Лабораторная работа: Сегментация изображений

17 февраля 2012 г.

1 Теоретические задачи

1.1

Необходимо провести сегментацию изображения размером 4*4 пикселя (см. рис.1) с помощью метода разрастания регионов и последовательного сканирования сверху-вниз и слева-направо. Считаем, что черные пиксели имеют интенсивность 0, серые пиксели имеют интенсивность 127, белые пиксели имеют интенсивность 254. Какой критерий однородности области необходимо применить, чтобы в результате сегментации каждый пиксел был отдельным сегментом? Чтобы в результате сегментации осталось два сегмента, как на рисунке 2?

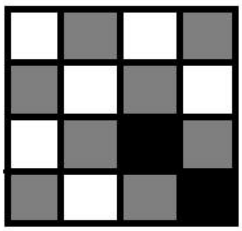


Рис.1

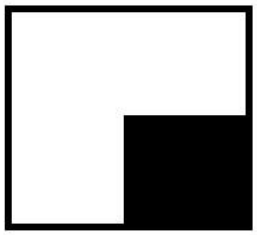


Рис.2

1.2

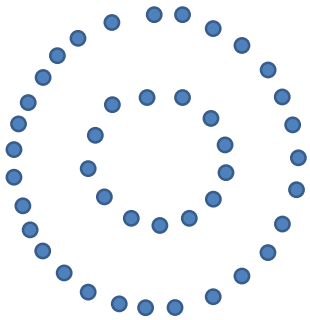
Пусть решается задача бинарной сегментации черно-белого изображения. Будет ли метод k-средних эквивалентен пороговой бинаризации по средней яркости изображения?

1.3

Объясните, почему "эффективный метод"сегментации имеет асимптотику времени работы $O(n \log n)$, где n - число ребер в графе, моделирующем изображение.

1.4

Каким будет результат кластеризации такого множества точек (см.рисунок) с помощью метода k-средних при $k = 2$?



Как нужно определить функцию веса на рёбрах, чтобы метод NCut отделил точки внутреннего круга от точек внешнего круга?

1.5

В методе NCut возникает задача нахождения собственных векторов нормализованной матрицы близости. Объясните смысл двух минимальных собственных значений этой матрицы. Почему ноль будет собственным значением? Какой смысл у собственного вектора, соответствующего нулевому собственному значению?

1.6

Пусть в методе NCut мы хотим для вычисления близости двух пикселей использовать сходство текстур их окрестностей. Пусть признаки текстуры вычисляются по некоторому окну фиксированной ширины вокруг пиксела. Приведите аргументы за и против использования малой ширины окна, большой ширины окна.

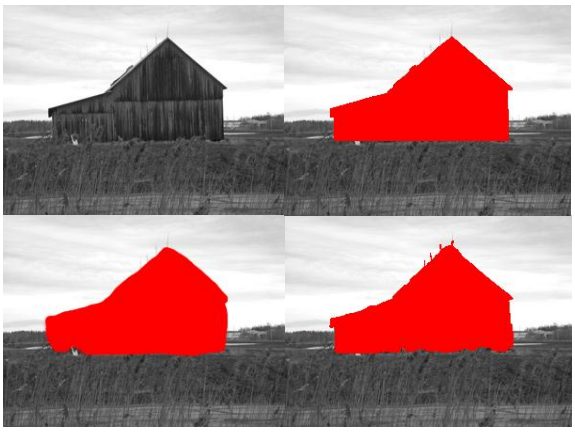
2 Практическая работа

Практическая часть заключается в реализации простого алгоритма сегментации с помощью k-средних и более сложного алгоритма интерактивной сегментации на основе модели активных контуров. Для проверки качества сегментации будут использоваться изображения, сегментированные вручную.

2.1 Данные и метрика качества

Для простоты будем решать задачу бинарной сегментации для черно-белых изображений. На каждом из тестовых изображений изображен один объект, необходимо каждому пикселу приписать метку: "foreground"или "background". Результатом алгоритма сегментации является маска, каждому пикселу которой приписана одна из этих двух меток.

Каждое изображение размечено вручную, причем не одним, а тремя асессорами. Пиксел считается принадлежащим объекту, если так посчитало по крайней мере двое асессоров.



Метрикой качества алгоритма будем считать F-меру (среднее гармоническое между точностью и полнотой), усредненную по всем изображениям. Все изображения находятся в папке *data*, в ней каждая подпапка соответствует одному изображению. В каждой подпапке есть исходное изображение *src.png* и набор отсегментированных вручную изображений в папке *human_seg*. Реализованные вами алгоритмы должны сохранять результаты сегментации каждого изображения в его папке, в подпапке с именем, одинковых для всех изображений. Например, алгоритм k-средних может создать в папке каждого изображения подпапку *kmeans_result* и сохранить свои результаты в ней. Для того, чтобы проверить качество того или иного алгоритма, необходимо запустить функцию *ComputeFMeasure*, указав ей первым параметром путь до папки *data*, а вторым - имя подпапки, в которой алгоритм сохранял результаты.

```
ComputeFMeasure('data', 'kmeans_result')
```

2.2 Алгоритм k-средних

В этой части необходимо реализовать один из простейших алгоритмов сегментации методом k-средних. Так как решается задача бинарной сегментации, значение k всегда будет равно двум.
Кластеризовать нужно будет величины интенсивностей, которые могут принимать всего 256 возможных значений. Можно этим воспользоваться и кластеризовать не пиксели, а столбцы в гистограмме интенсивностей изображения. Благодаря такой модификации время кластеризации не будет зависеть от размеров изображения.
Основная часть кода уже написана и находится в папке *Kmeans*. Основную логику содержит скрипт *Kmeans.m*, для получения рабочего алгоритма вам необходимо дополнить код, соответствующий одной итерации метода. Помимо основного скрипта есть вспомогательная функция *KmeansSegmentation*, запускающая сегментацию всех тестовых изображений и сохраняющая результаты. В качестве первого аргумента *KmeansSegmentation* необходимо указать путь до папки *data*, а вторым - имя подпапки, в которой сохранять результаты, например, *kmeans_result*.

```
KmeansSegmentation('../data', 'kmeans_result')
```

Задание: Дополните код одной итерации алгоритма k-средних в скрипте *Kmeans.m*

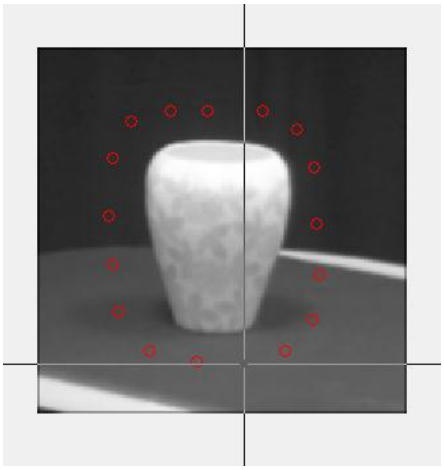
```
while(true)
  OldCentroids = Centroids;
  % you should assign closest centroids labels here
  ...
  % you should recalculate centroids here
```

```
  ...
  % stop if converge
  if(Centroids == OldCentroids)
      break;
  end;
end
```

Запустите метод на всех тестовых изображениях. Проанализируйте результаты работы метода. На каких изображениях метод дает хорошие результаты? неудовлетворительные результаты?

2.3 Алгоритм на основе модели активных контуров

В этой части необходимо реализовать более сложный алгоритм интерактивной сегментации на основе модели активных контуров. Интерактивность заключается в том, что перед началом сегментации пользователь выделяет вручную выделяет объект на изображении, тем самым инициализируя алгоритм сегментации.



Основная часть кода уже написана и находится в папке *Snake*. Вам необходимо дополнить код скрипта *iterate.m*. Для того, чтобы запустить алгоритм на всех тестовых изображениях необходимо запустить функцию *SnakeSegmentation*, указав ей в качестве параметров путь до папки *data* и имя подпапки для сохранения результата.

```
SnakeSegmentation('../data', 'snake_result')
```

Алгоритм имеет большое число параметров, в работе нас будут интересовать несколько из них:

1. α - коэффициент штрафа за "длинный"контур
2. β - коэффициент штрафа за контур с большой кривизной
3. κ - коэффициент чувствительности к градиентам интенсивности изображения

Ключевым местом в алгоритме является итерационный процесс решения системы уравнений:

$$x_t = (A+\gamma I)^{-1}\left(\gamma x_{t-1} - \kappa f_x\left(x_{t-1}, y_{t-1}\right)\right) \quad (1)$$

$$y_t = (A+\gamma I)^{-1}\left(\gamma y_{t-1} - \kappa f_y\left(x_{t-1}, y_{t-1}\right)\right) \quad (2)$$

Задание:
Дополните код одной итерации процесса решения системы уравнений

```
%moving the snake in each iteration
for i=1:N;

    % you should calculate new snake position here
...


    %Displaying the snake in its new position
    imshow(image,[]);
    hold on;

    plot([xs; xs(1)], [ys; ys(1)], 'r-');
    hold off;
    pause(0.001)
end;
```

Запустите метод на всех тестовых изображениях. Подберите оптимальные значения параметров (скрипт *SnakeSegmentation.m*). Попробуйте "поотключать"чувствительность к градиентам, штраф за длину. Проанализируйте результаты работы метода. На каких изображениях метод дает хорошие результаты? неудовлетворительные результаты?