ОБ ОДНОМ ИЗ АЛГОРИТМОВ ПОИСКА ОБЪЕКТОВ ПО БИОМЕДИЦИНСКИМ ДАННЫМ

А.Н. Решетников

Нижегородский госуниверситет им. Н.И. Лобачевского
E-mail: ANReshetnikov@mail.ru

Постановка задачи

На вход рассматриваемому алгоритму подаются данные компьютерной томографии или электронной микроскопии (всюду далее – CT- и EM-данные соответственно), представляющие собой трёхмерный массив вокселов. Каждому из элементов данного массива приписано определённое целочисленное значение плотности $density[i, j, k] \in Z$. При этом известны размеры каждого воксела вдоль осей **X**, **Y** и **Z** $ScaleX$, $ScaleY$ и $ScaleZ$ в некоторых метрических единицах. Таким образом, объём 1 воксела исходных данных равен $ScaleX \cdot ScaleY \cdot ScaleZ$ кубических единиц. Отметим, что данный 3D-массив можно интерпретировать двумя способами: как трёхмерное изображение и как последовательность полутоновых 2D-изображений (слоёв) одинаковой ширины и высоты.

По представленным данным требуется найти всевозможные объекты, при отсутствии какой бы то ни было априорной информации о них. Данная задача имеет важное практическое значение как один из промежуточных этапов в трёхмерной реконструкции объектов по таким данным и их последующем анализе конечным пользователем.

Многие из алгоритмов, решающие поставленную задачу, включают в себя 2 этапа:

1) сегментацию исходных данных для получения деталей искомых объектов или их начальных приближений;
2) обнаружение самостоятельных объектов по найденным сегментам.

Сегментация исходных данных

Учитывая специфику исходных биомедицинских данных (воксельная модель представления и скалярное поле значений плотности), для решения задачи сегментации использовался ***метод k-средних*** (или ***центров тяжести***) [1].

Для более детального представления сегментов, относящихся к разным объектам, предлагается использовать один из возможных нисходящих методов иерархической сегментации. Его суть состоит в том, что после сегментации исходных данных по значениям плотности можно ввести в рассмотрение сегменты более высокого уровня детализации, а именно, ***сегменты по расстоянию*** между вокселами и ***связные области*** (в пространственном смысле). Указанные сегменты в общем случае образуют иерархическую структуру, изображённую на рис. 1. При этом сегментацию по расстоянию можно осуществить итеративным алгоритмом, основанным на методе *k*-средних, рассматривая в качестве объектов пары индексов $\{(i, j)\}$, описывающих положение вокселов на одном и том же слое. В качестве меры рассогласования между вокселами x_1 и x_2 используется обычное евклидово расстояние, порождённое парами индексов. Связные области являются самым высоким уровнем детализации исходных данных. Для их поиска ис-

пользуется итеративный алгоритм разметки областей [2] в рамках конкретного сегмента по признаку 8-связности пикселов.

В случае послойной сегментации исходных данных (каждого слоя как 2D-изображения в отдельности) для дальнейшей идентификации объектов необходимо определить, какие сегменты на разных изображениях относятся к одним и тем же объектам. Для того, чтобы судить об этом, необходимо уметь сравнивать любые 2 сегмента друг с другом.

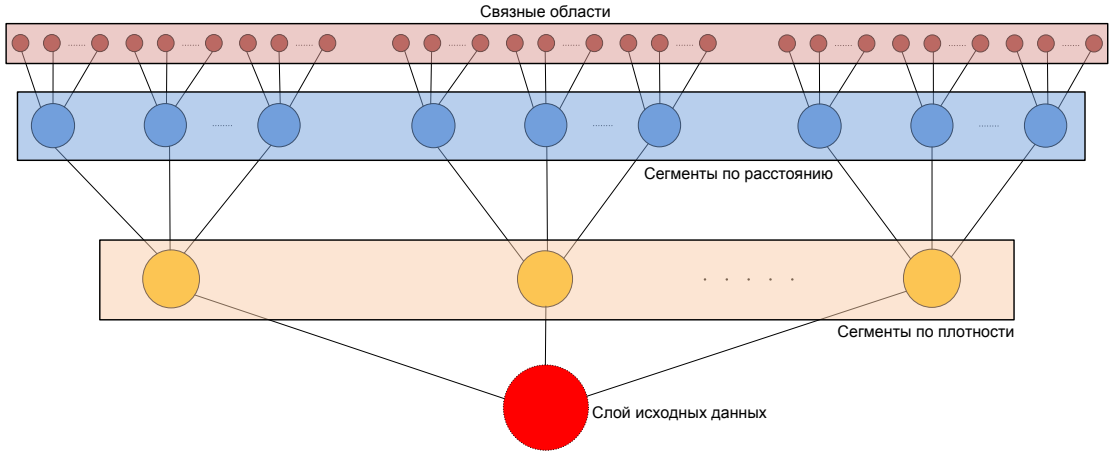

Рис. 1. Иерархическая структура сегментов после декомпозиции

Для построения дескриптора сегмента использовались следующие его атрибуты:

1) среднее значение плотности: $\bar{\mu} = \frac{1}{|S|}\sum_{i \in S} \mu_i$, (1)

где μ_i – значение плотности воксела с индексом i;

2) среднеквадратическое отклонение плотности: $\sigma_\mu = \frac{1}{|S|}\sum_{i \in S} \left(\mu_i - \bar{\mu}\right)^2$; (2)

3) геометрический центр масс $\left(\bar{x}, \bar{y}\right)$: $\bar{x} = \frac{1}{|S|}\sum_{i \in S} x_i$, $\bar{y} = \frac{1}{|S|}\sum_{i \in S} y_i$, (3)

где $\left(x_i, y_i\right)$ – координаты воксела с индексом i;

4) дисперсии координат вокселов (X и Y):

$$\sigma_X = \frac{1}{|S|}\sum_{i \in S} \left(x_i - \bar{x}\right)^2, \quad \sigma_Y = \frac{1}{|S|}\sum_{i \in S} \left(y_i - \bar{y}\right)^2; \quad (4)$$

5) объём сегмента в используемых кубических единицах:

$$V = |S| \cdot ScaleX \cdot ScaleY \cdot ScaleZ\ . \quad (5)$$

Атрибуты, вычисляемые по формулам (1) – (5), образуют признаковое описание сегмента S, или, по-другому, его ***дескриптор***, который далее будем обозначать $d(S)$.

Обнаружение объектов по найденным сегментам

Если исходные данные рассматривать как 3D-изображение, то результатом их сегментации является разбиение исходного множества вокселов на подмножества, соответствующие различным искомым объектам. Однако в том случае, когда применяется 2D-сегментация каждого слоя данных по отдельности, для идентификации объектов необходимо различить, какие сегменты на разных слоях относятся к одному и тому же или разным объектам. После этого для каждого объекта можно построить множество

точек, состоящее из соответствующих сегментов. Рассмотрим более детально метод, позволяющий решить данную проблему.

С помощью дескрипторов, которые ставятся в соответствие сегментам по формулам (1) – (5), можно определить степень рассогласования между сегментами s_1 и s_2 как расстояние ***Махаланобиса*** между их дескрипторами $d(s_1)$ и $d(s_2)$:

$$f(s_1, s_2) = \sqrt{\sum_{i=1}^{n} \frac{(d_i(s_1) - d_i(s_2))^2}{\sigma_i^2}}, \quad (6)$$

где

n – количество рассматриваемых атрибутов сегмента,

d_i – значение i - ого атрибута,

σ_i – среднеквадратическое отклонение i -го атрибута.

Определение. Последовательность сегментов на соседних слоях $(s_{i_{k_0}}, s_{i_{k_0+1}}, \ldots, s_{i_{k_0+K-1}})$, где i_j – индекс сегмента на j - м слое, k_0 – начальный слой, а K – количество слоёв, назовём ***соответствием*** между данными сегментами.

Таким образом, задача идентификации объектов сводится к поиску всевозможных соответствий между сегментами, расположенными на разных слоях исходных данных. Поиск таких соответствий можно рассматривать как многошаговый ***динамический процесс***, в котором каждый момент времени определяется номером соответствующего слоя данных, состояниями являются сегменты, распределённые по слоям, а управлениями – переходы с сегментов текущего слоя на соседний. Следует отметить, что множество финальных состояний совпадает с множеством всех предварительно найденных 2D-сегментов, поскольку объект может начинаться и заканчиваться на любом слое исходных данных.

В процессе поиска соответствий рассматриваются сегменты самого верхнего уровня детализации, то есть листья иерархической структуры, описанной в разделе «Сегментация исходных данных». Все сегменты делятся на ***занятые*** (те, которые уже включены в одно из найденных соответствий) и ***свободные***. В самом начале работы алгоритма все сегменты помечаются как свободные. Как только сегмент оказался включён в какое-либо соответствие, он получает статус «занят». Поиск очередного соответствия начинается с первого свободного сегмента на слое с наименьшим индексом. Для того чтобы не рассматривать слишком маленькие и/или большие детали, полученные в результате иерархической декомпозиции сегментов, можно ввести ограничения на их объём *V*, который должен принадлежать диапазону $[V_{\min}; V_{\max}]$.

Определим стоимость перехода с текущего слоя на соседний как рассогласование между дескрипторами соответствующих сегментов (состояний), тогда значение функции $f(s_1, s_2)$ вычисляется по формуле (6), где s_1 – сегмент текущего слоя, а s_2 – сегмент соседнего слоя.

Тогда критерием качества найденного соответствия можно считать суммарное рассогласование между всеми соседними сегментами, образующими соответствие:

$$F = \sum_{k=k_0}^{k_0+K-2} f(s_{i_k}, s_{i_{k+1}}). \quad (7)$$

Таким образом, поиск соответствий между сегментами сводится к решению задачи минимизации функции F при перемещении от начального слоя к конечному по соответствующим сегментам (рис. 2).

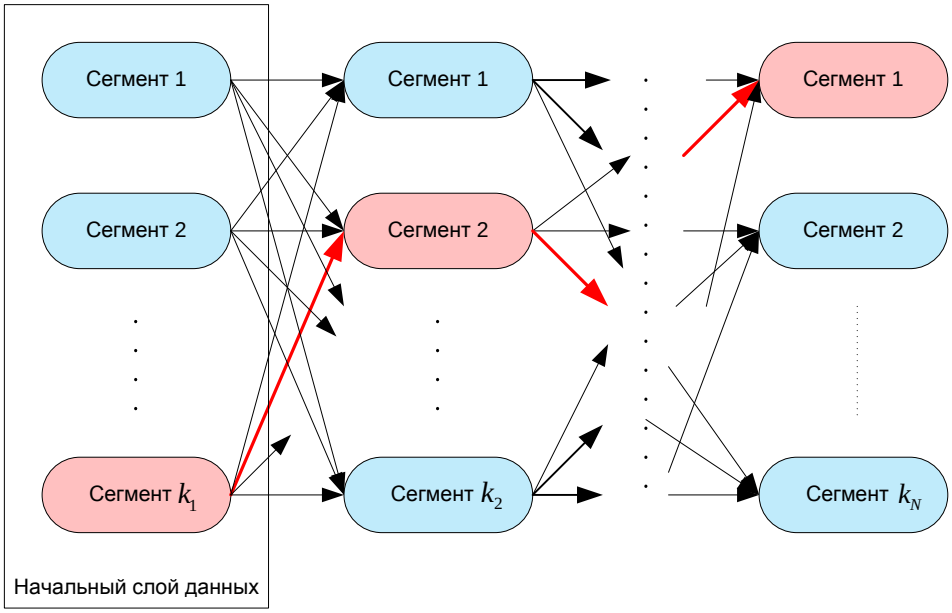

Рис. 2. Схема поиска соответствий между 2D-сегментами

Поскольку функция F аддитивна по промежутку слоёв, на которых расположены сегменты, и удовлетворяет принципу Беллмана в форме достаточного условия [3], для решения рассматриваемой задачи можно применить алгоритм пошаговой минимизации этой функции, основанный на методе рекуррентных соотношений Беллмана [3].

Приведём более подробное описание данного алгоритма применительно к рассматриваемой задаче, используя следующие обозначения:

s – массив сегментов,
IsSegmentUsed – булевы переменные (флаги) использования сегментов,
layer_index – индекс текущего слоя данных (служебная переменная),
Disparity – текущее значение минимальной стоимости переходов по соответствию,
N – общее количество слоёв исходных данных.

Шаг 1. Полагаем для всех сегментов **IsSegmentUsed = false**.
Шаг 2. В цикле по **I** от 0 до **N-1** и по сегментам **s[I][J]**, принадлежащим слою **I**, если ***IsSegmentUsed[I][J] == false***, выполняем ***Шаг 3 – Шаг 5***.
Шаг 3. Полагаем **layer_index = I, Disparity = 0**.
Шаг 4. Создаём новое соответствие и добавляем в него сегмент с индексом **J** на слое **layer_index**.
Шаг 5. Пока **layer_index<N**, выполняем ***Шаг 6 – Шаг 10***. После этого переходим на ***Шаг 2***.
Шаг 6. В цикле по **K** по сегментам слоя **layer_index + 1** выбираем тот, объём которого принадлежит заданному диапазону $[V_{\min}; V_{\max}]$ и который доставляет минимум величины **Disparity + *F(s[layer_index][J], s[layer_index+1][K])*.** Обозначим его **L**.
Шаг 7. Добавляем сегмент с индексом **L** в созданное соответствие.
Шаг 8. Полагаем **Disparity = Disparity + F(s[layer_index][J], s[layer_index+1][L])**.
Шаг 9. IsSegmentUsed[layer_index+1][L] = true.
Шаг 10. Увеличиваем **layer_index** на 1 и переходим на ***Шаг 5***.

Конец алгоритма

На рис. 3 в верхнем ряду показаны несколько последовательно расположенных слоёв одной из томограмм, а в нижнем – результаты сегментации каждого из слоёв. При этом одинаковыми цветами раскрашены сегменты, образующие один и тот же объект.

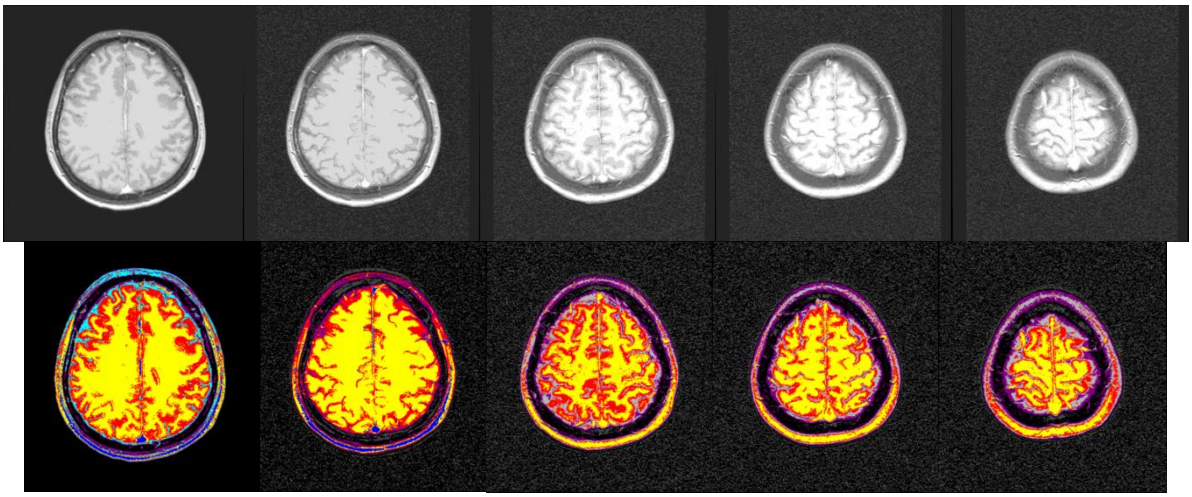

Рис. 3. Пример результатов поиска соответствий между 2D-сегментами

Повышение производительности вычислений

Вышеописанный алгоритм использовался при разработке программной системы Neocortex. В ходе этой разработки были реализованы как последовательная, так и параллельная (для GPU, с использованием технологии OpenCL) версии сегментации методом *k*-средних для решения поставленной задачи, которые используются для получения сегментов первого и второго уровней декомпозиции. При этом на каждом из этих двух этапов декомпозиции было получено ускорение от 4 до 10 раз в сравнении с последовательными реализациями [4].

Дальнейшее повышение производительности связано с разработкой параллельной схемы разметки связных областей (третий уровень декомпозиции сегментов) и алгоритма поиска соответствий между сегментами. Для этого целесообразно использовать возможности как центрального, так и графических процессоров.

Литература

1. Lloyd S. Least square quantization in PCM’s. Bell Telephone Laboratories Paper, 1957.
2. Форсайт Д. Компьютерное зрение. Современный подход / Д. Форсайт, Ж. Понс. – Пер. с англ. – М.: Издательский дом «Вильямс», 2004. – 928 с.
3. Беллман Р. Динамическое программирование. – М.: Изд-во иностранной литературы, 1960.
4. Решетников А.Н. Параллельная реализация сегментации компьютерных томограмм методом центров тяжести с использованием технологии OpenCL // Высокопроизводительные параллельные вычисления на кластерных системах: Доклад на XI Всероссийской конференции, Н. Новгород, 1-3 ноября 2011. – С. 258–261.

Thank you for evaluating AnyBizSoft PDF Splitter.

A watermark is added at the end of each output PDF file.

To remove the watermark, you need to purchase the software from

http://www.anypdftools.com/buy/buy-pdf-splitter.html