**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**

**Бек В.А.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc179492837)
2. [Описание задачи (формализация задачи) 4](#_Toc179492838)
3. [Теоретическая часть 6](#_Toc179492839)

[**1.** **Обработка изображений** 6](#_Toc179492840)

[**2. Классификация космических объектов** 6](#_Toc179492841)

[**3.** **Параллельные вычисления** 7](#_Toc179492842)

[**4.** **Методы анализа объектов на изображениях** 8](#_Toc179492843)

[**5.** **Хранение и визуализация результатов** 8](#_Toc179492844)

1. [Основные шаги программы 9](#_Toc179492845)
2. [Описание программы 12](#_Toc179492846)
3. [Рекомендации пользователя 14](#_Toc179492847)

[**Примечания:** 14](#_Toc179492848)

1. [Рекомендации программиста 15](#_Toc179492849)
2. [Исходный код программы 15](#_Toc179492850)
3. [Контрольный пример 16](#_Toc179492851)
4. [Вывод 18](#_Toc179492852)

# Цель работы

Цель данной работы заключается в разработке эффективной программы для анализа больших объемов данных, получаемых с космического телескопа, с использованием методов параллельных вычислений. Существующие методы анализа изображений могут быть недостаточно быстрыми для обработки сотен тысяч космических изображений, содержащих миллионы пикселей. Поэтому в рамках данного проекта необходимо создать систему, способную одновременно обрабатывать множество изображений, обеспечивая тем самым ускорение анализа и сбор статистики о выявленных астрофизических объектах.

# Описание задачи (формализация задачи)

Задача анализа космических данных включает в себя следующие ключевые аспекты:

1. **Входные данные**: Набор изображений космических объектов, полученных телескопом, где каждое изображение представлено в виде матрицы пикселей. Изображения могут иметь разные форматы, такие как PNG или JPG.
2. **Обработка изображений**: Для каждого изображения необходимо выполнить несколько этапов обработки:
   * **Преобразование в градации серого**: Сначала изображения конвертируются в оттенки серого для упрощения анализа.
   * **Размытие**: Применение фильтра Гаусса для снижения шума в изображении.
   * **Бинаризация**: Преобразование размытого изображения в бинарное для выделения контуров объектов.
   * **Поиск контуров**: Выделение объектов на основе найденных контуров.
3. **Анализ объектов**: Каждый выделенный объект должен быть проанализирован для определения его характеристик:
   * **Определение площади объекта**: Площадь определяется с помощью функции **cv2.contourArea**.
   * **Определение координат центра**: Центр объекта вычисляется на основе его границ.
   * **Измерение яркости**: Яркость объекта рассчитывается как сумма значений пикселей в области, занимаемой контуром.
   * **Классификация объектов**: На основе площади и яркости объекта выполняется его классификация (например, как звезда, планета или яркая звезда) с помощью функции **classification**.
4. **Параллельная обработка**:
   * Для ускорения обработки необходимо реализовать параллельную обработку изображений, распределяя задачи по доступным ядрам процессора. Каждое изображение разбивается на части, которые обрабатываются в отдельных потоках или процессах.
   * Обеспечение безопасного взаимодействия между потоками или процессами для предотвращения гонок данных и других конфликтов.
5. **Сбор статистики**: Для каждого найденного объекта необходимо собрать статистические данные, включая:
   * Название изображения.
   * Индекс части изображения.
   * Координаты центра объекта.
   * Яркость.
   * Площадь.
   * Тип объекта.
6. **Вывод результатов**: Результаты анализа должны быть сохранены в Excel-файл, включающий информацию о всех найденных объектах. Необходимо также создать выходные изображения, на которых выделены обнаруженные объекты.

# Теоретическая часть

### **1.** **Обработка изображений**

Обработка изображений – это одна из ключевых областей компьютерного зрения, где используются методы анализа данных, такие как фильтрация, выделение контуров и сегментация. Для задачи анализа космических данных важными этапами являются:

* **Преобразование в градации серого**.

Изображения с цветовой информацией содержат избыточные данные для большинства задач анализа объектов. Преобразование в оттенки серого снижает количество информации и упрощает вычисления, сохраняя при этом необходимые данные для дальнейшей обработки.

* **Размытие с помощью фильтра Гаусса**.

Применение фильтра Гаусса для размытия помогает сгладить шумы на изображении, что особенно важно для космических снимков, так как шум может быть вызван множеством факторов (например, радиопомехи или дефекты сенсоров телескопов). Гауссово размытие улучшает качество обнаружения контуров объектов.

* **Бинаризация изображения**.

Это процесс преобразования градаций серого в чёрно-белое изображение (бинарное), где пиксели выше определённого порога (например, 180 из 255) считаются белыми, а ниже — чёрными. Этот этап позволяет лучше выделить объекты на фоне.

* **Поиск контуров**.

После бинаризации выделение контуров осуществляется с помощью таких алгоритмов, как cv2.findContours, который находит границы объектов, используя анализ изменений интенсивности пикселей. Это важный этап для определения формы и размеров объектов.

### **2. Классификация космических объектов**

Космические объекты на изображениях можно классифицировать на основе их физических характеристик, таких как яркость и площадь. В основе классификации могут лежать следующие принципы:

* **Яркость**. Величина яркости объекта в изображении может быть связана с его физическими характеристиками, такими как мощность излучения звезды или отражающая способность планеты.
* **Площадь**. Размеры объектов (вычисляемые по площади контура) могут быть использованы для разделения звёзд, планет или других космических тел.

Классификация в данной задаче является упрощённой и основана на анализе яркости и площади, что позволяет разделить объекты на такие категории, как:

* **Звезда**. Малый объект с высокой яркостью.
* **Планета**. Объект большего размера, но с более низкой яркостью.
* **Яркая звезда**. Особо яркий объект с высокой интенсивностью светового потока.

### **3.** **Параллельные вычисления**

Параллельные вычисления играют ключевую роль в ускорении обработки больших объёмов данных. В контексте обработки космических изображений это особенно актуально, так как одно изображение может содержать миллионы пикселей, а объём данных для анализа велик.

#### **Модель параллельных вычислений:**

* **Параллелизм задач (Task-level parallelism)**: В рамках этого подхода каждое изображение или его часть может обрабатываться в отдельном потоке или процессе. Например, изображение можно разделить на сегменты, и каждый сегмент будет анализироваться независимо.
* **Использование потоков или процессов**. В Python для параллельной обработки часто используются библиотеки:
  + Threading для многопоточных задач, которые могут выполнять несколько операций одновременно, но при этом разделяют память.
  + Multiprocessing, которая лучше подходит для задач, требующих раздельной обработки данных с использованием независимых процессов.

#### **Преимущества параллельной обработки:**

* Ускорение работы за счёт распределения задач по нескольким ядрам процессора.
* Возможность обработки больших объёмов данных в разумные сроки, что критически важно для анализа изображений с высокими разрешениями.

#### **Проблемы параллельной обработки:**

* **Гонки данных**. Потоки или процессы могут одновременно обращаться к одним и тем же данным, что приводит к некорректным результатам. Для предотвращения этого необходимо использовать механизмы синхронизации, такие как блокировки (locks) и очереди (queues).
* **Перегрузка процессора**. Чрезмерное создание потоков может привести к снижению производительности. Необходимо учитывать количество доступных ядер и оптимизировать распределение задач.

### **4.** **Методы анализа объектов на изображениях**

Методы выделения и анализа объектов на изображениях включают:

* **Методы выделения контуров**. Они позволяют определить границы объектов на изображении. Используются такие алгоритмы, как метод Канни или поиск контуров через градиентные изменения интенсивности.
* **Морфологические операции**. Применяются для улучшения формы объекта (например, расширение или сужение контуров) с целью устранения мелких дефектов или заполнения пропусков в выделенных областях.
* **Анализ признаков объектов**. После выделения контуров необходимо анализировать такие характеристики, как площадь, периметр и компактность объекта. Эти признаки могут использоваться для классификации объектов или определения их физических свойств.

### **5.** **Хранение и визуализация результатов**

Для хранения результатов анализа используются структурированные данные, которые могут быть представлены в виде таблиц с полями для каждого объекта (координаты, площадь, яркость и тип объекта).

# Основные шаги программы

1. **Загрузка и предварительная обработка изображения**
   * На первом этапе программа загружает исходное космическое изображение, которое будет анализироваться. Это может быть изображение звёздного неба, космического объекта или иного астрономического объекта. Для этого используется библиотека OpenCV.
   * Изображение преобразуется в градации серого с целью упрощения дальнейших вычислений, так как цветовая информация в большинстве случаев не является критичной для задачи обнаружения объектов.
   * Для улучшения качества изображения и устранения шумов, которые могут привести к ошибочному обнаружению мелких объектов, применяется фильтр Гаусса. Это помогает сгладить шумы и делает границы объектов более явными.
   * Далее проводится бинаризация изображения с использованием порогового значения, что позволяет разделить пиксели на объекты и фон. Все пиксели, которые выше порогового значения, становятся "белыми" (объекты), а те, что ниже, — "черными" (фон).
2. **Выделение контуров объектов**
   * Для обнаружения границ объектов используется алгоритм поиска контуров (cv2.findContours), который работает с бинарным изображением, где объекты уже выделены на фоне. Этот алгоритм находит все замкнутые контуры, которые считаются потенциальными объектами.
   * После нахождения контуров программа проводит сортировку контуров по площади, что позволяет отсеять слишком маленькие или слишком большие объекты, которые не соответствуют заданным критериям (например, шумы или артефакты).
   * Кроме того, возможна фильтрация по форме объектов. Например, звезды можно выделить как относительно круглые объекты, а более сложные формы могут быть исключены как нерелевантные.
3. **Анализ и классификация объектов**
   * После выделения контуров каждого объекта программа начинает их анализ. Для этого вычисляются различные параметры: площадь объекта, периметр, момент инерции и другие характеристики.
   * Для каждого объекта также вычисляется его яркость, что позволяет отличать более яркие звезды от тусклых. Это важно для классификации объектов.
   * Классификация проводится на основе предварительно заданных порогов и правил. Например, если объект имеет круглую форму и высокую яркость, он может быть классифицирован как яркая звезда. Если объект менее яркий, он может быть классифицирован как обычная звезда. В зависимости от сложности задачи можно добавить другие классы, такие как планеты или астероиды.
4. **Параллельная обработка**
   * Для ускорения анализа изображения, особенно если оно большое или содержит множество объектов, возможно использование параллельных вычислений. Для этого можно разбить изображение на сегменты и обрабатывать их независимо друг от друга.
   * Использование библиотек Threading или Multiprocessing позволит выполнять обработку каждого участка изображения параллельно, что существенно снизит время выполнения программы.
   * Параллельная обработка может применяться на этапе поиска контуров, анализа объектов или других ресурсозатратных операций.
5. **Хранение результатов анализа**
   * После классификации все данные о найденных объектах сохраняются в структуру данных, например, в виде таблицы. В эту таблицу включаются такие параметры, как координаты объекта, его площадь, яркость, форма, а также результат классификации.
   * Сформированные данные записываются в файл Excel для удобного последующего анализа и работы. Это позволяет пользователю просматривать результаты в удобной форме и использовать их для дальнейших исследований.
6. **Визуализация результатов**
   * На оригинальном изображении программа выделяет каждый найденный объект, нанося контуры или рамки вокруг них. Это позволяет визуально оценить правильность работы алгоритма и классификации объектов.
   * Каждый объект может быть выделен различными цветами в зависимости от его классификации.
   * Программа отображает обработанное изображение с отмеченными объектами на экране, что помогает проверить результаты визуально и убедиться в корректности работы алгоритмов.
7. **Завершение программы**
   * После выполнения всех шагов программа завершает работу, закрывая все открытые окна с изображениями и освобождая ресурсы.

# Описание программы

Программная реализация написана на языке **Python 3.12.7** с использованием следующих библиотек: **cv2** для работы с изображениями, **multiprocessing** для многопоточной обработки, **numpy** для математических операций, openpyxl для работы с Excel-файлами, и tkinter для создания графического интерфейса. Программа организована в модуле, который фокусируется на обработке астрономических изображений, выделении контуров объектов и сохранении данных в Excel-файлы.

В процессе разработки программы использовалось 15 функций, каждая из которых имеет четко определенное назначение:

Таблица 1. parallel\_processing\_space.py

|  |  |  |
| --- | --- | --- |
| Функция | Описание | Возвращаемое значение |
| create\_interface | Создание графического интерфейса с кнопками для выбора изображений, пути сохранения и начала анализа. | root |
| choosephotos | Открытие диалогового окна для выбора директории с изображениями для анализа. | None |
| savePath | Открытие диалогового окна для выбора директории сохранения Excel-файла с результатами анализа. | None |
| analyze | Основная функция анализа изображений. Вызывает обработку всех изображений, сохранение результатов и уведомление пользователя о завершении процесса. | None |
| processAllphotos | Обработка всех изображений из указанной директории. | None |
| processphoto | Разбиение изображения на части, анализ каждой части изображения, объединение данных о всех обнаруженных объектах. | allObjectsData |
| splitphoto | Разделение исходного изображения на части для параллельной обработки. | Список частей изображения |
| analyzePhotoPart | Анализ отдельной части изображения: выделение контуров, вычисление площади, яркости объектов и их классификация. | Данные об объектах |
| classification | Классификация объектов по площади и яркости на основе заданных критериев (звезда, яркая звезда, планета). | Тип объекта |
| savephotoPart | Сохранение частей изображений с выделенными контурами объектов в отдельные файлы. | None |
| save | Сохранение всех результатов анализа в Excel-файл с возможностью настройки ширины столбцов. | None |
| create\_interface | Создание основного окна интерфейса и элементов управления. | None |
| choosephotos | Выбор директории для анализа изображений. | None |
| savePath | Выбор директории для сохранения Excel-файла. | None |
| analyze | Основная логика анализа изображений, вызов соответствующих функций для обработки фотографий и сохранения результатов. | None |

# Рекомендации пользователя

1. **Запуск программы:**
   * Дважды кликните на исполняемый файл программы, чтобы открыть графический интерфейс.
2. **Выбор изображений для анализа:**
   * Нажмите на кнопку **"Выбрать изображения"**.
   * В открывшемся окне выберите папку, где хранятся изображения для анализа. Программа поддерживает форматы **.jpg** и **.png**.
   * После выбора папки программа отобразит путь к выбранным изображениям в строке **"Выбраны изображения"**.
3. **Указание пути для сохранения результатов:**
   * Нажмите на кнопку **"Сохранить путь"**.
   * В диалоговом окне выберите папку, куда вы хотите сохранить итоговый файл с результатами в формате Excel.
   * Программа отобразит путь сохранения в строке **"Сохранить в"**.
4. **Начало анализа изображений:**
   * Нажмите кнопку **"Анализ"**. Программа начнет обрабатывать изображения, анализируя каждый фрагмент, выделяя объекты, такие как звезды и планеты.
   * Процесс обработки может занять некоторое время, в зависимости от количества и размера изображений.
5. **Завершение анализа:**
   * По окончании анализа появится всплывающее окно с уведомлением о том, что результаты успешно сохранены в выбранный Excel-файл.
6. **Просмотр результатов:**
   * Откройте указанный файл Excel (например, с именем statistic.xlsx) в выбранной папке. В таблице будут указаны:
     + Название изображения.
     + Номер фрагмента.
     + Координаты объектов.
     + Яркость объекта.
     + Площадь объекта.
     + Тип объекта (звезда, яркая звезда, планета).

## **Примечания:**

* Все изображения после анализа также сохраняются в папке **photo\_parts**, где каждый фрагмент изображения содержит выделенные объекты (звезды, планеты и т.д.).

# Рекомендации программиста

**Необходимые библиотеки:** Для корректной работы программы вам понадобятся следующие библиотеки Python:

* **OpenCV**: Для обработки изображений.

****

* **NumPy**: Для работы с массивами и числовыми данными.

****

* **OpenPyXL**: Для работы с файлами Excel.



* **tkinter**: Библиотека для создания графического интерфейса. Обычно она уже включена в стандартную библиотеку Python, но если вы используете минимальную установку, убедитесь, что она доступна.

**Структура проекта:**

* Убедитесь, что ваши изображения находятся в одной папке и имеют поддерживаемые форматы (.jpg, .png).
* Создайте папку для хранения выходных данных (например, photo\_parts), если она не создается автоматически.
* Проверьте, что у вас есть разрешения на запись в выбранной папке для сохранения Excel-файла и обработанных изображений.

# Исходный код программы

<https://github.com/Kliooo/Functional-programming>

# Контрольный пример

1. Запуск программы: Для запуска программы используйте файл **parallel\_processing\_space.py.** Программа запустит графический интерфейс (Рис. 1), в котором можно будет выбрать “Выбрать изображения” или “Сохранить путь” и “Анализ”.

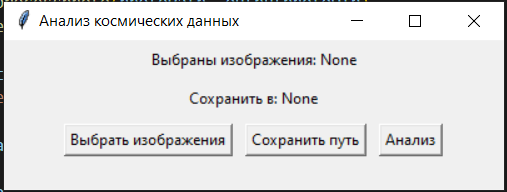


Рис 1. Графический интерфейс

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

3. Сохранить путь: При нажатии на кнопку: “ Сохранить путь” откроется проводник в котором будет необходимо выбрать папку в которую мы впоследствии будем сохранять получившиеся результаты.

4. Анализ: При нажатии на кнопку: “Анализ” запуститься процесс обработки картинок из выбранной папки, создастся папка: photo\_parts в которой будут части изображения и обведённые на них объекты, а также создастся xlsx файл в котором будет также представлена информация об этих объектах.

5. Итог: После завершения анализа появится окно(Рис. 2), уведомляющее о том, куда сохранены результаты.

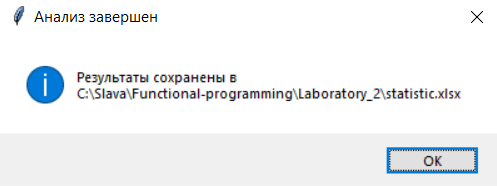


Рис 2. Окно завершения

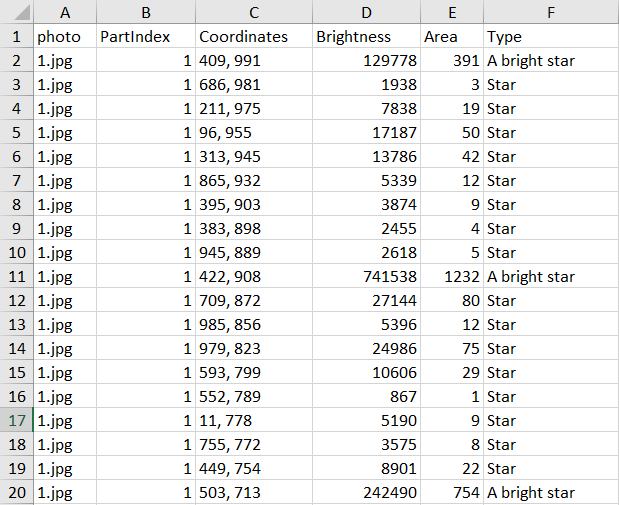


Рис 3. Файл statistic.xlsx

# Вывод

В рамках данной лабораторной работы была разработана программа для анализа изображений с целью классификации объектов на основе их размеров и яркости. Используя библиотеки OpenCV, NumPy и OpenPyXL, была реализована функция обработки изображений, которая включает в себя распознавание контуров, анализ яркости и классификацию объектов.

Программа позволяет пользователю выбрать папку с изображениями и сохранить результаты анализа в формате Excel. Графический интерфейс, созданный с использованием библиотеки tkinter, упрощает взаимодействие с пользователем, предоставляя интуитивно понятные кнопки для выбора изображений и запуска анализа.

Также была внедрена многопроцессорная обработка для повышения производительности программы, что позволяет обрабатывать большие объемы данных быстрее и эффективнее. В результате получена система, которая не только выполняет необходимые функции, но и демонстрирует принципы работы с библиотеками для компьютерного зрения и обработки данных.

Полученные результаты показывают, что разработанная программа эффективно справляется с поставленными задачами, и открывает возможности для дальнейших исследований и улучшений в области анализа изображений и автоматизации обработки данных.

# Источники

* **OpenCV**

<https://docs.opencv.org/4.x/>

*дата обращения: (09.10.2024)*

* **NumPy**

[*https://numpy.org/*](https://numpy.org/)

*дата обращения: (09.10.2024)*

* **tkinter**

[*https://docs.python.org/3/library/tkinter.html*](https://docs.python.org/3/library/tkinter.html%20)

*дата обращения: (09.10.2024)*

* **OpenPyXL**

[https://openpyxl.readthedocs.io/en/stable/](https://openpyxl.readthedocs.io/en/stable/%20)

*дата обращения: (09.10.2024)*