

## Алгоритм подсчета автомобилей в крупномасштабных системах видеонаблюдения\*

А.П. Широков<sup>1</sup>[0000-0001-7597-8511], Д.А. Купляков<sup>1,3</sup>[0000-0002-2957-3297],

А.С. Конушин<sup>1,2</sup>[0000-0002-6152-0021]

<sup>1</sup> Факультет вычислительной математики и кибернетики,  
Московский государственный университет им. М.В. Ломоносова, Москва, Россия  
arsenii.shirokov@graphics.cs.msu.ru

<sup>2</sup> НИУ Высшая Школа Экономики, Москва, Россия  
denis.kuplyakov@graphics.cs.msu.ru

<sup>3</sup> ООО «Технологии видеоанализа», Москва, Россия  
anton.konushin@graphics.cs.msu.ru

**Аннотация.** В работе рассматривается задача подсчета автомобилей в крупномасштабных системах видеонаблюдения. Применяется подход, основанный на сопровождении автомобилей и генерации событий пересечения траектории их движения с заданным сигнальным отрезком. За счет распределенности обработки и применения детекции не ко всем кадрам, а к их разреженному множеству, удается снизить требуемое число вычислительных ресурсов, увеличить производительность и получить алгоритм, способный качественно работать в реальном времени. Адаптируется и модифицируется подход, ранее предложенный для отслеживания людей [1]. Улучшение модуля оценки скорости и уточнение модели движения позволяют уменьшить частоту детекции в 3 раза. Предложенный алгоритм может работать при частоте детекции 3 Гц, сохраняя приемлемое качество.

**Ключевые слова:** компьютерное зрение, сопровождение в видео, подсчет автомобилей.

### 1 Введение

Автоматический подсчет автомобилей дает информацию, необходимую в ряде прикладных задач. Так, зная количество машин, проехавших по каждой дороге в конкретный промежуток времени, мы можем оптимизировать нагрузку на транспортное полотно, эффективно планировать ремонтные работы и проектировать новые дороги. Также полученную информацию можно использовать в различных маркетинговых исследованиях.

---

\* Публикация выполнена при поддержке гранта РФФИ №19-07-00844

Одним из возможных способов решения данной задачи является сопровождение автомобилей, построение их траекторий и подсчет количества пересечений этих траекторий с заданным сигнальным отрезком.

### 1.1 Обзор существующих методов

Современные методы сопровождения автомобилей используют подход к сопровождению через обнаружение. Общей чертой таких алгоритмов является то, что их можно условно разбить на 4 этапа: получение обнаружений, получение признаков обнаружений, построение матрицы стоимости, объединение обнаружений в траектории.

Отличаться же они могут, например, по типу детекций. Так, алгоритмы [4], [5], [7], [8] используют ограничивающие прямоугольники в плоскости кадра, а [6] параллелепипеды. Это различие является существенным, в первом случае мы можем судить только о пространственном расположении автомобиля, а во втором нам известно направление движения и часть траектории. Для более точного сопоставления обнаружений можно проводить вычисления не в плоскости кадра, а в координатах земли [13].

Также алгоритмы могут отличаться связями между этапами трекинга. В некоторых случаях все этапы реализованы отдельными модулями, которые оптимизируются отдельно ([7, 8, 9, 10]). А есть подходы, в которых 3 последних этапа объединяются в единую end-to-end сеть и оптимизируются совместно ([4], [5], [6]).

Еще одно отличие - информация, на основе которой принимается решение о сопоставлении новых обнаружений с траекториями. Могут быть рассмотрены связи между соседними обнаружениями [9], могут учитываться несколько последних обнаружений [5] или же на вход подается вся видеопоследовательность целиком, и есть возможность учитывать обнаружения на всех кадрах: предыдущих и последующих. А в некоторых работах предлагают строить траектории из обнаружений с помощью MCMC (Markov chain monte carlo) [12], [14].

### 1.2 Крупномасштабные системы

Системы дорожного видеонаблюдения содержат большое количество камер, что накладывает ограничения на стоимость установки и эксплуатации отдельно взятой камеры. Интерес представляет разработка алгоритма, способного работать именно в таких системах.

Полностью обрабатывать видеопоследовательность с одной камеры на одном локальном узле становится слишком дорого, поэтому возникает необходимость использовать распределенную систему с центром обработки данных. Все вычисления делать в ЦОД также невыгодно, так как тогда большая нагрузка приходится на каналы передачи информации.

Удачным выходом является распределенная система, описанная в [1] (рис. 1). В ней в ЦОД вычисляются только самые "дорогостоящие" операции, т.е. детекции. Все остальные этапы работы алгоритма вычисляются в локальных узлах,

расположенных возле камеры. Скорость работы системы определяется частотой обращений в ЦОД, т.е. частотой детекции. Поэтому выгодно считать детекции не на всех кадрах, а на разреженном их множестве.

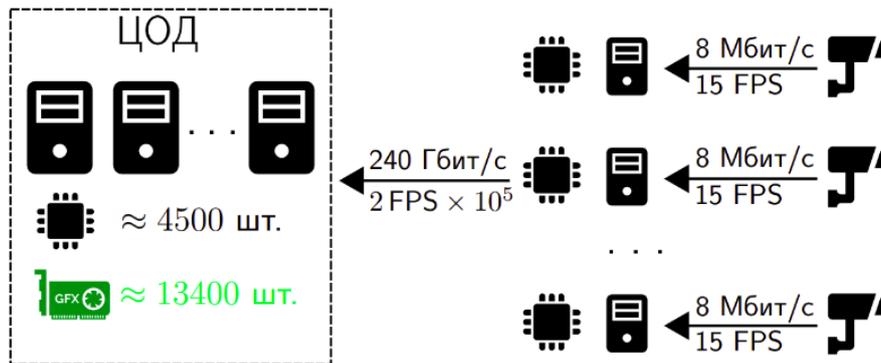


Рис. 1. Обработка видеопотоков в распределенной системе [1] при частоте детекций 2 Гц

## 2 Постановка задачи

В работе рассматривается задача автоматического подсчета автомобилей на интересующей части дороги.

**Формальная постановка задачи.** На вход подается видеопоток  $\{F_i\}$ ,  $F_i \in [0;255]^{W \times H \times 3}$ , кадров, полученных с неподвижной камеры. Сигнальный отрезок, заданный упорядоченной парой точек  $(L_a, L_b)$ .

На выходе — поток событий пересечения сигнального отрезка  $\{E_i\}$ , где  $E_i = (k_i, r_i, d_i)$  описывается номером кадра  $k_i$ , направлением пересечения  $r_i \in \{0, 1\}$  и ограничивающим изображением автомобиля прямоугольником  $d_i = (x_i, y_i, w_i, h_i)$ .

## 3 Предложенный метод

Предложенный алгоритм представляет собой модификацию алгоритма [1]. Он состоит из 4 этапов:

1. Получение детекции;
2. Построение матрицы стоимостей;
  - а. Оценка скорости с помощью визуального сопровождения;
  - б. Предсказание положения детекции на промежуточных кадрах;
3. Сопоставление траекторий и обнаружений;
4. Генерация событий.

Алгоритм был адаптирован к задаче подсчета автомобилей путем изменения детектора, также была добавлена фильтрация обнаружений, о которой подроб-

нее будет написано ниже. Был изменен и 2 этап: улучшена оценка скорости и изменена модель движения. Этапы 3 - 4 остались без изменения.

### 3.1 Получение детекции

По аналогии с базовым алгоритмом детекция производится не на всех кадрах, а только на ключевых:  $F_1, F_{1+\Delta}, F_{1+2\Delta}, F_{1+3\Delta}$  и т.д.,  $\Delta$  — параметр алгоритма, определяющий частоту детекции. Для получения набора ограничивающих изображения автомобилей в кадре прямоугольников используется детектор Detectron Faster R-CNN R-101-FPN [11], обученный на COCO 2017. Далее детекции проходят через ряд фильтров:

1. Фильтрация классов, не относящихся к автомобилям.  
Так как детектор обучен на COCO 2017, он выдает обнаружение неинтересующих нас классов, которые мы отсеиваем;
2. Фильтрация обнаружений, находящихся за пределами ROI (Region of interest).  
На этом этапе отсеиваем детекции, выходящие за пределы рассматриваемых участков дороги;
3. Фильтрация неуверенных обнаружений.  
Убираются детекции с уверенностью ниже заданного порога (0,6);
4. Последняя фильтрация связана с особенностью работы детектора. Чтобы исключить случаи, когда на 1 объект приходится несколько обнаружений (рис. 2(a)), соответствующих разным классам, используется алгоритм подавления не максимумов с высоким порогом (IOU  $\geq 0,95$ ).

**Таблица 1.** Параметры детектора [11]. *Box AP* - средняя точность по всем классам валидационной части COCO 2017; *производительность* – среднее время работы с одним изображением на GPU

Модель	Производительность (сек.)	Box AP
Faster R-CNN R-101-FPN	0,119	39,8



**Рис. 2.** (a) Визуализация работы детектора. На белом фургоне в центре кадра 2 обнаружения различных классов (b) Визуализация траекторий и их пересечений с сигнальным отрезком

### 3.2 Модификация оценки скорости

Важным этапом базового алгоритма является оценка скорости. Так как при разряжении ключевых кадров увеличивается расстояние между ними, то мы должны с достаточной точностью предсказывать положение автомобиля на промежуточном кадре, для чего нужна достаточно точная оценка скорости. С этой целью была предложена модификация модуля оценки скорости.

Для оценки скорости автомобиля на кадре  $F_N$  нужно найти смещение положения автомобиля за следующие  $\varepsilon$  кадров. Для этого выберем параметр алгоритма  $steps\_cnt$ , построим последовательность  $\{F_N, F_{N+\varepsilon/steps\_cnt}, F_{N+2\varepsilon/steps\_cnt}, \dots, F_{N+\varepsilon}\}$  и последовательно применим визуальный трекер к соседним парам кадров.

Отметим, что значение параметра  $steps\_cnt = 1$  соответствует работе оценки скорости в базовом алгоритме. Улучшение точности достигается за счет того, что визуальный трекер лучше работает на небольших смещениях, а использование промежуточных кадров уменьшает это смещение. Однако увеличение параметра  $steps\_cnt$  усложняет вычисления на локальном узле, визуальное сопровождение запускается для большего числа пар кадров.

### 3.3 Модель равноускоренного движения

Исходя из того, что автомобили часто меняют свою скорость, было решено изменить модель фильтра Калмана с равномерного движения на равноускоренное. Для этого достаточно добавить новые состояния, отвечающие за ускорение.

Таким образом, получим модель фильтра Калмана с состояниями  $x = (x, y, v_x, v_y, a_x, a_y, s, v_s, k)$ , где единственное скрытое состояние  $v_s$  - скорость изменения площади, а другие состояния:  $(x, y)$  - положение автомобиля,  $(v_x, v_y)$  - скорость движения,  $(a_x, a_y)$  - ускорение движения,  $s$  - площадь ограничивающего прямоугольника,  $k$  - отношение высоты и ширины ограничивающего прямоугольника. Матрица перехода представлена в формуле (1) ( $t$  - половина времени между соседними ключевыми кадрами):

$$\bar{x}^t = \begin{pmatrix} x + v_x t + \frac{a_x t^2}{2} \\ y + v_y t + \frac{a_y t^2}{2} \\ v_x + a_x t \\ v_y + a_y t \\ a_x \\ a_y \\ s + v_s t \\ v_s \\ k \end{pmatrix}. \quad (1)$$

Для оценки ускорения на кадре мы использовали следующий алгоритм:

1. Предположим,  $F_N$  - кадр, на котором нужно оценить ускорение.

2. С помощью визуального сопровождения найдем положение автомобиля с ключевого кадра  $F_N$  на  $F_{N+\varepsilon}$ .
3. По найденному смещению оценим начальную скорость  $v_0=(v_{0x}, v_{0y})$ .
4. Аналогично найдем смещение на кадре  $F_{N+\varepsilon}$  и оценим ускорение по формуле (2):

$$a_x = 2 \frac{x' - x - v_{0x}t'}{t'^2}, a_y = 2 \frac{y' - y - v_{0y}t'}{t'^2}, \quad (2)$$

где  $(x', y')$  положение автомобиля на кадре  $F_{N+\varepsilon}$ ,  $(x, y)$  положение автомобиля на кадре  $F_N$ ,  $t'$  – время между кадрами  $F_N$  и  $F_{N+\varepsilon}$ .

## 4 Экспериментальная оценка

### 4.1 Описание метрики

Для оценки качества полученного решения используется метрика, описанная в [1]. Разобьем видеопоследовательность на видеофрагменты, на которых произошло хотя бы 10 эталонных событий. Построим множество  $Q$  пар номеров кадров, ограничивающих эти видеофрагменты.

Для каждой пары  $(a, b)$  из  $Q$  посчитаем относительную ошибку подсчета  $E_{(a,b)}$  по формуле (3):

$$E_{(a,b)} = \frac{|FP_{(a,b)} - FN_{(a,b)}|}{GT_{a,b}}. \quad (3)$$

Итоговая метрика вводится, как средняя относительная ошибка на всех рассматриваемых отрезках и представлена в формуле (4):

$$E = \frac{\sum_{(a,b) \in Q} E_{(a,b)}}{|Q|}. \quad (4)$$

### 4.2 Описание набора данных для тестирования

Для тестирования базового алгоритма и его модификации были использованы наборы с ручной разметкой траекторий автомобилей для задачи сопровождения. Всего использовали 2 набора: UA-Detrac challenge [2] и GRAM Road-Traffic Monitoring [3], содержащие в общей сложности 103 видеопоследовательности, снятые с 27 различных камер. Встречались сцены различного уровня сложности: простые (камера расположена сверху и направлена вдоль движения, окклюзий не происходит) и достаточно сложные (постоянные окклюзии из-за столбов, разделителей полос, пересечения потоков автомобилей). Для каждой видеопоследовательности вручную был проведен сигнальный отрезок и получена эталонная разметка. В общей сложности наборы содержат 10,5 часов видео и примерно 7200 эталонных событий.

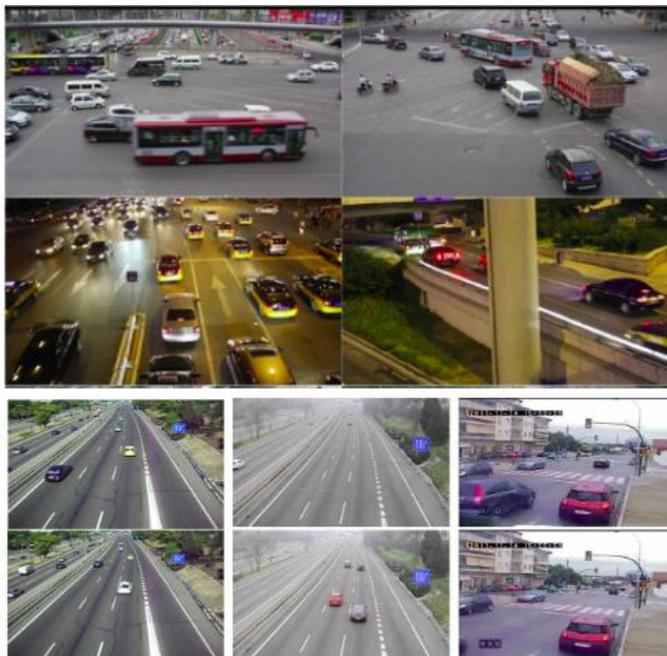


Рис. 3. Примеры сцен для тестирования

### 4.3 Качество подсчета автомобилей

Был проведен ряд экспериментов с каждой модификацией в отдельности и их объединением. Результаты приведены в таблице 2. Используемые обозначения:

- SORT - simple online and real time tracking[15];
- baseline - базовый алгоритм [1] с заменой детектора (см. раздел 3.1);
- ул. оц. скорости (steps\_cnt=N) - улучшение оценки скорости с указанием значения параметра (см. разд. 3.2);
- равноуск. движ. - модификация модели фильтра Калмана для учета равноускоренного движения (см. раздел 3.3).

Сравнив строку 2 со строками 3 и 4, можно заметить, что модификация оценки скорости дает улучшение точности. Аналогично, сравнив строки 2 и 5 можно убедиться в эффективности замены модели фильтра Калмана. Можно сделать вывод, что каждая из предложенных модификаций в отдельности улучшает качество алгоритма.

Наиболее значимый результат достигается при частоте детекции 3 Гц и применении обеих модификаций (последняя строка таблицы 2). Относительная ошибка составляет 5,4%, что незначительно хуже наилучшей ошибки базового алгоритма (5,2% при 10 Гц). Таким образом, удается сократить число обращений к ЦОД в  $\frac{10}{3}$  раз при сохранении приемлемого качества.

Стоит отметить, что предложенные модификации начинают работать в полной мере при частоте детекции 10 Гц и ниже. Связано это с тем, что при 20 Гц почти все кадры являются ключевыми и не хватает промежуточных кадров для задуманной работы модулей оценки скорости и ускорения. В таблице результаты соответствующих экспериментов не указаны.

Современные методы трекинга автомобилей обычно не преследуют цель работать с разреженными кадрами. Качество таких алгоритмов сильно падает при снижении частоты детекции. Чаще всего это связано с тем, что алгоритмы полагаются на малое смещение объекта в кадре. Высокая частота детекции приводит к выполнению этого условия. В строке 1 Таблицы 2 в качестве примера было рассмотрено одно из призовых решений [9] NVIDIA AI City Challenge, использовавшее для отслеживания автомобилей метод SORT [15].

**Таблица 2.** Результаты экспериментов. Указаны значения ошибки подсчета  $E$  (в процентах). Выделены лучшие результаты в столбце.

Метод / Частота детекции	20 Гц	10 Гц	5 Гц	3 Гц
SORT[15]	5,5	8,6	16,5	50,9
baseline	5,2	5,2	7,6	8,3
baseline + ул. оц. скорости (steps_cnt=2)	-	4,7	4,7	5,9
baseline + ул. оц. скорости (steps_cnt=3)	-	4,7	4,7	5,9
baseline + равноускор. движ	-	4,9	6,8	6,9
baseline + ул. оц. скорости (steps_cnt=3) + равноускор. движ	-	4,8	4,9	5,4

## 5 Заключение

Предложен алгоритм подсчета автомобилей, разработанный на основе [1]. Частота детекций была снижена более чем в 3 раза по сравнению с базовым методом, что позволяет сократить объем вычислительных ресурсов, требуемых для детекции, и увеличить скорость работы без потери качества.

В будущем планируется перейти от плоскости кадра к мировой системе координат за счёт автоматической калибровки камеры [13]. Это сделает модель движения фильтра Калмана более точной, что улучшит качество трекинга.

## Литература

1. Kuplyakov, D.A., Shalnov, E.V., Konushin, V.S. et al. A Distributed Tracking Algorithm for Counting People in Video. Program Comput Soft 45, 163–170 (2019).
2. Wen L. et al. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking //arXiv preprint arXiv:1511.04136. – 2015.

3. Guerrero-Gómez-Olmedo R. et al. Vehicle tracking by simultaneous detection and viewpoint estimation //International Work-Conference on the Interplay Between Natural and Artificial Computation. – Springer, Berlin, Heidelberg, 2013. – C. 306-316.
4. Chu P., Ling H. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking //Proceedings of the IEEE International Conference on Computer Vision. – 2019. – C. 6172-6181.
5. Sun S. J. et al. Deep affinity network for multiple object tracking //IEEE transactions on pattern analysis and machine intelligence. – 2019.
6. Li C. et al. TrackNet: Simultaneous Object Detection and Tracking and Its Application in Traffic Video Analysis //arXiv preprint arXiv:1902.01466. – 2019.
7. Pirsiavash H., Ramanan D., Fowlkes C. C. Globally-optimal greedy algorithms for tracking a variable number of objects //CVPR 2011. – IEEE, 2011. – C. 1201-1208.
8. Ooi H. L. et al. Multiple object tracking in urban traffic scenes with a multiclass object detector //International Symposium on Visual Computing. – Springer, Cham, 2018. – C. 727-736.
9. Kumar A. et al. A semi-automatic 2D solution for vehicle speed estimation from monocular videos //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. – 2018. – C. 137-144.
10. López-Sastre R. J. et al. Boosting multi-vehicle tracking with a joint object detection and viewpoint estimation sensor //Sensors. – 2019. – T. 19. – №. 19. – C. 4062.
11. Girshick, R., Radosavovic, I., Gkioxari, G., Doll'ar, P., He, K.: Detectron. <https://github.com/facebookresearch/detectron> (2018)
12. Kuplyakov D., Shalnov E., Konushin A. Markov chain monte carlo based video tracking algorithm // Programming and Computer Software. — 2017. — Vol. 43, no. 4. — P. 224–229.
13. Shal'nov E. V., Gringauz A. D., Konushin A. S. Estimation of the people position in the world coordinate system for video surveillance // Programming and Computer Software. — 2016. — Vol. 42, no. 6. — P. 361–366.
14. Benfold B., Reid I. Stable multi-target tracking in real-time surveillance video //CVPR 2011. – IEEE, 2011. – C. 3457-3464.
15. Bewley A. et al. Simple online and realtime tracking //2016 IEEE International Conference on Image Processing (ICIP). – IEEE, 2016. – C. 3464-3468.