# Сеточная векторизация

Векторизации подвергается пиксельное монохромное изображение типа контурной карты.

Пиксели контуров будем называть черными, все остальные - белыми.

## Определение компонент связности

На изображение накладывается сетка с квадратными ячейками. Узлы сетки это вершины графа. Если на отрезке, соединяющем два соседних узла, нет ни одного черного пикселя, между этими узлами существует ребро. Во всех иных случаях (есть черные пиксели или узлы не соседние) ребер между вершинами нет.

На первом этапе определяем компоненты связности графа. Одна компонента связности - это множество вершин, лежащих внутри замкнутого контура исходного изображения. Но поскольку наша цель - аппроксимировать этот контур ломаной, попутно с определением компонент связности находим точки, принадлежащие этой ломаной. Такими точками будут черные пиксели, обнаруженные нами в процессе определения ребер графа.

На каждом отрезке между двумя соседними узлами сетки таких точек будет не более одной (первый черный пиксель, найденный при прохождении отрезка). Помимо местоположения этих точек, важно знать их принадлежность к контуру (компоненте связности), точнее к паре смежных контуров.

Поскольку во время обнаружения точек результат определения компонент связности еще не готов, временно будем привязывать точки к паре узлов сетки (будем называть эту пару узлов охраняющими). По окончании определения к.с. графа по этой привязке можно будет определить, к каким контурам принадлежит конкретная точка.

## Алгоритм определения к.с.

Просматриваются все узлы слева направо, сверху вниз. Для каждого узла определяется его смежность с верхним и с левым соседом, если таковые имеются (узлы смежные, если на соединяющей их линии нет черных пикселей).

Если текущий узел не смежен ни с одним соседом, он один составляет новую временную компоненту связности.

Если текущий узел смежен только с одним соседом, он включается в компоненту соседа.

Если текущий узел смежен с обоими соседями, он включается в компоненту левого соседа,

а компонента верхнего соседа объединяется с компонентой левого.

Во время выполнения процедуры все компоненты связности считаются временными, но по окончании процедуры все временные компоненты становятся постоянными.

## Создание разделительных точек

Попутно с определением к.с. создаем разделительные точки, из которых позднее будем строить контуры векторного изображения. В процессе выяснения смежности мы проверяем все пиксели на отрезке, соединяющем узлы. Когда на этом отрезке встречается черный пиксель, создаем разделительную точку с координатами этого пикселя и ссылками на два узла по концам отрезка.

## Распределение точек по контурам

На следующем этапе точки контуров, найденные ранее, выстраиваются в ломаные, аппроксимирующие линии исходного изображения.

Все множество точек разбиваем на подмножества таких точек, что относятся к одной компоненте связности. Для этого просматриваем все точки и для каждой определяем ее к.с. опосредованно через охраняющие точку узлы.

Точки на замкнутых линиях удвоятся и попадут в две разные компоненты, точки на незамкнутых линиях - только в одну.

## Сборка контуров

Неупорядоченное множество точек, относящихся к одному контуру, следует выстроить в линию. Для этого возьмем любую точку и сделаем ее первой точкой контура. Следующей точкой контура будет ближайшая к ней и т.д., пока все точки не будут выстроены.

Легко понять, что расстояние между соседними точками контура не превышает диагонали ячейки сетки. Если расстояние до ближайшей точки превышает диагональ ячейки, то что-то пошло не так.

Точки замкнутого контура обязательно выстроятся в линию, т.е. расстояние между первой и последней будет не больше диагонали ячейки.

Если контур не замкнутый, то таким алгоритмом можно выстроить лишь часть линии, от первой точки до одного из концов линии.

## Незамкнутые контуры

Описанная процедура находит точки, принадлежащие и незамкнутым контурам тоже.

Одна компонента связности может содержать несколько незамкнутых линий. Сборка точек в незамкнутые линии отличается тем, что процедуру поиска ближайших точек нужно повторять дважды, для первой и второй части линии. Нужно иметь в виду, что линий в одной компоненте связности может быть более одной.

Для построения второй части линии нужно повторить процедуру поиска ближайших точек, еще раз начав с первой выбранной точки.

## Разное

Из-за того, что линии матричного изображения бывают толстыми, некоторые узлы сетки оказываются целиком окруженными черными пикселями. Это приводит к возникновению единичных компонент связности и утрате разделительных точек.