# 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 к компонентой левого.

Во время выполнении процедуры все компоненты связности считаютя временными,

но по окончании процедуры все временные компоненты становятся постоянными.

Создание разделительных точек

-------------------------

Попутно с определением к.с. создаем разделительные точки, из которых поздее 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.

class YPount

Dot Dot1, Dot2;

Point Point;

class Dot

Comp Comp;

class Comp: List<Node>

class YContur: IContur

List<YPoint> ypoints;

Dot[,] dots

List<Comp> comps;

На втором этапе точки контуров, найденные ранее, выстраиваются в ломаные, апроксимирующие линии исходного изображения.

Распределение точек по контурам

-------------------------------

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 раздвоятся и попадут в две разные компоненты, точки на незамнутых 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.

Если расстояние до ближайшей точки превышает гипотенузу ячейки сетки, то что-то пошло не так.

Если контур замкнутый, то все точки выстроятся в замкнутую линию, т.е. расстояние между первой и последней будет не больше гипотенузы.

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

не имеет смежных соседей, он окрашивается в новый цвет.

Если текущий узел не имеет смежных соседей, он окрашивается в новый цвет.

Если у текущего