Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Санкт-Петербургский государственный университет»

Кафедра системного программирования

Русинов Павел Александрович

Создание алгоритма детектирования объектов на основе разномасштабных haar-классификаторов

Бакалаврская работа

Допущен к защите.

Зав. кафедрой:

д. ф.-м. н., профессор Терехов А. Н.

Научный руководитель:

к. ф.-м. н., доцент Вахитов А.Т.

Рецензент:

Гуревич Л.С.

Санкт-Петербург

2015

SAINT-PETERSBURG STATE UNIVERSITY

Department of Software Engineering

Rusinov Pavel

Object detection algorithm based on multiscale haar-classifiers

Bachelor's Thesis

Admitted for defence.
Head of the chair:
professor A. Terehov

Scientific supervisor:
doc. A. Vakhitov

Reviewer:
L. Gurevich

Saint-Petersburg
2015

Оглавление

Введение

Алгоритм детектирования объектов на изображениях, использующий признаки Хаара, разработанный Виолой и Джонсом зарекомендовал себя как довольно точный и быстрый. Он находит применение при детектировании лиц[1], пешеходов[5], номеров машин[6], также используется в сочетании с другими методами компьютерного зрения[7]. В случае работы с изображениями высокого разрешения при ограниченных вычислительных ресурсах (возможно мобильных или встраиваемых системах) скорость работы детектора может стать препятствием для реализации работы системы в реальном времени. К настоящему моменту предложены некоторые методы увеличения вычислительной эффективности алгоритма и способы их применения: [2],[3],[4]. В [4] рассматривается построение системы видео-наблюдения в высоком разрешении. Для ускорения детектирования лиц предлагается выделение переднего плана и фильтрация по цвету, после чего область поиска детектора ограничивается найденными областями. В [3] модифицировали алгоритм обучения детектора для уменьшения числа слабых классификаторов в конечном классификаторе, это также позволяет ускорить алгоритм.

В данной работе предлагается увеличить производительность детектора за счет прохода классификатором первой стадии по разреженной сетке. Окна которые были допущены на первой стадии, а также соседние для них обрабатываются последующими стадиями. Классификатор первой стадии обучается с использованием уменьшенного числа признаков Хаара, которые являются устойчивыми к сдвигу, таким образом позволяя ему срабатывать на окнах в которые искомый объект попал не целиком.

Модифицированный алгоритм предполагается использовать при детектировании номеров машин.

Постановка задачи

Целью данной работы является оптимизация детектора Виолы-Джонса, позволяющая ускорить его работу за счет разреженного прохода скользящим окном. Основываясь на изложенной выше цели, были поставлены следующие задачи.

1. Изучить схожие по теме работы
2. Разработать модификацию детектора Виолы-Джонса
3. Изучить исходный код OpenCV и реализовать предложенный подход
4. Подготовить тестовую и обучающую выборки
5. Обучить исходный и модифицированный детектор
6. Сравнить их производительности

Обзор литературы

На данный момент предложено несколько методов увеличения производительности исходного алгоритма Виолы-Джонса.

В [3] предлагается использовать эволюционные алгоритмы для оптимизации ансамбля классификаторов. В качестве основы используются AdaBoost и каскадная структура. В методе обучения акцент делается на вес каждого слабого классификатора, а не на веса тренировочных примеров. В конечном итоге для заданных detection rate и false positive rate должен быть найден минимальный набор классификаторов. Для этого после AdaBoost добавляется дополнительный шаг: evolutionary pruning. Слабые классификаторы, выбранные AdaBoost, могут иметь зависимости. С помошью evolutionary pruning выполняется поиск избыточных слабых классификаторов, чтобы таким образом уменьшить их общее число при сохранении заданных параметров качества детектирования. Тестовый набор состоял из 6000 изображений лиц и 6000 изображений их не содержащих. Удалось достичь уменьшения времени работы алгоритма до 57.6% относительно исходного (Виола-Джонс), при этом detection rate оставался на том же уровне. Эффективность данного подхода зависит от того насколько много избыточных слабых классификаторов было после применения AdaBoost и в общем случае позволяет ускориться не более чем в 2 раза.

В [4] область поиска для детектора ограничивается выбранными регионами, уменьшение области поиска выполняется за счет препроцессинга, состоящего из двух шагов: цветовой сегментации и выделения переднего плана. Вначале берется разреженная сетка пикселей изображения для того чтобы уменьшить время поиска. Цветовая сегментация и выделение переднего плана используются для поиска регионов изображения с возможным присутствием лиц, затем детектор (Виола-Джонс) используется чтобы подтвердить или опровергнуть эти гипотезы. Эксперименты показали,

что предложенный метод уменьшает время детекции до 42 ms, в то время как исходный алгоритм (Viola and Jones) требует 565 ms. Такое улучшение позволяет использовать алгоритм в приложениях реального времени на изображениях высокого разрешения. Недостатком данного алгоритма является необходимость цветного изображения.

В [7] используется Haar/AdaBoost + HOG/libSVM система для детектирования пешеходов. При обработке изображения выполняются следующие шаги: препроцессинг, первичное детектирование haar-классификатором, окончательное детектирование HOG. Препроцессинг подразумевает преобразование входного изображение из цветного в черно-белое, эквализацию для увеличения контраста и затем уменьшение в два раза. На полученном низком разрешении работает haar, затем сдетектированные участки изображения обрабатываются HOG/libsvm классификатором на изображении исходного размера. По сравнению с исходным методом (только HoG/libSVM) удалось сократить время работы до 33% при сохранении примерно такой же точности распознавания.

В [8] алгоритм Виолы и Джонса реализован для мобильных и встраиваемых систем (используются арифметические операции с числами с фиксированной точкой, вместо чисел с плавающей). Библиотека OpenCV была использована как базовая платформа. Удалось добиться увеличения производительности с 3.81 fps до 5.84 fps для разрешения 352x288 и с 1.5 до 2 для 512x512. Данный подход дает одинаковый в процентном отношении прирост производительности для любого каскада и разрешения.

Таким образом в [4], [7] перед работой основного детектора добавляется дополнительный шаг препроцессинга, результаты работы которого используются далее, что позволяет уменьшить время работы. В [3] выполнятся оптимизация самого детектора, уменьшающая количество слабых классификаторов. В [8] предложено модифицировать реализацию.

Ни в одной из работ не было рассмотрено препроцессинга с использованием разреженного прохода скользящим окном на первой стадии каскада, также имеются ограничения по приросту производительности и случаям использования. Поэтому данный подход был изучен в этой работе.

Алгоритм

Детектор Виолы-Джонса

Виола и Джонс разработали алгоритм детектирования объектов, который использует признаки Хаара. Детектор представляет из себя каскад классификаторов, каждый из которых является комитетом слабых классификаторов. В качестве слабого классификатора используется признак Хаара с некоторым пороговым значением. Входное изображение обрабатывается с помощью метода скользящего окна.

1. Метод скользящего окна: подразумевает проход по всему изображению окном размера, соответствующего размеру входа для детектора с некоторым шагом. При каждом сдвиге окна часть изображения, попавшая в это окно обрабатывается детектором. В случае срабатывания детектора считается, что обнаружен искомый объект. Так как искомые объекты могут быть разных размеров входное изображение масштабируется в большую и меньшую стороны и также обрабатывается.

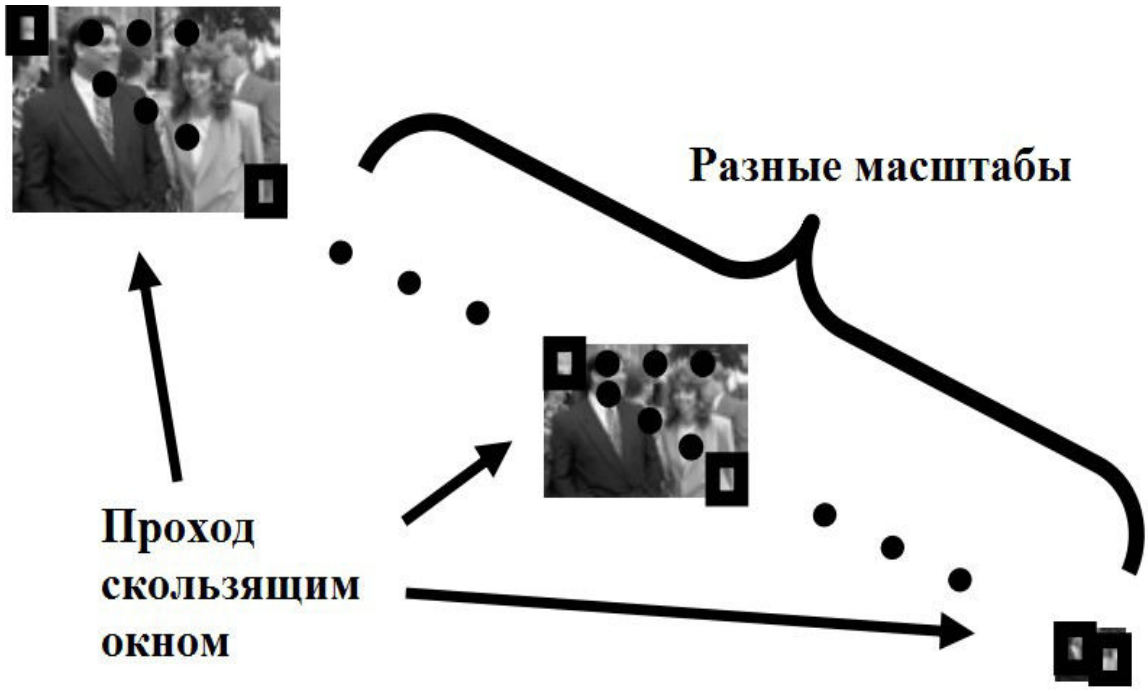

Рис. 1: Метод скользящего окна

2. Признаки Хаара: Каждый признак представляется некоторым шаблоном, координатами относительно координат окна поиска и размером. Некоторые шаблоны представлены на рис. 1. Значением признака является взвешенная сумма двух компонент: суммы значений пикселей черного прямоугольника и суммы значений пикселей всей области признака; веса двух компонент противоположны по знаку, а абсолютные значения обратно пропорциональны их площади.

$$HAAR = BC \cdot \sum_{pix(i,j) \in B} pix(i,j) - WC \cdot \sum_{pix(i,j) \in W} pix(i,j)$$

B – черный прямоугольник, BC – соответствующий коэффициент, W, WC – аналогично для белого прямоугольника.

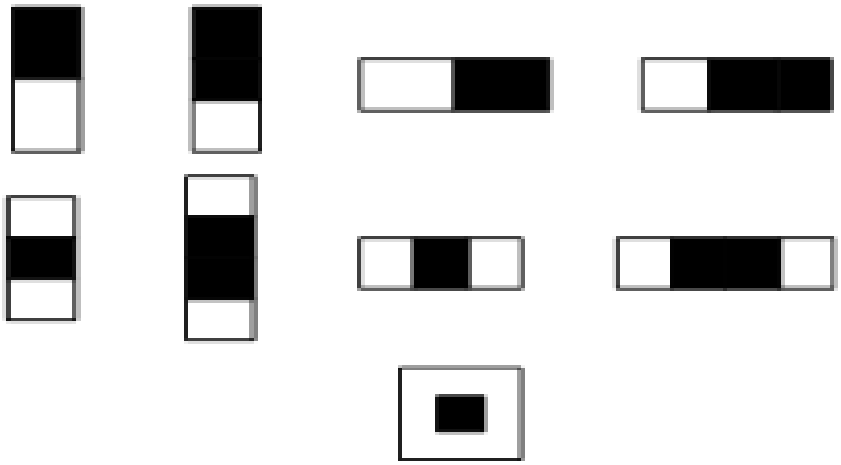

Рис. 1: Некоторые шаблоны признаков Хаара

3. Для быстрого вычисления значения признака Хаара используется интегральное представление изображения. Интегральное изображение вычисляется следующим образом: каждый пиксель содержит сумму значений всех пикселей исходного изображения, которые расположены выше и левее. Таким образом сумму пикселей исходного изображения в любой прямоугольной области можно вычислить за 4 обращения к интегральному изображению: I(bl) – I(br) – I(tl) + I(tr), где I(.) - значение интегрального изображения в пикселе, bl – левый нижний пиксель прямоугольной области, br – правый нижний, tl – верхний левый, tr – верхний правый.

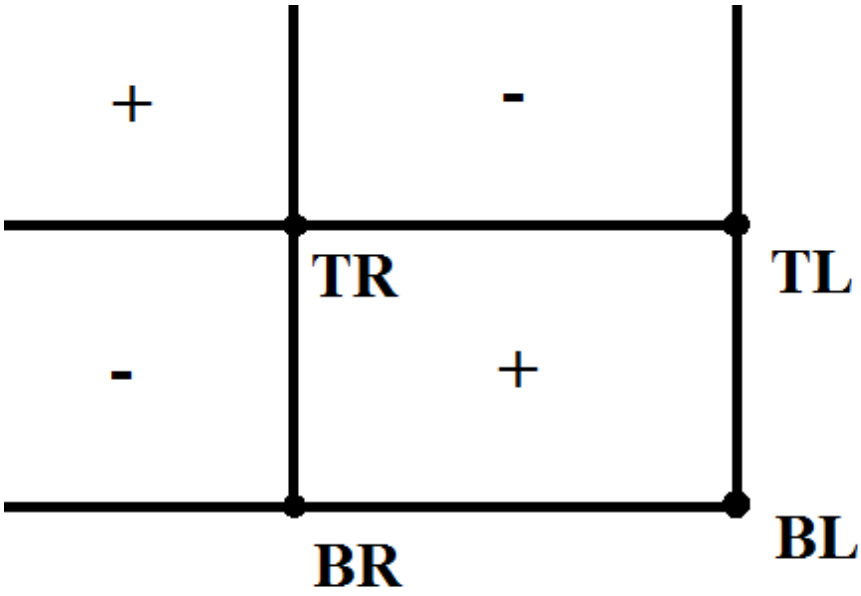

Рис. 3: Вычисление суммы пикселей в прямоугольной области за 4 обращения к интегральному изображению

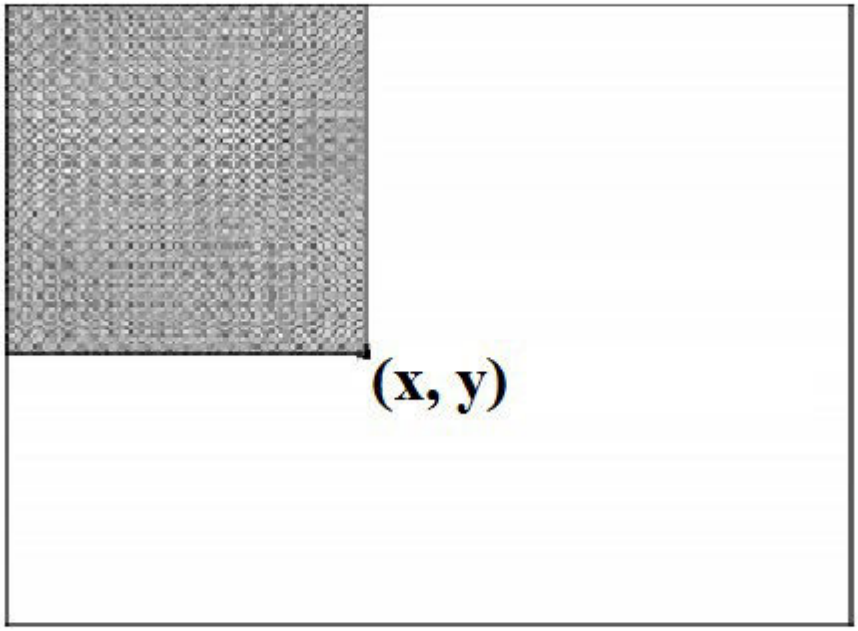

Рис. 4: Вычисление значения в точке (x, y) интегрального изображения

4. AdaBoost алгоритм: Набор всех возможных признаков для окна поиска является избыточным и вычислительно очень затратным. Поэтому для построения классификатора необходимо выбрать небольшое число признаков, которые наилучшим образом разделяют обучающую выборку. Adaptive boosting – алгоритм построения сильного классификатора как линейной комбинации слабых классификаторов.

Даны примеры $(x_{1,}, y_1), \ldots, (x_n, y_n)$, *где* $y_i = 0, 1$
для отрицательных и положительных примеров соответственно.
Инициализировать веса $w_{(1,i)} = \frac{1}{2\mathrm{m}}, \frac{1}{2l}$ *для* $y_i = 0, 1$ *соответственно, где* m *и* l *являются числом отрицательных и положительных примеров соответственно.*
Для $t = 1, \ldots, T$:

1. *Нормализовать веса,* $\frac{w_{(t,i)}}{\sum_{j=1}^{n} w_{(t,j)}} \rightarrow w_{(t,i)}$

2. *Для каждого признака,* j, *обучить классификатор* h_j, *который использует один признак. Ошибка оценивается в соответствии с весом* w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$

3. *Выбрать классификатор* h_t *с наименьшей ошибкой* ϵ_t.

4. *Обновить веса:* $w_{(t+1,i)} = w_{(t,i)} * \beta_t^{(1-\epsilon_i)}$, *где* $\epsilon_i = 0$, *если пример* x_i *классифицирован верно,* $\epsilon_i = 1$ *в противном случае, и* $\beta_t = \frac{\epsilon_t}{1} - \epsilon_t$

Итоговый сильный классификатор:

$$h(x) = \begin{cases} 1, \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t h_t(x) \\ 0, \text{в противном случае,} \end{cases}$$

где $\alpha_t = \log \frac{1}{\beta_t}$

Таблица 1: алгоритм AdaBoost

5. Каскадный детектор: Конечный детектор получается объединением в каскад нескольких классификаторов, обученных с помощью AdaBoost алгоритма. Каждый последующий в каскаде классификатор обучается с некоторой точностью срабатывания, используя в качестве отрицательных примеров те, которые были допущены предыдущим. При детектировании на входном изображении большая часть окон отвергается на первой стадии, что позволяет экономить время. Далее

если окно принимается всеми стадиями каскада, оно отмечается как содержащее искомый объект.

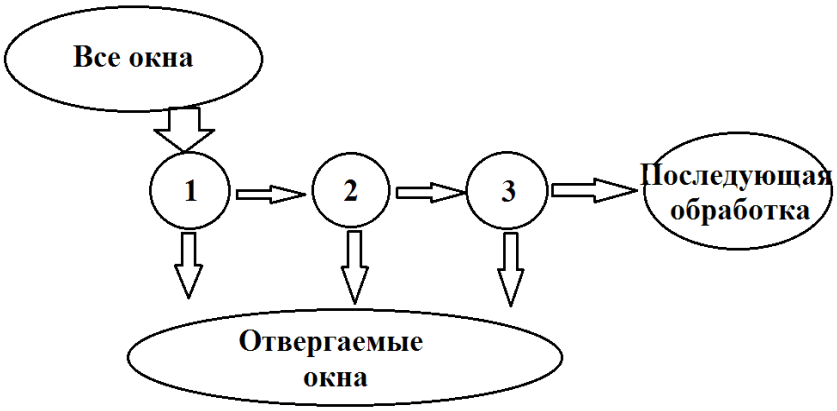

Рис. 5: Схема работы каскадного детектора

Предлагаемые изменения алгоритма

Для уменьшения времени обработки детектором одного изображения было предложено следующее: выполнять проход окном поиска с разреженным шагом для первой стадии каскада; в случае допуска окна первой стадией классификатора последующие стадии обрабатывают как это окно, так и соседние для него.

Чтобы сохранить качество детектирования на том же уровне, классификатор первой стадии необходимо обучить таким образом, чтобы он при разреженном проходе не пропустил искомые объекты, то есть он должен допускать окна в которые искомый объект попал не целиком, оказался сдвинут. Поэтому классификатор первой стадии предлагается обучать с использованием признаков Хаара, значение которых при сдвиге окна изменяется меньше. Чтобы найти такие признаки возможно проделать следующее: взять несколько изображений, соответствующих тем, на которых будет в дальнейшем работать детектор (в случае детектирования номеров машин отличия только в проезжающих машинах, фон, занимающий существенную часть — почти одинаковый, поэтому можно взять небольшое

число примеров) и вычислить среднее разницы значений признака в соседних окнах для каждого изображения после чего взять среднее значение по всем изображениям. Вычисленные значения для каждого признака делятся на его площадь, так как большие по площади признаки имеют большее среднее разницы. То есть из всех возможных признаков для данного окна выбираются наиболее устойчивые к сдвигу, после чего AdaBoosting работает только с ними и строит классификатор первой стадии. Классификаторы последующих стадий обучаются по исходному алгоритму.

Реализация

В качестве базовой реализации алгоритмов обучения и детектирования были взяты модули, содержащиеся в библиотеке компьютерного зрения с открытым исходным кодом OpenCV.

К алгоритму детектирования, содержащемуся в модуле opencv_objdetect, была реализована модификация, позволяющая включать на первой стадии разреженный проход, обрабатывая соседние окна при допуске на этой стадии.

В модуле opencv_traincascade была реализована функциональность, позволяющая вычислять среднее значение полной вариации по нескольким изображениям, деленное на площадь признака. Также реализованы методы для только признаков с наименьшими полученными значениями в обучении первой стадии каскада. Так как при обучении последующих стадий используются предыдущие было добавлено отображение номеров признаков, соответствующих выбранным из устойчивых в номера, соответствующие всем.

Была реализована программа для тестирования, которая загружает детекторы, видео, обрабатывает покадрово каждым из детекторов видео и замеряет среднее время обработки кадра.

Эксперименты

В качестве объектов детектирования для изучения свойств предложенной модификации алгоритма были выбраны номера машин.

Выбор устойчивых признаков

Был выполнен подсчет среднего разницы для шаблонов признака Хаара. Для этого были использованы следующие данные: 10 изображений дороги с проезжающими машинами (5 ночных и 5 дневных), размер изображений 2752x700, размер окна 48x12, взяты все шаблоны входящие в такое окно. На рис. 6 — распределение получившихся значений.

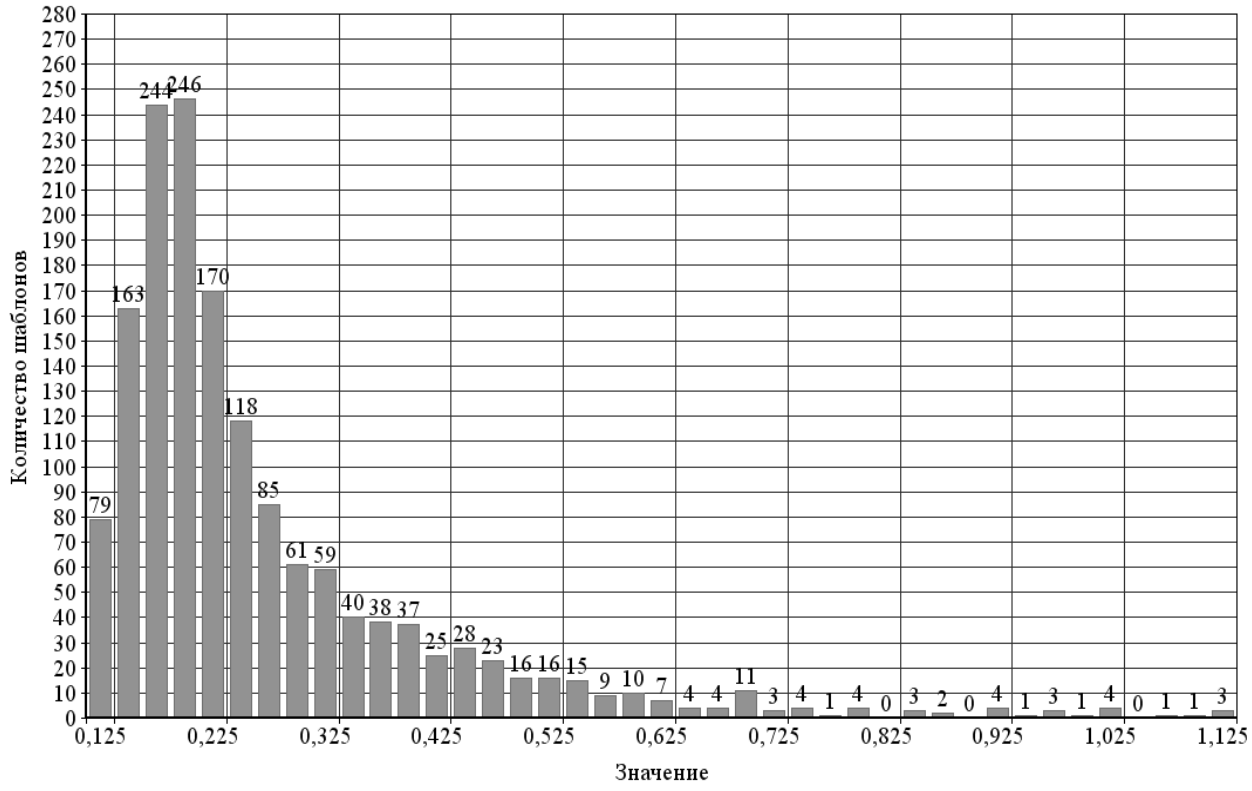

Рис. 6: Полученное распределение значений среднего разности для шаблонов признаков

Обучение детекторов

Обучение проводилось с помощью с помощью модуля изOpenCV

opencv_traincascade, в котором была реализована возможность использовать только некоторые признаки Хаара при обучении первой стадии каскада.

Были обучены детекторы, классификаторы первой стадии которых использовали 8%, 15%, 30% и 45% шаблонов признаков с наименьшими значениями (выбраны исходя из результатов описанного выше эксперимента), был обучен детектор с использованием исходного алгоритма. Каждый каскад состоял из 5 классификаторов и обучался со следующими параметрами: minHitRate = 0.995 (минимальное число допускаемых положительных примеров из тренировочной выборки при обучении классификатора стадии), maxFalseAlarm = 0.25 (максимальное число допускаемых отрицательных примеров из тренировочной выборки при обучении классификатора стадии, в каждой последующей стадии рассматриваются только примеры, допущенные на предыдущей), maxWeakCount = 100 (максимальное число слабых классификаторов в стадии каскада). В качестве тренировочной выборки было взято 4575 изображений номеров, 22000 изображений без номеров (части изображений дороги с проезжающими машинами). Также с использованием тех же параметров был обучен детектор для окон вдвое меньшего размера относительно исходного (размер который используют все детекторы, упомянутые выше).

Тестирование

Все детекторы были протестированы на 2 видео разрешением 2448x2050: ночном из 2001 кадра, содержащем 4448 номеров и дневном из 2500 кадров, содержащем 5885 номеров (каждый кадр может содержать 0 — 7 номеров). Пример кадра из ночного видео представлен на рисунке 7. Полученные результаты представлены в таблице 1. Модифицированный метод детектирования показал увеличение производительности примерно в 3 раза. Детекторы, обученные с использованием устойчивых признаков оказались лучше детектора (original cascade sparse), обученного с использованием всех

признаков.

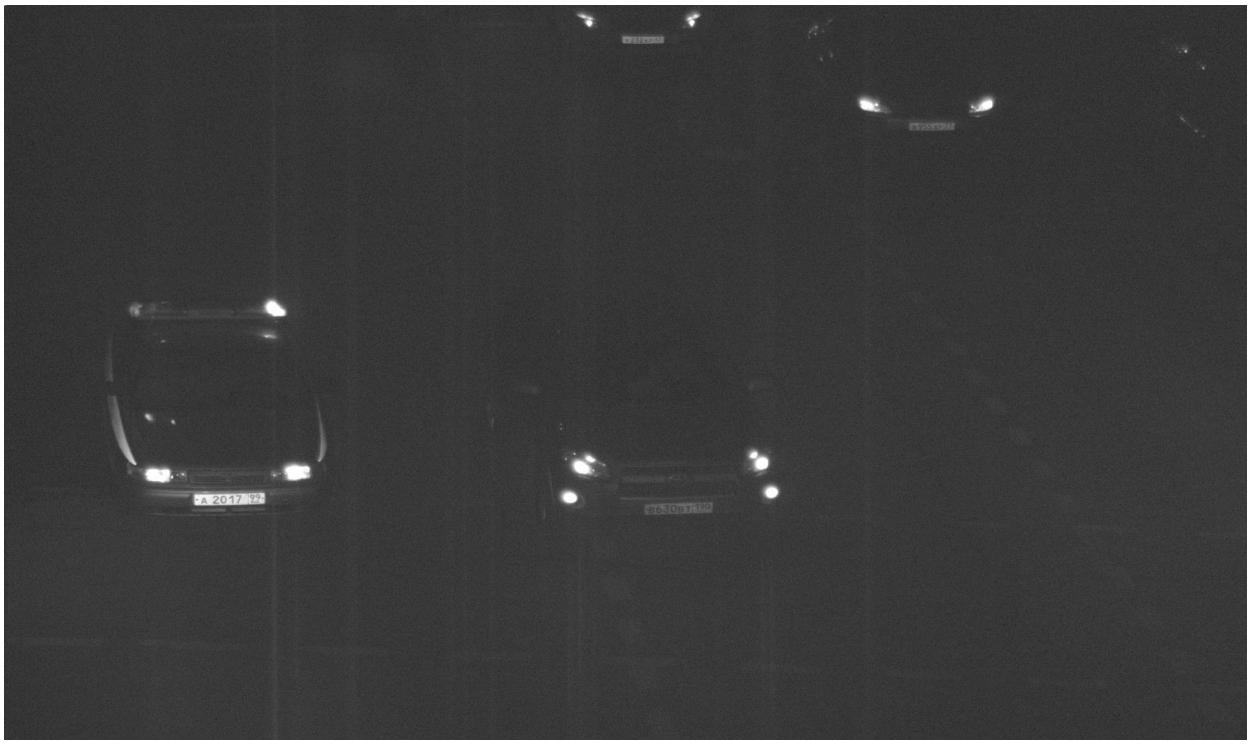

Рис. 7: пример кадра из ночного видео

Результаты

Дневное видео	HitRate	FalseAlarm	Среднее время
Cascade 8%	0.962076	0.000553113	0.546745
Cascade 15%	0.971113	0.000722372	0.465474
Cascade 30%	0.967035	0.000540243	0.527026
Cascade 45%	0.95582	0.000440805	0.51293
Original cascade sparse	0.946984	0.000975156	0.426023
Original cascade	0.988275	0.000387577	1.30343
Small cascade	0.937617	0.00088331	0.417890

Таблица 2.1: результаты на дневном видео

Ночное видео	HitRate	FalseAlarm	Среднее время
Cascade 8%	0.961341	0.000414019	0.522376
Cascade 15%	0.96173	0.00062928	0.43336
Cascade 30%	0.962811	0.000415254	0.511232
Cascade 45%	0.948757	0.00034693	0.496322
Original cascade sparse	0.930595	0.000842991	0.420927
Original cascade	0.988541	0.000338668	1.30537
Small cascade	0.921486	0.000837546	0.41233

Таблица 2.2: результаты *на ночном видео*

В таблицах 2.1, 2.2 представлены результаты тестирования детекторов: cascade 8%, 15%, 30%, 45% - обучены с соответствующим числом устойчивых признаков; original cascade sparse – обучен по исходному алгоритму, работает по модифицированному, использующему разреженный проход на первой стадии каскада; original cascade, cascade small – обучение и детектирование работают по исходному алгоритму, но cascade small для вдвое меньших окон. HitRate – процент сдетектированных номеров из тестовой выборки, FalseAlarm – отношение числа сдетектированных окон, не содержащих номеров к общему числу обработанных окон. Среднее время — среднее время обработки одного кадра видео.

Сравнение с базовым алгоритмом

По результатам проведенного тестирования можно сделать вывод о том, что разреженный проход на первой стадии каскада (используется в cascade 8%, cascade 15%, cascade 30%, cascade 45%, original cascade sparse) позволяет уменьшить среднее время обработки одного кадра примерно в 3 раза по сравнению с детектором, работающим в обычном режиме (original cascade).

При этом детекторы, при обучении первой стадии которых использовались устойчивые признаки (cascade 8%, cascade 15%, cascade 30%, cascade 45%) показывают компенсацию падения качества детектирования по сравнению с каскадом, обученным по исходному методу (original cascade sparse). Детектор, работающий с окнами, уменьшенными вдвое, также как и детекторы, использующие разреженный проход работает примерно в 3 раза быстрее, но уступает им в качестве детектирования.

Заключение

В работе представлен алгоритм детектирования объектов на основе разномасштабных haar-классификаторов, являющийся модификацией алгоритма Виолы-Джонса. Данный алгоритм предполагает обучение первой стадии детектора с использованием устойчивых признаков Хаара и разреженный проход на этой стадии при детектировании.

Реализовано обучение детектора с использованием устойчивых признаков Хаара. Было проведено тестирование модифицированного алгоритма на двух видео, объекты детектирование — номера машин. Удалось показать, что такой подход дает ускорение примерно в 3 раза, а использование устойчивых признаков позволило добиться некоторой компенсации потери качества. В дальнейшем можно опробовать алгоритм при детектировании других объектов.

Литература

1. P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 1. Kauai, HI, USA: IEEE Comput. Soc, 2001, pp. I–511–I–518.

2. H. Schneiderman, “Feature-centric evaluation for efficient cascaded object detection,” in Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Confer-ence on, vol. 2, 2004, pp. II–29–II–36 Vol.2

3. J. Jun-Su and K. Jong-Hwan, “Fast and robust face detection using evolutionary pruning,” Evolutionary Computation, IEEE Transactions on, vol. 12, no. 5, pp. 562–571, 2008

4. Cheng, Xin, Lakemond, Ruan, Fookes, Clinton B., & Sridharan, Sridha (2012) Efficient real-time face detection for high resolution surveillance applications. In 6th International Conference on Signal Processing and Communication Systems (ICSPCS’2012), 12 - 14 December 2012, Radisson Resort, Gold Coast, Qld.

5. Q. Chen, N. D. Georganas, and E. M. Petriu. Real-time vision-based hand gesture recognition using haar-like features. In Proc. of the IEEE Instrumentation and Measurement Technology Conf., pages 1 – 6, May 2007.

6. L. Dlagnekovin License Plate Detection Using AdaBoost .La Jolla: Comput. Sci. Eng. Dept., Univ. California San Diego, Mar. 2004.

7. W. Xing, Y. Zhao, R. Cheng, J. Xu, S. Lv, X. Wang, “Fast Pedestrian Detection Based on Haar Pre-Detection”, International Journal of Computer and Communication Engineering, Vol. 1, pp. 207-209, 2012

8. S. Boggarapu, T.Sreenivasu, “An Optimized Implementaion of Haar like Feature Based Object Detection”, International Journal of Innovative Research and Development, Vol. 2, pp. 37-47, 2013