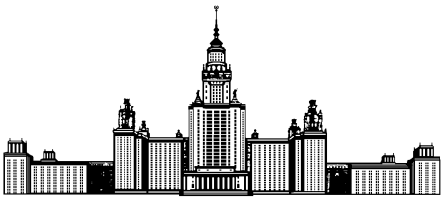
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

ДИПЛОМНАЯ РАБОТА СТУДЕНТА 517 ГРУППЫ

«Трекинг объектов на видео при помощи фильтра частиц»

Выполнил:

студент 5 курса 517 группы

Нижибицкий Евгений Алексеевич

Научный руководитель:

д.ф-м.н., профессор

Дьяконов Александр Геннадьевич

Москва, 2014

Содержание

1 Введение **2**

1.1 Определения и обозначения 3

2 Видеотрекинг при помощи фильтра частиц **6**

2.1 Модель состояний 7

2.2 Модель перехода 7

2.3 Инициализация 8

2.4 Модель наблюдения 9

2.5 Ресэмплинг 14

2.6 Борьба с наложениями 15

3 Программная реализация **15**

4 Вычислительные эксперименты **17**

4.1 Используемая мера качества 17

4.2 Данные для экспериментов 17

4.3 Результаты 20

4.4 Обсуждение и выводы 30

4.5 Сравнение с другими работами 31

5 Заключение **34**

Список литературы **35**

1 Введение

Задача трекинга объектов на видео является неотъемлемой частью многих прикладных областей, таких как построение систем видеонаблюдения, отслеживания дорожного трафика (в частности, наблюдения за определенными транспортными средствами в потоке), создание интерфейсов человек-компьютер, программ для передачи и сжатия видео и других.

При сопровождении объектов на видео необходимо обрабатывать большое количество потоковых данных, что затратно с вычислительной точки зрения, а значит и с точки зрения затрачиваемого на обработку времени. Особенно это справедливо, если распределение положений отслеживаемого объекта не нормально, мультимодально или вообще произвольной природы.

За последние годы было предложено множество успешных подходов по решению данной задачи. Тем не менее, многие из них накладывают определенные ограничения на обрабатываемые данные, как то статичный фон и фиксированный ракурс [11], знание о типе наблюдаемого объекта [15] или даже наличие множества камер [18]. Большая часть предлагаемых подходов уделяют мало внимания возможности проводить анализ в реальном времени, хотя и некоторые из них наоборот ориентированы на строгие ограничения по ресурсам — примером последнего являются приложения в робототехнике [5].

Целью данной работы является исследование случая отслеживания на видео объектов произвольной природы в отсутствии дополнительных предположений, таких как наличие стационарного фона при неподвижности камеры и других выше упомянутых случаев. Для проверки соответствия этим требованиям будет проведено тестирование на специализированном наборе данных, покрывающем многие «сложные» случаи работы с объектами, изменяющими свои размеры, форму, цвет, перекрывающимися другими объектами и пр. В качестве удовлетворительной производительности будет рассматриваться такая, которая позволяет проводить анализ в реальном времени на современном ноутбуке.

Если при трекинге объекта во время каждого фиксированного кадра рассматривать его возможные положения как распределение вероятностей по всем возможным точкам позиционирования и по всем возможным размерам описывающих его окон с центрами в соответствующих точках (при отслеживании объекта нашей целью является как можно точнее выделить объект прямоугольной рамкой), то нашей основной задачей во время каждого кадра является аппроксимация этого распределения, на основе которого уже можно будет вычислять искомую описывающую рамку.

Одними из наиболее подходящих для задачи моделирования изменяющихся во времени произвольных распределений являются последовательные методы Монте-Карло, или *фильтры частиц*, которые оценивают плотности при помощи конечных наборов *частиц*, или *сэмплов*, реализующих распределение.

1.1 Определения и обозначения

В основе фильтров частиц лежит модель, описывающая изменение скрытых переменных — в нашем случае реальных значений, характеризующих положение объекта в кадре, — и получение наблюдаемых переменных — итоговых изображений из видео, а также метод аппроксимации плотности с помощью реализующей её выборки, соответствующей задаче *фильтрации* — задаче определения реального значения скрытых переменных в текущий момент времени на основе полной истории значений наблюдаемых переменных.

1.1.1 Модель пространства состояний (SSM)

Модель пространства состояний состоит из марковского процесса $\{x_t\}_{t\geq 1}$ и процесса измерений $\{y_t\}_{t\geq 1}$ так, что

$$\begin{aligned} p(x_{t+1}|x_t) &= f_{\theta,t}(x_{t+1}|x_t, u_t), \\ p(y_t|x_t) &= h_{\theta,t}(y_t|x_t, u_t), \\ p(x_1) &= \mu(x_1), \end{aligned}$$

где x_t отвечает за состояние, u_t — входной сигнал, y_t — наблюдаемое измерение, θ — любые неизвестные (статичные) параметры.

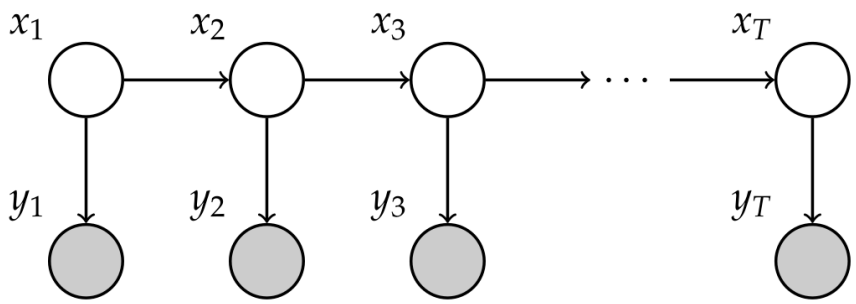


Рис. 1: Визуализация SSM.

Байесовская сеть на рис. 1 описывает совместное распределение всех задействованных переменных:

$$p(x_{1:T}, y_{1:T}) = \prod_{t=1}^{T} p(x_t|\text{pa}(x_t)) \prod_{t=1}^{T} p(y_t|\text{pa}(y_t)),$$

где "$z_{1:T}$" — сокращенная запись для "$z_1, \ldots, z_T$", а за $\text{pa}(z)$ обозначены все родители z с учётом ориентации дуг в графе.

Воспользовавшись равенствами для процесса выше, получим

$$p(x_{1:T}, y_{1:T}) = \mu(x_1) \prod_{t=1}^{T} f_{\theta,t}(x_{t+1}|\text{pa}(x_t)) \prod_{t=1}^{T} h_{\theta,t}(y_{t+1}|\text{pa}(y_t)).$$

На рис. 2 приведены возникающие в связи с этой моделью задачи оценивания плотностей.

Задача	Функция плотности
Фильтрация	$p(x_t\|y_{1:t})$
Предсказание	$p(x_{t+1}\|y_{1:t})$
Предсказание на k шагов	$p(x_{t+k}\|y_{1:t})$
Совместное сглаживание	$p(x_{1:T}\|y_{1:T})$
Маргинальное сглаживание	$p(x_{1:t}\|y_{1:T}), t \leq T$

Рис. 2: Функции плотности для наиболее часто решаемых задач вывода состояний.

Из всех функций плотности нас будет интересовать функция фильтрации — именно её мы будем оценивать в дальнейшем.

1.1.2 Sampling Importance Resampling

SIR — метод Монте-Карло, который позволяет получать независимую выборку из целевого распределения, плотность которого известна лишь с точностью до константы:

$$\pi(z) = \frac{\widetilde{\pi}(z)}{C_\pi},$$

где только $\widetilde{\pi}(z)$ мы умеем вычислять, C_π — неизвестная нормировочная константа, $\pi(z)$ — реальная плотность целевого распределения.

Для этого используется т.н. *предложенное распределение*, плотность которого мы точно знаем и которое по нашим соображением в чем-то походит на нужное нам распределение $\pi(z)$ — например, в случае некоторого неизвестного унимодального симметричного распределения можно использовать в качестве предложенного нормальное распределение.

Алгоритм SIR:

1. Генерируем выборку $\{\tilde{z}^i\}_{i=1}^N$ из предложенного распределения $q(z)$.
2. Вычисляем веса $\widetilde{w}^i = \frac{\tilde{\pi}(\tilde{z}^i)}{\tilde{q}(\tilde{z}^i)}$, $i = 1, \ldots, N$.
3. Нормализуем веса $w_k^i = \frac{\widetilde{w}^i}{\sum_{j=1}^N \widetilde{w}^j}$, $i = 1, \ldots, N$.
4. Для $i = 1, \ldots, N$ генерируем новый сэмпл z^i такой, что

$$\mathbb{P}(z^i = \tilde{z}^j) = w^j, j = 1, \ldots, N.$$

Алгоритм не возвращает выборку целевого распределения, но выборка $\{z^i\}_{i=1}^N$ совместно с нормализованными весами $\{w^i\}_{i=1}^N$ позволяет получить эмпирическое приближение целевой функции распределения:

$$\hat{\pi}(z) = \sum_{i=1}^N w^i \delta_{z^i}(z).$$

1.1.3 Фильтр частиц

Если применить SIR для рассмотренной ранее задачи фильтрации, получим алгоритм фильтра частиц для $p(x_{1:t}|y_{1:t})$:

1. Генерируем N начальных сэмплов $\widetilde{x}_1^i \sim \mu(x_1)$, веса устанавливаем равными $\widetilde{w}_0^i = 1/N$, $k = 0$.

2. Для всех k от 1 до t

   (a) Вычисляем веса $\widetilde{w}_k^i = p(y_k|\tilde{x}_k^i)\widetilde{w}_{k-1}^i$.

   (b) Нормализуем веса $w_k^i = \frac{\widetilde{w}_k^i}{\sum_{j=1}^N \widetilde{w}_k^j}$.

   (c) Для $i = 1, \ldots, N$ генерируем новый сэмпл x_t^i такой, что $\mathbb{P}(x_t^i = \tilde{x}_t^j) = w^j, j = 1, \ldots, N$.

   (d) Генерируем новую выборку из N элементов из предложенного распределения, $\tilde{x}_{k+1}^i \sim f(x_{k+1}|x_k^i)$.

2 Видеотрекинг при помощи фильтра частиц

Распределение $p(X)$ возможных состояний отслеживаемого объекта приближается набором взвешенных частиц (сэмплов) $S_t = \{s_t^j\}$, $j \in \{1 \ldots N\}$ (в экспериментах $N = 1000$, что позволяет укладываться в необходимые временные рамки для большинства рассматриваемых далее алгоритмов), где каждая частица $s_t^j = (x_t^j, \pi_t^j)$ состоит из вектора состояний x_t^j и вектора весов π_t^j. Набор частиц обновляется при переходе от одного кадра к другому по следующей рекурсивной процедуре:

1. Новый набор S_t получается из предыдущего S_{t-1}, где сэмпл s_{t-1}^i из старого набора выбирается с вероятностью, пропорциональной его весу π_{t-1}^i;

2. Для каждого сэмпла новое состояние x_t^j получается сэмплированием из модели движения $p(X_t|X_{t-1} = x_{t-1}^i)$;

3. Измерение в новом кадре Z_t используется при обновлении весов π_t^i с помощью подсчета правдоподобия наблюдения, т.е. $\pi_t^j = p(Z_t|X_t = x_t^j, Z_0, Z_1, \ldots, Z_{t-1})$.

Правдоподобие зависит от всех предыдущих кадров $Z_0, Z_1, \ldots, Z_{t-1}$, если модель наблюдения адаптируется со временем. Для статической модели мы имеем $\pi_t^j = p(Z_t|X_t = x_t^j, Z_0)$.

2.1 Модель состояний

Начальная позиция объекта, как и рассматриваемые в будущем вероятные позиции, определяются рамкой выделения на изображении, т.е. каждая позиция-кандидат, представленная в виде частицы, определяется некоторым вектором $\mathbf{x}$, по которому как минимум можно получить значения x, y, w, h, отвечающие за позицию центра и ширину/высоту рамки в абсолютных значениях.

В большинстве работ для представления рамки изображения все четыре координаты x, y, w и h содержатся в упомянутом векторе $\mathbf{x}$, но при этом могут также храниться и относительные значения — отношение величин к высоте/ширине всего кадра. Тем не менее, например в [15] вместо ширины и высоты рамки использовался лишь один коэффициент масштабирования s относительно её исходной ширины и высоты.

В [4, 9, 18] помимо текущих координат объекта и его размера так же хранились скорости перемещения по обеим координатам. А в [9] и вовсе для каждого отдельного сэмпла хранился свой классификатор-ансамбль C с отдельной историей обучения.

В данной работе мы будем учитывать как положение и масштаб рамки, так и скорость ее перемещения для каждого сэмпла:

$$x = (x, y, v_x, v_y, s)^\top.$$

2.2 Модель перехода

Во время шага перехода положение частиц в пространстве состояний изменяется в соответствии с динамической моделью перехода. Для предсказания новых положений частиц используется авторегрессионная модель первого порядка — новое состояние $\mathbf{x}_t$, основанное на предыдущем состоянии $\mathbf{x}_{t-1}$, получается за счет добавления нормального шума ко всем показателям в случае, когда мы не моделируем скорость

отслеживаемого объекта:

$$\begin{aligned} x_t &= x_{t-1} + N(0, \sigma_x^2), \\ y_t &= y_{t-1} + N(0, \sigma_y^2), \\ w_t &= w_{t-1} + N(0, \sigma_w^2), \\ h_t &= h_{t-1} + N(0, \sigma_h^2). \end{aligned}$$

В случае же наличия скоростей в модели, шум накладывается на них самих, тогда как координаты получаются простым выводом:

$$\begin{aligned} v_{x,t} &= v_{x,t-1} + N(0, \sigma_x^2), \\ v_{y,t} &= v_{y,t-1} + N(0, \sigma_y^2), \\ x_t &= x_{t-1} + v_{x,t}, \\ y_t &= y_{t-1} + v_{y,t}, \\ w_t &= w_{t-1} + N(0, \sigma_w^2), \\ h_t &= h_{t-1} + N(0, \sigma_h^2). \end{aligned}$$

2.3 Инициализация

В данной работе рассматривается инициализация, при которой начальное положение вместе с рамкой для объекта (x, y, w, h) получается из информации, передаваемой алгоритму. Веса частиц при инициализации принимаются равными между собой — $\pi_0^j = \frac{1}{J}$. В случае использования первых моментов, начальные скорости (v_x, v_y) могут быть либо приняты нулевыми, либо получены из некоторого нормального распределения.

После того, как все частицы были проинициализированы и произошел шаг перехода, необходимо вычислить апостериорное распределение текущего состояния, плотность которого аппроксимируется с помощью частиц.

Начиная с этого момента важность частицы, определяемая её весом, равна правдоподобию отвечающего ей наблюдения $p(Z_t|X_t = x_t^j, Z_0, Z_1, \ldots, Z_{t-1})$, которое вычисляется на основе модели наблюдения.

2.4 Модель наблюдения

Чтобы вычислить вес каждой частицы, нужно оценить правдоподобие наблюдения, отвечающего ей, т.е. оценивать схожесть региона, отвечающего частице, с шаблоном — таким же регионом для отслеживаемого объекта. Для этого мы будем извлекать признаки из регионов изображения и сравнивать их между собой. Далее под шаблоном также будут пониматься значения признаков для целевого объекта.

В данной работе будут рассмотрены признаки, зарекомендовавшие себя в задаче отслеживания объектов, моделирующие представления объекта с разной стороны, а значит подходящие для различных сложностей в исследуемых видео — одни модели используют цветовые признаки, другие же моделируют текстуру, контур или иные характеристики объекта. После обзора признаков мы рассмотрим способ подсчета правдоподобия, учитывающий схожесть по нескольким признакам.

2.4.1 Гистограммы значений цветовых каналов

Простейший способ создания цветовой модели региона изображения — рассчитать для каждого канала изображения, сколько раз встречается данное значение интенсивности, собрать полученные значения в гистограммы, а затем объединить все три поканальные гистограммы в одну, пронормировав — для 8-битного RGB-изображения мы для каждого региона получим вектор из 768 значений-признаков.

Измерить расстояние между векторами можно с помощью Евклидовой метрики:

$$\rho_E(\hat{h}, h(\mathbf{x_t})) = \sqrt{\sum_{i=1}^{768}(\hat{h}_i - h_i)^2},$$

где $\hat{h}$ — гистограмма для шаблона, а $h(\mathbf{x_t})$ — гистограмма для региона частицы $\mathbf{x_t}$, или с помощью расстояния Бхаттачарйа [2]:

$$\rho_B(\hat{h}, h(\mathbf{x_t})) = \sqrt{\sum_{i=1}^{768}\hat{h}_i \cdot h_i}.$$

Для оптимизации многократного подсчёта гистограмм можно воспользоваться интегральными изображениями, описанными далее. При этом мы будем считать количество пикселей с интенсивностью, попадающей в один из K равных диапазонов.

2.4.2 Признаки Хаара

Признаки Хаара впервые были описаны в [17], где они использовались в алгоритме для распознавания лиц, и на сегодняшний день применяются во многих алгоритмах классификации, т.к. обладают большей дискриминативной способностью, чем значения пикселей сами по себе.

Для каждого прямоугольного региона изображения в [17] суммировали значения интенсивности всех пикселей в нём, но при этом вместо прямого подсчета значений в циклах они предложили использовать *интегральные изображения*, у которых в каждой точке (x, y) содержится сумма интенсивностей всех пикселей левее и выше данной точки. Выгода от использования интегрального изображения в том, что, рассчитав его единожды за $O(w \times h)$, в дальнейшем можно находить такие суммы интенсивностей для любых прямоугольников за $O(1)$.

Чтобы получить интегральное изображение I на основе исходного F, для каждого пикселя необходимо вычислить значение по формуле

$$I(x, y) = F(x, y) + I(x - 1, y) + I(x, y - 1) - I(x - 1, y - 1),$$

где $I(x, -1) = I(-1, y) = 0$. А это, очевидно, можно сделать за один проход по результирующему изображению с помощью динамического программирования.

После того, как было получено интегральное изображение, чтобы посчитать сумму интенсивностей для прямоугольника с верхним левым углом (x_1, y_1) и нижним правым углом (x_2, y_2), нужно воспользоваться формулой:

$$\sum_{x=x_1}^{x_2} \sum_{y=y_1}^{y_2} F(x, y) = I(x_2, y_2) - I(x_2, y_1 - 1) - I(x_1 - 1, y_2) + I(x_1 - 1, y_1 - 1).$$

Данное выражение эквивалентно подсчету суммы для региона D изображения F с помощью вычисления $(A + B + C + D) - (A + B) - (A + C) + A$ для изображения I (см. рис. 3).

Таким образом, мы создаем интегральное изображение для всего кадра, а затем можем вычислять сумму для любого прямоугольного региона в кадре за константное время. Мы будем использовать интегральное изображение для вычисления средних

Рис. 3: Слева: исходное изображение F. По центру: интегральное изображения для изображения F. Справа: 4 региона, которые используются при подсчете суммы для региона D изображения F.

значений в каждом из трех каналов RGB:

$$(red_i, green_i, blue_i) = \sum_{x,y \in R_i} \frac{(red(x,y), green(x,y), blue(x,y))}{P_i},$$

где P_i — количество пикселей в прямоугольнике R_i. Подобные признаки использовались в [19], а использование именно цветных каналов появилось в [12].

В конце мы хотим определять правдоподобие частиц-кандидатов, поэтому нужно уметь сравнивать цветовую информацию из шаблона с информацией, полученной для частиц. Каждой частице соответствует прямоугольник на изображении R_i, который получается на основе вектора $\mathbf{x}_i$, отвечающего ей. Чтобы вычислить разницу между регионом, полученным для частицы и шаблоном для отслеживаемого объекта, воспользуемся Евклидовой метрикой:

$$\varepsilon_{HR}(\hat{c}, c(\mathbf{x}_t)) = \sqrt{\sum_{v=1}^{V}((\hat{red} - red_{(\mathbf{x}_t,v)})^2 + (\hat{green} - green_{(\mathbf{x}_t,v)})^2 + (\hat{blue} - blue_{(\mathbf{x}_t,v)})^2)},$$

где $\hat{c}$ — цветовая информация из шаблона, $c(\mathbf{x}_t)$ — цветовая информация для частицы, состоянию которой отвечает вектор $\mathbf{x}_t$, а v — один из типов признаков, изображенных ниже на рис. 4. Средние значения цветов для трёх правых признаков получаются вычитанием интегральных сумм для тёмных областей из таковых для светлых.

2.4.3 Гистограммы направленных градиентов

Признаки на основе цветов подходят для многих задач трекинга, даже когда происходят частичные наложения. Тем не менее, они плохо себя показывают в ситуации,

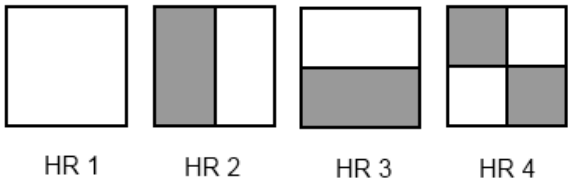


Рис. 4: Четыре прямоугольника Хаара для извлечения средних значений по каналам.

когда на фоне присутствуют похожие цвета. Было предложено множество других типов признаков для использования вместе с цветовыми. В [7] показали, что комбинация цветовой модели вместе с моделью контуров позволяет получить более быстрое и стабильное отслеживание объекта.

Для получения информации о контурах мы будем использовать гистограммы направленных градиентов (Histogram of Oriented Gradients, HOG). Они широко используются в распознавании жестов и нахождении объектов. По своей природе они устойчиво себя ведут в условиях изменения освещенности и в случае схожести фона и объекта по цветовой модели.

Для нахождения границ мы сначала переводим RGB-изображение в оттенки серого, а затем используем операторы Собеля K_x и K_y [16]:

$$G_x(x,y) = K_x * I(x,y), G_x(x,y) = K_y * I(x,y) \text{ (под } * \text{ понимается свертка).}$$

Тогда сила (резкость перехода) и ориентация границы вычисляются по формулам:

$$\begin{aligned} S(x,y) &= \sqrt{G_x^2(x,y) + G_y^2(x,y)}, \\ \theta &= \arctan(G_y(x,y)/G_x(x,y)). \end{aligned}$$

Чтобы избавиться от шума, можно применить порог T к значению $S(x,y)$ (используем $T = 100$). Затем границы распределяются по K «корзинам» в соответствии с их направлениями, после чего значения их сил суммируются, и получается нужная нам гистограмма. В оригинальной работе [3] лучше всего показало себя $K = 9$.

Для эффективного вычисления гистограммы для прямоугольного региона можно построить K интегральных изображений для каждого из «каналов» (корзин). Схожесть между шаблоном и гистограммой для каждой частицы вычисляются аналогично предыдущим признакам.

2.4.4 Локальные бинарные шаблоны

Одним из способов описания объектов являются т.н. локальные признаки, главная идея которых заключается в том, чтобы характеризовать не всё изображение целиком (к примеру, его общий цвет) а какие-то локальные свойства. Поиск ребер в точке, осуществляемый с помощью фильтров Собеля, является одним из примеров такого подхода. Другим примером данного подхода является извлечение локальных бинарных шаблонов (Local Binary Patterns, LBP), которые в каком-то смысле характеризуют текстуру изображения в каждой конкретной точке, для чего и были описаны впервые в [14].

Главная идея данного метода состоит в извлечении локальной структуры путём сравнивания каждого пикселя с его соседями — для каждого соседа мы получим число, которое будет равно 1, если интенсивность соседа больше рассматриваемого центрального пикселя, и 0 в противном случае. В итоге, если объединить полученные значения, например по часовой стрелке, то текстуру в окрестности каждого пикселя будет описывать бинарный вектор вроде 00010011, который и называется локальным бинарным шаблоном (см. рис. 5).

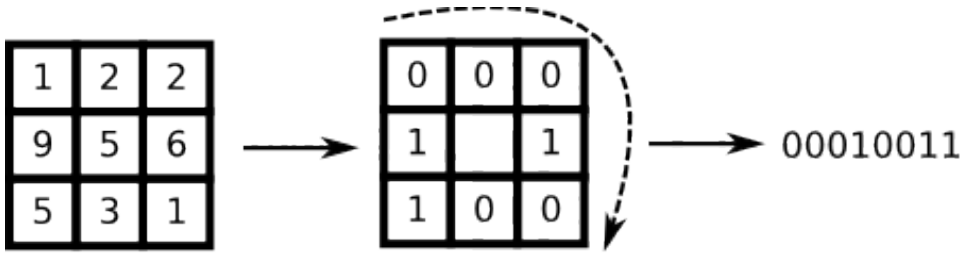


Рис. 5: Получение локального бинарного шаблона.

Так же как и в случае с гистограммами направленных градиентов, эти признаки можно разделить на 8 отдельных «корзин» и с помощью техники интегральных изображений получать для каждого участка изображения гистограммы с 8 числами, отвечающими за количества тех или иных локальных особенностей в данном участке. Схожесть между шаблоном и гистограммой для каждой частицы вычисляются аналогично предыдущим признакам.

2.4.5 Правдоподобие наблюдения

Распределение правдоподобия на основе одной метрики $\rho(\cdot, \cdot)$ можно получить по формуле:

$$p(\mathbf{z_t}|\mathbf{x_t}) \propto \exp\{-\rho^2(\hat{h}, h(\mathbf{x_t}))/\lambda\},$$

где λ – параметр, подбираемый отдельно для каждой пары признака и соответствующей ему метрики.

Самым простым способом объединить схожесть по нескольким признаком в общий показатель является перемножение соответствующих правдоподобий. Тем не менее, не всегда это может быть разумным — в некоторых случаях выделение одних признаков является более затратным с вычислительной точки зрения, чем других, а значит для них меньше количество «кандидатов», которые можно проверить на схожесть с шаблоном за определенное время. Для разрешения данной проблемы в [19] была преложена каскадная схема наподобие таковой в [17], где только для некоторых «самых правдоподобных» по первой части признаков частиц производился подсчет правдоподобия по второй части — для не попавших же в эти некоторые частицы такое правдоподобие устанавливалось равным минимум из полученных значений.

В данной работе рассматриваются признаки, не сильно отличающиеся друг от друга по вычислительным затратам ввиду использования описанной техники интегральных изображений, поэтому общее правдоподобие наблюдение вычисляется как произведение правдоподобий, соответствующих используемым признакам:

$$p(\mathbf{z_t}|\mathbf{x_t}) \propto \Pi_f \exp\{-\rho_f^2(\hat{h}_f, h_f(\mathbf{x_t}))/\lambda_f\}.$$

2.5 Ресэмплинг

После описанных шагов предсказания на основе модели движения и коррекции на основе модели наблюдения алгоритм фильтра частиц совершает ресэмплинг, описанный ранее — мы получаем новый набор частиц на основе текущего, где частицы с большим весом дублируются чаще, а веса новых частиц принимаются равными между собой. Таким образом мы избегаем ситуации, когда маловероятные части-

цы получают все меньший вес и ситуации сильного изменении положения объекта характеризуются моделью как невозможные.

2.6 Борьба с наложениями

Несмотря на то, что все описанные признаки в каком-то смысле устойчивы к частичным перекрытием объекта на видео, полные перекрытия все еще являются проблемой и заслуживают отдельного решения. При полном перекрытии сравнение кандидатов с «реальным» объектом не имеет смысла, т.к. реального объекта на сцене просто нет. А значит в этом случае ресэмплинг фильтра частиц будет вести себя некорректно. Чтобы справиться с этой ситуации, можно добавить специальный режим алгоритма «Потеря цели», который будет вводиться, если минимальное из расстояний между признаками регионов и шаблона больше некоторого порога. В этом случае фильтр частиц пропускает шаг ресэмплинга. Это позволит всем частицам продолжать находиться там же или двигаться с той же скоростью, что и объект до потери цели. После того, как первая частица превысит нужный порог, алгоритм переходит в обычный режим. Тем не менее, если это состояние длится слишком долго, алгоритм может уже не суметь восстановить «наведение на цель», а значит потребуется реинициализация, которую можно выполнить равномерным расположением частиц по всему кадру.

3 Программная реализация

Для тестирования алгоритмов с использованием описанных признаков была написана программа с графическим интерфейсом, позволяющая загружать видео из датасета, выбирать алгоритм с признаками и количество используемых частиц, запускать алгоритмы и наблюдать результат их работы во встроенном просмотрщике видео с возможностью отображения истинной и оцененной разметок, а так же положения всех частиц. На основе истинной разметки программа также позволяет сохранять в файл отчет с качеством работы алгоритма по каждому кадру видео для последующего сравнения графиков между собой.

Все рассмотренные алгоритмы с учетом различных признаков были реализованы на языке C++. Для высокоуровневой работы с изображениями и видео использовалась библиотека OpenCV, которая написана также на C и C++, что влечет высокую скорость работы и эффективность её использования в сторонних приложениях. В частности, с помощью этой библиотеки происходила загрузка видео и разбор по кадрам на отдельные изображения, а для изображений применялись встроенные функции для подсчета градиентов и интегральных изображений.

Для создания интерфейса использовались средства для создания графического интерфейса из Qt — кроссплатформенного инструментария для разработки программного обеспечения на языке C++, что позволяет запускать итоговую программу на множестве платформ, поддерживаемых Qt и OpenCV — Windows, Mac OS, ОС семейства Linux.

Для демонстрации работы алгоритмов, оперирующих изображениями OpenCV, был написан графический виджет для Qt, позволяющий отображать их в окне любой программы с графическим интерфейсом на Qt.

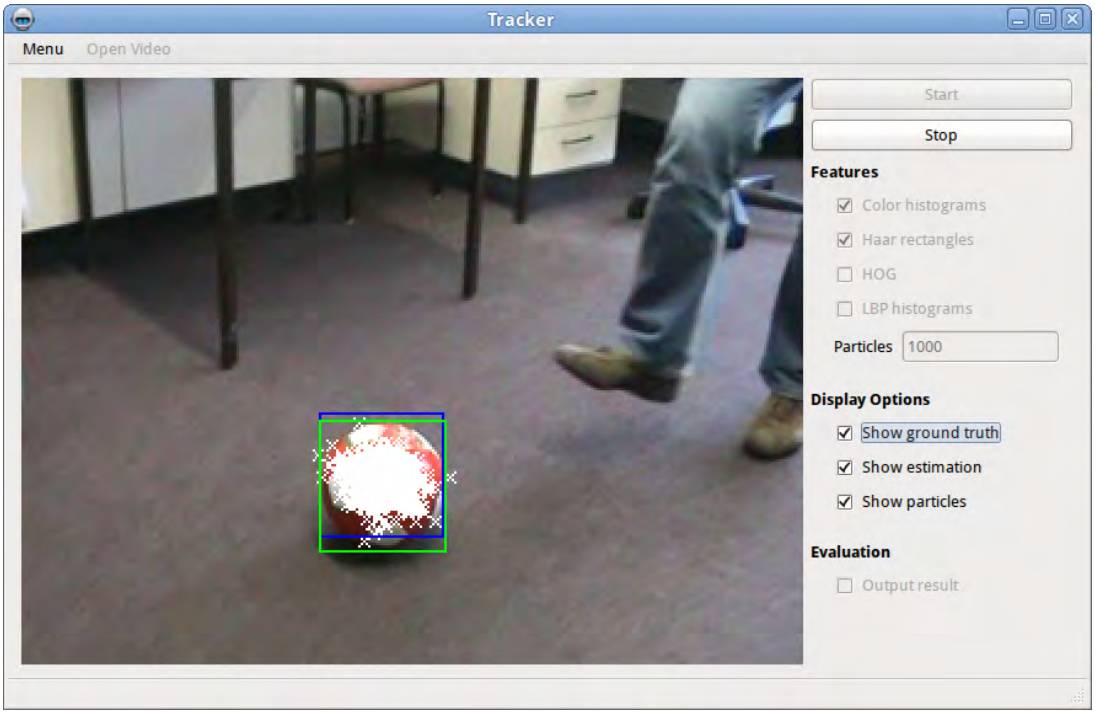


Рис. 6: Окно работающей программы.

Изображение с окном работающей программы можно увидеть на рис. 6.

4 Вычислительные эксперименты

4.1 Используемая мера качества

Для того, чтобы определять, насколько выделенный нами регион совпадает с реальным, нужно учитывать не только то, насколько близок центр рассчитанного нами выделения к реальному центру отслеживаемого объекта, но и то, насколько велика разница между реальными и вычисленными размерами объекта.

Для экспериментов в данной работе использовалась мера качества, предложенная в [9], определяющая долю пересечения двух областей в их объединении. Так же как и в упомянутой работе, будем считать, что качеству выше 33% соответствует правильное определение положения объекта.

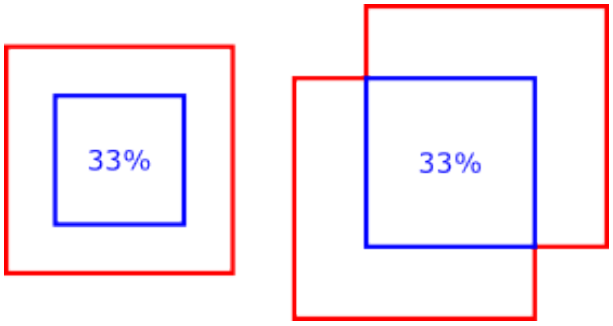


Рис. 7: Мера качества — доля пересечения регионов в их объедении.

4.2 Данные для экспериментов

Для многих областей применения трекинга существуют свои наборы видео, отражающие ту или иную специфику — например, видео с воздушных дронов для отслеживание машин или статичное видео с камеры наблюдения.

Нам же важно, чтобы данные отвечали начальному требованию о произвольности природы как отслеживаемых объектов, так и появляющихся осложнений на видео — таких как наложения, частичное изменение текстуры, плохое освещение и др.

Один из отвечающих этим требованием наборов был предложен в [9] группой ученых из университета Бонна — Bonn Benchmark on Tracking (BoBoT) [10].

4.2.1 Bonn Benchmark on Tracking (BoBoT)

BoBot является датасетом общей направленности и не принадлежит к какой-либо конкретной области. Он доступен для скачивания на сайте создателей [10], где также предоставлен исходный код программы на `Java` для оценивания качества алгоритмов на основе файлов истинной разметки и разметки, полученной ими самими. Тем не менее, в данной работе оценивание производилось собственным алгоритмом на `C++` для возможности интеграции в итоговую программу.

На рис. 8–16 приведены первые кадры каждого из используемых видео из данного датасета с их оригинальными названиями — мы так же дальше будем именовать их по порядку латинскими буквами, как в первоисточнике.

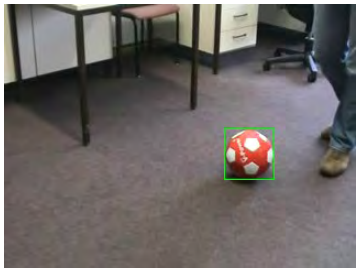


Рис. 8: ball (A).

Рис. 9: cup (B).

Рис. 10: juice box (C).

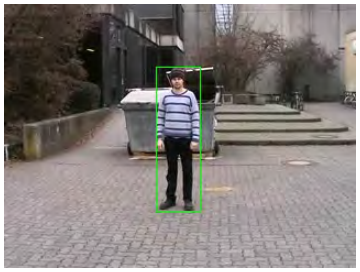


Рис. 11: person (D).

Рис. 12: person (E).

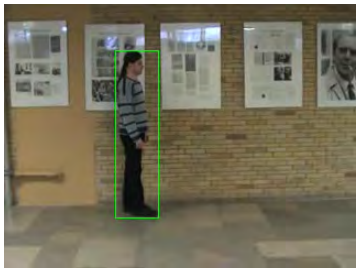


Рис. 13: person (F).

Рис. 14: Rubik's Cube (G).

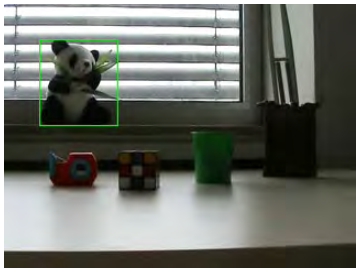


Рис. 15: panda (H).

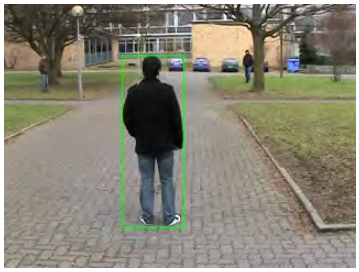


Рис. 16: person (I).

Видео A. Красно-белый мяч пасуют от одного человека к другому и обратно. Изменяется как положение отслеживаемого объекта, так и его представление, т.к. мяч при этом вращается.

Видео B. Синюю с цветными картинками кружку проносят на фоне стены с досками с большим количеством объявлений. Все время сильно изменяются вид фона.

Видео C. Снимают коробку из под сока среди других объектов, стоящих на столе. При этом камера все время движется в плоскости изображения, а в конце удаляется.

Видео D. Человек сначала поворачивается на месте, а затем идёт в одну сторону, продолжая поворачиваться. При этом незначительно изменяется вид отслеживаемого объекта — цвет и текстура в целом сохраняются.

Видео E. В центре кадра все время находится человек, стоящий на месте. При этом камера монотонно движется сначала в одну, а затем в другую сторону так, что на части видео объект оказывается наполовину перекрыт чёрной колонной.

Видео F. Камера следует параллельно движущемуся человеку, идущему в правую относительно оператора сторону. При этом человек трижды оказывается перекрыт чёрной колонной, находящейся между человеком и камерой, и единожды — другим человеком, идущим навстречу по той же причине.

Видео G. Камеру сначала проносят вокруг стола, сохраняя в центре кадра кубик Рубика, лежащий среди прочих объектов, а затем приближают к нему. По причине такого вращения объекта относительно камеры изменяется его цветовое представление (сначала видны одни паттерны 3×3 на гранях кубика, затем другие).

Видео H. Камера статично снимает игрушечную панду на фоне окна рядом с другими объектами на столе. При этом сначала закрывают жалюзи, уменьшая освещение, а затем включается/выключается лампа накаливания с жёлтым оттенком.

Видео I. Камера следует позади человека, идущего по пешеходной тропе с поворотом в конце. При этом позади него время от времени перебегают с десяток других людей, перекрывающих его на увеличивающиеся по ходу видео промежутки времени.

4.3 Результаты

В данном пункте рассмотрены результаты экспериментов по использованию трёх групп различных комбинаций рассмотренных ранее признаков. Предпосылки к разделению комбинаций признаков по этим группам, а также комментарии по результатом и выводы о том, как добавление тех или иных признаков из группы помогает избегать соответствующих сложностей на видео, будут освещены в следующем пункте.

4.3.1 Одиночные признаки

На рисунках 17–25 изображены графики с распределением качества по кадрам для алгоритмов, который для подсчета правдоподобия используют только один из рассмотренных ранее признаков — гистограммы цветов (color), гистограммы направленных градиентов (hog), прямоугольники Хаара (hr) и локальные бинарные шаблоны (lbp). Пунктирными линиями отображён тот порог, по которому мы определяем успешность отслеживания объекта в данный момент времени.

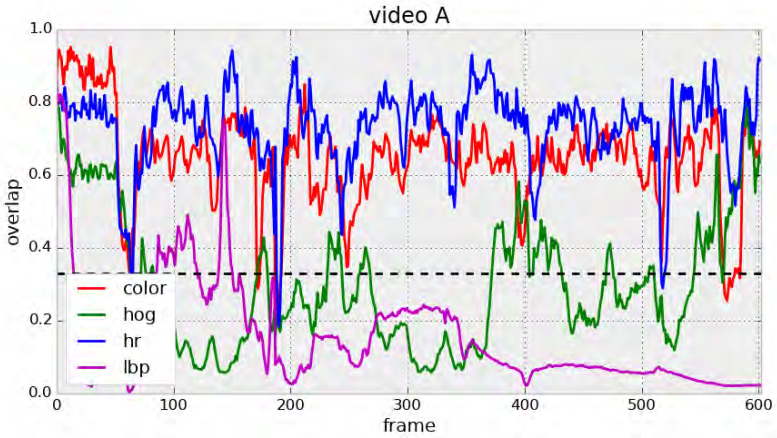


Рис. 17: Видео A: распределение качества по кадрам для одиночных признаков.

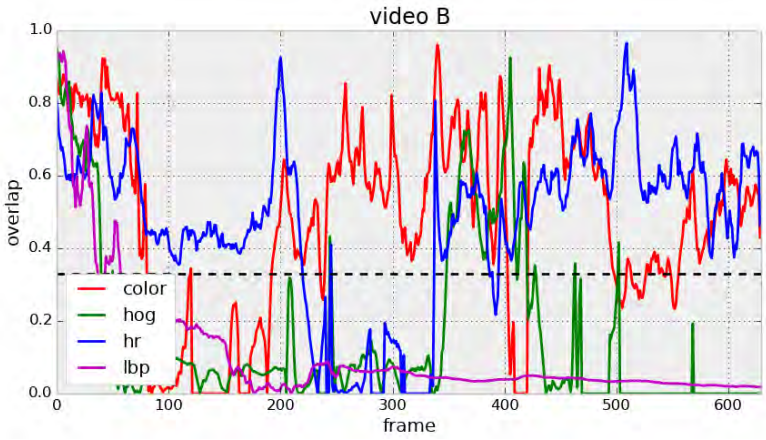


Рис. 18: Видео B: распределение качества по кадрам для одиночных признаков.

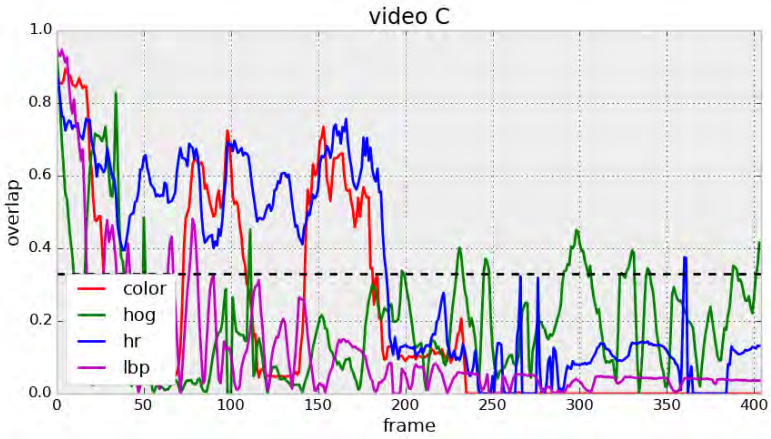


Рис. 19: Видео C: распределение качества по кадрам для одиночных признаков.

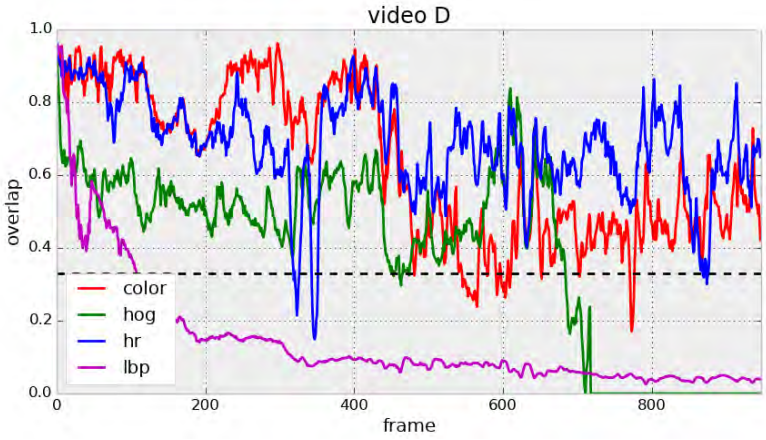


Рис. 20: Видео D: распределение качества по кадрам для одиночных признаков.

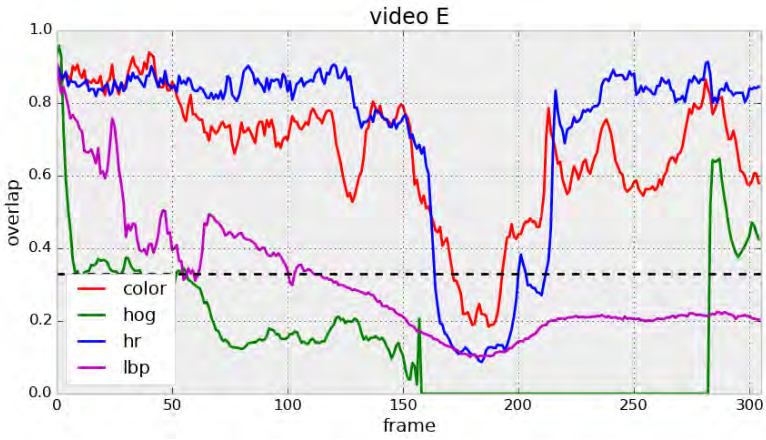


Рис. 21: Видео E: распределение качества по кадрам для одиночных признаков.

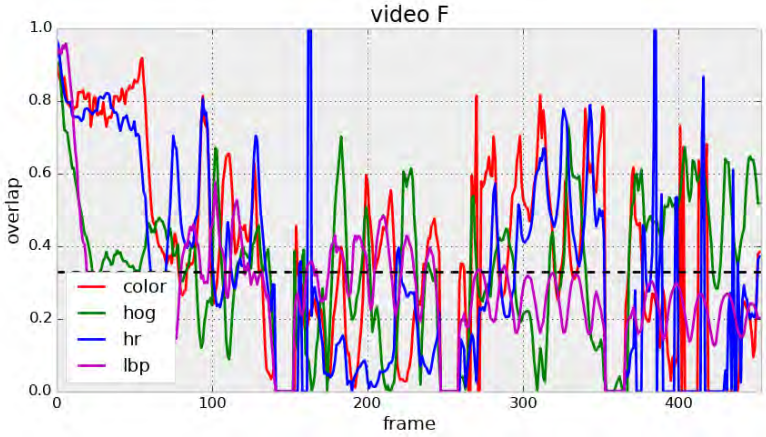


Рис. 22: Видео F: распределение качества по кадрам для одиночных признаков.

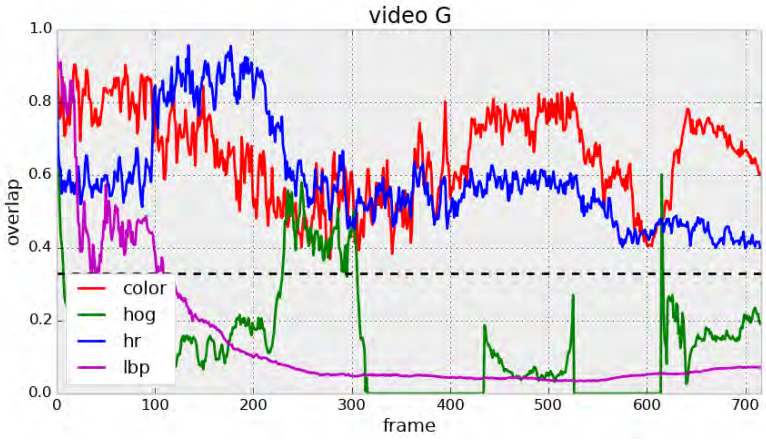


Рис. 23: Видео G: распределение качества по кадрам для одиночных признаков.

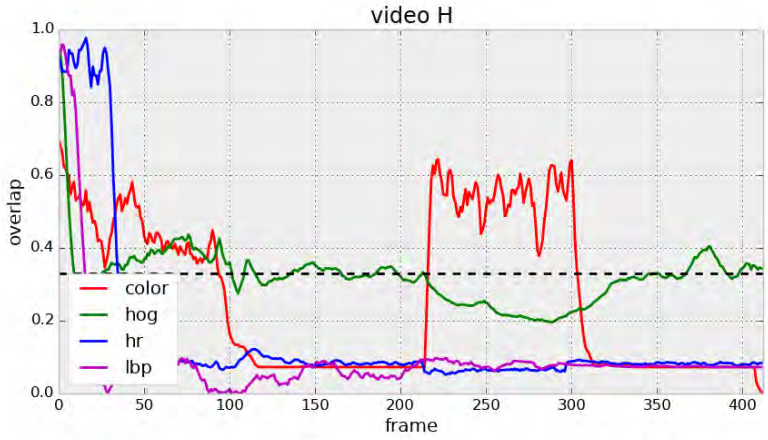


Рис. 24: Видео H: распределение качества по кадрам для одиночных признаков.

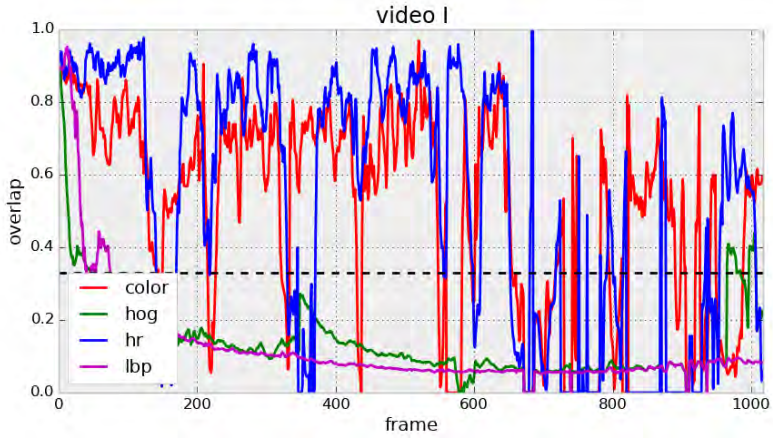


Рис. 25: Видео I: распределение качества по кадрам для одиночных признаков.

4.3.2 Комбинация цветовых признаков

На рисунках 26–34 изображены графики с распределением качества по кадрам для алгоритмов, который для подсчета правдоподобия используют оба набора цветовых признаков, а так же их комбинацию.

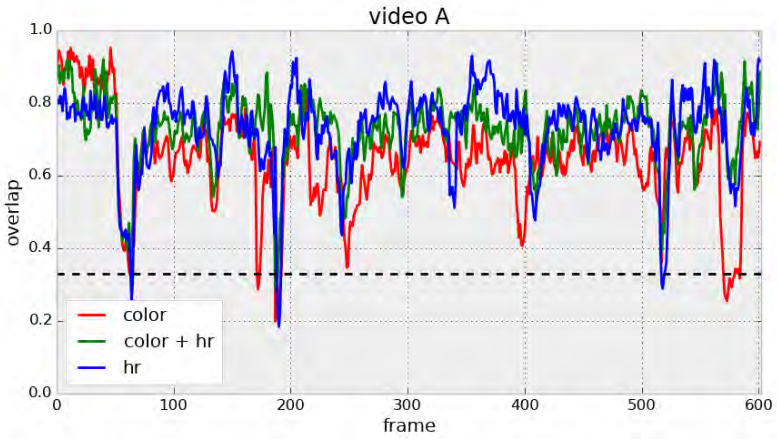


Рис. 26: Видео A: распределение качества по кадрам для цветовых признаков.

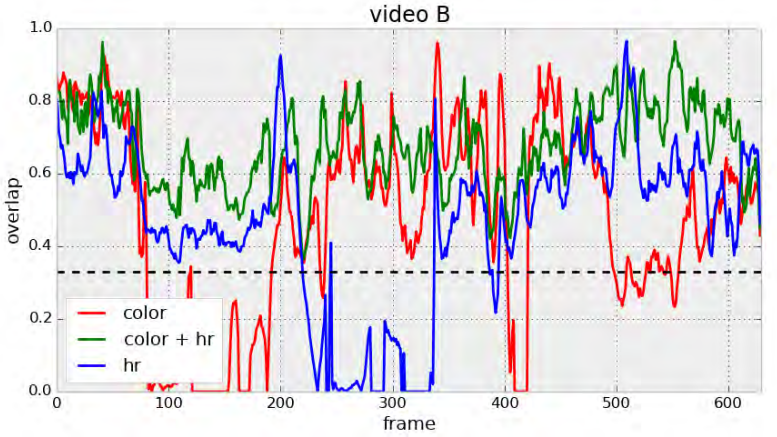


Рис. 27: Видео B: распределение качества по кадрам для цветовых признаков.

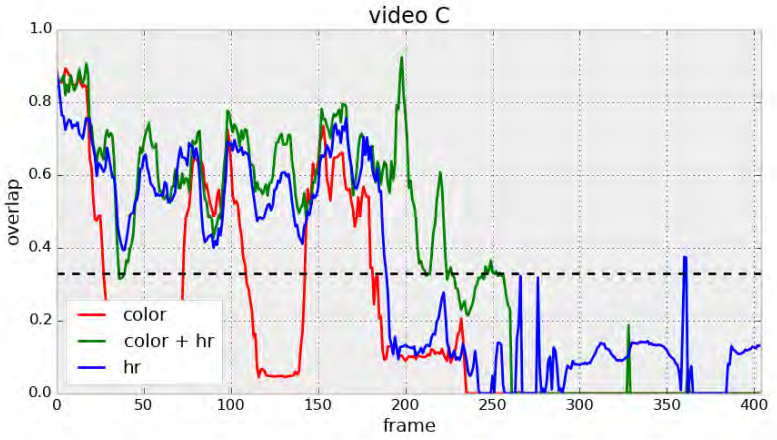


Рис. 28: Видео C: распределение качества по кадрам для цветовых признаков.

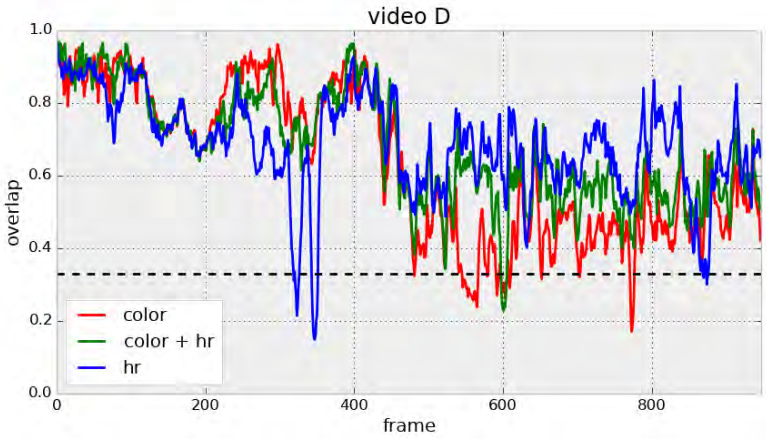


Рис. 29: Видео D: распределение качества по кадрам для цветовых признаков.

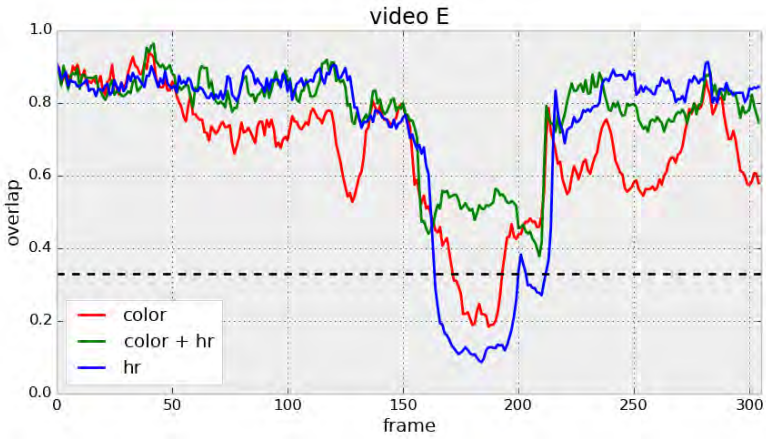


Рис. 30: Видео E: распределение качества по кадрам для цветовых признаков.

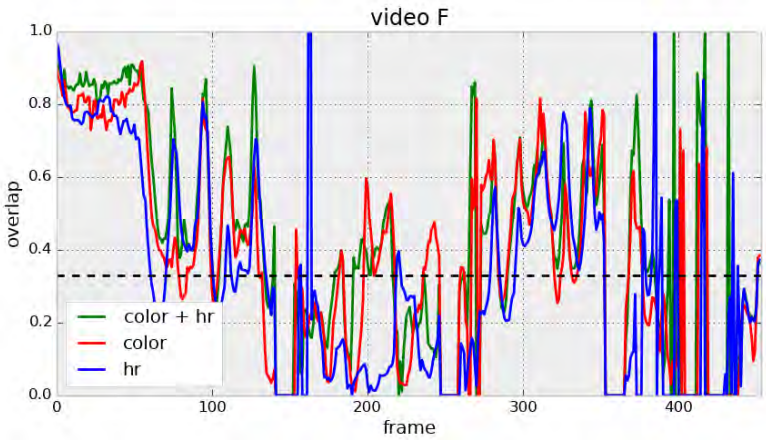


Рис. 31: Видео F: распределение качества по кадрам для цветовых признаков.

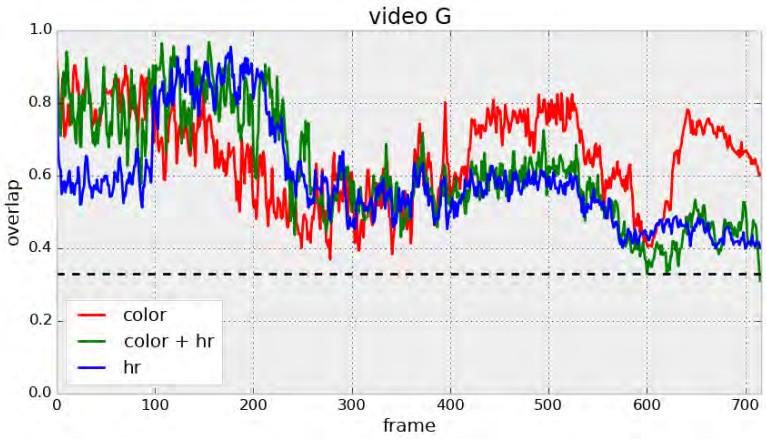


Рис. 32: Видео G: распределение качества по кадрам для цветовых признаков.

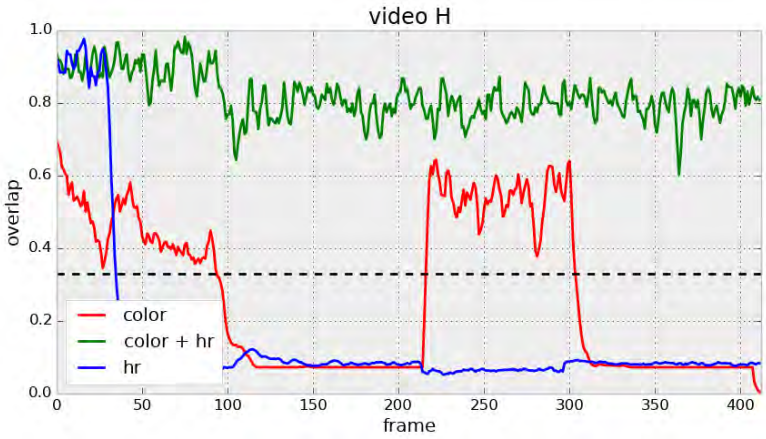


Рис. 33: Видео H: распределение качества по кадрам для цветовых признаков.

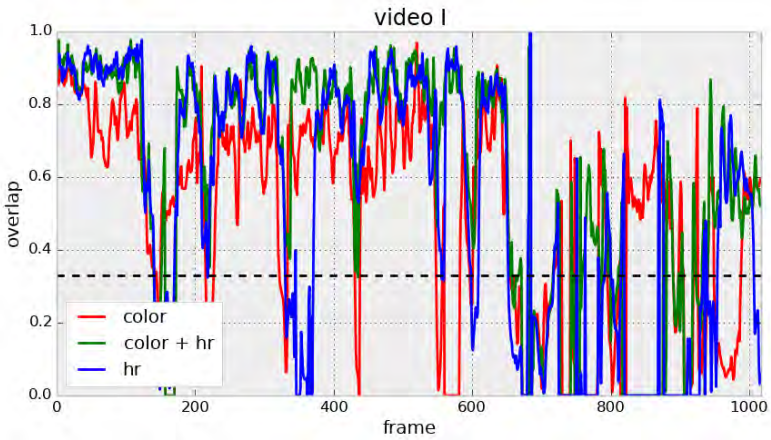


Рис. 34: Видео I: распределение качества по кадрам для цветовых признаков.

4.3.3 Добавление структурных признаков

На рисунках 35–43 изображены графики с распределением качества по кадрам для алгоритмов, которые вместе с рассмотренной в предыдущем подпункте комбинацией цветовых признаков учитывают один или оба признака, определяющих структурные (или текстурные) характеристики объектов, в сравнении с алгоритмом, таковые не использующим.

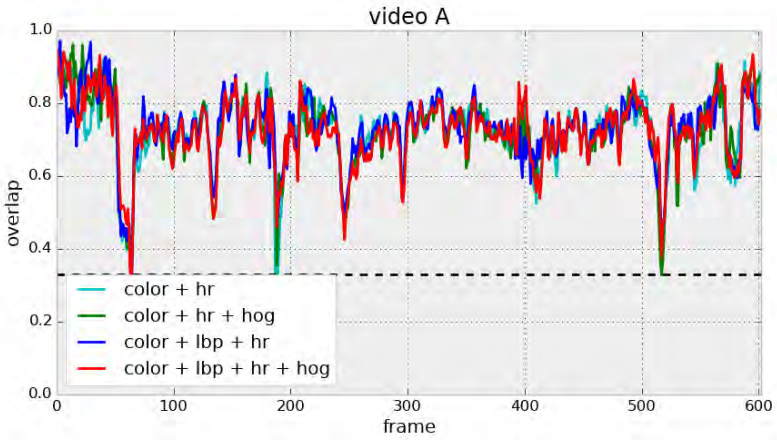


Рис. 35: Видео A: распределение качества по кадрам для структурных признаков.

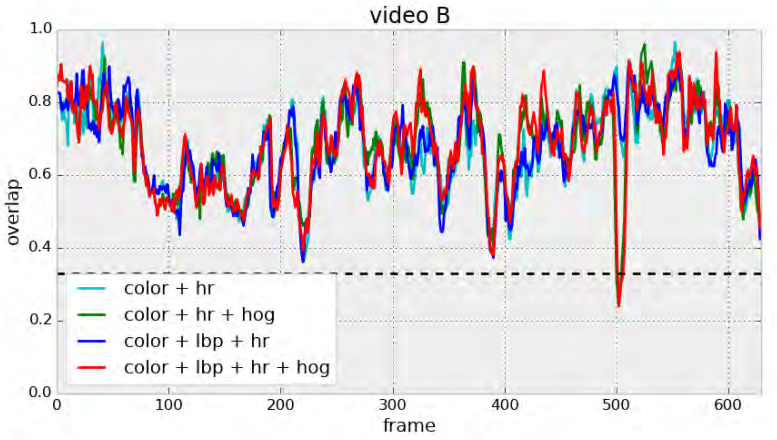


Рис. 36: Видео B: распределение качества по кадрам для структурных признаков.

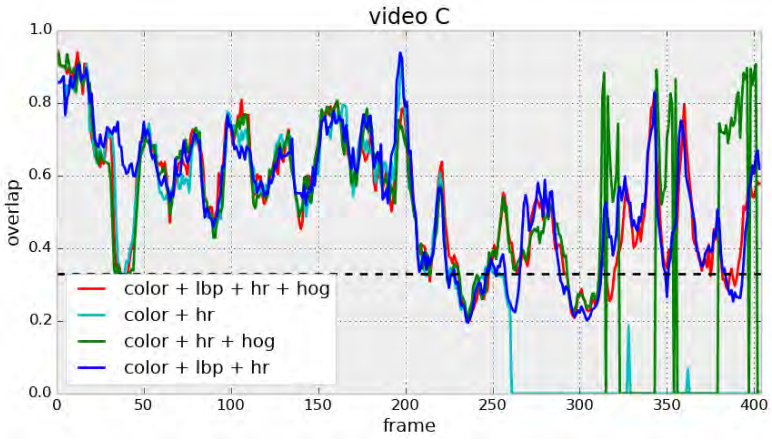


Рис. 37: Видео C: распределение качества по кадрам для структурных признаков.

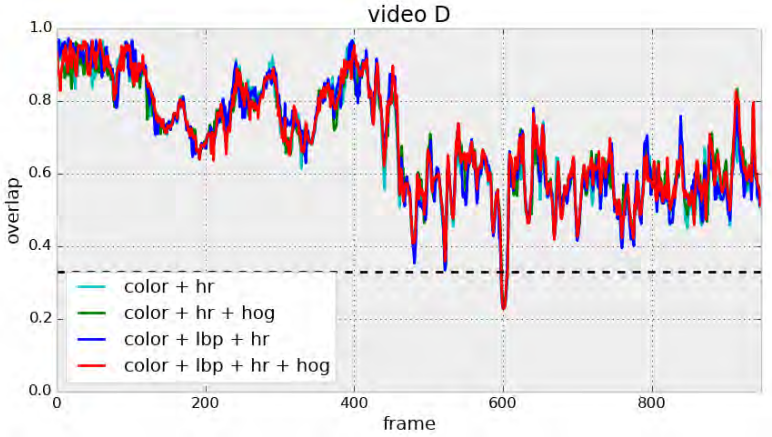


Рис. 38: Видео D: распределение качества по кадрам для структурных признаков.

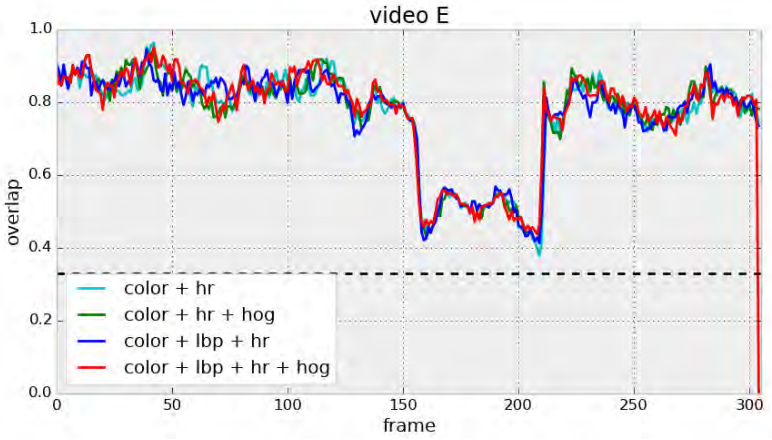


Рис. 39: Видео E: распределение качества по кадрам для структурных признаков.

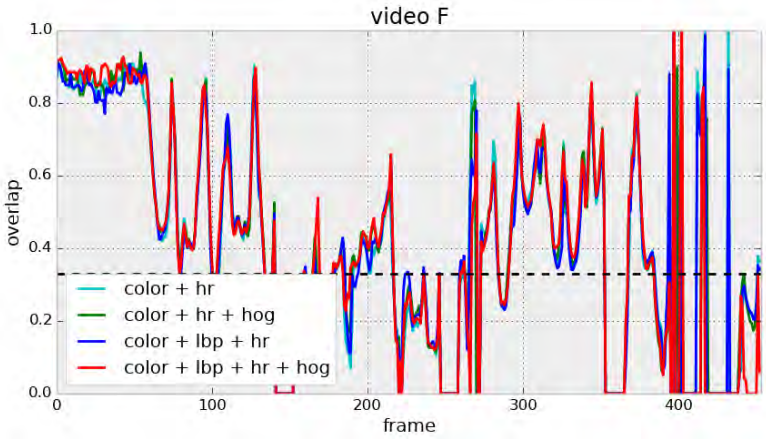


Рис. 40: Видео F: распределение качества по кадрам для структурных признаков.

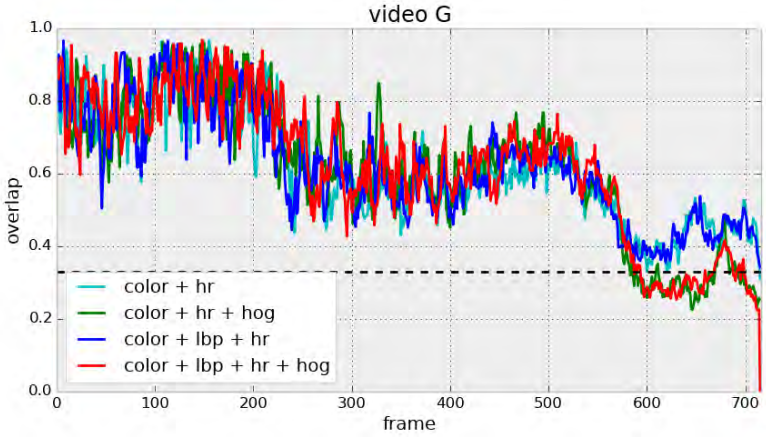


Рис. 41: Видео G: распределение качества по кадрам для структурных признаков.

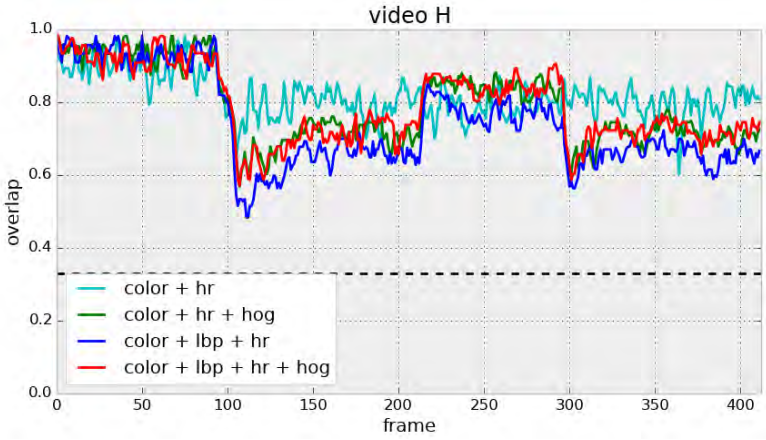


Рис. 42: Видео H: распределение качества по кадрам для структурных признаков.

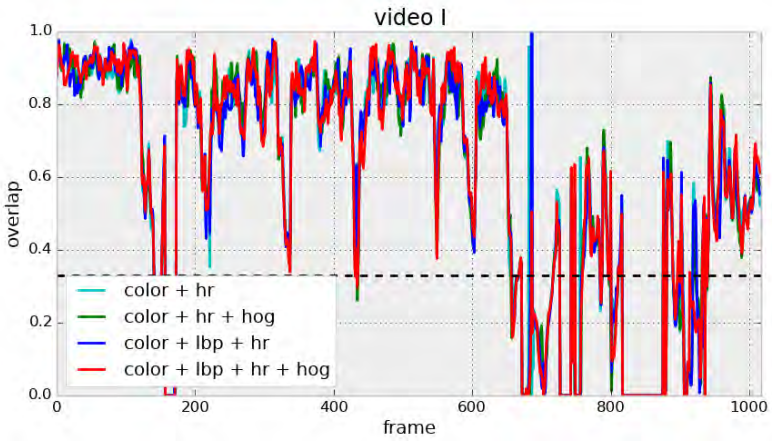


Рис. 43: Видео I: распределение качества по кадрам для структурных признаков.

4.4 Обсуждение и выводы

В первой части экспериментов были рассмотрены алгоритмы, использующие только одну характерную группу признаков. По соответствующим графикам (рис. 17–25) можно сделать выводы, что самодостаточными для трекинга признаками являются цветовые гистограммы и прямоугольники Хаара. Лишь в некоторых видео значимый результат также показывали гистограммы градиентов. Использование локальных бинарных шаблонов видится осмысленным только в композиции с другими признаками.

При использовании этих двух групп признаков (color, hr) значимые отличия в результатах проявились в анализе двух видео. На видео B с кружкой на очень пёстром и разнообразном по текстуре фоне оба алгоритма на основе этих признаков теряют на некоторое время цель, но в разное время — один алгоритм находит похожую на кружку синюю сплошную часть доски, другой видит похожие с шаблонными перепады в цветах на графиках и фотографиях на доске. На видео H признаки Хаара быстро находят схожую по перепадам область и там же и остаются, тогда как цветовые гистограммы точно находят объект только при фиксированном типе освещения.

Т.к. оба типа признаков имеют схожую вычислительную сложность, было решено сравнить их качество по отдельности с композицией (рис. 26–34), а затем исследо-

вать, как улучшает качество композиции добавление двух оставшихся «несамодостаточных» признаков (рис. 35–43).

Использование композиции позволяет нам использовать получать от обеих групп признаков, что подтверждается экспериментами — на видео B при использовании композиции мы никогда не уходим от отслеживаемой кружки к схожим частям фона и добиваемся 100%-го качества в смысле отношения длины удачного трекинга к общей длине видео. На видео E композиция лучше справляется с появлением колонны, и не захватывает ее в качестве предполагаемого объекта. Самый сильный же эффект от использования композиции достигается на видео H — мы получаем 100%-е качество трекинга с постоянным пересечением с реальной областью на уровне 80% при том, что каждый признак сам по себе совсем не справлялся с трекингом на данном видео.

При добавлении к изученной композиции структурных признаков на видео C стабильно лучше себя показали алгоритмы с добавлением LBP-признаков, что привело к очень высокому качеству трекинга для такой сложной задачи с значительными движением камеры и изменением масштаба объекта. На видео G добавление HOG уменьшило пересечения в конце где-то на четверть, что привело к вылету из минимальной зоны 33%-го качества.

4.5 Сравнение с другими работами

На рис. 44 и 45 приведены графики качества для одной из недавних работ [15], основанной на использовании метода каскадов для прямоугольников Хаара и гистограмм направленных градиентов, использующей BoBoT для экспериментов. На графиках присутствует качество отдельно для признаков и для их общей каскадной композиции.

Как видно из приведенных графиков, помимо того, что построенные базовые алгоритмы, в отличие от данной работы, не приводят к положительному результату на большинстве видео, даже построенные композиции не справились с «цветастым» и «разнофоновым» видео B, с перемещающейся камерой на видео C, а также полностью потеряли человека на видео E и F.

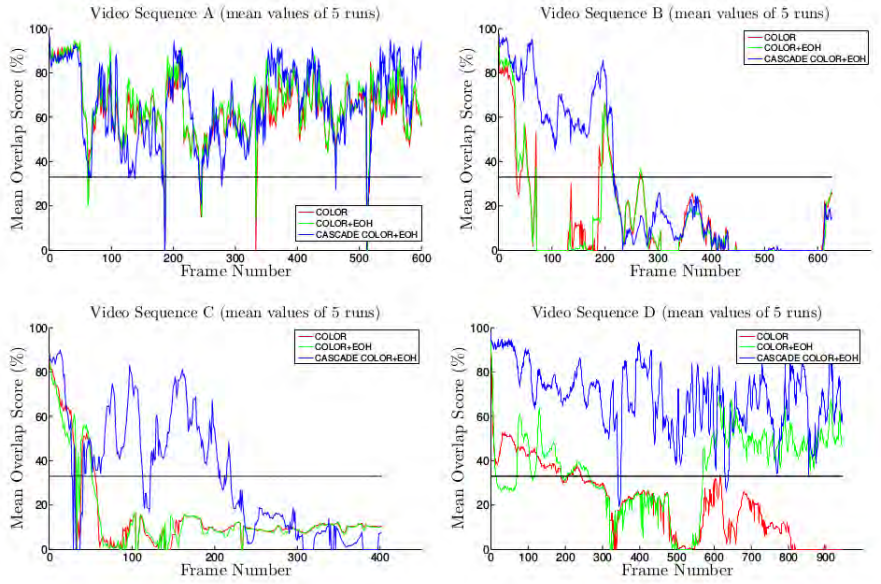


Рис. 44: Графики качества в работе на основе метода каскадов.

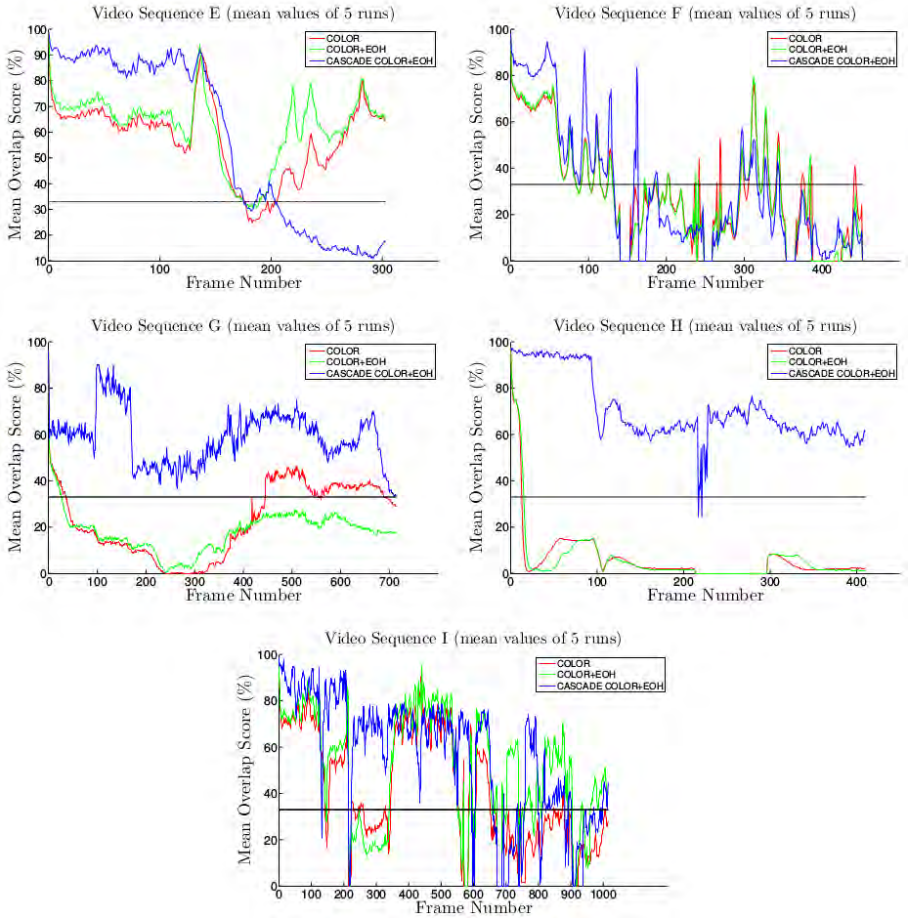


Рис. 45: Графики качества в работе на основе метода каскадов.

На рис. 46 приведены графики качества для оригинальной работы, в которой был впервые представлен используемый бенчмарк. В работе рассматривались методы построения ансамблей простых классификаторов для каждой частицы на основе прямоугольников Хаара с помощью алгоритма Gentle AdaBoost [9].

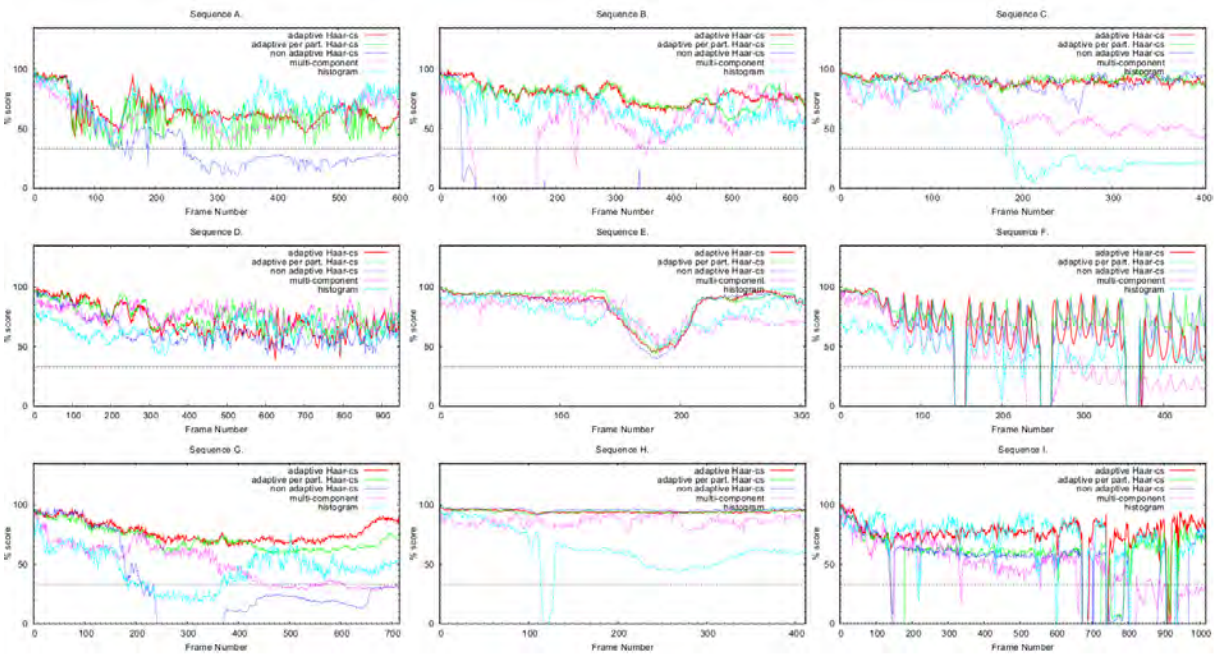


Рис. 46: Графики качества в работе с применением бустинга.

Голубым, сиреневым и синим линиями на этих графиках соответствуют алгоритмы на основе цветовых гистограмм и бустинга признаков Хаара. Красным и зеленым линиями с более стабильным и высококачественным трекингом объектов соответствует более сложная адаптивная схема с применением того же бустинга, учитывающая историю наблюдений каждой частицы.

Из графиков можно сделать вывод, что простые «цветовые алгоритмы» и даже неадаптивная схема с применением бустинга порой не позволяли добиться приемлемых результатов для сложных с точки зрения текстуры и перемещения камеры видео B и C соответственно — в отличие от методов, предложенных в данной работе.

5 Заключение

В работе изучены способы выделения признаков из изображений для задачи трекинга объектов на видео, а также приведены способы их оптимального многоразового подсчета, построены алгоритмы с использованием композиций на их основе.

Из проведенных экспериментов можно заключить, что наиболее универсальным в рамках рассматриваемых видео оказался композиционный алгоритм, основанный на использовании цветовых интегральных признаков, цветовых гистограмм и локальных бинарных шаблонов. Тем не менее, простая комбинация из двух цветовых групп признаков также дает высокие результаты, компенсируя недостатки каждой группы в отдельности. На рис. 47 приведена сравнительная таблица с результатами экспериментов для всех рассмотренных алгоритмов с указанием их качества в смысле отношения количества моментов удачного отслеживания объектов к общему количеству кадров в видео.

Группа признаков	A	B	C	D	E	F	G	H	I
color	0.966	0.732	0.252	0.947	0.931	0.556	**1.000**	0.441	0.719
hr	0.986	0.804	0.475	0.977	0.855	0.439	**1.000**	0.084	0.649
hog	0.367	0.184	0.178	0.714	0.200	0.569	0.117	0.415	0.074
lbp	0.102	0.081	0.091	0.114	0.347	0.256	0.146	0.038	0.066
hr + color	0.993	**1.000**	0.564	**0.991**	**1.000**	0.602	0.997	**1.000**	0.779
hr + color + lbp	**0.998**	**1.000**	**0.836**	**0.991**	**1.000**	**0.613**	**1.000**	**1.000**	**0.782**
hr + color + hog	0.996	0.992	0.764	**0.991**	**1.000**	**0.613**	0.853	**1.000**	0.770
hr + color + lbp + hog	0.996	0.992	0.863	0.990	0.996	**0.613**	0.863	**1.000**	0.772

Рис. 47: Сравнительная таблица всех рассмотренных алгоритмов.

С помощью рассмотренных композиций признаков достигнуто качество трекинга, сравнимое с более продвинутыми методами, основанными на построении сложных ансамблей с помощью бустинга, и превышающее результаты в схожей работе с использованием методов каскадов.

В рамках работы также создана реализация программы с графическим интерфейсом для демонстрации работы всех изученных алгоритмов с возможностью выбора признаков, которые необходимо учитывать в модели наблюдения.

Список литературы

[1] *Arulampalam M. S., Maskell S., Gordon N.* A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking // *IEEE Transactions on Signal Processing.* — 2002. — Vol. 50. — Pp. 174–188.

[2] *Bhattacharyya A. K.* On a measure of divergence between two statistical populations defined by their probability distributions // *Bulletin of CalMathSoc.* — 1943. — Vol. 35, no. 1. — Pp. 99–109.

[3] *Dalal N., Triggs B.* Histograms of oriented gradients for human detection // In CVPR. — 2005. — Pp. 886–893.

[4] *Esther K. N., Koller-meier E., Gool L. V.* A color-based particle filter. — 2002. — Pp. 53–60.

[5] *Fox D., Thrun S., Dellaert F., Burgard W.* Particle filters for mobile robot localization // Sequential Monte Carlo Methods in Practice / Ed. by A. Doucet, N. de Freitas, N. Gordon. — New York: Springer Verlag, 2000. — To appear.

[6] *Grabner H., Grabner M., Bischof H.* Real-time tracking via on-line boosting. — 2006.

[7] *Isard M., Blake A.* Icondensation: Unifying low-level and high-level tracking in a stochastic framework // Proceedings of the 5th European Conference on Computer Vision-Volume I - Volume I. — ECCV '98. — 1998. — Pp. 893–908.

[8] *Kalal Z., Matas J., Mikolajczyk K.* Online learning of robust object detectors during unstable tracking // In International Conference on Computer Vision. — 2009.

[9] *Klein D. A., Schulz D., Frintrop S., Cremers A. B.* Adaptive real-time video-tracking for arbitrary objects // IEEE Int. Conf. on Intelligent Robots and Systems (IROS). — Oct 2010. — Pp. 772–777.

[10] *Klein D. A., Schulz D., Frintrop S., Cremers A. B.* BoBoT - Bonn Benchmark on Tracking. — `http://www.iai.uni-bonn.de/~kleind/tracking/`. — 2010.

[11] *Li H., Xiong S., Duan P., Kong X.* Multitarget tracking of pedestrians in video sequences based on particle filters // *Adv. MultiMedia.* — jan 2012. — Vol. 2012.

[12] *Maggio E., Cavallaro A.* Multi-part target representation for colour tracking // Proc. of IEEE Int. Conference on Image Processing (ICIP). — Genoa, Italy: 11–14 September 2005.

[13] *Murphy K. P.* Machine Learning: A Probabilistic Perspective. — The MIT Press, 2012.

[14] *Pietikäinen M., Ojala T., Xu Z.* Performance evaluation of texture measures with classification based on kullback discrimination of distributions // 12th IAPR International Conference on Pattern Recognition (ICPR 1994). — 1994. — Pp. 582–585.

[15] *Samuelsson O.* Video Tracking Algorithm for Unmanned Aerial Vehicle Surveillance. — June 2012.

[16] *Sobel I., Feldman G.* A 3x3 Isotropic Gradient Operator for Image Processing. — Never published but presented at a talk at the Stanford Artificial Project.

[17] *Viola P., Jones M.* Rapid object detection using a boosted cascade of simple features. — 2001. — Pp. 511–518.

[18] *Xu M., Orwell J., Jones G.* Tracking football players with multiple cameras // In: IEEE International Conference on Image Processing, IEEE Computer Society Press, Los Alamitos. — 2004. — Pp. 2909–2912.

[19] *Yang C., Duraiswami R., Davis L.* Fast multiple object tracking via a hierarchical particle filter // Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01. — ICCV '05. — 2005. — Pp. 212–219.