Прослеживание объектов в видеопотоке, оптимизированное с помощью быстрого преобразования Хафа

А.А. Котов, И.А. Коноваленко, Д.П. Николаев

Аннотация. В работе описан метод прослеживания в видеопотоке объектов, содержащих множество концентрических дуг, основанный на применении структурного тензора и голосующей схемы в пространстве центров дуг. На основе предлагаемого метода реализован алгоритм прослеживания автомобильных колес в рамках задачи построения автоматического классификатора транспортных средств. Алгоритм использует дополнительное предположение об осесимметричности объекта. Произведена оптимизация алгоритма по скорости с помощью быстрого преобразования Хафа. Проведен ряд численных экспериментов, подтверждающих эффективность применения данного подхода.

Ключевые слова: зрительный интеллект, прослеживание объектов в видеопотоке, структурный тензор, быстрое преобразование Хафа.

Введение

В данной работе рассматривается проблема автоматического выделения и сопровождения движущихся объектов на цифровых видеоизображениях. Эта задача часто возникает при разработке систем видеонаблюдения и систем машинного зрения, предназначенных для мобильных технических средств. Решение этой проблемы в основном требует априорных моделей прослеживаемых объектов. Эти модели могут быть самыми разнообразными по своей сложности, начиная от цветовых гистограмм и заканчивая трехмерными сетками, состоящими из тысяч полигонов. Априорные знания позволяют робастно прослеживать объекты, особенно когда применяется сразу несколько моделей. Алгоритмы бустинга [1] могут обеспечивать робастное сопровождение объектов в реальном времени, но требуют как больших по объему положительных и отрицательных выборок для обучения, так и значительное время обучения. Системы, основанные на использовании характерных черт, например, описанная в работе [2], также способны робастно работать в реальном времени, но требуют значительных априорных знаний.

Для устойчивого слежения за объектами без использования сложной априорной модели, необходимо выделить черты объекта, устойчивые к аффинному преобразованию и изменению освещения. Эффективные и робастные схемы обнаружения характерных черт, такие как SIFT [3], не могут быть применены для прослеживания в реальном времени из-за своей вычислительной сложности. Хуан и др. [4] использовали шаблоны или паттерны областей близкой интенсивности. Сато и др. [5] применяли цветовые гистограммы. Оба подхода успешно справляются с заслонением, но ни тот, ни другой не способны побороть тени и изменения цвета, вызванного меняющимся освещением. Для сопровождения объектов в различных условиях освещения необходимо выделить

[1]Исследование выполнено при финансовой поддержке РФФИ в рамках научных проектов 13-01-12106-офи_м, 13-07-00870-а.

черты, которые непосредственно не зависят от яркости и цветовой информации. В исследуемой задаче такой чертой являются множественные концентрические дуги.

1. Постановка задачи

Цель работы — обеспечение сопровождения как движущихся, так и остановившихся объектов, содержащих множество концентрических дуг, в предположении, что объект движется в плоскости изображения, не меняя своих размеров. Примерами подобных объектов являются колеса автомобилей, движущихся параллельно плоскости изображения.

Входными данными задачи является последовательность видеокадров, среди которых выделены так называемые начальные кадры. На начальных кадрах заданы прямоугольники, выделяющие объекты, за движением которых нам и необходимо проследить.

Выходные данные представляют собой множество пронумерованных прямоугольников, описанных около сопровождаемых объектов. Причем одному объекту на разных кадрах соответствуют прямоугольники с одним и тем же номером — идентификатором.

Для решения поставленной задачи автоматического сопровождения объектов, содержащих концентрические дуги, предлагается сначала найти общий центр этих дуг — так называемую реперную точку, а затем с помощью полученной информации организовать процедуру межкадрового прослеживания.

2. Обзор аналогичных работ

Итак, мы рассматриваем объекты, содержащие множество концентрических дуг, откуда следует, что они имеют центрально-симметричную структуру.

Задача обнаружения симметрии на цифровых изображениях является популярной в области компьютерного зрения на протяжении последних десятилетий. Обобщенное преобразование симметрии (Generalized Symmetry Transform) [6] может обнаруживать как осевую, так и центральную симметрию на различных масштабах. Детектор, построенный в статье [7], может обнаруживать искаженную проективным преобразованием симметрию, но предложенный алгоритм имеет высокую вычислительную стоимость. Огава предложил использовать преобразование Хафа для нахождения симметрии на контурном изображении [8]. В то время как радиальная симметрия используется в системах реального времени [9], детекторы осевой симметрии из-за своей высокой вычислительной стоимости применяются в основном в офлайн приложениях. Тем не менее, авторы статей [10-11] предложили свой алгоритм обнаружения осевой симметрии и на его основе построили трекер осесимметричных объектов [12]. Для уточнения положения объекта авторы использовали фильтр Калмана. Получившаяся в результате их работы система устойчива к частичному заслонению объекта, может прослеживать прозрачные объекты (например, бутылку). Более того, трекер успешно сопровождает поворачивающиеся в пространстве и удаляющиеся или приближающиеся к камере объекты.

3. Нахождение реперной точки

Как говорилось ранее, предполагается, что прослеживаемый в видеопоследовательности объект движется в плоскости изображения, не меняя своих размеров. Например, таким свойством обладают автомобильные колеса, изображенные на кадре, полученном с камеры, снимающей вид автомобиля сбоку.

Алгоритм нахождения реперной точки объекта заключается в следующем: сначала для входного изображения строится центрированный вариант структурного тензора Яне [13]. Структурный тензор есть не что иное, как набор симметричных матриц определенного вида для каждого пиксела исходного изображения. Соответствующие матрицы имеют следующий вид:

$$\begin{pmatrix} \widetilde{\left(\frac{\partial \mathrm{I}}{\partial \mathrm{x}}\right)^2} - \left(\widetilde{\frac{\partial \mathrm{I}}{\partial \mathrm{x}}}\right)^2 & \widetilde{\left(\frac{\partial \mathrm{I}}{\partial \mathrm{x}}\frac{\partial \mathrm{I}}{\partial \mathrm{y}}\right)} - \left(\widetilde{\frac{\partial \mathrm{I}}{\partial \mathrm{x}}}\right)\left(\widetilde{\frac{\partial \mathrm{I}}{\partial \mathrm{y}}}\right) \\ \widetilde{\left(\frac{\partial \mathrm{I}}{\partial \mathrm{x}}\frac{\partial \mathrm{I}}{\partial \mathrm{y}}\right)} - \left(\widetilde{\frac{\partial \mathrm{I}}{\partial \mathrm{x}}}\right)\left(\widetilde{\frac{\partial \mathrm{I}}{\partial \mathrm{y}}}\right) & \widetilde{\left(\frac{\partial \mathrm{I}}{\partial \mathrm{y}}\right)^2} - \left(\widetilde{\frac{\partial \mathrm{I}}{\partial \mathrm{y}}}\right)^2 \end{pmatrix}, \quad (1)$$

где $\partial I/\partial x$ и $\partial I/\partial y$– горизонтальный и вертикальный градиенты исходного изображения соответственно, которые могут быть получены, например, с помощью свертки исходного изображения с ядрами (-1 0 1) и $(-1\ 0\ 1)^{\mathrm{T}}$ соответ-

ственно. Символом «~» обозначено усреднение по окну, в качестве которого можно применить гауссовское сглаживание изображения.

Таким образом, массив вычисленных структурных тензоров может быть представлен в виде трехканального изображения, в котором первый канал соответствует левому верхнему элементу матрицы, третий – правому нижнему, а второй – оставшимся двум.

Построенная матрица является неотрицательно определенной, это следует в частности из неравенства Коши-Буняковского. Это в свою очередь означает, что данная матрица имеет неотрицательные собственные значения. Направление собственного вектора, соответствующего наибольшему собственному значению, совпадает с направлением, в котором максимальна дисперсия производной яркости изображения в соответствующей точке, то есть той, для которой была построена матрица.

Рассмотренный выше структурный тензор дает нам так называемую диаграмму направленностей.

Через каждую точку области интереса (прямоугольного участка исходного изображения, в котором локализован объект) построим отрезок прямой, ограниченный данной областью интереса, вдоль направления, соответствующего этой точке. Рассмотрим точку изображения с максимальным числом проходящих через нее построенных прямых. По построению эта точка и будет центром концентрических дуг. Таким образом, найдя точку с максимальным порядком пересечения, мы автоматически получаем положение реперной точки.

Рассмотренную выше процедуру нахождения реперной точки можно алгоритмизировать ниже изложенным образом.

Для области интереса:

1. Построить структурный тензор.

2. Найти собственные векторы структурного тензора, соответствующие наибольшему собственному значению.

3. Создать «пустое» вспомогательное скалярное (одноканальное) изображение с размерами, соответствующими области интереса, то есть положить значение интенсивности каждого пиксела равным нулю.

4. Через каждую точку вспомогательного изображения «прочертить» прямую с некоторым весом вдоль направления, соответствующего собственному вектору из пункта 2, путем увеличения значения яркости всех пикселов, через которые проходит данный отрезок, на величину, равную этому самому весу. О выборе веса поговорим чуть ниже.

5. Сгладить вспомогательное изображение гауссовским фильтром. Этот шаг нужен для уточнения положения реперной точки.

6. В результате значение интенсивности каждого пиксела вспомогательного изображения будет с точностью до сглаживания равно сумме весов проходящих через него прямых. Таким образом, реперной точкой будет самый "яркий" пиксел, то есть аргмаксимум яркости вспомогательного изображения.

Для повышения быстродействия предложенного алгоритма будем строить прямые не через каждую точку области интереса, а с некоторым шагом h по вертикали и по горизонтали. При этом на каждом построенном отрезке сохраняются все точки (без разрежения), уменьшается лишь число рассматриваемых отрезков – их становится в h^2 раз меньше (с точностью до округления). Другими словами, мы проредили прямые, наложив на область интереса сетку с шагом h.

Очевидно, что точность нахождения положения реперной точки падает при увеличении шага сетки h. С другой стороны, значение шага h не имеет смысла брать меньше σ – параметра гауссовской фильтрации, применяемой для построения структурного тензора.

Теперь обсудим выбор весовой функции. Очевидно, что если для какой-либо точки изображения собственные значения структурного тензора равны, то весовая функция должна обращаться в нуль, так как в этом случае текстура в текущем положении окна не имеет выраженного направления и, как следствие, не содержит изображения дуг. Это утверждение можно обобщить. Выражение

$$\Delta = \sqrt{\lambda_{max}} - \sqrt{\lambda_{min}} \qquad (2)$$

характеризует степень направленности текстуры. Поэтому имеет смысл рассматривать весовую функцию w, график которой изображен на Рис.1.

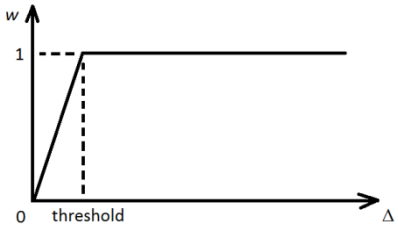

Рис.1. График зависимости весовой функции от разности корней собственных значений

Эксперименты, показали, что в случае изменения яркости изображения в диапазоне от нуля до единицы, оптимальное значение порога *(threshold)* лежит в окрестности 0.01.

На Рис. 2-Рис. 6 визуализирован каждый из этапов работы алгоритма. На Рис. 2 изображено исходное изображение объекта. На Рис. 3 штрихами показаны направления собственных векторов структурного тензора, соответствующих наибольшим собственным значениям. Результат голосования прямых в пространстве центров дуг приведен на Рис. 4. Крестиком обозначено положение реперной точки на сглаженной картине голосования (Рис. 5) и на исходном изображении (Рис. 6).

4. Межкадровое прослеживание

Изначально нам известно положение объекта, то есть описанный вокруг него прямоугольник, на первом кадре видеопоследовательности. Мы выбираем в качестве области интереса локализованный прямоугольный участок изображения, содержащий объект - в ней мы будем искать реперную точку. Область интереса можно построить путем расширения прямоугольника, описанного вокруг объекта, на Δx влево и вправо; и на Δy вверх и вниз. В частном случае, когда Δx и Δy равны нулю, область интереса, то есть область, в которой мы будем искать реперную точку, совпадает с областью изображения, ограниченной описанным вокруг объекта прямоугольником. Определившись с областью интереса, производим в ней процедуру нахождения реперной точки, описанную в предыдущем разделе.

При нахождении объекта на втором и всех последующих кадрах видеопотока считаем известным положение объекта, найденное на предыдущем кадре. Сначала выбираем область интереса, в которой будем искать реперную точку. Это делаем тем же самым способом, что и на первом кадре, только в качестве описанного вокруг объекта прямоугольника, берем

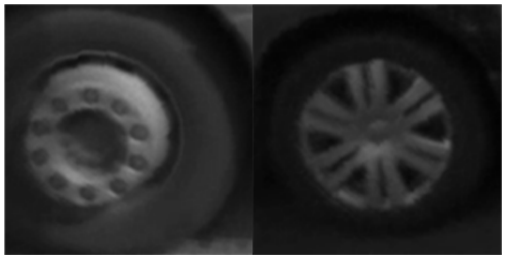

Рис. 2. Исходное изображение колеса

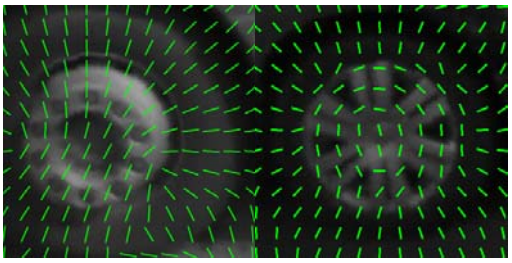

Рис. 3. Структурный тензор

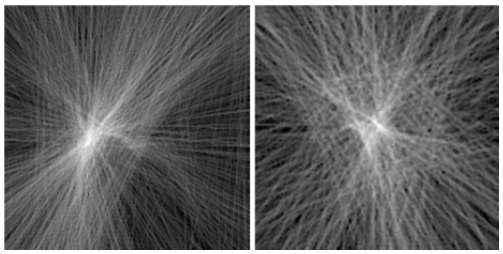

Рис. 4. Результат голосования прямых

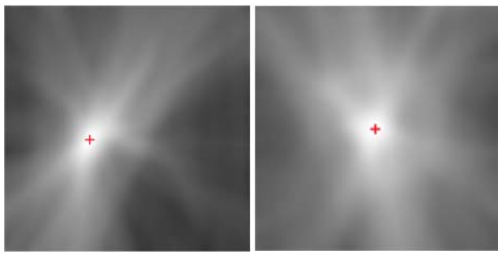

Рис. 5. Сглаженная картина голосования

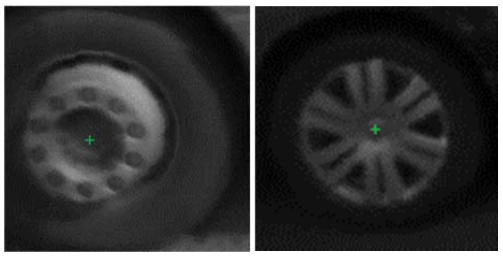

Рис. 6. Положение реперной точки

положение объекта на предыдущем кадре. После этого повторяем процедуру нахождения центра концентрических дуг объекта, то есть реперной точки, в построенной области интереса.

На Рис. 7 продемонстрирован результат работы алгоритма (показан каждый четвертый кадр видеопоследовательности).

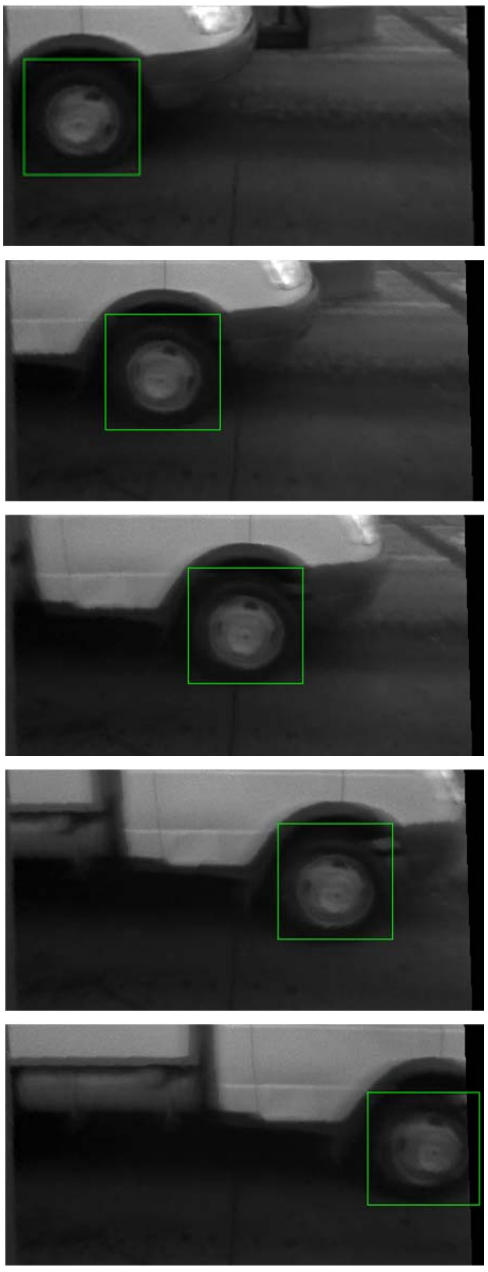

Рис. 7. Визуализация работы алгоритма (показан каждый четвертый кадр видеопоследовательности)

5. Ускорение процедуры голосования с помощью быстрого преобразования Хафа

Оценим асимптотику временной сложности описанного выше алгоритма построения картины голосования по уже имеющемуся структурному тензору.

Для простоты вычислений предположим, что мы работаем с квадратным изображением размером $n \times n$ пикселов. Пусть с учетом выбора шага построения структурного тензора для получения картины голосования нужно построить k прямых, где k имеет порядок n^2. Причем для построения каждой прямой мы должны «пробежаться» по каждому пикселу, через который она проходит. Таким образом, построения одной прямой занимает $O(n)$ операций. Следовательно, для построения всей картины потребуется $O(kn)$, то есть $O(n^3)$ операций.

Данная процедура может быть ускорена с помощью быстрого преобразования Хафа [14]. Голосующие прямые образуют пучок и проходят через одну точку, положение которой нам и необходимо найти. Постараемся найти эту точки без проведения процедуры голосования. Каждая голосующая прямая задается двумя коэффициентами (сдвигом и наклоном). Другими словами, каждая прямая задает точку в двумерном пространстве коэффициентов. Причем, прямым, проходящим через одну точку, в пространстве коэффициентов соответствуют точки, лежащие на одной прямой. Таким образом, для нахождения точки схода прямых достаточно найти прямую, которую задают точки, соответствующие этим прямым, в пространстве коэффициентов. Это можно сделать с помощью быстрого преобразования Хафа [14].

Опишем алгоритм подробнее и оценим его асимптотику временной сложности. Для начала нам нужно накидать точки в пространство коэффициентов. Количество точек равно количеству голосующих прямых, то есть равно k. Напомним, что k имеет порядок n^2, где n – сторона изображения области интереса. Быстрое преобразование Хафа выполняется за $O(n^2 \log n)$ операций. Итого получается $O(k) + O(n^2 \log n) = O(n^2) + O(n^2 \log n) = O(n^2 \log n)$ операций.

Следует отметить, что не существует непрерывного взаимно-однозначного отображения прямой в пространство коэффициентов. Поэтому,

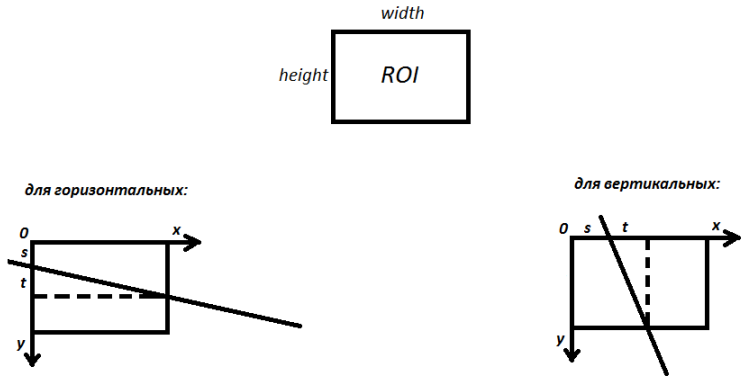

Рис. 8. Закон отображения прямых в пространстве коэффициентов

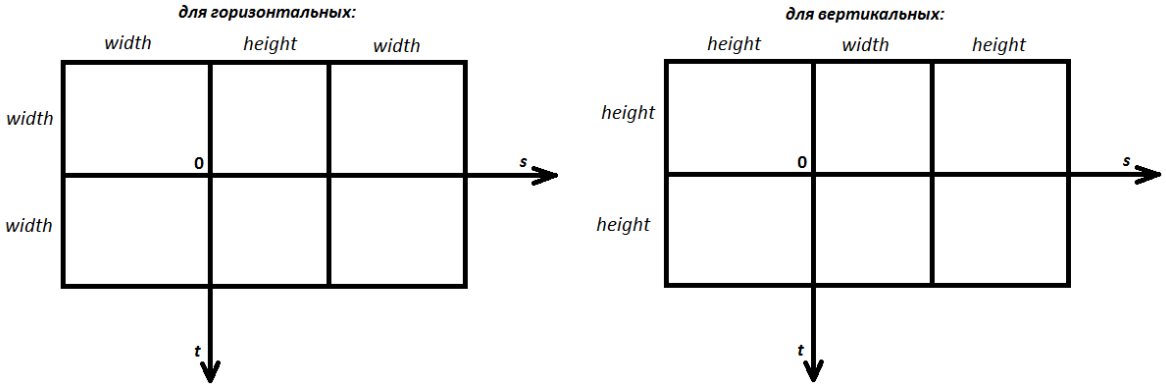

Рис. 9. Положение начала координат в пространстве коэффициентов

нам, как и в быстром преобразовании Хафа придется отдельно работать с преимущественно горизонтальными и преимущественно вертикальными прямыми.

Рис. 8 демонстрирует закон отображения прямых в пространство коэффициентов *s* (*shift* или *сдвиг*) и *t* (*tangent* или *наклон*) для преимущественно горизонтальных и вертикальных прямых.

Как можно понять из Рис. 8, для преимущественно горизонтальных прямых уравнение можно записать в виде:

$$x = (y - s)\frac{width}{t}, \quad (3)$$

а для вертикальных:

$$y = (x - s)\frac{height}{t}. \quad (4)$$

Можно заметить, что для преимущественно горизонтальных прямых значение *t* изменяется в диапазоне *[– width, width]*, а значение *s* в диапазоне*[– width, height + width]*. Аналогично, для преимущественно вертикальных прямых: значение *t* изменяется в диапазоне *[– height, height]*, а значение *s* в диапазоне *[– height, width + height]*. Таким образом, для пространства коэффициентов преимущественно вертикальных прямых достаточно заготовить изображение размером *2·width × (2·width + height)*, а для преимущественно вертикальных прямых – размером *2·height × (2·height + width)*. На Рис. 9 отмечено положение начала координат в пространстве коэффициентов на изображениях, соответствующих преимущественно горизонтальным и преимущественно вертикальным прямым.

Как видно на Рис. 10 вертикальные прямые дают больше шума из-за элементов автомобильного кузова. Это подтверждается Рис. 12.

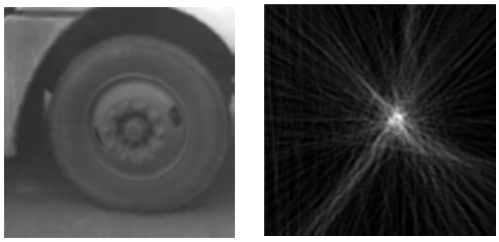

Рис. 10. Исходное изображение колеса (слева) и визуализация голосующих прямых (справа)

Рис. 11. Пространство коэффициентов для преимущественно горизонтальных прямых

Рис. 12. Пространство коэффициентов для преимущественно вертикальных прямых

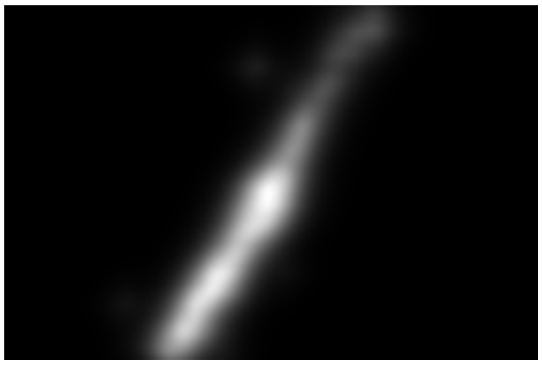

Рис. 13. Сглаженное пространство коэффициентов для преимущественно горизонтальных прямых

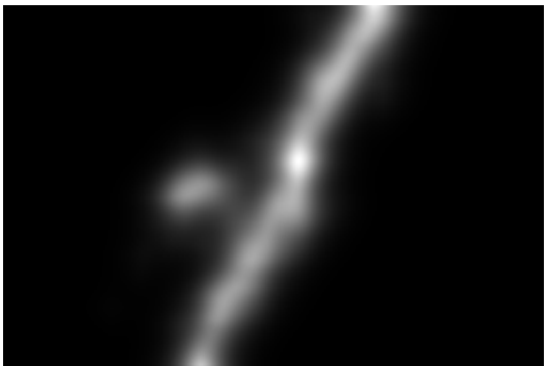

Рис. 14. Сглаженное пространство коэффициентов для преимущественно вертикальных прямых

Здесь мы имеем дело с коррелированным неаддитивным шумом. Как показали авторы работы [15], именно в этом случае лучше всех других распространенных методов решения задачи линейной регрессии работает именно преобразование Хафа. Но перед тем как его применить, нужно сгладить полученные картины пространства коэффициентов гауссовским фильтром. На Рис. 13 и Рис. 14 изображены пространства коэффициентов, сглаженные фильтром Гаусса со значением $\sigma = 10$.

Теперь можно применить к полученным изображениям быстрое преобразование Хафа. Так какточки в пространстве коэффициентов в силу его построения образуют преимущественно вертикальную прямую с отклонением от оси Y по часовой стрелке, то и применять нужно быстрое преобразованиеХафа для преимущественно вертикальных прямых с отклонением от оси Y только по часовой стрелке. На Рис. 15 и Рис. 16 изображен результат применения быстрого преобразования Хафа к сглаженному пространству коэффициентов.

Для нахождения положения прямой в пространстве коэффициентов необходимо найти аргмаксимум яркости препарата, полученного в результате применения быстрого преобразования Хафа. Положение максимума (x^*, y^*) взаимно однозначно определяет прямую в пространстве коэффициентов, интеграл вдоль которой максимален. Соответствующая прямая определяется с точностью до пары целочисленных точек (x_1, y_1) и (x_2, y_2), лежащих на границе изображения, соответствующего пространству коэффициентов. С учетом сдвига начала координат относительно изображения, проиллюстрированного на Рис. 9, координаты данной пары точек в пространстве коэффициентов для случая преимущественно горизонтальных прямых имеют вид:

$$s_i = x_i - width, \qquad t_i = y_i - width, \tag{5}$$

где $i \in \{1,2\}$.

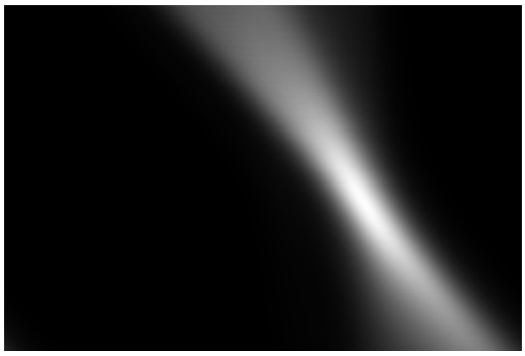

Рис. 15. Преобразование Хафа сглаженного пространства коэффициентов для преимущественно горизонтальных прямых

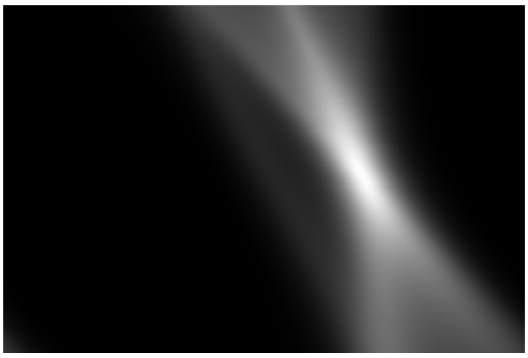

Рис. 16. Преобразование Хафа сглаженного пространства коэффициентов для преимущественно вертикальных прямых

Аналогично для случая преимущественно вертикальных прямых:

$$s_i = x_i - height, \qquad t_i = y_i - height, \tag{6}$$

где $i \in \{1,2\}$.

Из Рис. 8 видно, что преимущественно вертикальные прямые дают хорошую оценку для координаты *X* точки схода, а горизонтальные – для *Y*. Для нахождения *X* точки схода достаточно найти значение *s* точки пересечения прямой в пространстве коэффициентов для преимущественно вертикальных прямых с прямой $t = 0$. Аналогично: для нахождения *Y* точки схода достаточно найти значение *s* точки пересечения прямой в пространстве коэффициентов для преимущественно горизонтальных прямых с прямой $t = 0$.

Из уравнения прямой, проходящей через две точки:

$$\frac{t - t_1}{t_2 - t_1} = \frac{s - s_1}{s_2 - s_1} \tag{7}$$

следует, что искомые значения s определяются по формуле:

$$s = s_1 - \frac{s_2 - s_1}{t_2 - t_1} t_1. \tag{8}$$

Описанным выше образом можно определить координаты точки схода голосующих прямых без проведения процедуры голосования. На Рис. 17 изображена сглаженная картина голосования, на которой черной точкой отмечена точка схода, координаты которой были найдены с помощью быстрого преобразования Хафа, то есть без построения картины голосования.

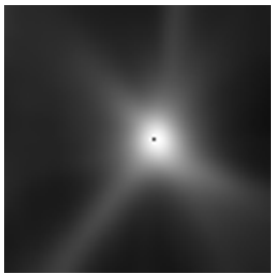

Рис. 17. Положение точки схода, найденной с помощью быстрого преобразования Хафа

6. Применение алгоритма

Поставленная проблема автоматического прослеживания в видеопотоке объектов, содержащих концентрические дуги, возникла у авторов как подзадача нахождения автомобильных колес в рамках задачи построения автоматического классификатора транспортных средств (*АКТС*). За подсчет количества колесных осей в *АКТС* отвечает интегратор колес (подсистема *wheel integrator*), который инициализируется набором координат и размеров колес на различных кадрах проезда и запускается в конце каждого проезда. Алгоритм, реализующий эту подсистему, описан в статьях [16] и [17]. В качестве инициализатора интегратора колес выступает колесный модуль (*wheel module*), который в свою очередь состоит из двух подмодулей: детекторы колес (*wheel detectors*) и трекер колес (*wheel tracker*). Подсистема *wheel detectors* находит колеса на каждом кадре проезда автомобиля, она представлена в виде различных детекторов колес, среди которых – детектор, основанный на алгоритме Виолы и Джонса [1] (с помощью которого обычно

ищут лица на изображениях), и детектор эллипсов на контурном препарате, построенном с помощью детектора границ Канни [18]. Подсистема *wheel tracker* реализована на основе алгоритма, описанного в этой работе. В качестве инициализатора этой подсистемы, т.е. источника колес, выступает модуль *wheel detectors*.

Проблема интеграции трекера колес в *wheel module*, заключается в том, что в отличие от подсистемы *wheel detectors*, которая в свою очередь является статической в том смыслеМ, что не является функцией состояния и не зависит от работы на предыдущем кадре, подсистема *wheel tracker* в этом плане является динамической и зависит от состояния на предыдущем кадре. Здесь мы имеем дело с двумя проблемами.

Первая – это накопление ошибки, которое связано с ложной инициализацией колес. Эту проблему можно обойти несколькими способами. В первом из них предлагается не запускать трекер колес на каждом срабатывании детекторов колес, а кластеризовать эти срабатывания по признаку пересечения, и инициализировать трекер только теми кластерами, размер которых не меньше заданного. Другими словами, предлагается прослеживать только «плотные» срабатывания колес (Рис. 18, справа). Таким образом, «слабые» срабатывания (Рис. 18, слева) не будут инициализировать ложные объекты. Другой способ состоит в использовании уровня достоверности (*confidence*) трекера, за который в нашем случае логично взять значение интенсивности вспомогательного изображения в реперной точке. То есть трекер будет прекращать прослеживать объект, если он в достаточной степени не является объектом, содержащим множество концентрических дуг, т.е. значение яркости очередной реперной точки ниже заданного. Это в частности касается и первого кадра работы алгоритма.

Второй проблемой является устойчивость построенной нами динамической системы. Трекер колес в этом плане можно рассматривать, как марковский процесс, так как текущее состояние зависит только от состояния на предыдущем кадре и никак не зависит от состояний на всех кадрах, предшествующих предыдущему. Таким образом, трекер может «срываться» с колеса. Это может происходить по разным причинам, среди наиболее распространенных – загораживание части колеса каким-либо объектом, в том числе и тень, падающая на колесо. Здесь опять можно выделить несколько путей решения данной проблемы. Первый способ заключается в отказе сопровождения объектов с уровнем *confidence* ниже, чем заданная доля аналогичного уровня достоверности, полученного на первом кадре прослеживания данного объекта. Второй подход более интеллектуальный и позволяет не терять объект. Данный способ основан на предсказании положения колеса каким-либо образом (например, методом наименьших квадратов) и поиска объекта как в области интереса, построенной на основании положения объекта на предыдущем кадре, так и в области, полученной путем предсказания.

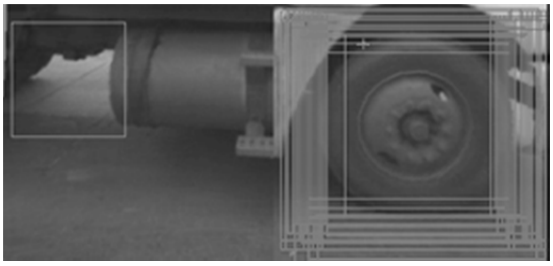

Рис. 18. Визуализация работы детекторов колес; слева – ложное одиночное срабатывание, справа – многократное истинное

Эксперименты показали, что горизонтальная координата реперной точки колеса почти всегда совпадает с горизонтальной координатой центра колеса. Однако подобное утверждение для вертикальной координаты оказалось неверным. Этот факт объясняется тем, что за счет присутствия в области интереса, изображения дорожного покрытия, элементов кузова и автомобильного крыла изображение колеса не является центрально-симметричным, но, тем не менее, имеет вертикальную ось симметрии. Поэтому при построении трекера колес предложенный алгоритм был слегка модифицирован, а именно – добавилось уточнение горизонтальной координаты колеса на основе положения реперной точки.

7. Эксперимент и результаты

Для подтверждения эффективности предлагаемого подхода был проведен численный эксперимент.

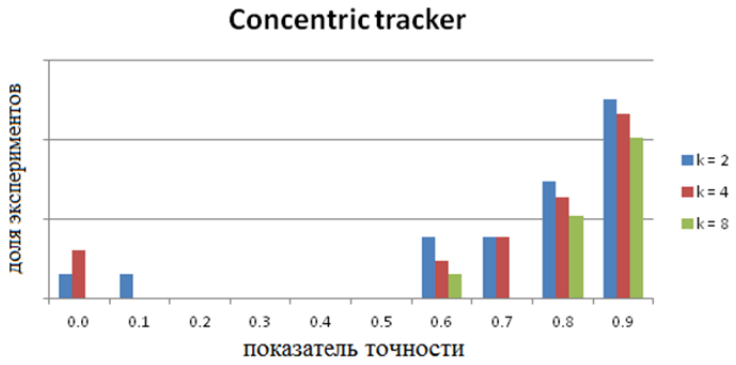

Рис. 19. Гистограмма с результатом работы трекера концентрических объектов

Сперва был выбран стенд изображений, состоящий из проездов автомобилей различного класса, то есть с разным размером колес и различным количеством колесных осей. Затем для каждого проезда вручную были размечены положение каждого колеса. В итоге получилась выборка из 64 проездов колес.

Основанная на предлагаемом алгоритме система прослеживания автомобильных колес тестировалась следующим образом. Трекер по очереди обрабатывал каждый кадр видеопоследовательности, при этом каждое очередное истинное положение колеса инициализировало в системе новый объект и добавляло его в очередь для прослеживания. Причем на каждом кадре производилось сопоставление реального положения колеса с положением соответствующего сопровождаемого объекта, который был добавлен в систему k кадров назад. Сопоставление положений объектов заключалось в определении показателя точности, то есть отношения площади пересечения описанных вокруг колес прямоугольников к площади одного из прямоугольников. Очевидно, что получаемое таким образом значение принадлежит отрезку *[0, 1]*. При этом – чем больше это значение, тем лучше работает трекер. Если система за соответствующие k кадров теряла объект, то в этом случае показатель точности приравнивался нулю. Таким образом, для данного конкретного значения параметра k мы можем построить гистограмму, изображающую зависимость количества прослеженных объектов от соответствующего интервала группировки значений показателя точности.

На Рис. 19 изображены гистограммы для значений параметра k, равным 2, 4 и 8.

Описанный способ построения метрики качества работы системы позволяет сравнивать между собой разные реализации трекера.

Сравним трекер, основанный на предлагаемом в данной работе алгоритме, с корреляционным трекером. Суть последнего заключается в том, что на каждом следующем кадре видеопоследовательности в некой области интереса, ограничивающей предсказанное положение объекта (полученное на предыдущем кадре), производится поиск изображения максимально коррелирующего с изображением колеса на предыдущем кадре. Другими словами, решается следующая задача оптимизации:

$$X^* = \operatorname{argmax}_{X' \in U(prediction)} r_{XX'}, \qquad (9)$$

где *U(prediction)* – область интереса вокруг предсказания, то есть положения колеса на предыдущем кадре, X – эталонное изображение колеса, полученное на предыдущем кадре, а r_{XY} – линейный коэффициент корреляции Пирсона:

$$r_{XY} = \frac{cov_{XY}}{\sigma_X \sigma_Y} = \frac{\sum(X-\overline{X})(Y-\overline{Y})}{\sqrt{\sum(X-\overline{X})^2 \sum(Y-\overline{Y})^2}}, \qquad (10)$$

где символом «–» обозначены средние значения яркости изображений:

$$\overline{X} = \frac{1}{n}\sum_{t=1}^{n} X_t, \ \overline{Y} = \frac{1}{n}\sum_{t=1}^{n} Y_t. \qquad (11)$$

На Рис. 20 можно видеть гистограммы, на которых изображены результаты работы сразу нескольких версий трекера – и описываемых в данной статье (концентрической и основанной на преобразовании Хафа), и корреляционной. Следует отметить, что с ростом значения параметра k качество работы корреляционного трекера падает. Этот факт объясняется тем, что корреляцион-

ный трекер со временем начинает терять объект, и соответственно пытается проследить уже не за изображением колеса, а за неким его смещенным положением, с каждым новым кадром еще больше теряя реальный объект.

Посмотрим, как поведет себя корреляционный трекер, который на каждом кадре за эталонное изображение колеса принимает не участок предыдущего кадра, соответствующий предыдущему положению колеса, а изображение колеса, полученное на первом кадре, когда объект добавлялся в очередь для прослеживания. Гистограммы, изображенные на Рис. 20, также содержат результаты тестирования данной версии корреляционного трекера. Как видно из гистограммы, второй вариант корреляционного трекера, который работает по первому кадру, значительно опережает своего аналога, работающего по предыдущему кадру. Тем не менее, даже эта версия корреляционного трекера уступает трекеру, основанному на поиске центра концентрических дуг. Это проявляется в большой доле колес, которые полностью теряются корреляционным трекером, то есть тех, для которых показатель точности равен нулю. Этот факт в свою очередь объясняется тем, что не всегда изображение колеса на первом кадре соответствует его изображению на последующих. В том смысле, что соответствующие изображения очень слабо коррелируют. Для этого может быть несколько причин. Самой распространенной среди них является большое количество проездов, содержащих частично или полностью затененные колеса. Также есть колеса с дисками, содержащие радиально направленные ребра жесткости. При движении автомобиля колесо вращается, поэтому два изображения одного и того же колеса на разных кадрах могут быть даже анти-коррелированы. Выше описанные проблемы также в какой-то степени относятся и к версии корреляционного трекера, который работает по предыдущему кадру. Примеры тени от столба и колеса с ребрами жесткости изображены на Рис. 21 и Рис. 22 соответственно.

Как видно из Табл. 1-Табл. 3 и Рис. 20 версия, основанная на быстром преобразовании Хафа, является не только самой быстрой, но и самой точной.

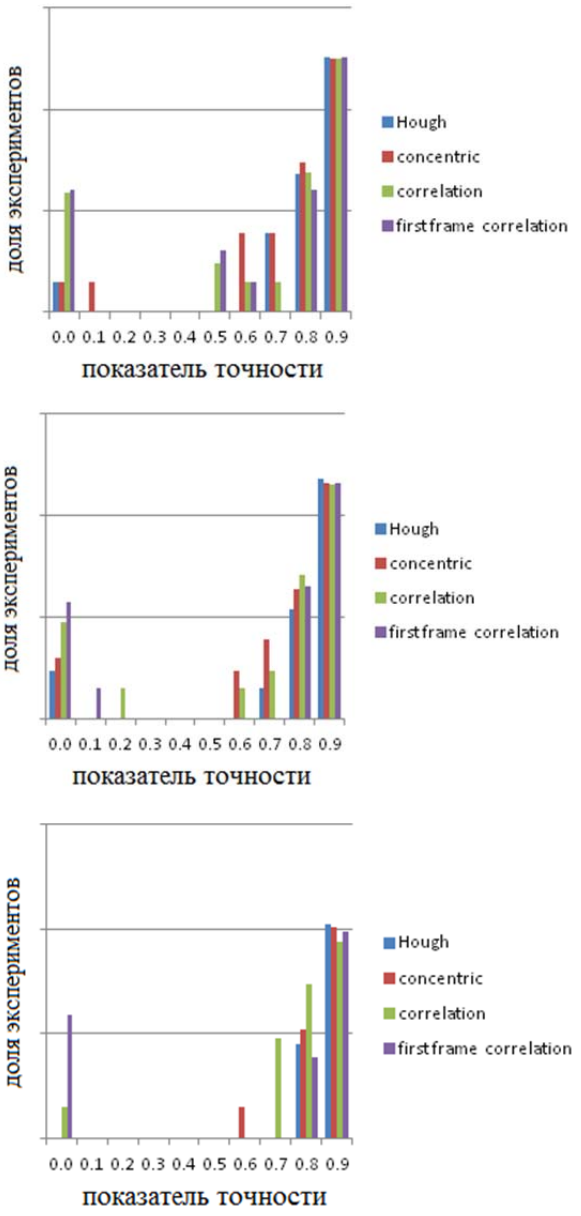

Рис. 20. Сравнение различных версий трекера для значений параметра k= 2 (вверху), k=4 (посередине) и k=8 (внизу

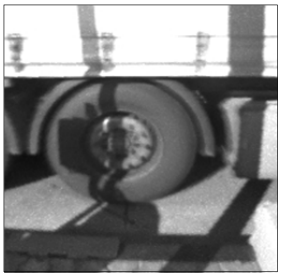

Рис. 21. Пример тени от столба, падающей на колесо

Рис. 22. Колесо с ребрами жесткости

Табл. 1. Сравнение различных версий трекера для k = 2

Точность	Hough	concentric	Correlation	first frame correlation
0.0	2	2	15	16
0.1	0	2	0	0
0.2	0	0	0	0
0.3	0	0	0	0
0.4	1	0	1	0
0.5	0	0	3	4
0.6	0	6	2	2
0.7	6	6	2	0
0.8	23	30	24	16
0.9	331	317	316	325

Табл. 2. Сравнение различных версий трекера для k = 4

Точность	Hough	concentric	Correlation	first frame correlation
0.0	3	4	9	14
0.1	0	1	0	2
0.2	0	0	2	0
0.3	0	0	0	0
0.4	0	0	0	0
0.5	0	0	0	0
0.6	0	3	2	0
0.7	2	6	3	0
0.8	12	19	26	20
0.9	228	212	203	209

Табл. 3. Сравнение различных версий трекера для k = 8

Точность	Hough	concentric	Correlation	first frame correlation
0.0	0	0	2	15
0.1	0	0	0	0
0.2	0	0	0	0
0.3	0	0	0	0
0.4	0	0	0	0
0.5	0	0	1	1
0.6	0	2	0	0
0.7	0	1	9	0
0.8	8	11	30	6
0.9	110	104	76	96

Заключение

В данной работе изложен алгоритм автоматического прослеживания в видеопотоке объектов, содержащих множество концентрических дуг, основанный на построении структурного тензора и применении быстрого преобразования Хафа.

Описаны алгоритмы, реализующие каждый из этапов метода.

Предложенный метод был реализован в виде программы на языке C++.

Приведены результаты работы алгоритма на реальных данных, показывающие временную эффективность этого метода, а также его адекватность поставленной задаче.

Предложенная методика автоматического сопровождения концентрических объектов на видеопоследовательностях, получаемых от различных двумерных сенсоров, может быть использована в различных системах ситуационного анализа изображений динамически меняющихся сцен, таких как системы оценки транспортных потоков, охранные системы, системы обнаружения препятствий для мобильных технических средств и другие технические системы аналогичного назначения.

Литература

1. P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in Proceedings of IEEE Computer Society Conference on Computer Vis ion and Pattern Recognition (CVPR), vol. 1, no. 1. IEEE Computer Society, 2001, pp. 511–518.
2. S. Khan, M. Shah, Object based segmentation of video using color, motion and spatial information, in CVPR (2), 2001, pp. 746–751.
3. D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision, vol. 60, no. 2, p. 91, November 2004.
4. Y. Huang, T. S. Huang, H. Niemann, A region-based method for model-free object tracking, in ICPR (1), 2002, pp. 592–595.
5. Y. Satoh, T. Okatani, K. Deguchi, A color-based tracking by kalman particle filter, in ICPR (3), 2004, pp. 502–505.
6. D. Reisfeld, H. Wolfson, Y. Yeshurun, Context-free attentional operators: the generalized symmetry transform, Int. J. Comput. Vision , vol. 14, no. 2, pp. 119–130, 1995.
7. R. K. K. Yip, A hough transform technique for the detection of reflectional symmetry and skew-symmetry, Pattern Recognition Letters , vol. 21, no. 2, pp. 117–130, 2000.
8. H. Ogawa, Symmetry analysis of line drawings using the hough transform, Pattern Recognition Letters, vol. 12, no. 1, pp. 9–12, 1991.
9. G. Loy, A. Zelinsky, Fast radial symmetry for detecting points of interest, IEEE Trans. Pattern Anal. Mach. Intell., vol. 25, no. 8, pp. 959–973, 2003.
10. W. H. Li, A. Zhang, L. Kleeman, Fast global reflectional symmetry detection for robotic grasping and visual tracking, in Proceedings of Australasian Conference on Robotics and Automation, M. M. Matthews, Ed., December 2005.
11. W. H. Li, A. M. Zhang, L. Kleem2an, Real time detection and segmentation of reflectionally symmetric objects in digital images, Accepted for publication at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Beijing, China, October 2006.
12. W. H. Li, L. Kleeman, Real time object tracking using reflectional symmetry and motion, in Proceedings of the 2006 IEEE/RSJ, International Conference on Intelligent Robots and Systems, October 2006.
13. Яне Б. Цифровая обработка изображений. М.: Техносфера, 2007.

14. Nikolaev D., Karpenko S., Nikolaev I., Nikolaev P. Hough transform: underestimated tool in the computer vision field // Proceedings of the 22th European Conference on Modelling and Simulation.- 2008.- P. 238-246
15. Безматерных П., Ханипов Т., Николаев Д.П. Решение задачи линейной регрессии с помощью быстрого преобразования Хафа // Информационные технологии и системы (ИТиС'12). Петрозаводск, 2012. М.: ИППИ РАН, 2012. С. 354-359.
16. A.Grigoryev, T.Khanipov, D.Nikolaev. Determination of axle count for vehicle recognition and classification. 8th Open German-Russian Workshop «Pattern Recognition and Image Understanding»: Workshop proceedings. — Nizhny Novgorod, 2011. — p. 89–91.
17. Григорьев А.С., Николаев Д.П., Ханипов Т.М. Определение количества осей транспортного средства по видеоряду проезда. Труды 54-й научной конференции МФТИ «Современные проблемы фундаментальных и прикладных наук»: Часть IX. Инновации и высокие технологии. — М.: МФТИ, 2011. — с. 31-32.
18. J. Canny. A computational approach to edge detection, Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI-8(6):679–698, Nov. 1986.

Антон Александрович Котов. Окончил МФТИ в 2013 году. Автор одной печатной работы. Область научных интересов: зрительный интеллект, алгоритмы обработки изображений. E-mail: kotov.anton.1@gmail.com

Иван Андреевич Коноваленко. Младший научный сотрудник ИППИ РАН. Окончил МФТИ в 2014 году. Автор 14 печатных работ. Область научных интересов: статистика, анализ данных, зрительный интеллект. E-mail: alatkon@yandex.ru

Дмитрий Петрович Николаев. Заведующий сектором ИППИ РАН. Окончил МГУ в 2000 году. Кандидат физико-математических наук. Автор 122 печатных работ. Область научных интересов: быстрые алгоритмы обработки изображений, зрительный интеллект. E-mail: dimonstr@iitp.ru