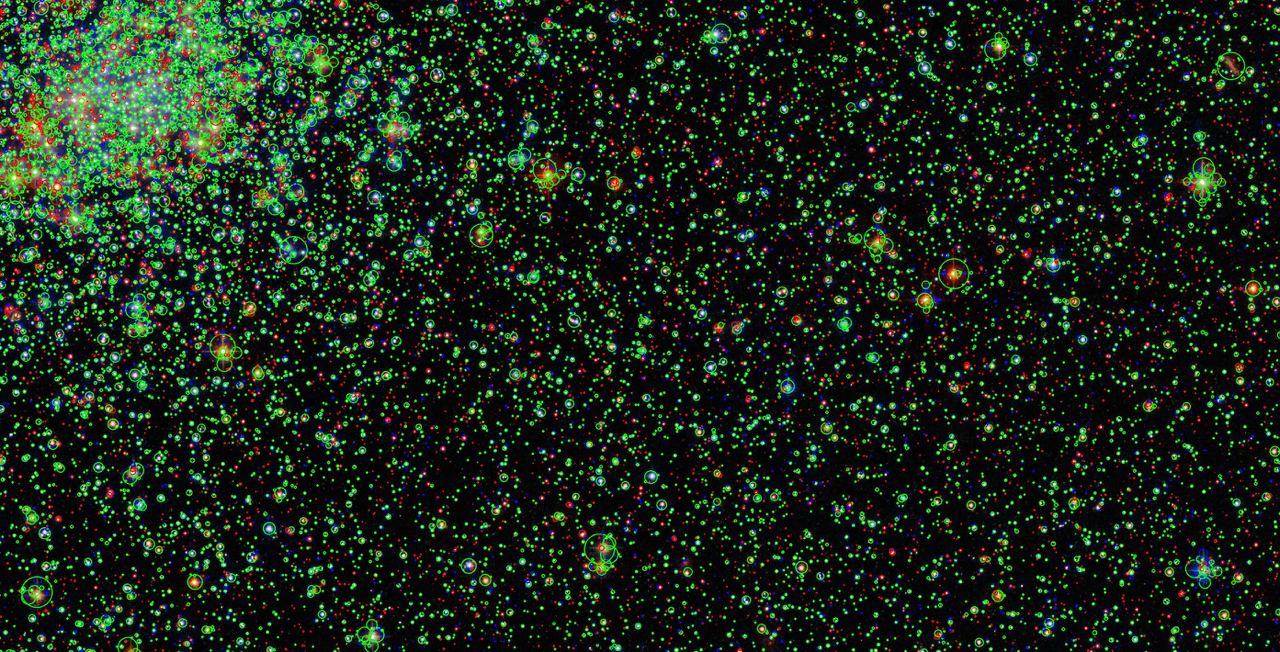


Рис 3 итог обработки

# **Вывод**

В ходе выполнения данной лабораторной работы была разработана программа для анализа астрономических изображений, направленная на классификацию объектов по их размерам и яркости. Для реализации обработки изображений использовались библиотеки OpenCV, NumPy и Pandas, что позволило эффективно выделять контуры, анализировать яркость и классифицировать объекты.

Программа предоставляет пользователю удобный интерфейс для выбора папок с изображениями и сохранения результатов в формате CSV. Графический интерфейс, построенный с использованием библиотеки tkinter, делает взаимодействие с программой простым и понятным, благодаря кнопкам для выбора папок и запуска обработки.

Для повышения производительности программы была внедрена многопроцессорная обработка данных, что значительно ускоряет обработку больших объемов изображений. Таким образом, была создана система, которая эффективно решает поставленные задачи, демонстрируя возможности работы с компьютерным зрением и обработкой данных.

# **Источники**

* **OpenCV** <https://docs.opencv.org/4.x/> дата обращения: (20.12.2024)
* **NumPy** <https://numpy.org/> дата обращения: (20.12.2024)
* **tkinter** [https://docs.python.org/3/library/tkinter.html](https://docs.python.org/3/library/tkinter.html%20)  дата обращения: (20.12.2024)
* **Pandas** https://pandas.pydata.org/ дата обращения: (20.12.2024)