Быстрый Алгоритм Обнаружения Пешеходов по Видеоданным

Алексей Казаков, Александр Бовырин
Факультет вычислительной математики и кибернетики
Нижегородский государственный университет им. Н.И. Лобачевского, Нижний Новгород, Россия
kazakovdz@yandex.ru, alexander.bovyrin@itseez.com

Абстракт

В работе представлен алгоритм поиска пешеходов на изображении в реальном времени. Алгоритм основан на применении метода классификации AdaBoost совместно с HOG признаками, при этом размер окна поиска динамически изменяется в зависимости от положения линии горизонта на изображении.

Ключевые слова: *обнаружение пешеходов, каскадный алгоритм, HOG признак, линейная интерполяция.*

1. ВВЕДЕНИЕ

Задача поиска людей на изображении является важнейшей составляющей разнообразных прикладных систем компьютерного зрения. Подобные системы можно разделить на 3 группы: (1) системы видеонаблюдения, в которых камера зафиксирована и следит за определённым участком местности; (2) системы помощи водителю в управлении автомобилем, в котором установлена камера, направленная на дорогу по ходу движения; (3) системы поиска роботом человека, в которых камера встроена в робота и её положение относительно системы координат робота может изменяться. Далее для удобства будем использовать термин «пешеход» вместо более общего «человек», тем самым выделяя основной способ применения нашей системы, в качестве составляющей комплексов помощи водителю в управлении автомобилем.

Обнаружение пешеходов считается сложной и до сих пор не решённой задачей компьютерного зрения. Причиной этого является широкое разнообразие во внешнем виде пешеходов, зависящем от множества факторов, таких как: поза, одежда, освещение, фон. В качестве алгоритма машинного обучения в системе обнаружения пешеходов применялся AdaBoost на основе деревьев заданной высоты. В качестве тренировочных данных мы использовали базу “Daimler Chrysler Benchmark Dataset”, состоящую из 15660 изображений пешеходов размером 48x96 пикселей и 6744 изображений не содержащих пешеходов. Тестовая база была представлена 27-ми минутным видео, которое было снято с камеры установленной на автомобиле и разбито на 21790 изображений 640x480. Каждому тестовому изображению соответствует запись в файле с разметкой, содержащей информацию о количестве, положениях и размерах пешеходов, присутствующих на изображении. Полностью видимые, вертикально стоящие пешеходы, на расстояниях от 12 до 37 метров, помечены в разметке как те, обнаружение которых обязательно.

2. ОБЗОР СИСТЕМ ОБНАРУЖЕНИЯ ПЕШЕХОДОВ

Приведём краткий обзор нескольких, на наш взгляд, наиболее значимых работ. С более полным обзором работ по обнаружению пешеходов можно ознакомиться в статьях [2] и [4]. Кроме того, в работе [2] приводится сравнение качества и скорости работы, описанных систем на базе “Daimler Chrysler Benchmark Dataset”.

2.1 Каскадный алгоритм основанный на Haar wavelet признаках

Авторы работы [2] использовали для обнаружения пешеходов каскадный алгоритм Виолы - Джонса, основанный на Haar wavelet признаках. Подробное описание этого метода приведено в работе [8]. Кратко опишем суть каскадного алгоритма. В алгоритме поиск пешеходов осуществляется сканированием изображения скользящим окном (окном обнаружения), плотно заполненном Haar wavelet признаками. В алгоритме используются признаки различных типов, различных масштабов и положений (рис. 1). Внутри окна обнаружения таких признаков генерируется более 100,000.

Рисунок 1: Типы Haar wavelet признаков. Чёрная закраска соответствует положительному весу, белая - отрицательному.

Итоговый классификатор представляет собой цепочку уровней (рис. 2), каждый из которых тренируется с помощью алгоритма AdaBoost [3] на Haar-вейвлетах . В качестве слабого классификатора, используемого в AdaBoost, применялись деревья решений единичной высоты. Критерий качества каждого уровня – 50% ложных срабатываний при 99.5% обнаруженных пешеходов. Натренированный каскадный классификатор состоит из 15 уровней, использующих от 15 до 727 признаков.

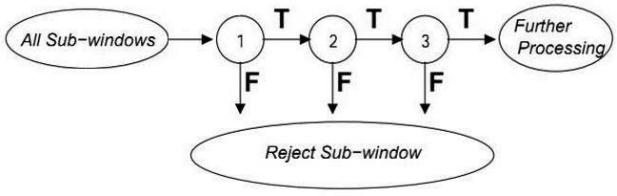

Рисунок 2: Схема каскадного классификатора. Каждое окно обнаружения проходит через цепочку уровней, каждый из которых отсекает большую долю негативных примеров. На следующие уровни поступают окна, не отсечённые на предыдущем. Критерии отсечения уровней задаются при тренировке.

2.2 HOG алгоритм

В 2004 году в работе [1] было предложено использовать HOG (Histograms of Oriented Gradients) дескрипторы в задаче обнаружения пешеходов. Идея этого подхода в том, что форма и вид объектов на изображении могут хорошо

описываться распределением относительных величин градиентов функции интенсивности, характеризующих направление границ объектов.

На практике такой подход реализуется следующим образом. Всё входное изображение разбивается на маленькие регионы - «ячейки», каждой из которых соответствует одномерная гистограмма направлений градиентов (ориентации рёбер). Каждый бин этой гистограммы соответствует определённому диапазону направлений и содержит суммарную (по пикселям ячейки) магнитуду градиентов соответствующего направления. Совокупность всех таких гистограмм образует общее представление о рёбрах на изображении. Для инвариантности к перепадам освещения производится нормировка значений в полученных гистограммах относительно некоторой области, содержащей рассматриваемую ячейку. Для этого соседние ячейки объединяются в группы - «блоки», а значения в гистограммах всех ячеек внутри блока нормируются на суммарное значение магнитуды градиентов по рассматриваемому блоку. Нормализованные блоки называются HOG дескрипторами, а множество таких блоков, локализованных в компактном регионе изображения, образуют «окно поиска». Итоговый вектор признаков образуется из конкатенации дескрипторов внутри окна поиска. В качестве алгоритма машинного обучения используется линейная машина опорных векторов (Linear SVM) [6]. Отметим, что натренированной моделью линейного SVM является плоскость, наилучшим образом разделяющая объекты от негативов в обучающей выборке.

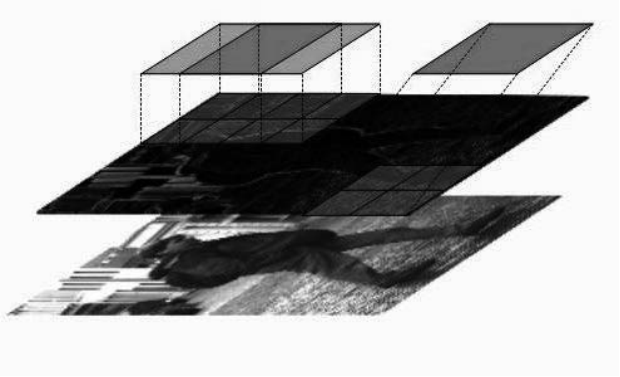

Рисунок 3: Схема формирования HOG признака. Градиенты, вычисленные на входном изображении (синего цвета) дискретизируются в гистограммы по ячейкам (жёлтого цвета). Ячейки группируются в перекрывающиеся между собой блоки (зелёного цвета). Конкатенация всех гистограмм всех блоков внутри окна образует итоговый вектор признаков.

В работе [1] рекомендуется использовать следующие параметры метода как оптимальные: [-1, 0, 1] - фильтр для вычисления градиентов на изображении; количество бинов в гистограммах – 9; размер ячейки в пикселях – 8 x 8; количество ячеек в блоке – 2 x 2; тип нормализации ячеек в блоке – квадратичная с усечением малых значений; шаг блоков в пикселях – 8 x 8 (соответствует размеру ячейки).

Согласно обзору [2], описанный HOG алгоритм обладает самым высоким качеством обнаружения, значительно опережая каскадный алгоритм, основанный на Haar wavelet признаках

2.3 Каскадный алгоритм, основанный на HOG признаках

Авторы работы [8] совместили лучшие стороны, описанных выше методов (скорость работы каскадного алгоритма и качество обнаружения HOG алгоритма), в каскадном алгоритме, основанном на HOG признаках. В качестве признаков использовались блоки, состоящие из 4 ячеек, заполненных HOG признаками, дискретизированными по 9 бинам. Таким образом, каждый признак (блок) представляется 36-ти размерным вектором HOG признаков внутри блока. Для быстрого вычисления HOG признаков для каждого изображения строится интегральная гистограмма градиентов [5], позволяющая вычислять признак за несколько простейших операций. В рассматриваемом алгоритме окно обнаружения заполняется блоками трёх типов в зависимости от соотношения ширины блока к высоте (1:2, 1:1, 2:1) . Размер и положение блоков сильно варьируются внутри окна обнаружения. Внутри окна обнаружения размером 64x128 пикселей генерируется 5,031 блоков, из которых для тренировки случайным образом выбираются 5% (250) блоков.

Каждый уровень каскада тренировался (как и в каскадном алгоритме на основе Haar wavelet признаков) с помощью алгоритма AdaBoost. Однако в качестве слабого классификатора, используемого в AdaBoost, применялась линейная SVM.

Критерий качества каждого уровня – 70% ложных срабатываний при 99.75% обнаруженных пешеходов. Натренированный каскадный классификатор состоит из 30 уровней.

Применение данного подхода позволило существенно ускорить процесс обнаружения пешеходов при сохранении высокого качества обнаружения, близкого к результатам HOG алгоритма Далала.

3. МОДИФИКАЦИЯ КАСКАДНОГО АЛГОРИТМА

Следуя работе [8], мы попытались совместить каскадный алгоритм Виолы-Джонса с HOG признаками, внеся следующие изменения. В работе [8] в алгоритме AdaBoost в качестве слабого классификатора использовались линейные SVM. Слабый классификатор формировался по одному 36-размерному векторному признаку (блоку). Мы разбили векторные признаки на скалярные, а в качестве слабого классификатора использовали деревья решений заданной высоты. Таким образом, каждый уровень нашего каскада представляет собой ансамбль деревьев заданной высоты, каждый узел которых соответствует разбиению по одному из скалярных признаков по всем блокам. Для окна обнаружения 48x96 пикселей генерировались блоки в диапазоне размеров от 16x16 до 48x96. Шаг блоков внутри окна составлял 8 пикселей, а сами блоки согласно соотношению ширины и высоты трёх типов: вытянутые вверх (1:2), квадратные (1:1), вытянутые по ширине (2:1). Общее количество блоков - 894, а общее количество признаков – 32184 (894x36). Как и в работе [8] для ускорения вычисления HOG признаков для каждого обрабатываемого изображения строилась интегральная гистограмма [5], позволяющая вычислять признак за несколько простейших арифметических операций.

Тренировка алгоритма осуществлялась на данных, состоящих из 15660 пешеходов и 10000 фрагментов

изображений, не содержащих пешеходов. Критерий качества каждого уровня – 50% ложных срабатываний при 99.75% обнаруженных пешеходов. В алгоритме AdaBoost использовались деревья решений высоты 1(stump), 2, 3 и 4. Интегральные гистограммы содержат 9 бинов, отвечающих направлениям градиентов от 0 до 180 градусов. Натренированные каскадные классификаторы состоят из 18 уровней, использующих от 1 до 39 признаков.

Реализация нашего каскадного алгоритма находится в библиотеке компьютерного зрения OpenCV [9].

4. ИСПОЛЬЗОВАНИЕ ИНФОРМАЦИИ О ПОЛОЖЕНИИ ЛИНИИ ГОРИЗОНТА

В модификации каскадного алгоритма мы используем данные о положении линии горизонта на изображении, а так же считаем известными минимальный и максимальный размеры пешеходов. Мы используем допущение, что размеры пешеходов линейно растут от минимальных (на дальнем плане, вблизи линии горизонта) до максимальных (на ближнем плане) (рис. 4). Для обнаружения пешеходов различных размеров применяется масштабирование поискового окна. Пусть **min**, **max** – размеры поисковых окон, соответствующих минимальному и максимальному размерам пешеходов. Поисковое окно большего размера получается умножением размеров текущего окна на величину **factor**, обозначающую отношение соседних по размеру окон. Количество **n** поисковых окон различных размеров в диапазоне [**min**, **max**] определяется величиной $\left\lceil log_{factor} \frac{max}{min} \right\rceil$.

Участок входного изображения от линии горизонта (**yHor**) до низа (**H**) разбивается на **n** равных полос. Каждое **i** – е поисковое окно сканирует заданное количество полос (**l**) изображения. Точнее, **i** – му окну достаётся вертикальный фрагмент изображения в диапазоне **[yHor + i * h**, **yHor + (i + l)*h]**, где **h = (H - yHor)/n** высота одной полосы. Таким образом, каждая полоса сканируется с помощью серии из поисковых окон. Это необходимо для гибкости системы обнаружения к незначительным изменениям рельефа дороги.

Применение описанного подхода позволило значительно сократить количество поисковых окон. Совместно с использованием интегральных гистограмм для быстрого вычисления HOG признаков это позволило ускорить процесс обнаружения на порядок. Кроме того, за счёт уменьшения области поиска пешеходов, уменьшилось количество ложных срабатываний алгоритма, что привело к повышению качества работы системы в целом.

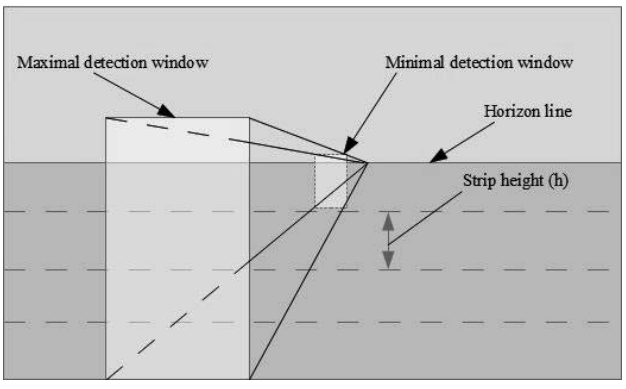

Рисунок 4. Схема поиска пешеходов использующая информацию о положении линии горизонта на изображении. Для определения предполагаемого размера пешеходов применяется линейная интерполяция размеров пешеходов от минимального до максимального.

5. ЭКСПЕРИМЕНТЫ

Для сравнения качества систем обнаружения пешеходов применяется подход, подробно описанный и использованный в работе [2]. В этом подходе к системам обнаружения предъявлены естественные требования, формирующие двухкритериальную задачу оптимизации. С одной стороны должно совершаться как можно меньшее количество ошибок второго рода, то есть, как можно большая доля объектов (detection rate - DR) должна быть обнаружена. С другой стороны должно минимизироваться количество ошибок первого рода (false positives per frame - FPPF), т.е. количество ложных срабатываний. Для визуализации качества работы системы используются ROC (Receive operating characteristic) – кривые, описывающие множество Парето, нашей двухкритериальной задачи оптимизации. Каждая точка ROC– кривой соответствует некоторой паре (FPPF, DR) при фиксированном пороге классификатора системы обнаружения. В качестве тестовых данных, как и в работе [2], использовались 21790 размеченных изображений из базы “Daimler Chrysler Benchmark Dataset”.

Все эксперименты проводились на CPU – IntelCore i7-2600 (3.4GHz), RAM - 8Gb, OS – Windows 7 Professional (64-bit).

На рисунке 4 приведено сравнение каскадных классификаторов, натренированных с использованием деревьев решений различных высот (1, 2, 3 и 4) в качестве слабых классификаторов. В таблице 1 указаны соответствующие им конфигурации параметров и скорость работы.

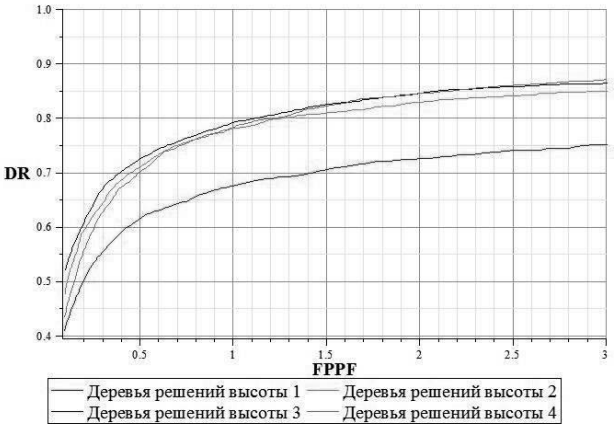

Рисунок 5: Сравнение качества работы каскадных алгоритмов на основе деревьев решений различных высот.

Таблица 1. Сравнение скорости работы каскадных алгоритмов на основе деревьев решений различных высот

	Деревья решений высоты 1	Деревья решений высоты 2	Деревья решений высоты 3	Деревья решений высоты 4
Шаг масштаба	1.05	1.05	1.05	1.05
Шаг по изображе-нию	(4, 4)	(4, 4)	(4, 4)	(4, 4)

Время на обработку изображения (сек)	~0.140	~0.151	~0.156	~0.162

С увеличением высоты деревьев до 3 качество обнаружения улучшается. При дальнейшем увеличении высоты деревьев рост качества прекращается. Затрачиваемое время с ростом высоты деревьев увеличивается не значительно. По этим причинам в дальнейших экспериментах будем использовать каскадный классификатор, использующий деревья решений высоты 3.

На рисунке 5 и в таблице 2 представлено сравнение качества и времени каскадного алгоритма с его модификацией, использующей информацию о положении линии горизонта на изображениях. Линия горизонта на тестовых изображениях размечена вручную, однако допускается модификация, в которой линия горизонта задаётся одинаковой на всех тестовых данных, согласно направлению камеры.

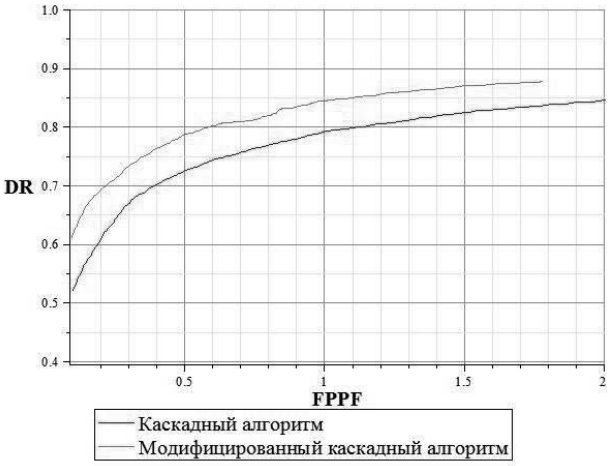

Рисунок 6: Сравнение качества работы каскадного алгоритма с модифицированным каскадным алгоритмом, использующим информацию о положении линии горизонта.

Таблица 2. Сравнение скорости работы каскадного алгоритма с модифицированным каскадным алгоритмом, использующим информацию о положении линии горизонта.

	Каскадный алгоритм	Модифицированный каскадный алгоритм
Шаг масштаба	1.05	1.05
Шаг по изображению	(4, 4)	(4, 4)
Время на обработку изображения (сек)	~0.156	~0.034

Использование информации о положении линии горизонта не только позволило существенно ускорить процесс обнаружения, но и улучшило качество обнаружения.

На рисунке 7 и в таблице 3 приведено сравнение HOG алгоритма из работы [1] с модифицированным каскадным алгоритмом, использующим информацию о положении линии горизонта на изображении. Для тестирования HOG алгоритма использовался программный код, реализованный авторами этого алгоритма – Н.Далалой и Б.Триггсом.

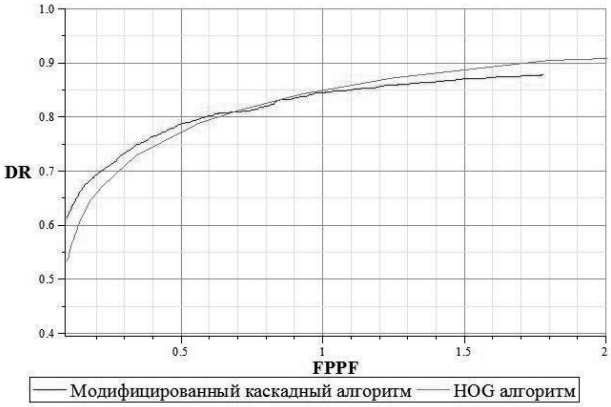

Рисунок 7: Сравнение качества работы модифицированного каскадного алгоритма с HOG алгоритмом в реализации Н.Далала, Б.Триггса.

Таблица 3. Сравнение скорости работы модифицированного каскадного алгоритма с HOG алгоритмом в реализации Н.Далала и Б.Триггса.

	Каскадный алгоритм	HOG алгоритм
Шаг масштаба	1.05	1.05
Шаг по изображению (в пикселях)	(4, 4)	(4, 4)
Время на обработку изображения (сек)	~0.034	~1

Каскадный алгоритм, использующий информацию о положении линии горизонта существенно опережает HOG алгоритм по скорости работы и сравним с ним по качеству обнаружения.

6. ЗАКЛЮЧЕНИЕ

Реализован каскадный алгоритм на основе HOG признаков, использующий в качестве слабого классификатора в алгоритме AdaBoost деревья решений заданной высоты. Применение информации о положении линии горизонта совместно с использованием интегральных гистограмм для ускоренного вычисления HOG признаков позволило увеличить скорость обнаружения до 30 кадров в секунду.

7. ССЫЛКИ

[1] N. Dalal and B. Triggs. *Histograms of oriented gradients for human detection* // Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

[2] M. Enzweiler and Dariu M. Gavrila. *Monocular Pedestrian Detection: Survey and Experiments* // Pattern Analysis and Machine Intelligence, 2009.

[3] Y. Freund, R.E. Schapire. *A Short Introduction to Boosting* // Journal of Japanese Society for Artificial Intelligence, 14(5): 771-780, September, 1999.

[4] D. Geronimo, A.M. Lopez, A.D. Sappa, and T. Graf. *Survey of pedestrian detection for advanced driver assistance systems //* Pattern Analysis and Machine Intelligence, 2010.

[5] F. Porikli. *Integral histogram: A fast way to extract histograms in Cartesian spaces //* Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

[6] V.N. Vapnik. *The Nature of Statistical Learning Theory* // Springer-Verlag, 1995.

[7] P. Viola, M. Jones. *Rapid object detection using a boosted cascade of simple feature* // Conference on Computer Vision and Pattern Recognition (CVPR), 2001.

[8] Qiang Zhu, Sai Avidan, Mei-Chen Yeh, and Kwang-Ting Cheng. *Fast Human Detection Using a Cascade of Histograms of Oriented Gradients* // Computer Vision and Pattern Recognition, 2006.

[9] www.opencv.org – Open Source Computer vision library.

Об авторах

Алексей Казаков - аспирант ВМК ННГУ имени Н.И. Лобачевского. Его адрес: kazakovdz@yandex.ru.
Александр Бовырин – директор по исследованиям и разработкам ООО «Аргус» . Его адрес: alexander.bovyrin@itseez.com