Выделение связных областей в цветных и полутоновых изображениях

Авторы:
Александр Вежневец

В статье произведен небольшой анализ существующих методов разбиения изображения на связные области и предложен модифицированный вариант алгоритма region growing.

Введение.

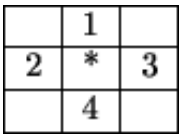
В задачах обработки и анализа изображений часто появляется необходимость разбить исходное изображение на некоторое множество связных (в пространственном смысле) областей, пиксели которых близки по некоторому признаку.

В данной статье будет произведен небольшой анализ существующих методов выделения связных областей и будет предложен расширенный и модифицированный вариант алгоритма роста регионов (region growing).

Основные понятия.

Связное множество (область) - множество пикселей, у каждого пикселя которого есть хотя бы один сосед, принадлежащий данному множеству.

Виды связности:
4 связность - соседями для пикселя считаются 4 пикселя: сверху, слева, справа, снизу.
8 связность - соседями для пикселя считаются 8 пикселей, т.е. все к нему прилежащие, в том числе и по диагонали.



4-связность

8-связность

Случай бинарного изображения.

Хотя основной в статье является тема сегментации цветного и полутонового изображения, тем не менее, следует рассмотреть и случай бинарного изображения, т.к. многие задачи для цветных изображений можно свести к решению задачи для бинарного изображения.

Итак, пусть у нас имеется бинарное изображение, бинарным считается изображение, состоящее из точек двух типов: фоновых точек и точек интереса. Пусть 0 - фон, 1 - объект. В данном случае разметка является однозначной при фиксированном виде связности. Существует 2 известных метода разметки: рекурсивный и итеративный.

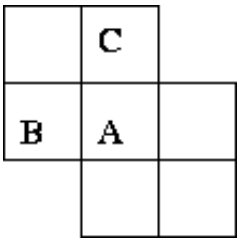
Рекурсивный алгоритм описывается следующим псевдокодом:

```
void Labeling(BIT* img[], int* labels[])
{
        // labels должна быть обнулена
        L = 1;
        for(y = 0; y < H; y++)
```

```
                for(x = 0; x < W; x++)
                {
                        Fill(img, labels, x, y, L++);
                }
}

void Fill(BIT* img[], int* labels[], int x, int y, int L)
{

if( (labels[x][y] = = 0) && (img[x][y] = = 1) )
        {
                labels[x][y] = L;
                if( x > 0 )
                        Fill(img, labels, x - 1, y, L);

if( x < W - 1)
                        Fill(img, labels, x + 1, y, L);

if( y > 0 )
                        Fill(img, labels, x, y - 1, L);

if( y < H - 1 )
                        Fill(img, labels, x, y + 1, L);
        }
}
```

Недостатком рекурсивного метода является медленная работа и большой расход памяти. Существует и итеративный метод, который в литературе часто встречается под названием "алгоритм последовательного сканирования". Опишем и его:

Начинаем обход изображения, для определённости, из левого верхнего угла сверху вниз, справа налево. При обходе пропускаем пиксели фона. Понятно, что при таком порядке обхода у текущего рассматриваемого пикселя верхний и левый сосед уже должны быть размечены.




```
if A = 0
        do nothing

else if (not B labeled) and (not C labeled)

increment label numbering and label A

else if B xor C labeled
        copy label to A

else if B and C labeled

if B label = C label
                copy label to A
        else
                copy either B label or C label to A

record equivalence of labels
```

После того как мы обошли всю картинку, нам нужно совершить ещё один обход и произвести переразметку с учётом эквивалентности областей.

Случай цветного и полутонового изображения.

Сведение к бинарному изображению простыми методами.

Данная группа методов несёт в себе следующую идею - изображение следует привести к бинарному простыми методами, учитывая яркость/ цвет пикселей, а далее применить известные для бинарных изображений

алгоритмы.

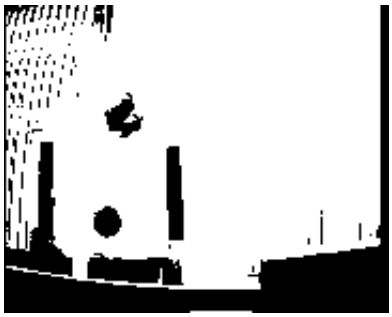
Алгоритмы, использующие только априорные знания (заданный порог яркости, заданные пороги цветовой сегментации и т.д.) крайне негибки. При изменении яркости, контрастности или цветового баланса пороги алгоритма нужно настраивать заново (вручную). Алгоритмы, адаптирующие свое поведение, основываясь на статистиках обрабатываемого изображения, обладают большей устойчивостью к изменению характеристик изображения.

Разберём работу метода на основе алгоритма k-средних.

Алгоритм k-средних в общем виде записывается следующим образом:
Дано:
Набор векторов $xi, \quad i = 1,:,p$;
k - число кластеров, на которые нужно разбить набор xi;
Найти:
k средних векторов $mj, \quad j = 1,:,k$ (центров кластеров);
отнести каждый из векторов xi к одному из k кластеров;

1. Случайным образом выбрать k средних $mj, \quad j = 1,:,k$;
2. Для каждого $xi, \quad i = 1,:,p$ подсчитать расстояние до каждого из $mj, \quad j = 1,:,k$,
   Отнести (приписать) xi к кластеру j', расстояние до центра которого mj' минимально;
3. Пересчитать средние $mj, \quad j = 1,:,k$ по всем кластерам;
4. Повторять шаги $2, 3$ пока кластеры не перестанут изменяться;

Применим алгоритм к одномерному пространству яркостей пикселей, положив $k = 2$. В данном случае каждому пикселю соответствует один вектор пространства - его яркость После разбиения на два кластера, всем пикселям с яркостью попавшей в первый кластер присвоим чёрный цвет, а другим - белый. Рассмотрим использование бинаризации с помощью метода k-средних для нахождения чёрных маркеров на белом участке с естественным фоном.


Главным достоинством этого подхода является скорость и простота. Однако такой подход не всегда дает положительные результаты - как видно на примере алгоритм отработал не достаточно хорошо - левая чёрная полоска слилась с фоном.

Остальные методы выделение связных областей можно разделить на две группы: алгоритмы, основанные на границах областей и алгоритмы, основанные на поиске регионов непосредственно.

Алгоритмы, основанные на границах областей.

Данная группа алгоритмов ищет регионы по их границам. Общая схема работы данной группы методов такова:

1. Обнаружение границ регионов.
2. Обнаружение регионов исходя из найденных границ.

Рассмотрим туже задачу нахождения чёрных маркеров на белом участке с естественным фоном, что и в предыдущем случае.



Выше приведены картинки, иллюстрирующие работу алгоритма, основанного на поиске границ. В данном случае для выделения границ используется высокочастотный фильтр, для бинаризации используется K-кластеризация, а выделение регионов на бинарном изображении происходит с помощью алгоритма последовательного сканирования.

Данный пример хорошо иллюстрирует достоинства и недостатки данного подхода. Подход существенно более гибок, чем простая бинаризация: изменяя параметры фильтра выделения границ, мы можем изменять чувствительность метода (в зависимости от контрастности фона и объекта). Так же алгоритм более стабилен к изменениям характеристик изображения(средняя яркость, фон, шум и т.д.), т.к. использует не фиксированный порог, а разницу яркости/цвета соседних пикселей (в данном случае энергию высоких частот).

Недостатки подхода:

1. Слабая устойчивость : достаточно небольшого разрыва границы для несрабатывания алгоритма С этим можно бороться, например, математической морфологией, но при этом обостряется следующий недостаток.
2. Выделение областей не является точным. Часть региона "отъедается" границей. Этот недостаток может быть несущественным для ряда применений (если интересуют, например, только положения центра региона).
3. Скорость работы методов невысока, т.к. требуется целых 3 этапа, каждый из которых не слишком быстрый.

Многие из этих недостатков являются следствием многоступенчатости метода. На каждом этапе мы пытаемся отбросить лишнюю, на наш взгляд, информацию и выделить полезную. Следует понимать, что каждый этап

работает с некоторой ошибкой: отбрасывается часть полезной информации и не вся лишняя уходит, при этом характер ошибки каждого этапа свой, но сильно зависящий от работы предыдущего.

Алгоритмы, основанные на поиске регионов непосредственно.

Данная группа пытается отыскать регионы непосредственно, в отличие от предыдущих методов. Приведём пример метода, который был разработан в нашей лаборатории.

Обозначим I - исходное изображение, (x,y) - координаты в I. I и пара (x,y) образуют множество троек $(I, (x,y))$. Пусть
$f(I, (x,y)) \rightarrow Rn$
- некоторая функция, отображающая картинку I и пару координат (x,y) в пространство Rn.
Пусть $v \in Rn$ - вектор в Rn, а $L(v)$ - норма в Rn.
Пусть C - класс элементов множества $(I, (x,y))$, а
$g(C) \rightarrow Rn$ - функция, отображающая класс C в Rn
Пусть $v1 = g(C)$, $v2 = f(I, (x,y))$,
тогда $L(v1 - v2)$ - мера близости класса C и вектора $(I, (x,y))$.
Ясно, что класс C задаёт область в изображении I.
Предположим, что у нас уже существует класс C, задающий связную областью в I. Тогда будем добавлять к классу C соседние (на изображении) ему элементы $(I, (x,y))$ если
$L(f(I, (x,y)) - g(C)) < \delta$,
где δ - некий выбранный нами порог.

Разберём простой случай (разбиение по яркости) :
$f(I, (x,y))$ - яркость пикселя с координатой (x,y),
$g(C)$ - средняя яркость пикселей класса,
$L(Rn)$ $(Rn = R1$ - одномерное пространство яркостей) - есть модуль числа.

Тогда пиксель будет отнесён к классу, если его яркость отличается от средней по классу меньше чем на δ. Такой выбор мер и отображений вполне полезен на практике, однако, общий метод позволяет получить и

более интересные результаты.

Для получения более интересных результатов в качестве можно выбрать пространство статистических характеристик пикселя, например, отклонение каждой из цветовых компонент от среднего значения по окрестности, дисперсия каждой цветовой компоненты в окрестности пикселя и т.д. Также вместо простого расстояния в качестве L можно выбрать какую либо другую норму.

Существуют известные рекурсивные алгоритмы для простого случая разбиения по яркости, так называемый алгоритм роста регионов(Region Growing), который по сути мало отличается от рекурсивного алгоритма разметки бинарного изображения. Этот алгоритм несложно обобщить для случая произвольных мер и отображений. Главный недостаток рекурсивного подхода - малая скорость и ошибка переполнение стека на больших картинках. Для преодоления данных проблем автором был разработан итеративный алгоритм применимый для общего случая.

Начинаем обход, для определённости, из левого верхнего угла.

1. Объявляем левый верхний пиксель изображения новым классом.
2. Для пикселей первой строки считаем отклонение от класса левого пикселя и, сравнивая с порогом, либо добавляем пиксель к классу соседа, либо заводим новый класс.
3. Первый пиксель каждой последующей строки сравниваем с классом верхнего пикселя. Далее будем сравнивать пиксель с классами двух соседей: левого и верхнего.
    1. Если отклонение от обоих классов больше порога, то заводим новый класс, если отклонение больше только для одного класса, то добавляем пиксель к тому классу отклонение от которого меньше порога.
    2. Если отклонение допустимо для обоих классов, то возможно 2 варианта:
        1. $L(g(C1) - g(C2)) < \delta$ - тогда объединяем эти 2 класса (если они не один и тот же класс) и добавляем к объединённому классу текущий пиксель.
        2. $L(g(C1) - g(C2)) > \delta$ - в этом случае добавляем пиксель к тому

классу, от которого отклонение минимально.

Данный алгоритм допускает различные внутренние реализации классов и отображений, но является однозначным относительно их. Стоит заметить, что алгоритм использует 4 связность, но возможна и реализация и для 8 связных областей.

Рассмотрим туже задачу нахождения чёрных маркеров на белом участке с естественным фоном, что и в предыдущем случае.

Данный пример является яркостной сегментацией с $\delta = 50$ (яркость от 0 до 255).
Достоинствами такого подхода являются:

- Широкий спектр применения. За счёт выбора отображений и меры можно приспособить алгоритм под различные задачи.
- Гибкость. Изменяя порог и меру можно эффективно менять чувствительность алгоритма.
- Скорость. Алгоритм работает существенно быстрей алгоритма с поиском границы метода.
- Устойчивость. Алгоритм более устойчив к ошибкам, чем методы, основанные на нахождение границ, т.к. при ошибке мы теряем не весь регион, а лишь его небольшую часть.
- Точность. Найденные регионы не "обкусаны", как при edge based алгоритмах.

Существуют также и другие методы, основанные на поиске регионов. Подробно о них можно прочитать в других источниках, например: [1], я же приведу лишь их краткое описание.

Слияние регионов (Region Merging):

Изображение первоначально разбивается на регионы некоторым простым методом. Далее для каждого региона рассматриваются его соседи, если они подходят под некоторый критерий схожести, то регионы сливаются. Процесс идёт пока на следующем шаге не один регион не изменится. Проблемы: нужно выбрать первоначальное разбиение, медленная работа.

Разбиение и слияние регионов (Region splitting & merging):

Изображение первоначально разбивается на регионы некоторым простым методом. Далее, если регион не является однородным - разбиваем его на некоторое количество подрегионов. Когда все регионы однородны, сливаем те из них которые являются соседями и вместе образуют однородный регион. Проблемы: зачастую возникают ограничения на форму регионов, также часто

Метод "водораздела" (Watershed Segmentation):

Метод рассматривает только полутоновое изображение. Полутоновое изображение можно воспринимать, как некий ландшафт - чем светлей пиксель, тем выше точка поверхности. Задача сегментации сводится к нахождению областей стабильного минимума отделенных друг от друга областями стабильного максимума. Подробней о методе можно прочитать по выше приведенной ссылке.

Литература:

Дополнительная информация
Ссылка:

Александр Вежневец. Выделение связных областей в цветных и полутоновых изображениях. *Компьютерная графика и мультимедиа*. Выпуск №1(5)/2003. http://cgm.computergraphics.ru/content/view/53

Выпуск:

Выпуск №1(5)/2003 [1]

1. http://cgm.computergraphics.ru/issues/issue4