УДК 004.932

АЛГОРИТМ КЛАСТЕРНО-ОРИЕНТИРОВАННОЙ ЛОКАЛИЗАЦИИ СВОЙСТВ ИЗОБРАЖЕНИЯ

Валюх А.А., Тонкошкур Ю.О.
Факультет физики электроники и компьютерных систем
Днепропетровского национального университета им. О. Гончара
E-mail: valiuh@gmail.com, tonkoshkur@gmail.com

Аннотация:

Валюх А.А., Тонкошкур Ю.О. Алгоритм кластерно-ориентированной локализации свойств изображения. *Рассматривается задача построения алгоритма поиска изображений по содержанию на основе использования цветовой кластеризации. Алгоритм базируется на построении набора дескрипторов описывающих исходное изображение и дальнейшем их сравнении с дескриптором, содержащимся в базе данных изображений на предмет полного или частичного их совпадения. Алгоритм защищён авторским свидетельством.*

Общая постановка проблемы

Поиск изображений по содержанию — раздел компьютерного зрения, решающий задачу поиска изображений, которые имеют требуемое содержание, в большом наборе цифровых изображений[1]. В качестве примера рассмотрим один из самых популярных Интернет сервисов посвященный поиску изображений по содержанию – TinEye (tineye.com). TinEye мгновенно анализирует исходное изображение, чтобы создать компактную цифровую подпись или «отпечаток пальца» для него[3]. TinEye ищет изображение в сети Интернет путем сравнения его отпечатков пальцев с другими проиндексированными TinEye изображениями. В данной работе описана работа алгоритма поиска изображений по содержанию на основе использования цветовой кластеризации .

Постановка задачи

С целью поиска всех копий изображения целесообразно использовать методы позволяющие выполнить хеширование исходного изображения с последующим сравнением полученного хэша, с хэшами других изображений. Однако поскольку, похожие друг на друга изображения могут несколько отличатся в первую очередь по цветовой гамме, а также по ряду других показателей, то применение для этих целей таких средства как преобразование хэширующими алгоритмами типа md5 и им подобным будет не эффективно. Это очевидно так как похожие «на глаз» изображения, но отличающиеся по своим показателям, будут давать разные хэши, и следовательно система будет обладать низкой обобщающей способностью. Необходим алгоритм построения хэша обладающего более высокой обобщающей способностью.

Данная работа посвящена алгоритму кластерно-ориентированной локализации свойств изображения В процессе реализации и тестирования алгоритма использовался язык Python (версия Python - 2.7) с модулями PIL, numpy, scipy и matplotlib для обработки изображения и визуализации работы алгоритма. Указанные компоненты были развернуты на системе следующей конфигурации Athlon XP 1800+ (1,53GHz), 1GB DDR OC Fedora 14.

Основная часть

На начальном этапе работы алгоритма проводится предварительная обработка входного изображения. Целью предварительной обработки изображения является во-первых

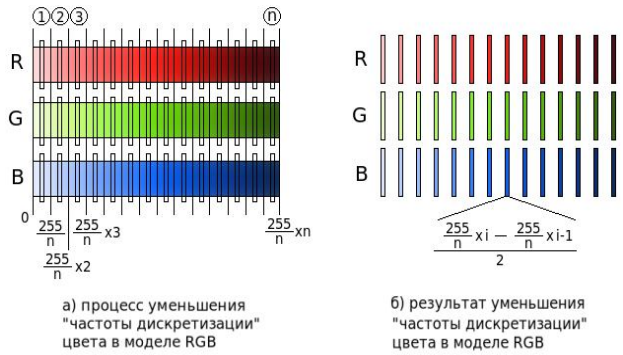
приведение всех изображений к единому масштабу. Для горизонтальных изображений - это 320x240, для вертикальных - 240x320. Во-вторых, сглаживание полученного изображения с целью уменьшения влияния мелких деталей изображения, а также удаления различного рода шумов связанных с качеством изображения. Наилучший результат сглаживания, удовлетворяющий описанным выше условиям, был получен с использованием медианного фильтра. Результат предварительной обработки входного изображения представленного на рис. 1а представлен на рис. 1б.

а) исходное изображение приведенное к разрешению 320x240

б) изображение после обработки медианным фильтром

Рис.1. Результаты предварительной обработки входного изображения

После предварительной обработки изображение представлено в цветовом пространстве RGB. В данном цветовом пространстве каждый пиксель кодируется комбинацией значений красного (R-компонента), зеленого (G-компонента) и синего (B-компонента) цветов. Каждая компонента в свою очередь представляет собой набор дискретных значений яркости (или оттенков) закодированных числами в диапазоне от 0 до 255 (см. рис. 3а). Передискретизация в обработке сигналов - изменение частоты дискретизации сигнала. Поскольку каждая из RGB компонент кодируется набором дискретных значений, то есть смысл говорить о так называемой передискретизации по цвету. В процессе этой операции происходит понижение «частоты дискретизации» цвета - то есть переход от полного набора значений оттенков по каждой компоненте к их частичным значениям. Сам процесс понижения «частоты дискретизации» представлен на рис.2а и рис.2б.



а) процесс уменьшения "частоты дискретизации" цвета в модели RGB

б) результат уменьшения "частоты дискретизации" цвета в модели RGB

Рис. 2 Процесс понижения «частоты дискретизации».

Суть процесса заключается в том, что весь диапазон значений оттенков по каждой компоненте разбивается на n равных частей (рис.2а). После чего из каждого поддиапазона выбираются их средние значения. В результате имеем урезанный «спектр» цвета для исходной модели RGB представленный на рис.3б. Всего цветов полученных на новом изображении после понижения «частоты дискретизации» цвета будет составлять $(3? n)!$.

После того как получены новые значения яркостей в R, G, и B компонентах производится кодирование всех их комбинаций степенями двойки. Так комбинация первая

комбинация будет соответствовать значению $2^0 = 1$, следующая комбинация будет соответствовать значению $2^1 = 2$, и т.д. Схематически это можно представить на рис.3.

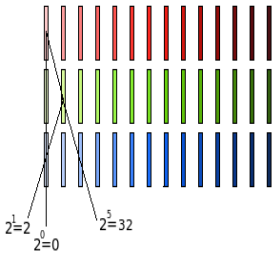
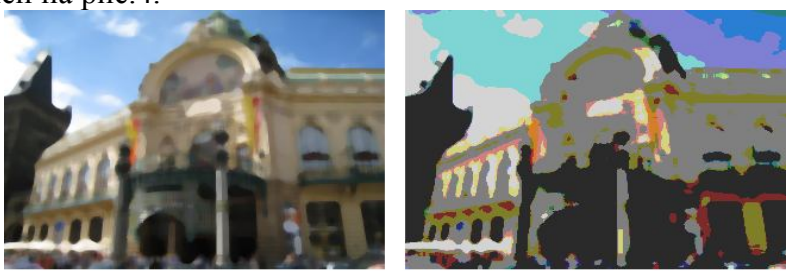


Рис.3. Кодирование новых комбинаций цветов степенями двойки

Полученные коды для комбинаций цветов будут в дальнейшем использованы в процессе создания дескрипторов изображения описываемых ниже. После получения новых значений яркостей в R, G, и B компонентах происходит предескретизация по цвету исходного изображения. Для этого каждый пиксель изображения из своего текущего значения в исходной модели RGB переводится в «урезанную» модель RGB (см. рис.2б). Перевод осуществляется по следующему принципу — для каждой из R, G, и B компонент определяется ее текущее значение. Вычисляется в каком из n диапазонов лежит ее значение и в зависимости от этого ей присваивается значение из середины этого диапазона. Полученные новые значения R, G, и B компонент являются значениями пикселя в передискретизированном изображении. Результат передискритизации изображения по цвету для n=3 представлен на рис.4.



а) изображение после обработки медианным фильтром

б) передискретизированное изображение

Рис.4. Результат передискритизации изображения по цвету для n=3.

Во время процесса разделения на слои происходит попиксельная обработка входного изображения полученного на предыдущем этапе (рис. 4б). При этом входное цветное изображение, полученное после передискритизации по цвету, поддается разбивке на бинарные слои (состоящие из двух цветов — черного (0) и белого(255)). Под слоем будем понимать черно-белое изображение, полученное из исходного в котором все пиксели одного из $(3? n)!$ цветов преобразовываются в белые, а все остальные, не принадлежащие данному цвету, преобразовываются в черные. Всего количество полученных на данном этапе слоев соответствует общему количеству цветов $(3? n)!$ — по одному слою на каждый цвет.

Результат разбиения на бинарные слои для некоторых цветовых кодов (1, 4096, 8192, 67108864) исходного изображения представлен на рис.5а-рис.5г.

Рис. 5. Результат разбиения исходного изображения на бинарные слои для цветовых кодов 1, 4096, 8192 и 67108864.

После того как были получены описанные выше слои, проводится их дальнейшая обработка с целью выделения границ кластеров на каждом слое. Для этого в каждом из слоев последовательно анализируются каждый пиксель до первого информационного пикселя (белый пиксель). При нахождении данного белого пикселя, запускается следующий алгоритм нахождения граничных кластеров — анализируется пространство окном размером 3x3, если в окне находится хотя бы один черный пиксель, то центральный пиксель считается граничным и заносится в массив, описывающий этот кластер. Далее окно сдвигается на следующий найденный белый пиксель и весь процесс повторяется. При прохождении всех пикселей данного кластера, все его найденные белые пиксели затираются черным цветом. После этого слой последовательно анализируется до нахождения следующего кластера. Описанный выше процесс повторяется до тех пор, пока недостигнут конец изображения.

Результат кластеризации для бинарных слоев представленных на рис.5а-рис.5г показан на рис.6а-рис.6г. При этом может быть использован любой другой алгоритм кластеризации отличный от использованного выше[2].

Рис. 6. Результат кластеризации бинарных слоев для цветовых кодов 1, 4096, 8192 67108864.

После этого на выходе алгоритма будет набор массивов границ для каждого кластера по всем изображениям слоям. Каждый из массивов имеет свой уникальный порядковый номер. Дальнейшая задача получить набор дескрипторов на основе координат границ каждого кластера и его цвета с целью описания исходного изображения.

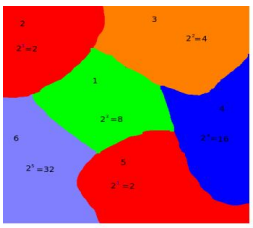
После того как получены массивы границ для каждого кластера, начинается этап формирования из них дескрипторов. Для этого для каждого кластера находят соседние кластера путем анализа граничных пикселей каждого из кластеров. Граничными кластерами считаются те кластера, которые имеют количество точек на общих границах не менее 10.. После нахождения всех соседствующих кластеров и получения кодов их цветов строится хэш (дескриптор) данной кластерной группы. Результирующий дескриптор имеет вид описываемый выражением (1).

$$clu\_num|neig\_1,neig\_2,..., neig\_M|clu\_color\_code,S1,S2...SN \quad (1)$$

где, clu_num — номер кластер в десятичной системе; neig_1,neig_2,...,neig_M — номера кластеров соседей окружающий данный кластер в десятичной системе; clu_color_code —

код цвета данного кластер; S1,S2...SN — сумма закодированных цветов соседних кластеров. При этом если цвета у соседей повторяются, то он разносятся в отдельные подклассы S1...SN.

Рассмотри построение дескрипторов на примере изображения представленного рис. 7а.



а) исходное изображение.

№	Дескриптор.
1	1|2,3,4,5,6|8,54,2
2	2|1,3,6|2,44
3	3|1,2,4|4,26
4	4|1,3,5|16,14
5	5|1,4,6|2,56
6	6|1,2,5|32,12

б) перечень его дескрипторов.

Рис.7. Пример описания кластера для данного изображения.

На данном рисунке: 1, 2, 3, 4, 5, 6 — это номера кластеров в десятичной системе; $2^1=2, 2^2=4, 2^3=8, 2^4=16, 2^5=32$ — коды цветовой гаммы изображения на рис. 7а. Построим дескриптор для кластера № 1. Согласно выражению (3) первым элементом дескриптора будет номер самого кластера, то есть в нашем случае №1. На следующем шаге определяем количество и порядковые номера соседей кластера № 1. В нашем случае это будут следующие кластера - 2, 3, 4, 5, 6. После этого необходимо будет построение так называемой hash-последовательности для цветов кластеров 2, 3, 4, 5, 6 окружающих кластер №1. В нашем случае первый элемент — цвет самого кластера №1 имеющего вид — $2^3=8$. Следующие элементы кода цветов соседних кластеров 2, 3, 4, 5, 6 имеют вид — $2^1+2^2+2^4+2^5, 2^1$. Полученная hash-последовательность при этом будет иметь вид — 8,54,2. Где первый элемент это цветовой код самого кластера №1, второй элемент это сумма цветов кластеров окружающих кластер № 1 и не содержащая повторений цвета, и последний элемент это сумма повторяющихся цветов не попавшая во второй элемент (в нашем случае вторая сумма представляет собой всего один равный 2^1). После всех рассмотренных выше операций имеем следующий дескриптор кластера №1:

$$1|2,3,4,5,6|8,54,2$$

Процесс построения дескриптора проводится по всем имеющимся кластерам. В нашем случае таковыми будут кластера с номерами 2,3,4,5,6. В результате будем иметь список дескрипторов представленных на рис. 7б. Список дескрипторов представленная в рис. 7б это фактически более сжатая форма исходного изображения пригодная для операций сравнения. Для реализации процедуры поиска изображения по контексту посредством использования алгоритма, необходимо провести сравнение набора цветовой hash-последовательностей дескрипторов исходного изображения с цветовой hash-последовательностью дескрипторов сравниваемого изображения.

Выводы.

Описанный выше алгоритм поиска изображений по содержанию на основе использования цветовой кластеризации базируется на построении набора дескрипторов описывающих исходное изображение и дальнейшем их сравнении с дескриптором содержащимися в базе данных изображений на предмет полного или частичного их совпадения. Дескрипторы изображения строятся по цветовым сегментам изображения. При этом в алгоритме не используется размер, площадь, форма сегментов. Вместо этого учитывается лишь их присутствие в определённой последовательности на изображении. Таким образом алгоритм является устойчивость к вращению изображения на различный угол поворота, а так же изменению размеров изображения. Так как при этого рода операциях

кластерная картина изображения не меняется. Имеется возможность поиска части одного изображения в другом изображении. Алгоритм устойчив к так называемому позиционному искажению. В случае если один и тот же объект находится на различных изображениях но под различным ракурсом, алгоритм выявит его.

Литература

1. Яневичь Ю.М. «Задачи приема сигналов и определения их параметров на фоне шумов»: курс лекций / Ю.М. Яневичь - СПбу.
2. Конушин А., Баринова О., Конушин В., Якубенко А., Велижев А. Лекции «Введение в машинное обучение. Часть 3 » М. МГУ ВМК, Graphics & Media Lab, 2009. сл.59-сл.125.
3. How does TinEye work? Режим доступа: http://www.tineye.com/faq#how