1. **маски Робертса/Собеля/Превита**
2. Преобразуем кортежи (R, G, B) в яркость: Y = 0.222R + 0.707G + 0.071B
3. Строим две матрицы (Gx и Gy) на основании маски (если маска выходит за границы изображения – заменяем значения пикселей за изображением нулями или дублируем пиксели рядом
4. **Робертса:**

Gx = z9 – z5,

Gy = z8 – z6

**Превитта:**

Gx = (z7 + z8 + z9) – (z1 + z2 + z3),

Gy = (z3 + z6 + z9) – (z1 + z4 + z7)

**Собеля:**

Gx = (z7 + 2 z8 + z9) – (z1 + 2 z2 + z3),

Gy = (z3 + 2 z6 + z9) – (z1 + 2 z4 + z7)

2. Рекурсивный/ Итеративный алгоритм

**Рекурсивный алгоритм:**

просмотр с точки (1,1);

•поочередно все точки строки: если все белые, переходим к следующей строке; иначе:

–а) помечается точка;

–б) определяется цвет четырех соседних точек;

–в) если цвет всех точек белый или все помечены, закончить, иначе начальная позиция перемещается в первую найденную соседнюю черную точку;

–г) повторяются пункты а, б и в;

•повторить до последней точки последней строки изображения.

**Итеративный алгоритм:** 

1 – Новый класс

2 – Класс, как в левом пикселе

3 – Класс, как в верхнем пикселе

4 – Класс, как в любом пикселе слева или сверху

**3. Алгоритм кратчайшего незамкнутого пути:**

1: Найти пару точек (i, j) с наименьшим ρij и соединить их ребром;

2: пока в выборке остаются изолированные точки

3: найти изолированную точку, ближайшую к некоторой неизолированной;

4: соединить эти две точки ребром;

5: удалить K − 1 самых длинных рёбер;

**4. Решающее дерево**

**IG(Q) = S0 -**

s0 = вычисляем энтропию исходного множества

Если s0 == 0 значит:

Все объекты исходного набора, принадлежат к одному классу

Сохраняем этот класс в качестве листа дерева

Если s0 != 0 значит:

Перебираем все элементы исходного множества:

Для каждого элемента перебираем все его атрибуты:

На основе каждого атрибута генерируем предикат, который разбивает исходное множество на два подмножества

Рассчитываем среднее значение энтропии

Вычисляем IG

Нас интересует предикат, с наибольшим значением IG

Найденный предикат является частью дерева принятия решений, сохраняем его

Разбиваем исходное множество на подмножества, согласно предикату

Повторяем данную процедуру рекурсивно для каждого подмножества