Победит #398

Сегментация пустот сплава WC-Co

Д.Г.Каграманян[1], Д.Ю.Камалова[2], Б.Б.Страумал[3], Л.Н.Щур[4]

13 апреля 2021 г.

[1]исследователь,dgkagramanyan@edu.hse.ru
[2]исследователь,dyukamalova@edu.hse.ru
[3]соруководитель,straumal@issp.ac.ru
[4]руководитель,levshchur@gmail.com

1 Аннотация

Для успешного решения задачи сегментации зерен сплава WC-Co, поставленной в работе [8], необходимо выделить пустоты микроструктуры (рис. 1) и затем составить граф из вершин, находящихся в углах пустот. При помощи полученного графа, который является периметром, можно посчитать статистическую информацию и выделить зерна сплава.

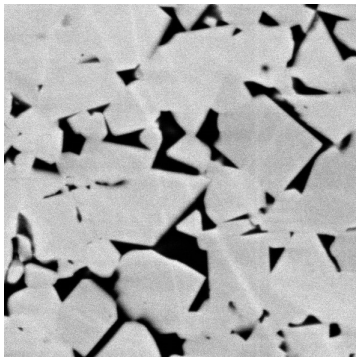

Рис. 1: Микроструктура сплава WC-Co. Белые участки - зерна, черные - связующее вещество(пустота)

2 Задача

Нужно на каждой пустоте разметить вершины углов и соединить их так, чтобы они образовали линию периметра. Иными словами нужно создать полигон на основе координат точек углов.

3 Предобработка данных

3.1 Первичная обработка изображений

Самой удачной комбинацией оказались последовательные шаги:

- комбинация левой и правой части исходного изображения
- медианный фильтр
- бинаризация Отсу
- карта градиентов

Итоговое изображение вычисляется по формуле 1.

$$\overline{\text{бинаризованное изображение}} + \text{карта градиентов} \tag{1}$$

Без использования в итоговом изображении градиентов плохо работает детектор углов, а без бинаризованного изображения теряются значения пикселей пустот и не получается выделить отдельную пустоту в индивидуальный регион.

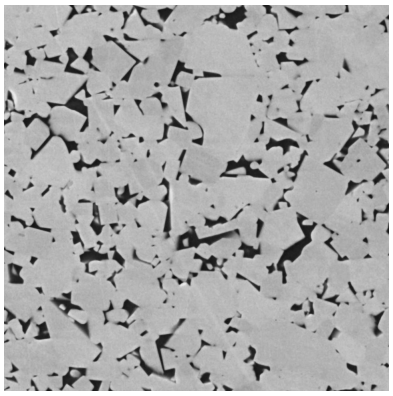

Рис. 2: Микроструктура после медианного фильтра

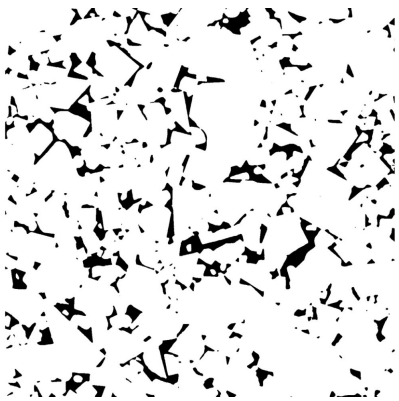

Рис. 3: Бинаризация Отсу

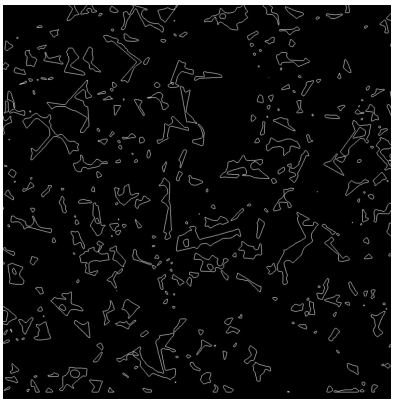

Рис. 4: Карта градиентов

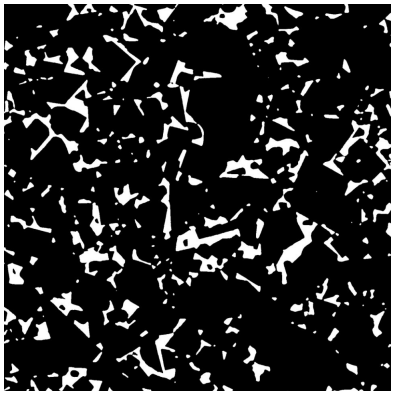

Рис. 5: Итоговое изображение

3.2 Разметка углов

После выделения пустот необходимо разметить вершины углов (рис. 6). Воспользуеся детектором Ши-Томаси [4].

Его принцип работы очень сложен и основан на матрицах свертки, анизитропии и аппроксимации границ при помощи гиперболы [7]. Результат работы алгоритма - массив координат неупорядоченных точек углов.

3.3 Разметка пустот

Основной принцип - по изображению проходит матрица свертки [1], которая выполняет следующие задачи:

- считает количество уникальных особенностей
- дает кажому пикселю свой класс, к которому он принадлежит

Как итог - каждый пиксель изображения имеет привязку к определенному классу пустоты (рис. 7).

3.4 Привязка углов к пустотам

Следующий этап обработки данных - привязка каждого угла к пустоте, на которой она находится. Это значит, что у каждой вершины может быть ровно одна привязка к пустоте, но у одной пустоты может быть много вершин.

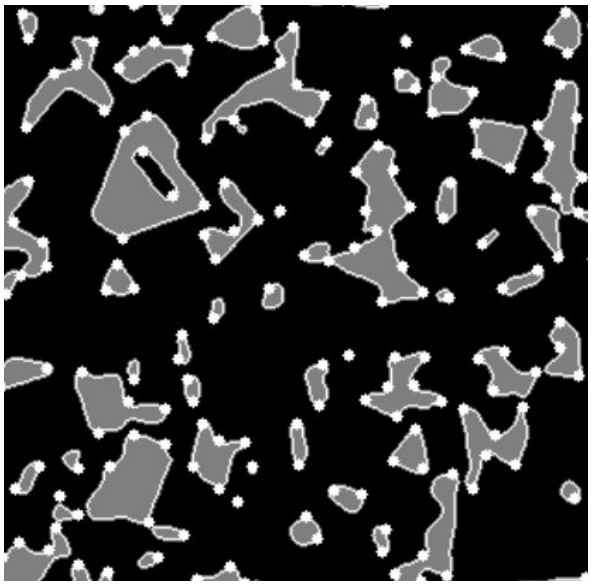

Рис. 6: Размеченные углы пустот (белые точки)

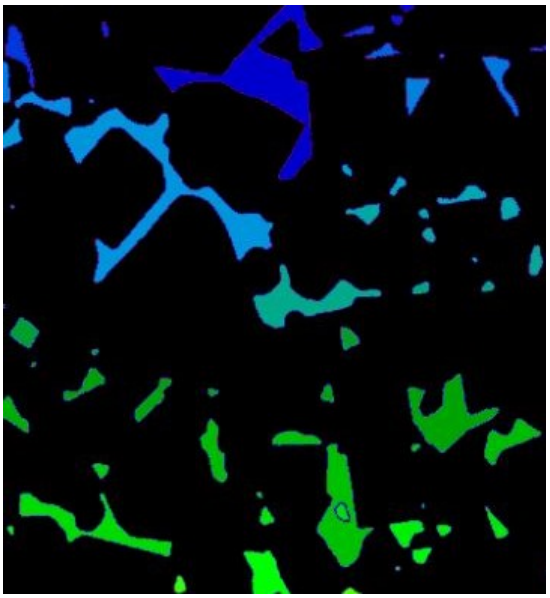

Рис. 7: Разметка пустот. Каждый цвет обозначает индивидуальный класс

4 Создание линий периметра

4.1 Обход всех вершин

Зададим метрики для ребер:

- mean(point1,point2,R) - среднее значение пикселей в прямоугольнике длины $\|point1 - point2\|$ и ширины 2*R
- dist(point1,point2)- расстояние $\|point1 - point2\|$ между двумя вершинами, образующими ребро

Основной принцип итеративного обхода - фиксирование одной вершины и обход всех остальных относительно первой. При успешном определении следующей точки, она фиксируется, а первая удаляется из списка доступных. Итерация продолжается до конца списка доступных вершин. Количество итераций можно строго описать в виде суммы переборов для каждой вершины региона (ур. 2). Сложность алгоритма O(N^2) .

$$(N-1) + (N-2) + .. = \sum_{i=1}^{N-1}(N-i) = \frac{1}{2}(N-1)N \tag{2}$$

где N - количество вершин(углов) в пустоте сплава.

Для вычисления метрики среднего значения 4.1 обозначим вектор ребра двух вершин $\overline{L} = (L_x; L_y) = (x_2 - x_1; y_2 - y_1)$. Получим координаты второй стороны (ширины) прямоугольника. Для этого пустим из точки $(x1; y1)$ вектор $\overline{\mathrm{A}} = (\mathrm{A}_x; \mathrm{A}_y) = (a_x - x_1; a_y - y_1)$.

Решим систему уравнений (ур. 3) и получим (ур. 4) координаты a_x и a_y второй точки вектора $\overline{A}$.

$$\begin{cases} (a_x - x_1)^2 + (a_y - y_1)^2 = R^2 \\ (a_x - x_1)L_x + (a_y - y_1)L_y = 0 \end{cases} \tag{3}$$

$$a_x = x_1 - \frac{RL_y}{\sqrt{{L_x}^2 + {L_y}^2}} \; a_y = y_1 + \frac{RL_x}{\sqrt{{L_x}^2 + {L_y}^2}} \tag{4}$$

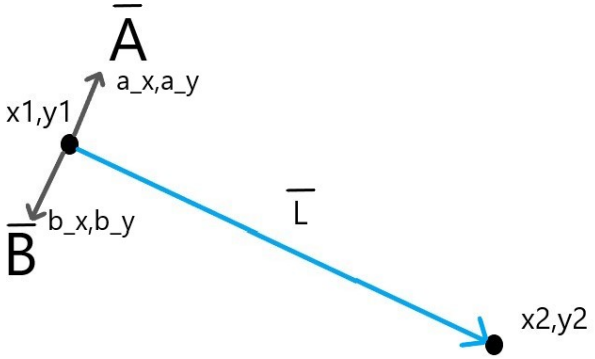

Рис. 8: Вектора , использвоанные при построении прямоугольника между точками

Для вектора $\overline{B}$ действия идентичны, как для вектора $\overline{A}$.

Получим координаты прямой между точками $(a_x; a_y)$ и $(b_x; b_y)$ при помощи алгоритма Брезенхема [9]. Пройдемся по каждой точке полученной прямой: построим из $(x^i; y^i)$ в $(x^i + L_x; y^i + L_y)$ новую прямую, которая будет являться i-тым слоем прямоугольника. Завершив проход по всем точкам, получим правильный и удобный для итерации массив точек прямоугольника. Алгоритм учитывает наклон фигуры.

4.1.1 Простое комбинирование метрик

Принцип работы - сложение или умножение двух метрик с весовыми коэффициентами. Расстояние между вершинами нормируется относительно максимальной или средней длины ребра в пустоте. Алгоритм итеративно ищет ту вершину, для которой суммарная метрика наименьшая.

Результат - успешная сегментация только на вершинах,находящихся на выпуклой части полигона (рис. 9). При наличии тонкой шейки или отклонения формы пустоты от эллипса происходит самопересечение (рис. 10). Изменение весовых коэффициентов дает прирост эффективности сегментации для одних регионов, но ухудшает для других.

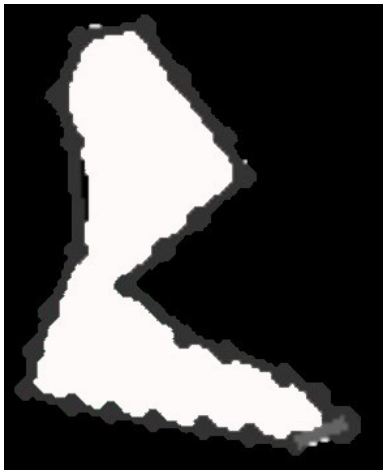

Рис. 9: Верное соединение вершин региона

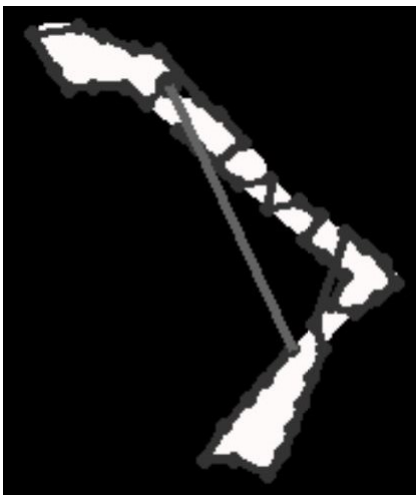

Рис. 10: Неверное соединение вершин региона

4.1.2 F-мера

Принцип работы - f мера (ур. 5) комбинация двух метрик A и B с различным значением коэффициента β, который задает вес одной метрики относительно другой. Расстояние между вершинами нормируется относительно максимальной или средней длины ребра в пустоте. Алгоритм итеративно ищет ту вершину, для которой f-мера наибольшая.

$$F_\beta = \frac{(1+\beta^2)AB}{\beta^2 A + B} \tag{5}$$

Результат - самопересечение,сегментации нет. При нормировке расстояние заведомо находится нулевом диапазоне значений из-за большого нормирующего значения. Подбор параметра β , как для весов в разделе 4.1.1, усложняется неопропорциональным влиянием на процесс сегментации.

4.1.3 Последовательное применение метрик

Принцип работы - сортировка(по возрастанию) вершин по расстоянию. Затем поиск до первого совпадения вершины с наиболее близким к 0.5 мет-

рики mean (4.1).

Результат - самопересечение,сегментации нет. Разметка углов на изображении может происходить с небольшим сдвигом в область региона. Появляется сдвиг среднего значения пикселей между вершинами. Нет единого ε для сравнения mean (4.1) с 0.5, на основе которого можно выявить вершину на границе. Для каждой вершины ε лежит в диапазоне от 0.1 до 0.3.

Будущее развите. При использовании динамического определения отклонения среднего значения писелей от 0.5 может быть достигнут хороший результат.

4.2 Аналитический подход

Не реализован.

Интересный, но сложный путь решения [2]. Создание полигона только на основе координат нетривиальная задача так как существует множество вариаций линии периметра, проходящей полинии. В поставленной задаче помимо координат есть само изображение, которое однозначно определяет форму пустоты.

4.3 Геометрическая аппроксимация

4.3.1 Convex hull

Простой и рабочий метод для создания грубого полигона [3]. Принцип работы схож с стягиванием вокруг вбитых гвоздей связанной нитки (рис. 11).

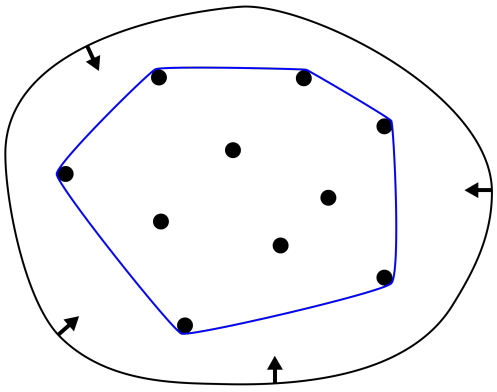

Рис. 11: Черные точки - "вбитые гвозди" , черная линия - связанная нитка, которая при стягивании занимает место синей линии

Результат работы алгоритма - неточное выделение контура пустот. Подходит только для выпуклых регионов(рис. 12).

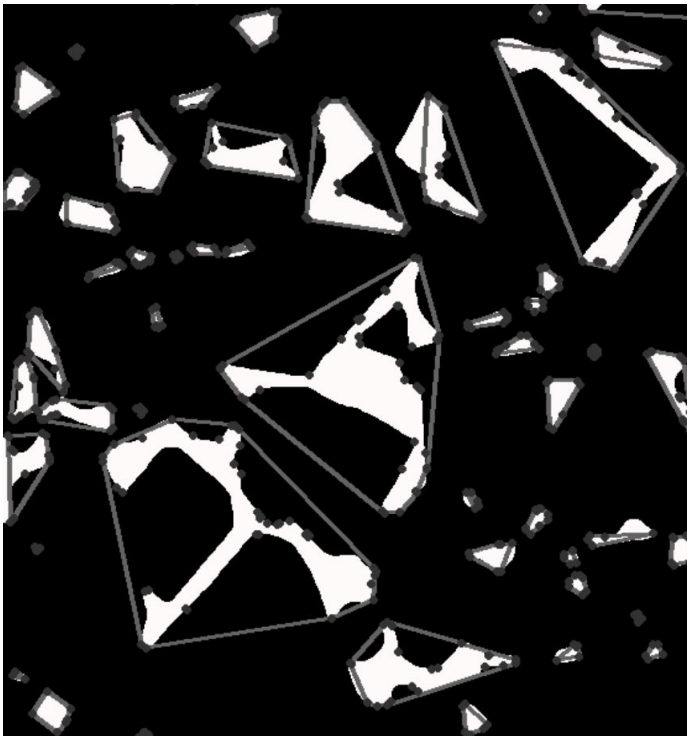

Рис. 12: Серые линии - периметр пустот, полученный в результате работы convex hull

4.3.2 Поиск контуров и линейная аппроксимация

Иной вариант решения задачи.Получим упорядоченный набор пикселей контуров пустот, а затем аппроксимируем их линейной функцией.

Реализуем последовательность нескольких алгоритмов:

- поиск границ детектором Кенни [10]
- получение упорядоченного набора пикселей контуров [12]
- линейная аппроксимация контура [5]

Сначала детектор границ Кенни выделяет границы у пустот (рис. 13). Затем полученные контуры обрабатываются на алгоритмом [12] и потом каждому пикселю контура присваивается порядковый номер. Последний этап - уменьшение количества точек и аппроксимация пикселей с помощью линейной функции (рис. 14).

Результат - достаточно точное выделение границ пустот при помощи прямых линий и отсутсвие необходимости в предварительной разметке углов.

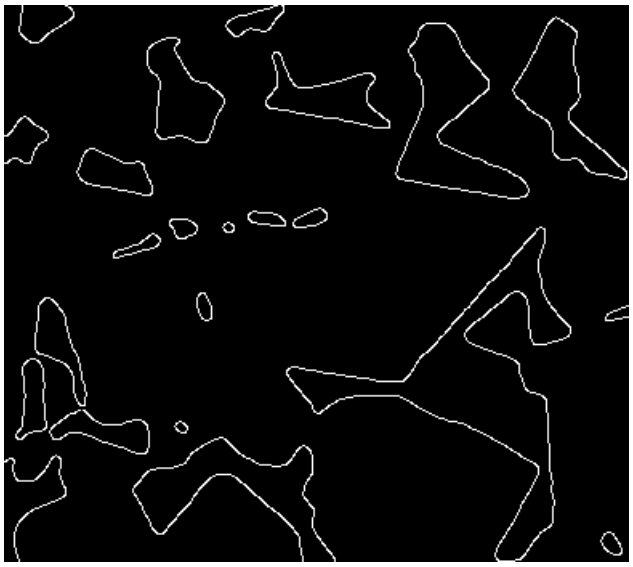

Рис. 13: Выделенные методом Кенни контуров пустот

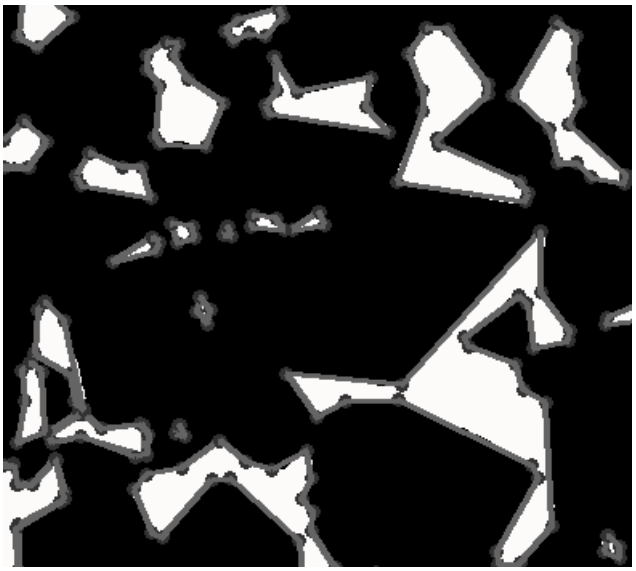

Рис. 14: Пустоты с линейно аппроксимированными границами

5 Выводы

Поставленная задача на деле оказалась нетривиальной. Множество популярных методов не смогли показать достаточный уровень точности определения границы.

Метод из главы 4.3.2 показал отличное решение поставленной задачи. При помощи комбинации нескольких алгоритмов удалось выделить границы пустот и получить упорядоченный и линейно приближенный набор пикелей контура.

Список литературы

[1] scipy.ndimage.label

[2] Hongyun Zhang ,Quanhua Zhao ,Yu Li - Generation of simple polygons from ordered points using an iterative insertion algorithm

[3] Convex hull

[4] Shi-Tomasi - Corner detection

[5] Ramer,Douglas,Peucker - Ramer–Douglas–Peucker algorithm

[6] Approximate and subdivide polygons

[7] lightsource - Детекторы углов

[8] Д.Г.Каграманян - Сегментация зерен сплава WC-Co готовыми инструментами

[9] Брезенхэм - Отрисовка линий в двумерного растра

[10] John F. Canny - Canny edge detector

[11] impersonalis - Выделение границ Канни

[12] Satoshi Suzuki,Keiichi Abe - Topological Structural Analysis of Digitized Binary Images by Border Following