Отчет по проекту на тему:

Поиск связных компонентов в трехмерном пространстве

Выполнял: Киселев Егор

25.05.2022

План:

1. Постановка задачи
2. Описание входных и выходных данных
3. Описание используемого алгоритма
4. Детали реализации
5. Планируемые результаты
6. Полученные результаты

1. Постановка задачи

В трехмерном пространстве, разделенном на кубические блоки, имеется связная область с бинарной маркировкой, то есть каждый блок отмечается 0 или 1. Требуется выделить и промаркировать уникальными идентификаторами связные компоненты, состоящие из блоков с маркировкой 1. Другими словами, нужно выделить изолированные наборы блоков, где каждый блок касается блока с той же маркировкой хотя бы одной гранью. В результате в исходном пространстве должна получится новая маркировка, где все блоки из одной связной компоненты одинаково промаркированы, а каждая компонента имеет свой уникальный маркер.

2. Описание входных и выходных данных

На вход программа может принимать трехмерный массив в текстовом формате. Где пространство размера $n \times m \times k$ описывается в виде строки с указанием размерностей в формате '$n\ m\ k$' и n двумерных массивов, состоящих из m строк, каждая из которых содержит k маркеров 0 или 1 разделенных пробелами.

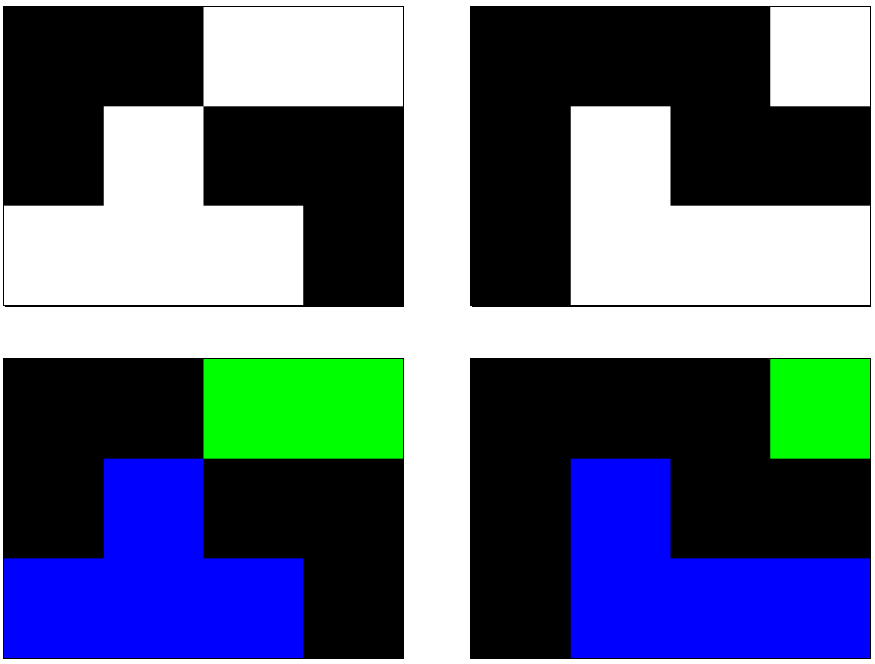
Например, пространство размера $2 \times 3 \times 4$ можно описать следующим образом:

```
2 3 4
0 0 1 1
0 1 0 0
1 1 1 0

0 0 0 1
0 1 0 0
0 1 1 1
```

Выходные данные описываются аналогичным образом, с изменением единиц на уникальные идентификаторы связных компонентов. Например, для примера выше результат будет следующим:

```
2 3 4
0 0 2 2
0 3 0 0
3 3 3 0

0 0 0 2
0 3 0 0
0 3 3 3
```

Также предусмотрено описание данных в виде n изображений в растровом формате размера $m \times k$, представляющих собой срезы трехмерного пространства, где каждый пиксель отвечает за один блок. На входе изображения двухцветные: черный - 0, белый - 1. На выходе цвет каждой связной компоненты уникален. Пример описания того же пространства, что и выше:



3. Описание используемого алгоритма

Для поиска связных компонентов будем использовать рекурсивынй алгоритм. Принцип его работы следующий:

1. Начинаем полный перебор всех блоков, поддерживая идентификатор текущей выделяемой компоненты, в качестве начального значения которого

возьмем 2, чтобы различать еще не посещенные блоки с маркировкой 1 от блоков уже выделенной первой компоненты.

2. Встретив блок с маркировкой 1, совершаем рекурсивный вызов для всех смежных блоков с той же маркировкой, помечая каждый посещенный блок идентификатором текущей компоненты.
3. После выхода из рекурсии вся текущая компонента будет промаркирована, поэтому можно инкрементировать текущий идентификатор и двигаться дальше до следующего непосещенного блока.

Преимущество данного алгоритма в его простоте и наглядности. Основной недостаток - ожидемо долгое время работы при больших размерах связных компонентов, в связи с большой глубиной рекурсии. Однако, поскольку обход компоненты по сути является поиском в глубину, одной из возможных оптимизаций является реализация его немного менее наглядной нерекурсивной версии, например, с помощью очереди.

4. Детали реализации

Для хранения идентификаторов блоков трехмерного пространства будем использовать реализованную нами библиотеку *M3i*.

Для чтения и сохранения изображений, воспользуемся библиотекой *stb*, а конктретно нам понадобится *stb_image* и *stb_image_write.*

5. Планируемые результаты

Программа, принимающая на вход пространство, описанное в одном из предложенных форматов, и возвращающая в том же формате аналогичное пространство с промаркированными связными компонентами.

Для отладки и визуализации работы алгоритма, в процессе поиска будет производиться вывод текущей маркировки пространства на каждом шаге в текстовом формате или в виде изображений. В результате можно будет наглядно наблюдать порядок обхода и выделения связных компонентов на любом этапе работы алгоритма.

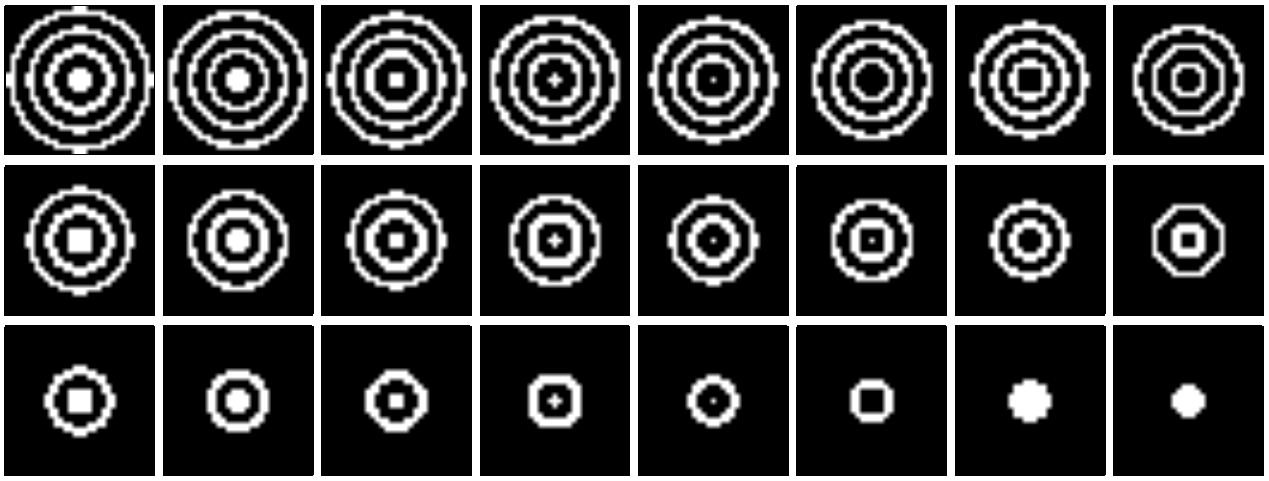
Также необходимо проверить как размер и количество связных компонентов влияет на время работы алгоритма.

6. Полученные результаты

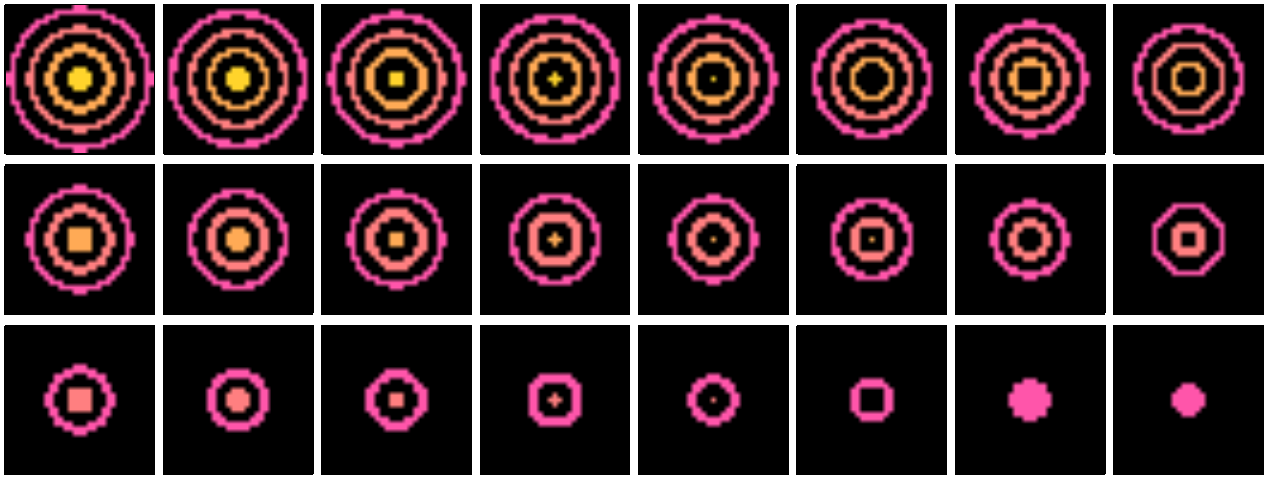
Реализован класс *CCLSolver,* способный считывать ввод и сохранять вывод в обоих форматах и решающий задачу рекурсивным алгоритмом. Код здесь

Вот результат работы алгоритма на пространстве размера $24 \times 31 \times 31$. Здесь связные компоненты представляют собой вложенные друг в друга полуэллипсоиды.

Входные данные:

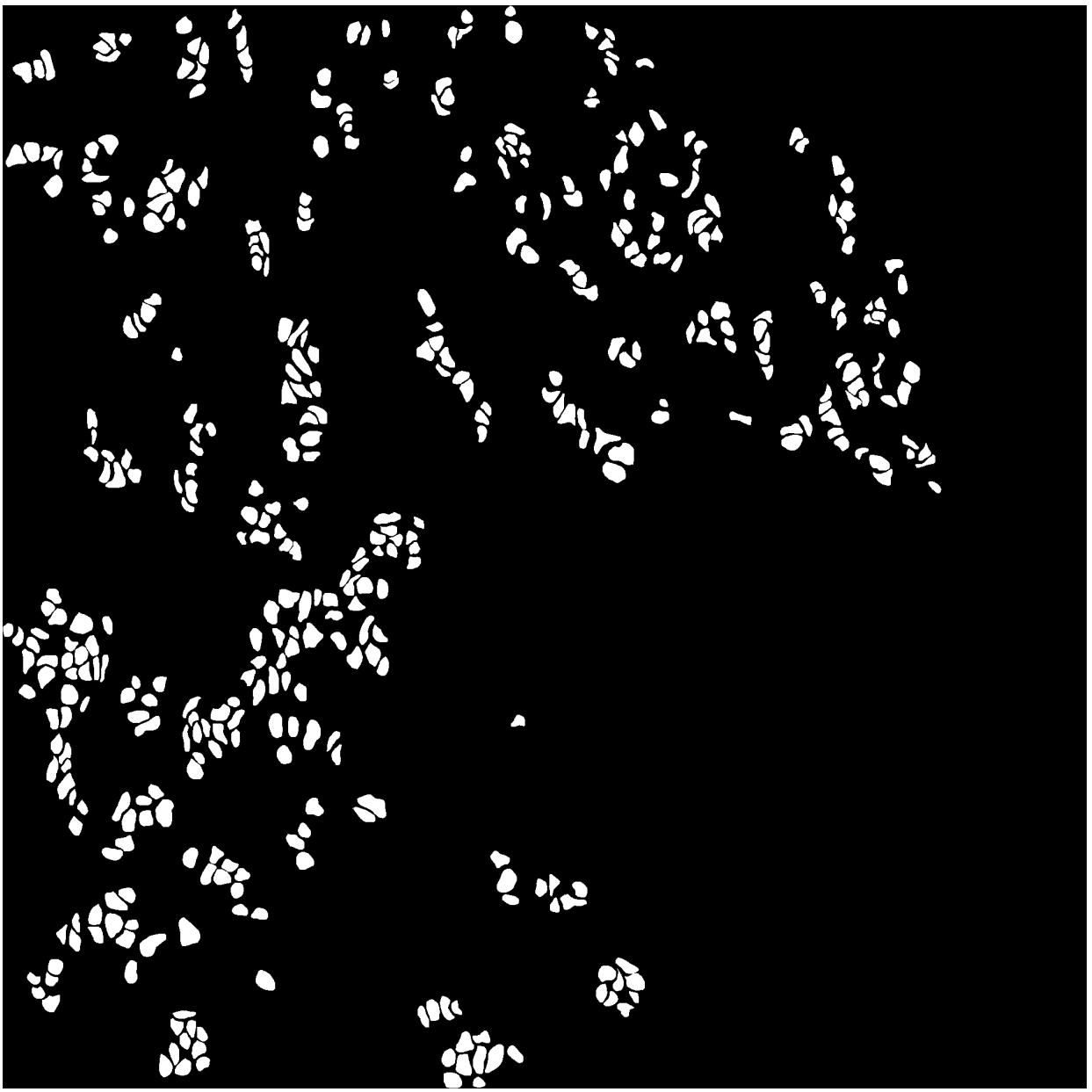


Выходные данные:

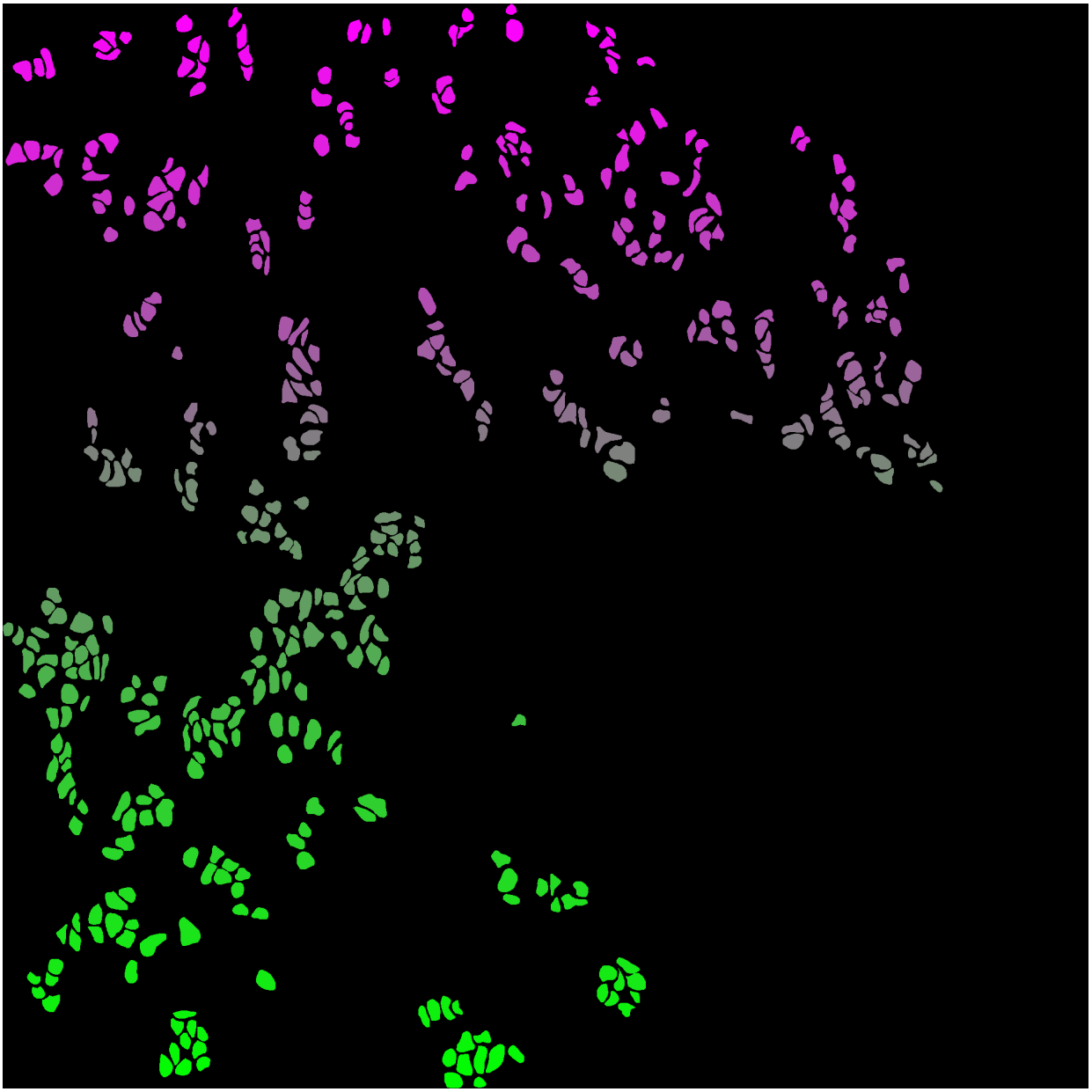


Вот более наглядный пример в двумерном пространстве размера $1 \times 2000 \times 2000$. Здесь уже больше связных компонентов. (источник изображения)

Входные данные:



Выходные данные:



Рассмотрим некоторые промежуточные этапы на примере следующего пространства размера $7 \times 50 \times 50$ с тремя связными компонентами: