УДК 519.68

С. И. Хашин

ОЦЕНКА КАЧЕСТВА СЕГМЕНТАЦИИ ИЗОБРАЖЕНИЯ

1. Как численно сравнить качество алгоритмов сегментации

При анализе изображений часто возникает задача разделения пикселей изображений на группы по некоторым признакам. Такой процесс называется сегментацией изображения. Сегментация — обширная и сложная область компьютерной графики, которой посвящено много книг и статей, см., например, библиографию в [2, 6].

Понятие качества сегментации в общем виде формализовать трудно. Наиболее общий подход, не зависящий от приложения, состоит в сравнении результатов рассматриваемого алгоритма с заранее заданной "оптимальной" сегментацией, сделанной обычно вручную. Например, *Berkeley Segmentation Dataset* [7] содержит свыше 12 000 таких ручных сегментаций для примерно 1000 высококачественных изображений.

В настоящее время имеется множество различных алгоритмов сегментации изображений [2,4,5,6].

В каждом конкретном приложении, например при разработке алгоритма видеосжатия [4] или в других областях [1], сегментация является лишь частью некоторого более общего процесса. Поэтому можно определить качество сегментации апостериори, по общим результатам работы всего алгоритма. Однако этот процесс очень медленный и в немалой степени зависит от других частей общего алгоритма, помимо собственно сегментации. Поэтому имеет смысл разработать отдельный, более простой метод определения качества сегментации, не связанный с конкретным приложением, а более универсальный.

Отметим, что одного параметра недостаточно для сравнения алгоритмов сегментации. В самом деле, сегментация из 10 сегментов не должна прямо сравниваться с сегментацией из 10 тыс. сегментов: критерии у них должны быть различны.

Предлагаемый алгоритм состоит из следующих шагов.

1. Рассмотрим пару полноцветных (TrueColor) изображений, не обязательно одинакового размера.

2. Из второго изображения вырежем круг фиксированного радиуса (или, более общо, фиксированную кусочно-гладкую фигуру), взятый произвольно.

3. Вклеим ее в первое изображение опять же в случайное место.

4. Сегментируем полученное изображение. Некоторые сегменты могут пересекаться как с внутренностью вклеенной фигуры, так и с внешностью. Назовем такие сегменты *ошибочными*. Часть точек ошибочного сегмента принадлежит внешности фигуры, часть — внутренности. Меньшую часть из них назовем *ошибочными* точками.

5. Качество сегментации будем обозначать двумя параметрами:
а) средний размер сегмента (в точках);
б) 100% − (доля ошибочных точек по отношению к площади фигуры, %).

2. Описание программы

Для нахождения численной оценки качества сегментации предлагается программа “segmTst.exe”. Она работает в двух режимах:

1) построение тестовых изображений;
2) нахождение качества сегментации.

Сначала программа запускается в режиме построения тестовых изображений, в результате чего получается каталог с набором из некоторого количества (не менее сотни) тестовых изображений. Затем программа запускается в режиме проверки качества. Исследуемая программа вызывается для построения сегментации каждого из тестовых изображений, результат должен быть получен в виде текстовой матрицы заданного формата.

2.1. Структура файла с сегментацией

Сегментацией изображения называется его разбиение на однородные области (сегменты) по некоторому признаку. С вычислительной точки зрения сегментацией изображения размера $mx * my$ будем называть целочисленную матрицу того же размера из неотрицательных чисел. Каждое число обозначает номер сегмента ($\geqslant 0$), к которому принадлежит точка. Такие матрицы сохраняются в текстовом файле вида

mx my комментарий s00 ... s10 ... s01 s11 ... .

Например, при сегментации файла размером $4 * 3$:

```
4 3; в первой строке файла заданы размеры матрицы по x (4)
и по y (3)
1 2 1 1
2 2 1 1
2 2 2 3,
```

где цифры в теле матрицы означают номер сегмента ($\geqslant 0$), к которому относится каждая точка. Сегменты не предполагаются связными, и их номера не обязаны идти подряд. Никакое упорядочение сегментов также не требуется.

2.2. Построение тестовых изображений

Запуск:

`segmTst.exe − c srcPath resPath cnt [mx my],`

где srcPath — каталог с исходными файлами (bmp-24), из которых строятся текстовые файлы, resPath — выходной каталог с результатами работы.

Из исходного каталога случайно выбираются cnt пар, из второго изображения пары вырезается заданная фигура (круг радиуса 50) в случайном месте и вклеивается в случайное место в первом изображении из пары. Если заданы два дополнительных параметра mx, my, то изображение масштабируется до указанного размера. Полученное склеенное

изображение записывается в каталог resPath под именем nnnn.bmp, где nnnn — номер пары. В этом же каталоге строится текстовый файл-описание со строчками вида

```
nnnn.bmp C mmmm xxxx yyyy,
```

где nnnn.bmp — имя файла;
C — признак типа обрабатываемой фигуры (circle);
mmmm — количество точек в фигуре;
xxxx — x-координата центра вклеенной фигуры на изображении;
yyyy — y-координата центра вклеенной фигуры на изображении.

Замечание. В предложенный выше алгоритм были внесены некоторые изменения. Если его использовать напрямую, то получающиеся "врезанные" круги оказываются в среднем чересчур резкими, т. е. слишком легкими для сегментации. Поэтому при выборе случайного места, из которого вырезается фигура, и случайного же места, куда она вставляется, отдается предпочтение точкам, у которых средняя длина цветового градиента в окрестности выше средней. На глаз фигуры все равно остаются очень заметными, но чуть менее.

2.3. Требования к сегментирующей программе

Мы планируем сравнивать различные алгоритмы сегментации, реализованные в виде готовых программ. Поэтому программы должны иметь некоторый унифицированный интерфейс. Сегментирующая программа (exe- или bat-файл) должна воспринимать 3 аргумента (3-й может игнорироваться):

```
zzz.exe src.bmp segm.txt nSegm,
```

где zzz.exe — программа сегментации (возможен как exe-, так и bat-файл);
src.bmp — файл, подлежащий сегментации (в формате bmp-24);
segm.txt — текстовый файл с результатами сегментации в формате, описанном выше;
nSegm — желательное количество сегментов.

Программа может и не обрабатывать третий аргумент и сама определять количество сегментов. Если же он обрабатывается, не обязательно выдавать в точности заданное количество, а хотя бы приближенно.

2.4. Нахождение качества программы сегментации

Запуск:

```
zzz.exe src.bmp segm.txt nSegm,
```

где resPath — каталог с искусственными изображениями (с вклеенной фигурой);
progName — исследуемая программа сегментации (exe- или bat-файл);
nSegm — рекомендуемое количество сегментов;
prot.txt — выходной текстовый файл с результатами работы.

Строки файла prot.txt имеют вид

```
nnnn.bmp C mmmm xxxx yyyy nSegm qq.qq,
```

где nnnn.bmp — имя файла;
C — признак обрабатываемой фигуры (circle);
mmmm — количество точек в фигуре;
xxxx — x-координата центра вклеенной фигуры на изображении;
yyyy — y-координата центра вклеенной фигуры на изображении;
nSegm — количество сегментов;
dd.dd — качество сегментации (%).

В конец этого же файла записывается небольшая таблица с общим итогом проверки качества сегментации: для каждого количества сегментов указывается среднее значение качества. Каждая строка таблицы имеет следующий вид:

```
nSegm_min nSegm_max dd.dd_mid,
```

где nSegm_min — минимальное количество сегментов в интервале;
nSegm_max — максимальное количество сегментов в интервале (превышает nSegm_min не более чем на 30%);
dd.dd_mid — среднее качество сегментации для интервала (%).

2.5. Пример: качество сегментирования на квадраты

Проверим работу на примере тривиального сегментирования — "на квадраты". Строим 100 файлов размером 640*480 со вклеенным кругом диаметра 100:

```
Segm_test.exe - C C:/bmp/ C:/bmp_res/ 100 640 480.
```

Вызов программы сегментации (Segm_by_squares.exe)

```
Segm_by_squares.exe src.bmp segm.txt nSegm,
```

где src.bmp — файл, подлежащий сегментации (в формате bmp-24);
segm.txt — текстовый файл с результатами сегментации в формате, описанном выше;
nSegm — желательное количество сегментов.

Программа, учитывая размеры изображения, подбирает размер квадрата так, чтобы получить количество сегментов, максимально близкое к заданному.

Для построенных изображений с вклеенными кругами находим качество сегментации с помощью программы сегментирования на 200 ... 5000 квадратов:

```
Segm_test.exe - T C:/bmp_res Segm_by_squares.exe 200
Segm_test.exe - T C:/bmp_res Segm_by_squares.exe 500
Segm_test.exe - T C:/bmp_res Segm_by_squares.exe 1000
Segm_test.exe - T C:/bmp_res Segm_by_squares.exe 2000
Segm_test.exe - T C:/bmp_res Segm_by_squares.exe 5000.
```

Вот полученный протокол работы:

```
N=100; SZ=3491.00+-0.00; eff=42.43+-11.44;
prog=Segm_by_squares.exe
N=200; SZ=1600.00+-0.00; eff= 61.02+-7.21;
prog=Segm_by_squares.exe
```

```
N=500; SZ=647.00+-0.00; eff=75.37+-3.54;
prog=Segm_by_squares.exe
N=1000; SZ=338.00+-0.00; eff=82.54+-2.26;
prog=Segm_by_squares.exe
N=2000; SZ=145.00+-0.00; eff=88.39+-1.52;
prog=Segm_by_squares.exe
N= 5000; SZ=64.00+-0.00; eff=92.34+-0.82;
prog=Segm_by_squares.exe
```

Представим протокол в виде таблицы.

Запрашиваемое количество сегментов	Средний размер сегмента	Эффективность
100	3491	42.43 ± 11.44
200	1600	61.02 ± 7.21
500	647	75.37 ± 3.54
1000	338	82.54 ± 2.26
2000	145	88.39 ± 1.52
5000	64	92.34 ± 0.82

3. Сегментация через триангуляцию

Рассмотрим следующий алгоритм сегментации изображения.

Шаг 1. Построение точек. Вначале ставим 4 угловые точки. Затем добавляем по одной (по мере убывания длины градиента), но не слишком близко к уже поставленным точкам, пока не получим нужное количество точек (а значит, и треугольников).

Шаг 2. Триангуляция Делоне построенной сети.

4. Сегментация с помощью кристаллизации

Рассмотрим следующий алгоритм сегментации изображения. Пусть задано некоторое количество сегментов N.

Шаг 1. Выбираем удвоенное количество точек ($2N$) — центров будущих сегментов. Точки выбираем в тех местах, где усредненное значение длины градиента как можно меньше, причем точки не должны быть близко друг к другу.

Шаг 2. Наращиваем все сегменты из их центров. Каждый раз присоединяем к сегменту соседние точки, которые отличаются по цвету наименьшим образом. В результате получаем сегментацию на удвоенное количество сегментов.

Шаг 3. Объединяем два сегмента в один, если у них достаточно большая общая граница и цвета не слишком сильно отличаются. Объединение производим до тех пор, пока не получим заданное количество сегментов.

5. Результаты сравнения алгоритмов сегментации

Приведем результаты сравнения эффективности трех методов сегментации: на квадраты, с помощью триангуляции и через кристаллизацию. В первой строке (N) указано задаваемое количество сегментов, во второй ($Size$) — средний размер получающихся сегментов, в третьей ($Square$) — эффективность сегментации на квадраты, в четвертой ($Triangle$) — эффективность сегментации с помощью триангуляции, в пятой ($Crystall$) — эффективность сегментации с помощью кристаллизации:

N :	100	200	500	1000	2000	5000
$Size$:	3030	1525	612	307	153	61
$Square$:	42.43	61.02	75.37	82.54	88.39	92.34
$Triangle$:	65.15	78.20	89.51	93.35	95.87	97.72
$Crystall$:	78.65	83.34	88.12	86.10	—	—

Библиографический список

1. *Вежневец А., Баринова О.* Методы сегментации изображений: автоматическая сегментация // Компьютерная графика и мультимедиа. 2006. Вып. № 4. URL: http://cgm.computergraphics.ru/content/view/147 (дата обращения: 28.02.2010).
2. *Гонсалес Р., Вудс Р.* Цифровая обработка изображений. М : Техносфера, 2006. 1072 с.
3. *Кручинин А. Е., Хашин С. И.* Сегментация изображения путем выделения непрерывных границ // Вестн. Иван. гос. ун-та. 2007. Вып. 3. С. 80—83.
4. *Хашин С. И.* Применение методов распознавания образов для сжатия видеоинформации // Математические методы распознавания образов : сб. докл. 13-й Всерос. конф. Зеленогорск : МАКС Пресс, 2007. С. 420—424.
5. *Хашин С. И., Хашина Ю. А.* Свойства r-сегментации изображения // Математика и ее приложения : журн. Иван. мат. о-ва. Иваново : Иван. гос. ун-т, 2007. Вып. 1. С. 93—98.
6. *Яне Б.* Цифровая обработка изображений. М. : Техносфера, 2007. 583 с.
7. Berkeley Segmentation Dataset. URL: http://www.eecs.berkeley.edu/Research /Projects/CS/vision/grouping/segbench (дата обращения: 28.02.2010).