электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

 # 09, сентябрь 2015

УДК 00.93'12

Сравнение алгоритмов распознавания, основанных на цветовых параметрах, на примере меню столовой МГТУ. им. Н.Э. Баумана

Бородин И.Д., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
Кафедра «Информационные системы и телекоммуникации»

Научный руководитель: Алфимцев А.Н, к.т.н, доцент.
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
Кафедра «Информационные системы и телекоммуникации»
alfim@bmstu.ru

Введение

Компьютерное зрение на сегодняшний день является одной из самых востребованных областей компьютерных технологий. Оно находит применение в медицине, промышленности, военных целях. Последнее время активно ведутся разработки по созданию автономных транспортных средств [1]. Ярким примером может служить проект Google по созданию беспилотного автомобиля. Ведутся исследования по созданию виртуальной реальности [2]. Также эта область является довольно сложной, с точки зрения алгоритмической обработки результатов [3].

В связи с ростом популярности этой отрасли, появляются средства для простого доступа к базовым методам, алгоритмам. Одним из самых простых способов попробовать компьютерное зрение в действии является пакет Computer Vision System Toolbox [4] для MatLab, который содержит набор средств для проектирования систем компьютерного зрения. Так же есть возможность познакомиться с машинным зрением в среде LabView [5]. Но для этих способов необходимо соответствующее программное обеспечение.

Другим способом является использование библиотек содержащих алгоритмы компьютерного зрения. Таких библиотек немало, они доступны для разных языков (но самым распространенным является C/C++). Самой популярной и часто используемой является библиотека OpenCV [6].

OpenCV — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом. Библиотека

реализована на C/C++. Также имеет реализации для большинства популярных языков программирования, таких как Java, Python, Ruby, Matlab, Lua и других. Преимуществом использования OpenCV является широкий выбор целевой платформы, языка реализации и благодаря тому, что библиотека распространяется свободно, нет ограничений по распространению готовых продуктов с ее использованием.

Довольно популярной темой является распознавание еды. Эта область является сложной, в связи с большим разнообразием пищевых продуктов. Много исследований на эту тему проводится в Японии. В Токийском университете Электро-Коммуникаций предлагаются разные подходы. В работе [7] предлагается использовать для распознавания Bag-of-features (BoF), цветовую гистограмму, градиентную гистограмму и метод Габора, результаты которых совместно обрабатываются с помощью многоядерных вычислений (Multiple Kernel Learning). Эти исследования продолжились, и была создана система для распознавания 85 блюд [8]. В работе [9] предложен метод для распознавания нескольких блюд на изображении. Для начала еда, разделяется на разные регионы, а затем происходит распознавание, используя, цвет, текстуру, градиент и SIFT. В работе [10] описывается мобильная система для распознавания еды. В ней так же для начала используется деление изображения по регионам. Для этого используется алгоритм GrubCut. Для распознавания используются цветовая гистограмма и алгоритм Bag-of-SURF.

Так же довольно много исследований на эту тему проводится в западных университетах. В Стэнфордском университете проводятся исследования [11], в которых опираются на алгоритмы Bag-of-SURF и Spatial Pyramid Matching. В Массачусетском технологическом институте [12] предлагается использовать bag-of-SIFT метод. Так же предложена система [13], которая позволяет определять к какому типу кухни (Американская, Индийская, Итальянская, Тайская или Мексиканская) относится то или иное блюдо. В работе [14] описывается создание 3D моделей блюд.

Так же в последнее время увеличивается интерес к внедрению информационных технологий в медицину. В работе [15] предлагается система для распознавания блюд и расчета калорий.

Недостатком этих работ является сильная национальная составляющая. Кроме того алгоритмы на основе цвета не сильно распространены среди них. Поэтому целью данной работы является анализ существующих алгоритмов и разработка оригинального алгоритма распознавания на основе цвета.

Далее в первом разделе статьи рассмотрим принципы работы популярных алгоритмов MeanShift и CamShift и их применение для распознавания блюд. Далее

подробно рассмотрим работу алгоритма распознавания блюд по цвету. В заключении сравним результаты работы алгоритмов и сделаем выводы.

Создание тестовой выборки

Перед тем как приступать к применению алгоритмов, необходимо создать базу образцов блюд, которые будут использоваться для поиска. Для создания базы использовались фотографии подносов с едой, сделанные на камеру мобильного телефона, в нескольких столовых МГТУ им. Баумана. На рисунке 1 представлены примеры таких изображений. Для каждого образца необходимо будет хранить размер прямоугольной области, соответствующий интересующей нас области на изображении, цветовую гистограмму, среднее значение цвета и дополнительные вспомогательные параметры.

Рис. 1. Примеры обедов студентов МГТУ им. Н.Э. Баумана

На изображении, содержащем образец еды, выделяем участок внутри тарелки. Воспользуемся методом OpenCv ROI. ROI (Region Of Interest — регион интересов — интересующая область изображения) — один из фундаментов OpenCV, который позволяет пользователю задать определённую прямоугольную область на изображении, и алгоритмы, применяемые к этому изображению, будут применяться только к выделенной

области. Далее вычисляем для выделенной области гистограмму. Для этого переводим изображение в цветовое пространство HSV (Hue Saturation Value) и разбиваем на отдельные каналы.

```
CvHistogram templateHueHist = new ColorHistogram().getHueHistogram(templateImage,
minSaturation);
```

Следующим действием вычисляем среднее значение цвета для выделенной области.

```
ColorRGB colorRGB = getMeanColor(templateImage, rect);
```

Сохраняем полученные значения в модель FoodTemplate. Таким образом, этот объект будет хранить в себе всю необходимую информацию, необходимую для алгоритмов поиска и вывода на экран.

```
public class FoodTemplate {

    private String name;

    private Rectangle rect;
    private int minSaturation;
    private CvHistogram templateHueHist;

    private int minDist;
    private ColorRGB targetColor;
    private double coef;
}
```

Алгоритм MeanShift

Принцип алгоритма MeanShift [16] показан на рисунке 2 и заключается в следующем. Имеется набор точек на изображении и окно поиска, находящееся в начальном положении. Определяется центр масс точек на изображении, и окно перемещается к этому центру с некоторым заданным шагом. Это действие повторяется до тех пор, пока центр масс точек на изображении не совпадет с центром окна поиска. При этом размеры этого окна остаются неизменными. На рисунке 3 показано как работает алгоритм на реальном изображении.

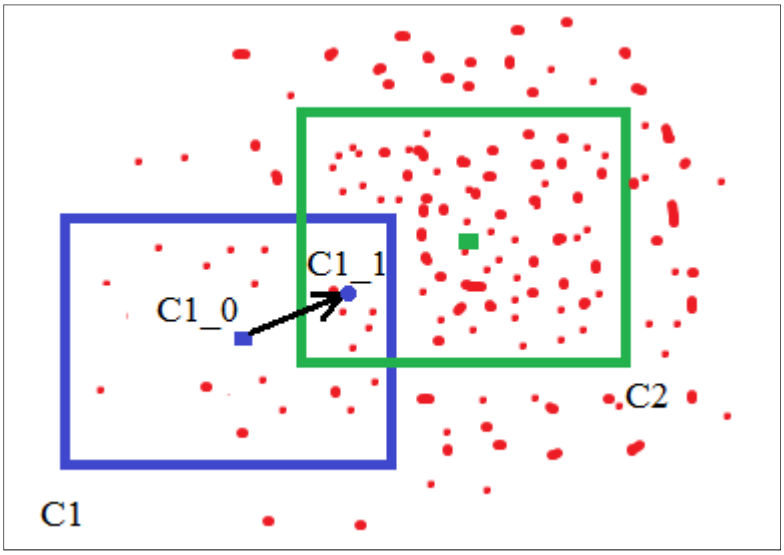

Рис. 2. Принцип работы алгоритма MeanShift

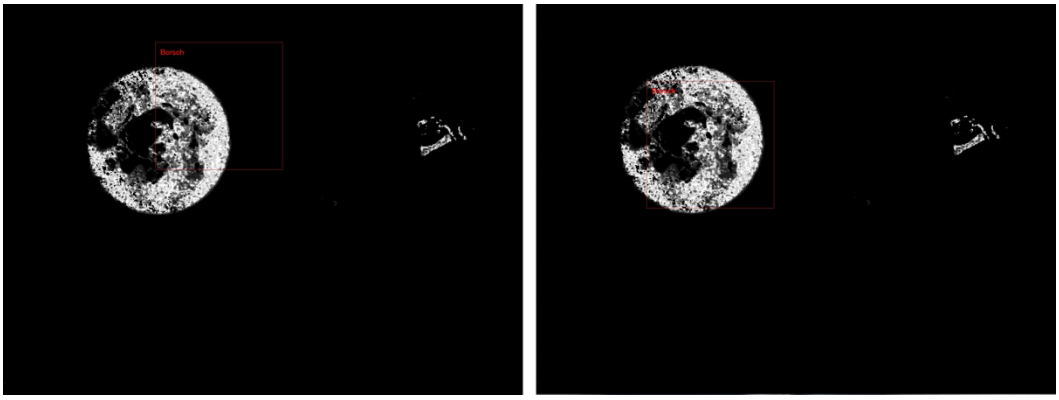

Рис. 3. Применение алгоритма MeanShift

Алгоритм CamShift

Алгоритм CAMshift (Continuously Adaptive Meanshift) [17], содержит в основе алгоритм MenaShift, с той разницей что окно поиска, масштабируется, подстраиваясь под разброс точек [18]. На рисунке 4 показан общий принцип работы, а на рисунке 5 применение алгоритма к реальному изображению.

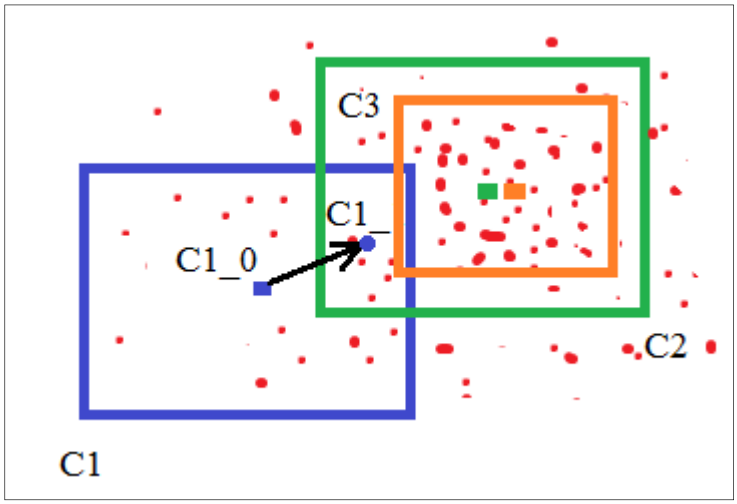

Рис. 4. Принцип работы алгоритма CamShift

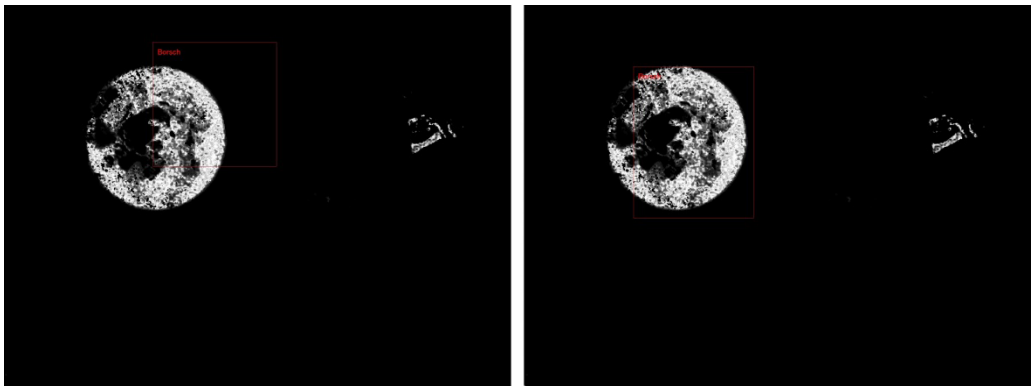

Рис. 5. Применение алгоритма CamShift

Применение алгоритмов

На вход оба алгоритма принимают изображение и образец блюда, поиск которого и будет производиться.

Первым действием входное изображение переводится в пространство hsv и на нем выделяется область соответствующую заданной гистограмме.

Далее к обработанному изображению применяется непосредственно алгоритм CamShift или MeanShift.

После применения этих алгоритмов, необходимо определить круглые области, соответствующим тарелкам на изображении. Для этого воспользуемся алгоритмом Hough Circle [19], встроенным в OpenCV. Для улучшения результатов перед применением алгоритма, дополнительно обработаем изображение. Поиск образцов, будем вести только внутри этих областей.

```
cvtColor(src, gray, CV_BGR2GRAY);
GaussianBlur(gray, gray, size, 1.5, 1.5, BORDER_DEFAULT);

CvMemStorage mem = CvMemStorage.create();
CvSeq circles = cvHoughCircles(gray.asIplImage(), mem, CV_HOUGH_GRADIENT, 1, 300, 80, 60,
100, 500);
```

Последним действием является проверка, что область найденная с помощью алгоритмов CamShift и MeanShift, находится внутри одной из круглых областей – тарелок, найденных на прошлом шаге.

```
for(int i = 0; i < circles.total(); i++){
   CvPoint3D32f circle = new CvPoint3D32f(cvGetSeqElem(circles, i));

   CvPoint2D32f point = new CvPoint2D32f();
   point.x(circle.x());
   point.y(circle.y());

   CvPoint circleCenter = cvPointFrom32f(point);

   int radius = Math.round(circle.z());

   if (distanceBetweenPoints(rectCenter, circleCenter) < radius) {
      return true;
   }
}
```

Оба алгоритма показывают идентичные результаты, однако зоны, которые они находят на изображении отличались по формам и размерам. Это хорошо видно на рисунке 6. На левом изображении отмечены области, найденные с помощью алгоритма MeanShift, на правом – CamShift.

Рис. 6. Области, найденные с помощью алгоритмов MeanShift и CamShift

Алгоритм поиска по цвету

Для начала представим общий принцип работы алгоритма. На рисунке 7 представлена его блок-схема.

Рис. 7. Блок схема алгоритма распознавания по цвету

Рассмотрим подробно работу эвристического алгоритма поиска по цвету на примере изображения, на рисунке 8. На вход подается изображение, на котором будет вестись поиск, и образец, который будет искаться.

Рис. 8. Входное изображение

Первым этапом поиска, является перевод изображения в черно-белое. В зависимости от того насколько цвет пикселя на изображении, отличается от среднего значения цвета образца он окрашивается в черный или белый цвет. Если разница меньше порогового значения, окрашиваем пиксель в белый цвет, иначе в черный цвет. Пороговое значение, позволяет выбирать приближение, к искомому значению цвета. Таким образом, после этого шага получаем черно-белое изображение, на котором область, наиболее близкая к искомому цвету окрашена в белый.

```
for (int y = 0; y < src.getHeight(); y++) {
   for (int x = 0; x < src.getWidth(); x++) {
      if (distance(src.getColor(x, y), foodTemplate.getTargetColor()) < foodTemplate.getMinDist()) {
         dest.set(x, y, 255);
      }
   }
}
```

На рисунке 9 показано входное изображение после первого этапа, для разных искомых образцов.

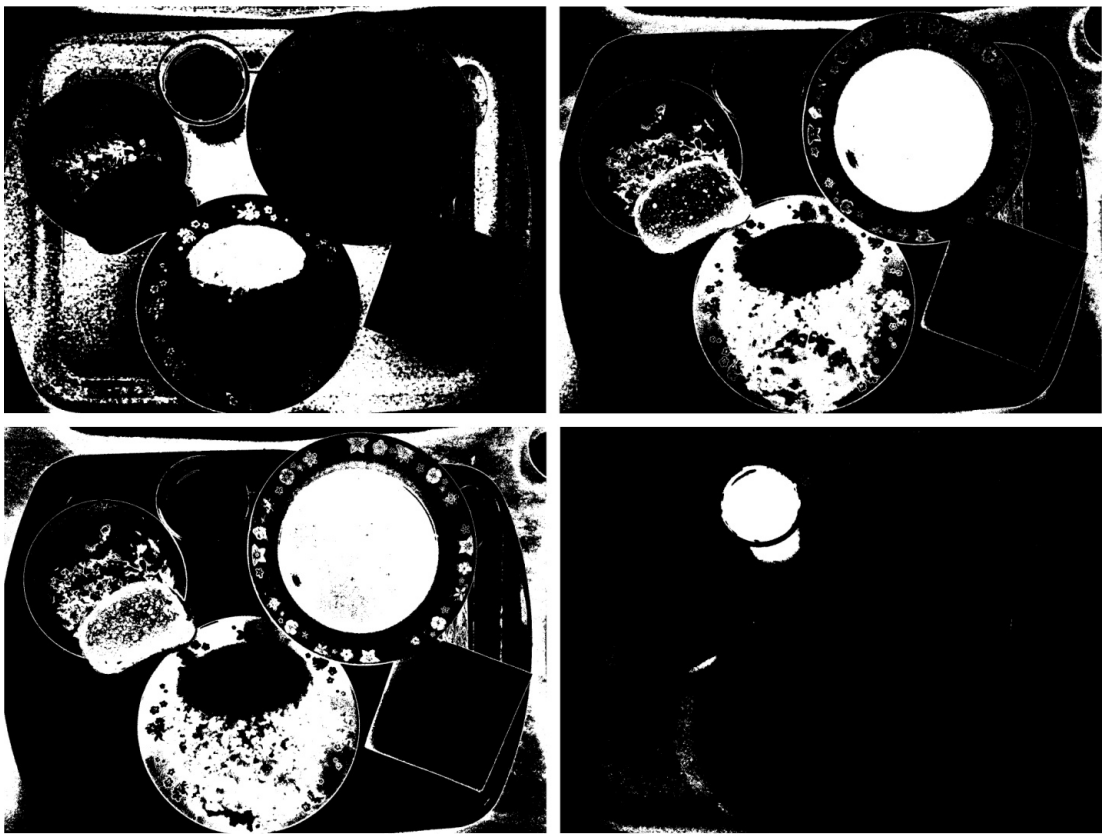

Рис. 9. Входное изображение после перевода в черно-белое представление для разных входных образцов

Следующим шагом является определение круглых областей, соответствующим тарелкам на изображении. Для этого воспользуемся алгоритмом Хафа (Hough Circle), встроенным в OpenCV. Для улучшения результатов перед применением алгоритма, дополнительно обработаем изображение. Поиск образцов, будем вести только внутри этих областей.

cvtColor(src, gray, *CV_BGR2GRAY*);
GaussianBlur(gray, gray, size, 1.5, 1.5, *BORDER_DEFAULT*);
CvMemStorage mem = CvMemStorage.*create*();
CvSeq circles = *cvHoughCircles*(gray.asIplImage(), mem, *CV_HOUGH_GRADIENT*, 1, 300, 80, 60, 100, 500);

Результат выполнения этого этапа представлен на рисунке 10.

Рис. 10. Результат нахождения окружностей (тарелок) на входном изображении

Последним этапом является непосредственно определение области на изображении, соответствующей образцу. Для этого прямоугольную область, которая хранится в модели FoodTemplate, будем перемещать внутри обнаруженных на прошлом этапе кругов, и высчитывать количество белых точек внутри области. Если значение достаточно близко к идеальному, считаем что эта область соответствует искомому образцу, и прекращаем поиск.

На рисунке 11 показан результат выполнения алгоритма. Найденные образцы выделяются и подписываются на изображении.

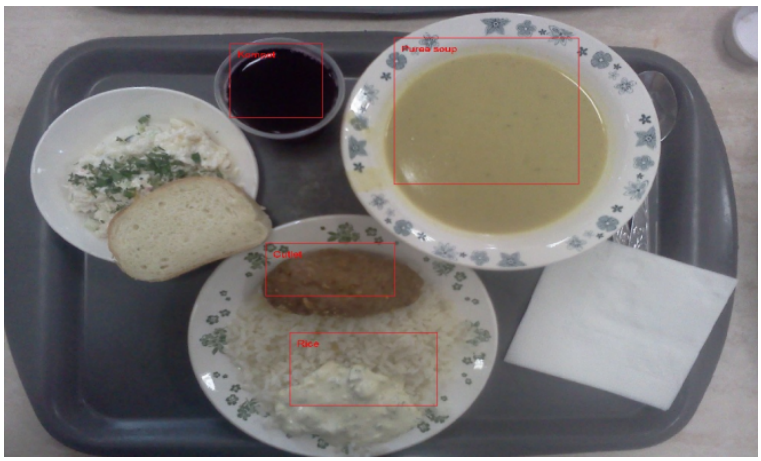

Рис. 11. Результат выполнения поиска по цвету

Тестирование алгоритмов проводилось на выборке из 18 изображений. На рисунке 12 показаны примеры выполнения алгоритма для разных входных изображений. В таблице представлены результаты эксперимента.

Рис. 12. Результаты работы эвристического алгоритма поиска по цвету

№	MeanShift				CamShift				Color Algorithm			
	TP	TN	FN	FP	TP	TN	FN	FP	TP	TN	FN	FP
p1	2	4	2	1	2	4	2	1	4	4	0	0
p2	1	4	2	3	1	4	2	3	2	3	1	0
p3	1	2	2	4	1	2	2	4	0	6	2	0
p4	2	5	0	1	2	5	2	4	0	6	2	0
p5	1	4	2	2	1	4	2	3	1	5	1	1
p6	1	3	2	3	1	3	2	3	2	4	1	1
p7	1	4	1	2	1	4	1	2	1	5	1	0
p8	2	4	0	2	2	4	0	2	2	4	1	0
p9	0	2	2	4	0	2	2	4	0	5	1	1
p10	1	3	1	3	1	3	1	3	0	6	2	0
p11	1	5	2	0	1	5	2	0	2	3	2	1
p12	1	4	1	2	1	4	1	2	0	6	2	0
p13	2	3	0	3	2	3	0	3	0	6	2	0
p14	2	5	0	1	2	5	0	1	0	6	2	0
p15	1	4	0	3	1	4	0	3	0	7	1	0
p16	1	4	1	2	1	4	1	2	0	6	2	0
p17	0	6	1	1	0	6	1	1	0	7	1	0
p18	0	7	0	1	0	7	0	1	0	8	0	0

В таблице представлены результаты работы алгоритмов на тестовой выборке из 18 образцов. Для каждого алгоритма представлены столбцы TP (верно определенные образцы), TN (верно проигнорированные образцы), FN (ошибка первого рода, когда положительный случай, детектируется как негативный) и FP (ошибка второго рода, когда отрицательный случай детектируется как положительный). В первом столбце перечислены номера тестовых изображений. В строке каждого тестового изображения можно увидеть, какой результат показали алгоритмы при применении к нему.

Таким образом, из столбца FP для ColorAlgorithm видно, что эвристический алгоритм поиска по цвету почти исключает ошибку второго рода (отрицательный результат принимается за положительный). Так же можно сказать, что при некоторых условиях (вторая часть выборки) этот алгоритм плохо определяет объекты (столбец TP), что связано с ошибками работы алгоритма Хафа.

Заключение

В работе рассмотрен метод распознавания блюд, входящих в меню столовой МГТУ им. Н.Э. Баумана, основанный на алгоритме определения по цвету. Для обучения достаточно иметь одно изображение образца, и выделить на нем область соответствующую ему. Данный метод позволил почти полностью исключить ошибку второго рода.

В дальнейшем планируется развить процедуру обучения, путем добавления дополнительных изображений. Кроме того, планируется на основе этой системы, сделать

экспертную систему, которая будет давать рекомендации, по поводу совместимости блюд между собой, на основе выбора пользователей, и медицинских диетических фактов.

Список литературы

1. Чигорин А. А., Конушин А. С. Эксперименты с обучением методов распознавания дорожных знаков на синтетических данных // Инженерный вестник. МГТУ им. Н.Э. Баумана. Электрон. журн. 2013. № 8. DOI:10.7463/0813.0603378.
2. Девайкин П. А., Шикуть А. В. Построение дополненной реальности на примере создания виртуальной примерочной // Инженерный. МГТУ им. Н.Э. Баумана. Электрон. журн. 2012. № 8. DOI: 10.7463/0813.0603378/.
3. Алфимцев А.Н. Нечеткое агрегирование мультимодальной информации в интеллектуальном интерфейсе // Программные продукты и системы. 2011. № 3. С. 44-48.
4. Computer Vision System Toolbox™User's Guide. Available at: http://cn.mathworks.com/help/pdf_doc/vision/vision_ug.pdf (accessed 07.10.2014).
5. Визильтер Ю. В., Желтов С. Ю., Князь В. А., Ходарев А. Н., Моржин А. В. Обработка и анализ цифровых изображений с примерами на LabVIEW IMAQ Vision. М.: ДМК Пресс, 2007. 464 с.
6. Dawson-Howe K. A Practical Introduction to Computer Vision with OpenCV. John Wiley & Sons Ltd., 2014. 234 p.
7. Joutou T., Yanai K. A food image recognition system with multiple kernel learning // IEEE International Conference on Image Processing. 7-10 Nov 2009. P. 285–288. DOI: 10.1109/ICIP.2009.5413400.
8. Hoashi H., Joutou T., Yanai K. Image recognition of 85 food categories by feature fusion // International Symposium on Multimedia. 13-15 Dec 2010. P. 296–301. DOI: 10.1109/ISM.2010.51.
9. Matsuda Y., Hoashi H., Yanai K. Recognition of multiple-food images by detecting candidate regions // IEEE International Conference on Multimedia and Expo. 09-13 July 2012. P. 1554–1564. DOI: 10.1109/ICME.2012.157.
10. Kawano Y., Yanai K. Real-time mobile food recognition system // IEEE Conference on Computer Vision and Pattern Recognition Workshops. 23-28 June 2013. P. 1-7. DOI: 10.1109/CVPRW.2013.5.
11. Konaje N.K. Food recognition and calorie extraction using Bag-of-SURF and Spatial Pyramid Matching methods / Computer Science department Stanford University. Available

at: http://cvgl.stanford.edu/teaching/cs231a_winter1415/prev/projects/food.pdf, accessed 17.10.2014.

12. Baxter J. Food recognition using ingredient-level features. Massachusetts Institute of Technology. Available at: http://jaybaxter.net/6869_food_project.pdf, accessed 07.10.2014.
13. Bettadapura V., Thomaz E., Parnami A., Abowd G.D., Essa I. Leveraging context to support automated food recognition in restaurants // IEEE Winter Conference on Applications of Computer Vision. 5-9 Jan 2015. P. 580-587. DOI: 10.1109/WACV.2015.83.
14. Puri M., Zhu Z., Yu Q., Divakaran A., Sawhney H. Recognition and Volume Estimation of Food Intake using a Mobile Device // IEEE Workshop on Applications of Computer Vision. Dec 2009. P. 1-8.
15. Pouladzadeh P., Villalobos G., Almaghrabi R., Shirmohammadi S. A Novel SVM Based Food Recognition Method for Calorie Measurement Applications // IEEE International Conference on Multimedia and Expo Workshops. 9-13 July 2012. P. 495 - 498. DOI: 10.1109/ICMEW.2012.92.
16. Mean Shift Theory. Avaliable at http://www.cse.psu.edu/~rcollins/CSE598G/introMeanShift_6pp.pdf, (accessed 12.10.2014).
17. CAMSHIFT Tracking Algorithm. Available at: http://www.gergltd.com/cse486/project5/, accessed at 12.10.2014).
18. John G. Allen, Richard Y. D. Xu, Jesse S. Jin Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces // In VIP '05: Proceedings of the Pan-Sydney area workshop on Visual information processing. 2004. P. 3-7.
19. Borovi˘cka J. Circle Detection Using Hough Transforms Documentation. Available on: https://files.nyu.edu/jb4457/public/files/research/bristol/hough-report.pdf, accessed 17.10.2014.