

Обзор алгоритмов трекинга объектов



Дмитрий АКИМОВ

Video Group
CS MSU Graphics & Media Lab



Содержание

- **Введение**
- Predator
- Particle Filter
- Graph Evolution-Based Tracking
- Заключение

Введение

Постановка задачи трекинга (1)



Примеры трекинга объектов

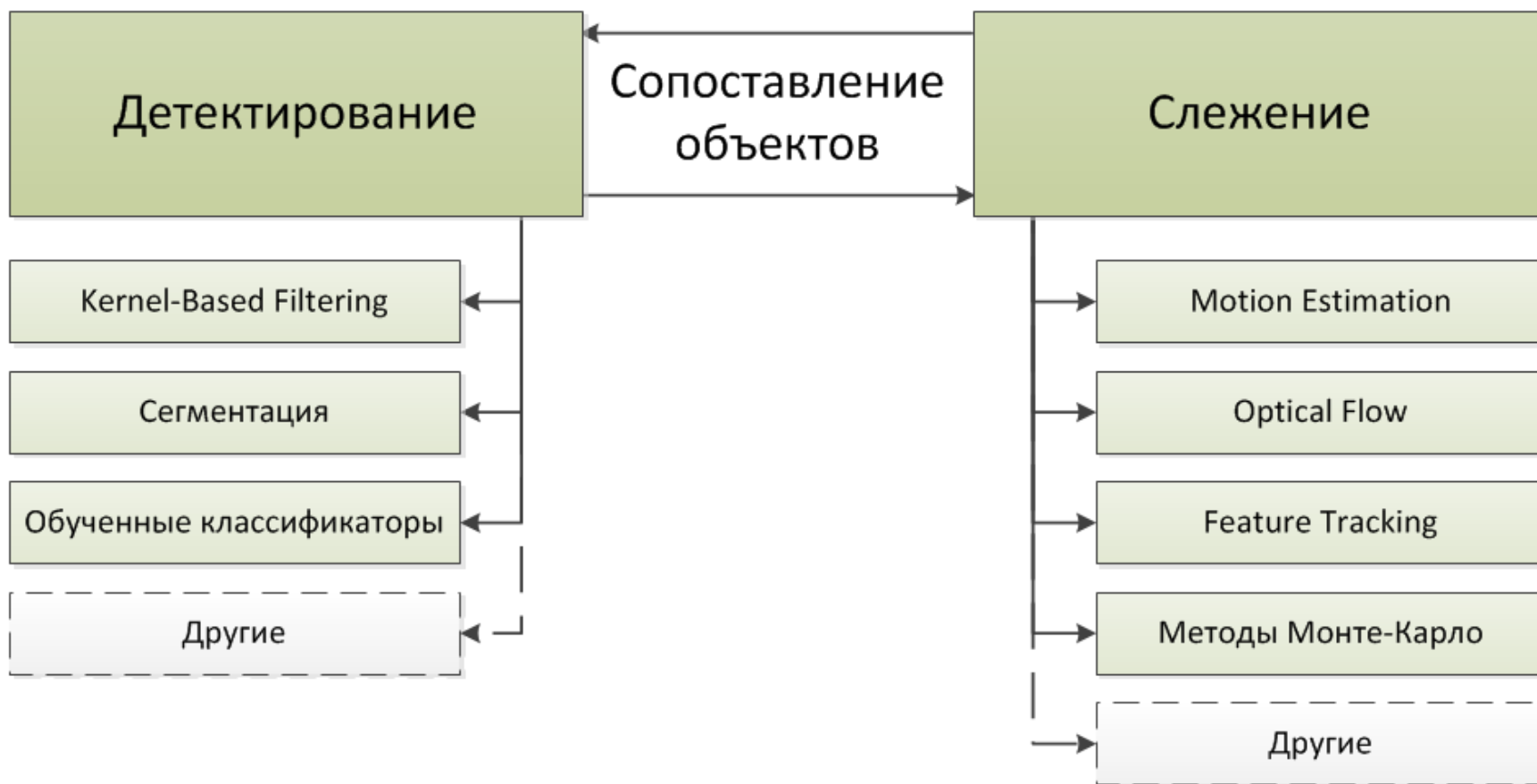
Введение

Постановка задачи трекинга (2)

- Дана последовательность кадров $\{ I_t(x, y) \}$
- Задан объект слежения (может быть несколько)
- Требуется:
 - обнаружить объект на каждом кадре
 - определить траекторию объекта

Трекинг

Частичная классификация



Слежение

Feature Tracking

■ Точечная особенность m : окрестность $\Omega(m)$ однозначно отличима от любой другой окрестности точки $n \in \Omega'(m)$, $\Omega' > \Omega$

Для областей объекта $W(t)$, $W(t+\Delta t)$:

1. Находятся точечные особенности (детектор Харриса)
2. Сопоставляются соседние по времени особенности (Lucas-Kanade метод)

Feature Tracking

Детектор Харриса

- Метод основан на поиске областей, похожих на угол
- Функция отклика угла для изображения $I(x, y)$:

$$R = \det M - k * (\text{trace} M)^2$$

$$M = \begin{bmatrix} \left(\frac{\partial I}{\partial x}\right)^2 & \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}$$

- По предложению Харриса $k = 0,4$

Feature Tracking

Метод Лукас-Канэде

Сопоставление точки x изображения I в окрестности $\Omega(x)$, требуется найти смещение Δx окрестности:

$$E = \sum_{\Omega(x)} (I(x + \Delta x, t) - I(x, t + \Delta t))^2$$

$$I(x + \Delta x) \approx I(x) + \Delta x \frac{\partial}{\partial x} I(x)$$

$$0 = \frac{\partial}{\partial \Delta x} E \approx \sum_{\Omega(x)} \frac{\partial}{\partial \Delta x} (I(x, t) + \Delta x \frac{\partial}{\partial x} I(x, t) - I(x, t + \Delta t))^2$$

$$\Delta x = \left[\sum_{\Omega(x)} \left(\frac{\partial I}{\partial \Delta x} \right)^T (I(x, t) - I(x, t + \Delta t)) \right] \left[\sum_{\Omega(x)} \left(\frac{\partial I}{\partial \Delta x} \right)^T \left(\frac{\partial I}{\partial \Delta x} \right) \right]^{-1}$$

Feature Tracking

Улучшения ЛК-метода

Улучшения стандартного ЛК-метода достигаются за счет усложнения функции E :

- Tomasi-Kanade – переформулировка задачи в линейных уравнениях (итеративное решение)
- Shi-Tomasi-Kanade – учет аффинных преобразований окрестности особой точки
- Jin-Favaro-Soatto – учет аффинных преобразований и изменения освещенности

Feature Tracking

Результаты (1)



KLT feature tracker

Feature Tracking

Результаты (2)



SIFT алгоритм (реализация для мобильных устройств)



Содержание

- Введение
- **Predator**
- Particle Filter
- Graph Evolution-Based Tracking
- Заключение

Predator: Camera That Learns

Zdenek Kalal, Jiri Matas, Krystian Mikolajczyk
University of Surrey, UK
Czech Technical University, Czech Republic

Первая ассоциация





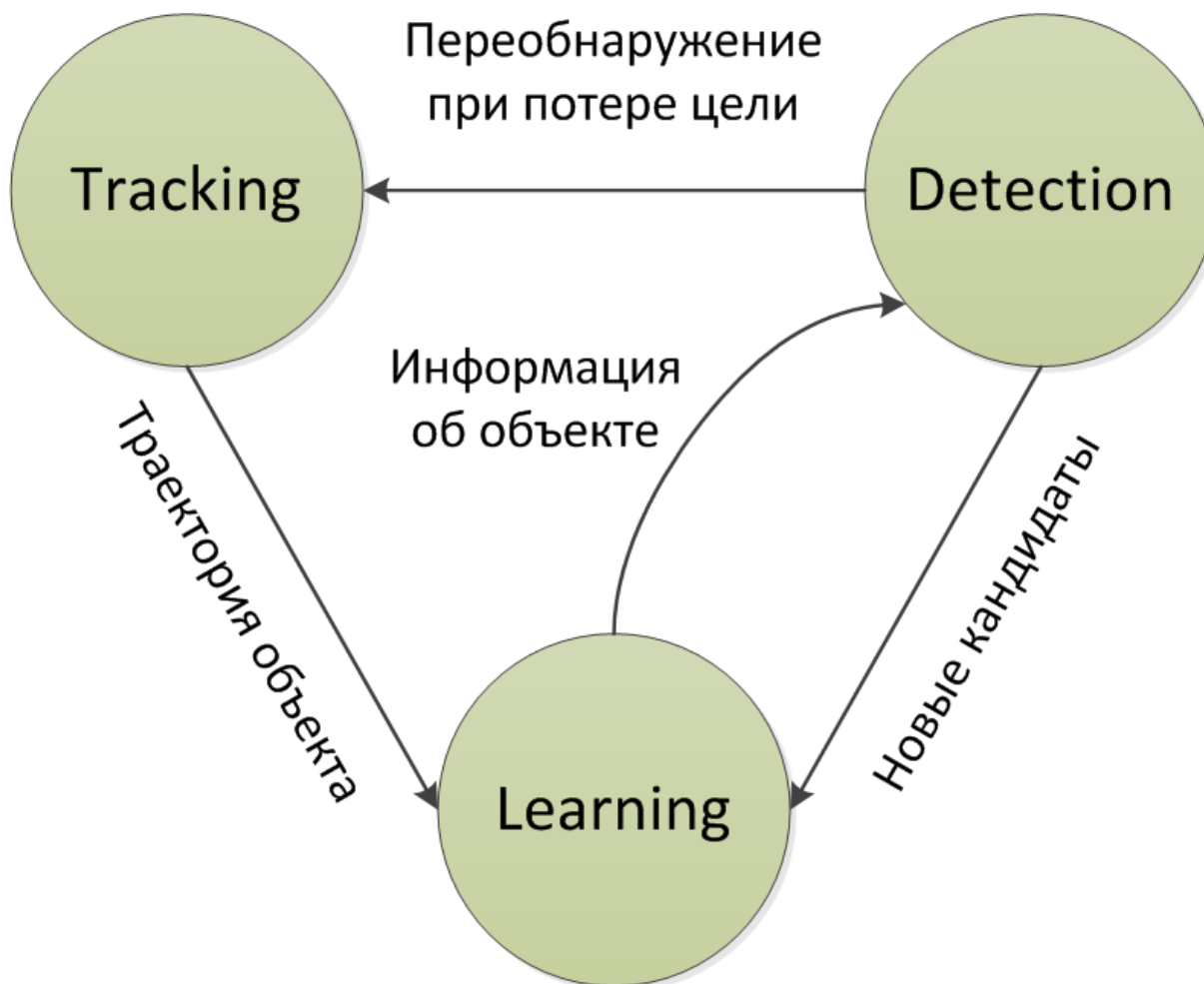
Predator

Решается задача трекинга одного объекта при следующих условиях:

- Работа в реальном времени (on-line, real-time)
- Слежение в течение продолжительного времени (long-term)
- Отсутствие заранее известной информации об объекте (no prior information; всё, что известно – bounding box объекта на первом кадре)

Predator

Предложенная схема



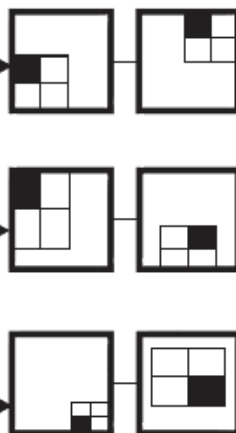
Predator

Этап детектирования

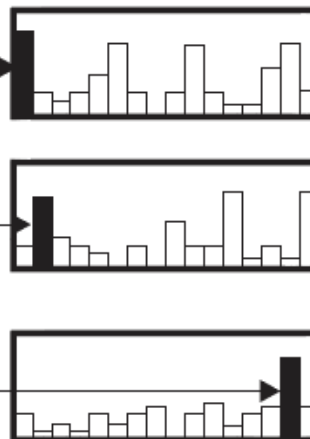
Scanning window



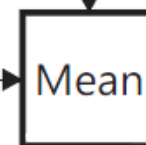
Features



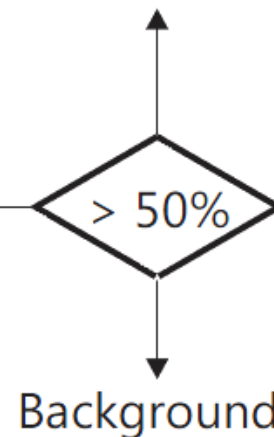
Posteriors



Mean



Object

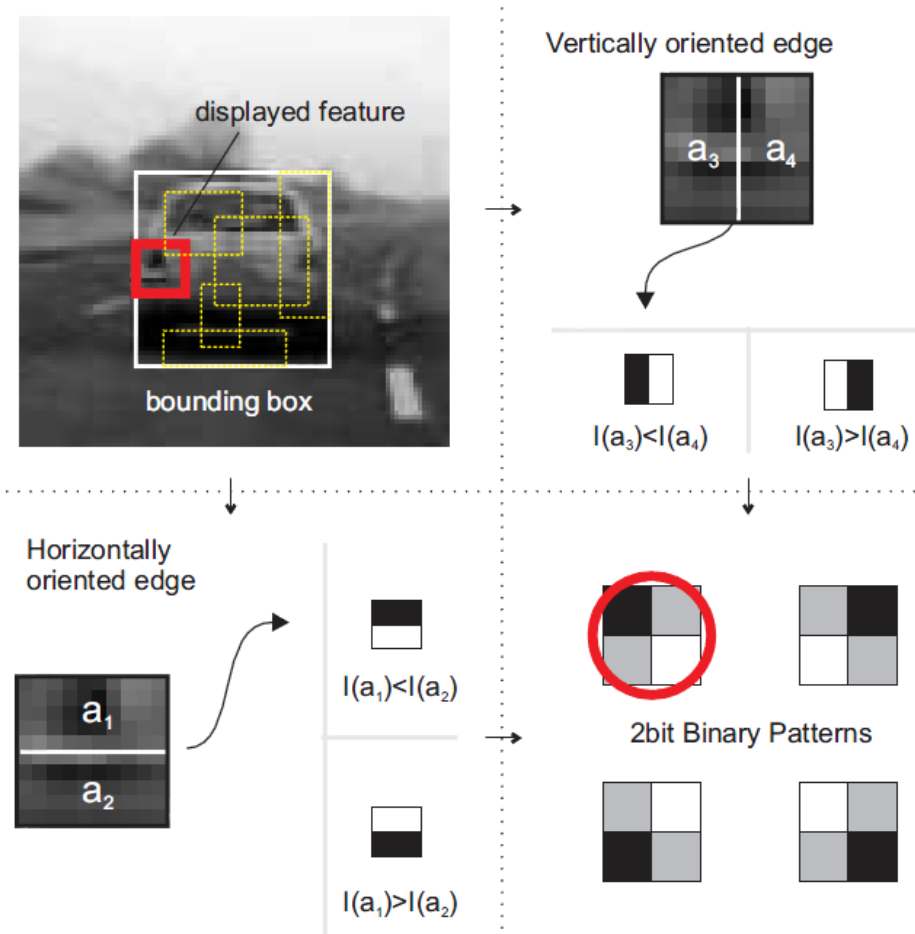


Концептуальная схема работы этапа детектирования

Этап детектирования

Вычисляемые особенности

- Выбираются случайные части окна
- Каждая часть кодируется как 2bitBP особенность
- Особенности случайно разбиваются на группы
- Каждая группа передается на вход классификатору



Вычисление 2bitBP особенности

Этап детектирования

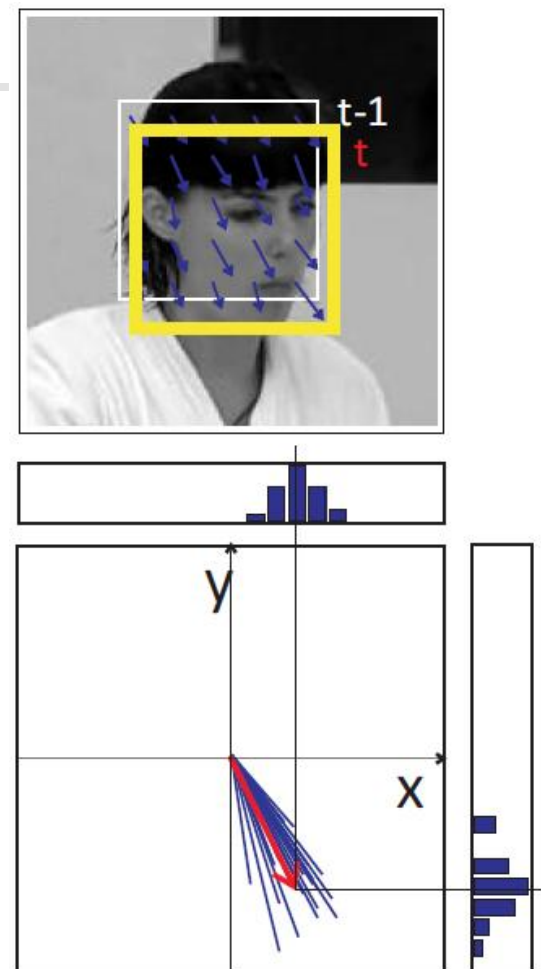
Классификатор

- Используется ferned random forest (быстрый, способен к модификации в процессе работы)
- Первоначальное обучение происходит на первом кадре
- На каждом кадре классификатор модифицируется после получения положительных и отрицательных примеров объекта

Predator

Этап трекинга

- Вычисляется движение между исходным положением и каждым новым кандидатом (использован классический LK-метод)
- Новым объектом считается ближайший к исходному положению



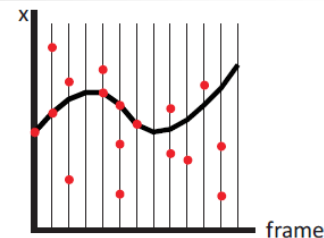
Распределение векторов движения feature points

Learning

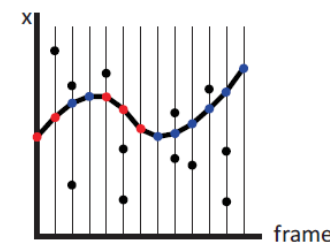
Positive-Negative Constraints

На каждом кадре:

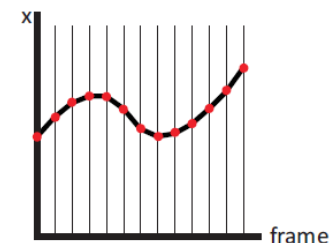
1. Находятся кандидаты
2. Определяется траектория
3. Все кандидаты в окрестности траектории считаются положительными примерами
4. Все остальные кандидаты считаются отрицательными
5. На основе P-N constraints модифицируется классификатор



Данные трекера и детектора



P-N constraints { • positive example
• negative examples



Модифицированная работа классификатора

P-N Constraints

Устойчивость метода

Precision/Recall-анализ метода:

$$\begin{bmatrix} \alpha(k+1) \\ \beta(k+1) \end{bmatrix} = M \cdot \begin{bmatrix} \alpha(k) \\ \beta(k) \end{bmatrix}$$

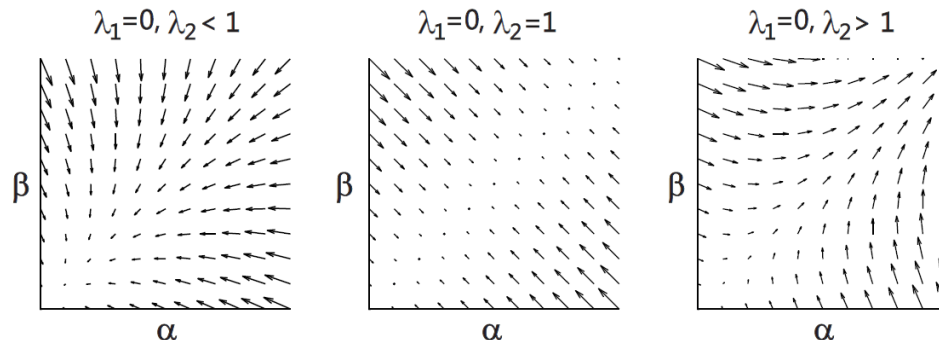
$$M = \begin{bmatrix} 1 - R^- & \frac{1 - P^+}{P^+} R^+ \\ \frac{1 - P^-}{P^-} R^- & 1 - R^+ \end{bmatrix}$$

$\alpha(k)$ – false negative

$\beta(k)$ – false positive

P^+, P^- – P-N constraints precision

R^+, R^- – P-N constraints recall

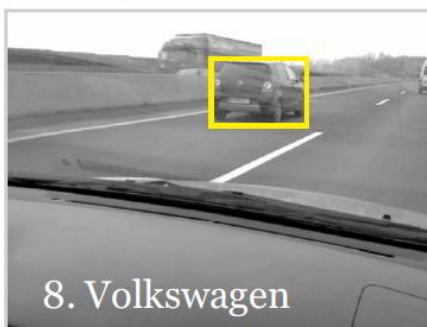


Динамика α, β срабатываний
в зависимости от собственных
значений матрицы M

Метод устойчив и минимизирует ошибки ($\lambda_2 < 1$)

Результаты

Тестовые видеопоследовательности



Результаты

Precision/Recall анализ

Sequence	Initial Detector	Final Detector	P-constraints	N-constraints	Eigenvalues
	Precision / Recall	Precision / Recall	P^+, R^+	P^-, R^-	λ_1, λ_2
1. David	1.00 / 0.01	1.00 / 0.32	1.00 / 0.08	0.99 / 0.17	0.92 / 0.83
2. Jumping	1.00 / 0.01	0.99 / 0.88	0.86 / 0.24	0.98 / 0.30	0.70 / 0.77
3. Pedestrian 1	1.00 / 0.06	1.00 / 0.12	0.81 / 0.04	1.00 / 0.04	0.96 / 0.96
4. Pedestrian 2	1.00 / 0.02	1.00 / 0.34	1.00 / 0.25	1.00 / 0.24	0.76 / 0.75
5. Pedestrian 3	1.00 / 0.73	0.97 / 0.93	0.98 / 0.78	0.98 / 0.68	0.32 / 0.22
6. Car	1.00 / 0.04	0.99 / 0.82	1.00 / 0.52	1.00 / 0.46	0.48 / 0.54
7. Motocross	1.00 / 0.00	0.92 / 0.32	0.96 / 0.19	0.84 / 0.08	0.92 / 0.81
8. Volkswagen	1.00 / 0.00	0.92 / 0.75	0.70 / 0.23	0.99 / 0.09	0.91 / 0.77
9. Car Chase	0.36 / 0.00	0.90 / 0.42	0.64 / 0.19	0.95 / 0.22	0.76 / 0.83

Результаты

Слабо текстурированный объект



Результаты

Схожие объекты



Predator

Выводы

- Достоинства:
 - Работа в реальном времени
 - Неограниченная длительность слежения
 - Отсутствие стадии off-line обучения
(не требуется априорная информация об объекте)
 - Стабильность к перекрытиям
- Недостатки:
 - Слежение только за одним объектом
 - Требуется ручная инициализация цели



Содержание

- Введение
- Predator
- **Particle Filter**
- Graph Evolution-Based Tracking
- Заключение

Particle Filter

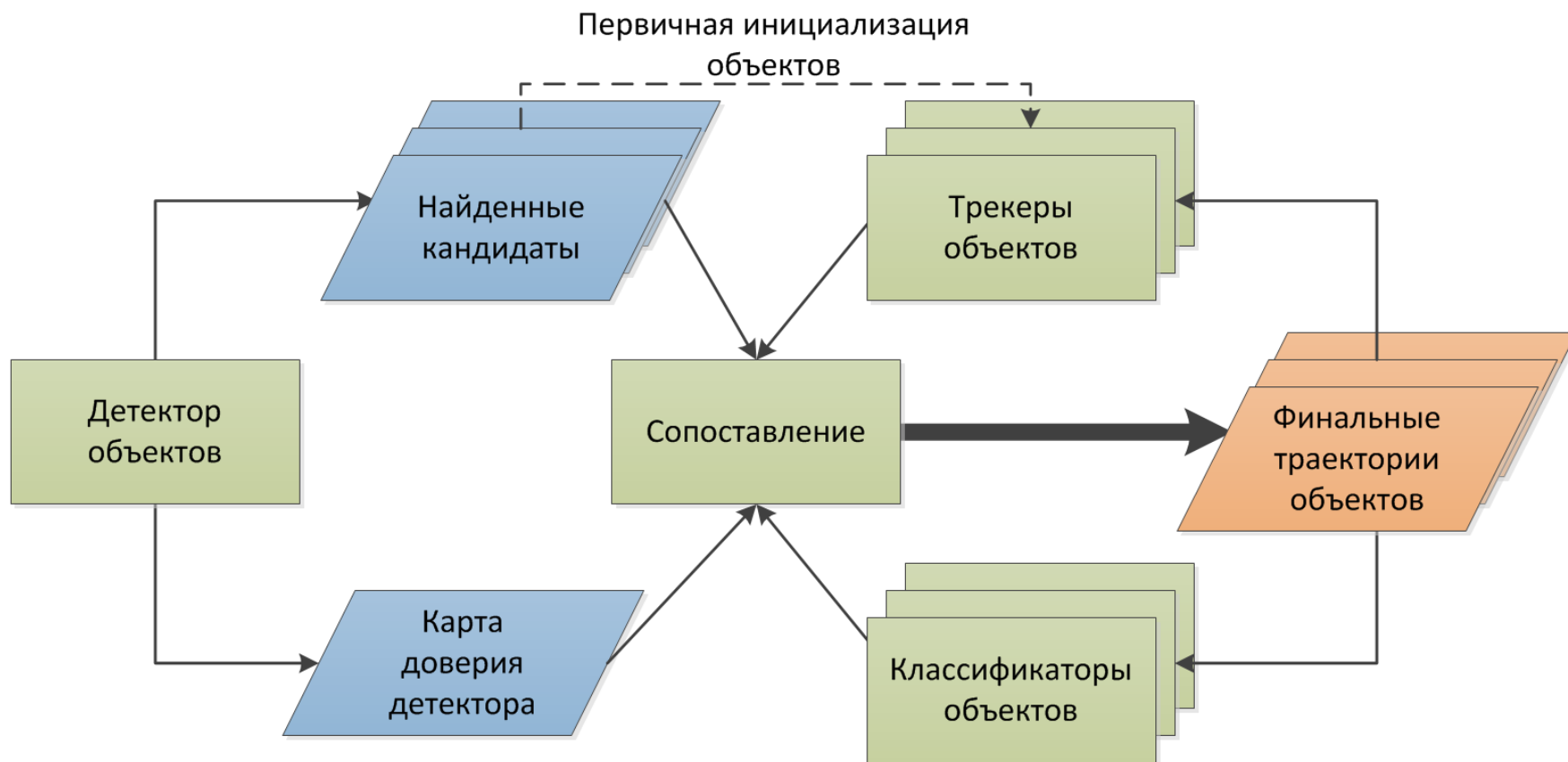
- В основе метода – классический подход Particle Filter Tracking
- Усложняется критерий взвешивания «испускаемых» частиц
- Область применения – on-line трекинг большого числа объектов



Визуализация процесса «испускания» частиц

Particle Filter

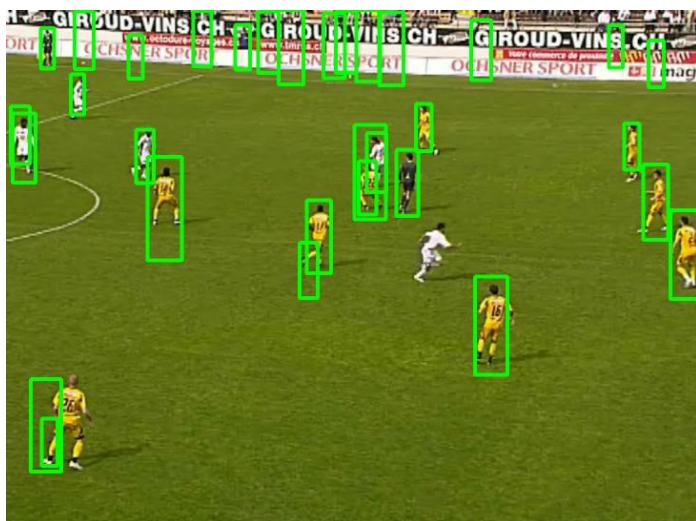
Схема алгоритма



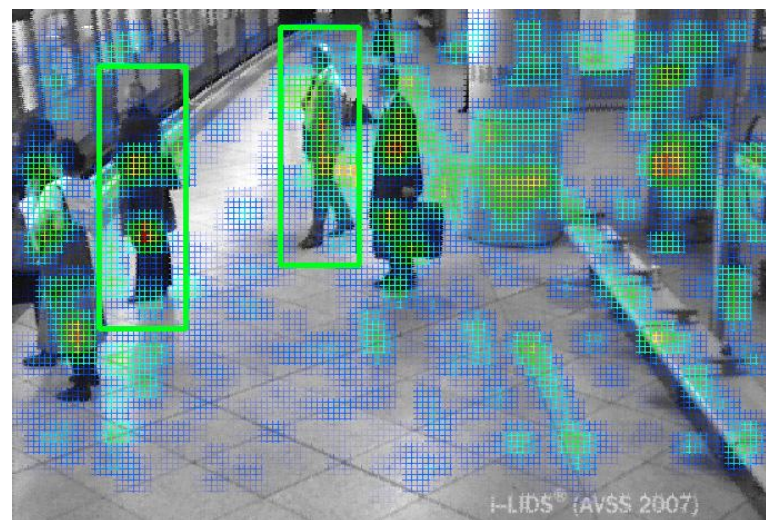
Particle Filter

Детектор объектов

Используется стандартный HOG (Histogram of Oriented Gradients) детектор людей



Обнаруженные кандидаты



Карта доверия
(вероятность обнаружения
объекта в конкретной точке)

Детектор объектов

Сопоставление объектов

Обнаруженные объекты d и объекты слежения tr требуется сопоставить

1. Для каждой пары (tr, d) вычисляется функция соответствия $s(tr, d)$
2. Жадным алгоритмом для каждого tr выбирается единственное d , пока $s(tr, d)$ не стало меньше допустимого значения
3. Каждое d соответствует единственному tr

Сопоставление объектов

Функция сопоставления

$$s(tr, d) = g(tr, d) \cdot (c_{tr}(d) + \alpha \cdot \sum_{p \in tr}^N p_{\mathcal{N}}(d - p))$$

$$g(tr, d) = p(size_d | tr) p(pos_d | tr)$$

$$= \begin{cases} p_{\mathcal{N}}\left(\frac{size_{tr} - size_d}{size_{tr}}\right) \cdot p_{\mathcal{N}}(|d - tr|) & \text{if } |\mathbf{v}_{tr}| < \tau_v \\ p_{\mathcal{N}}\left(\frac{size_{tr} - size_d}{size_{tr}}\right) \cdot p_{\mathcal{N}}(dist(d, \mathbf{v}_{tr})) & \text{otherwise.} \end{cases}$$

c_{tr} – классификатор объекта tr

v_{tr} – скорость объекта tr

p – «испускаемая» частица объекта tr

$p_{\mathcal{N}}$ – функция нормального распределения

α, τ_v – константы

$dist(d, v_{tr})$ – расстояние от d до линии, задаваемой скоростью v_{tr} и проходящей через положение объекта tr

Трекеры объектов

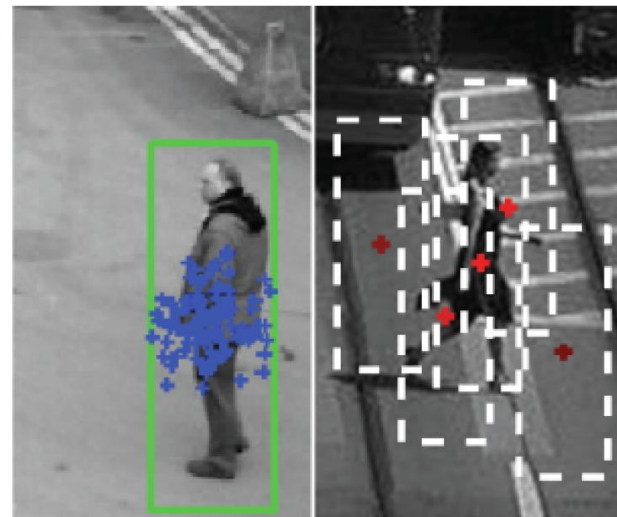
«Испускаемые» частицы (1)

- Объект задан положением (x, y) , скоростью (u, v) и областью bounding box
- «Испускаемая» частица – новое предполагаемое положение объекта:

$$(x, y)_t = (x, y)_{t-1} + (u, v)_{t-1} \cdot \Delta t + \varepsilon_{(x,y)}$$

$$(u, v)_t = (u, v)_{t-1} + \varepsilon_{(u,v)}$$

$\varepsilon_{(x,y)}$, $\varepsilon_{(u,v)}$ – функции случайного распределения



Пример «испускаемых» частиц

Трекеры объектов

«Испускаемые» частицы (2)

Новое положение объекта tr определяет частица p с наибольшим весом $w_{tr,p}$

$$w_{tr,p} = \underbrace{\beta \cdot \mathcal{I}(tr) \cdot p_{\mathcal{N}}(p - d^*)}_{\text{Вклад детектора}} + \underbrace{\gamma \cdot d_c(p) \cdot p_o(tr)}_{\text{Вклад карты доверия}} + \underbrace{\eta \cdot c_{tr}(p)}_{\text{Вклад классификатора}}$$

$$p_o(tr) = \begin{cases} 1 & \text{if } \mathcal{I}(tr) = 1 \\ \max_{tr': \mathcal{I}(tr')=1} p_{\mathcal{N}}(tr - tr') & \text{elif } \exists \mathcal{I}(tr') = 1 \\ 0 & \text{else} \end{cases}$$

d^* – ближайший к частице p обнаруженный объект

$\mathcal{I}(tr)$ – индикатор сопоставления объектов d^* и tr

$d_c(p)$ – карта доверия детектора

β, γ, η – константы

Particle filter

Результаты (1)



Particle filter

Результаты (2)



Particle filter

Результаты (3)



Particle Filter

Выводы

- Достоинства:
 - Работа в реальном времени
 - Отслеживание большого числа объектов
- Недостатки:
 - Ложные срабатывания детектора
 - Нестабильность траектории объектов (за счет случайного характера «испускания» частиц)



Содержание

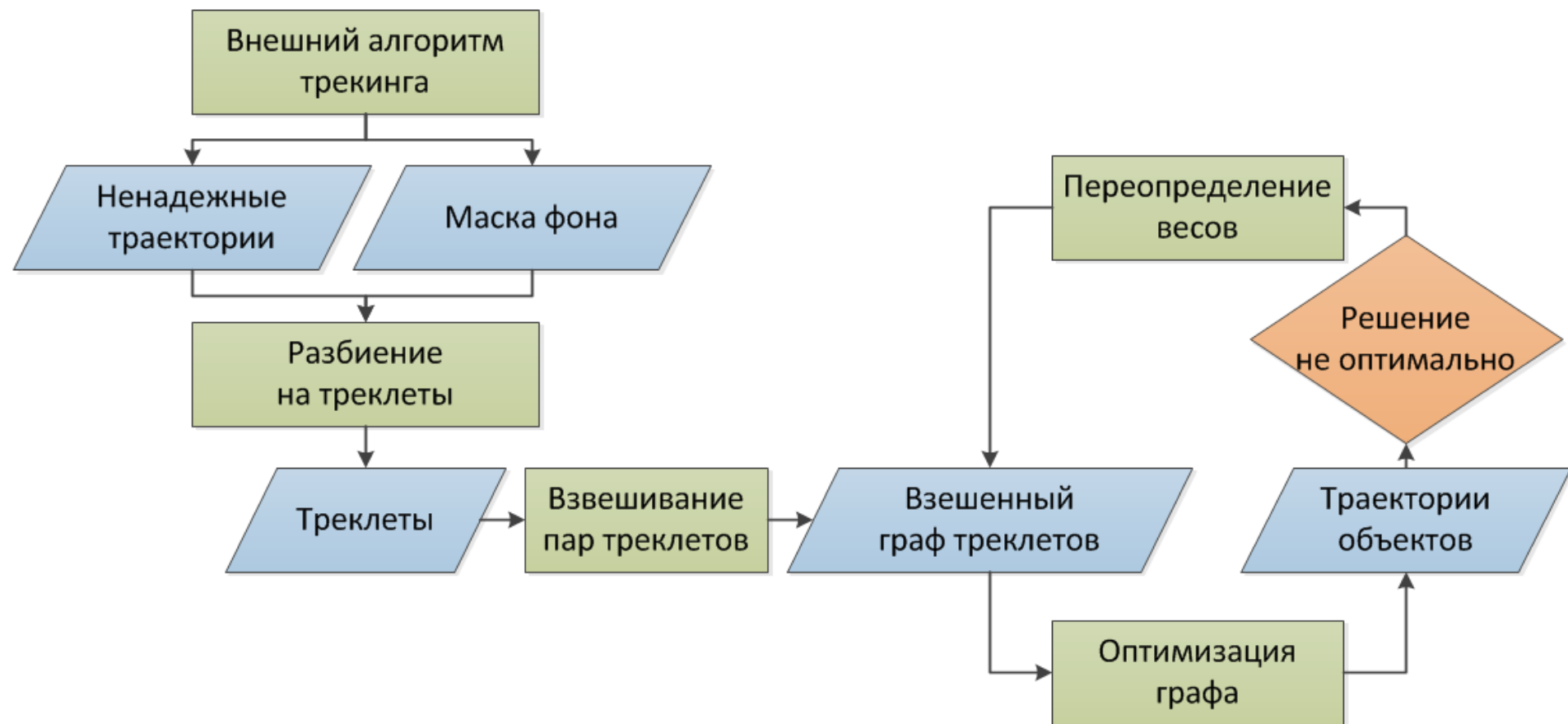
- Введение
- Predator
- Particle Filtering
- **Graph Evolution-Based Tracking**
- Заключение

GE-Based Tracking

- Рассматривается задача глобальной оптимизации траекторий нескольких объектов в сцене
- Каждая траектория строится из коротких треков (tracklets, треклеты), найденных другим методом трекинга
- Требуется найти оптимальное объединение треклетов в траектории объектов

GE-Based Tracking

Схема алгоритма



GE-Based Tracking

Разбиение на треклеты

- Полученные от внешнего трекера траектории разбиваются на короткие треклеты
- Траектория разрывается:
 - В точках низкого доверия к обнаруженному объекту
 - В точках, где область объекта содержит большое количество фона
 - В областях сближения с другими траекториями

Взвешивание треклетов

Конечная точка треклета

Вводится функция вероятности $q_{tl}(w|x)$ продолжения треклета из последней точки треклета x в точку w :

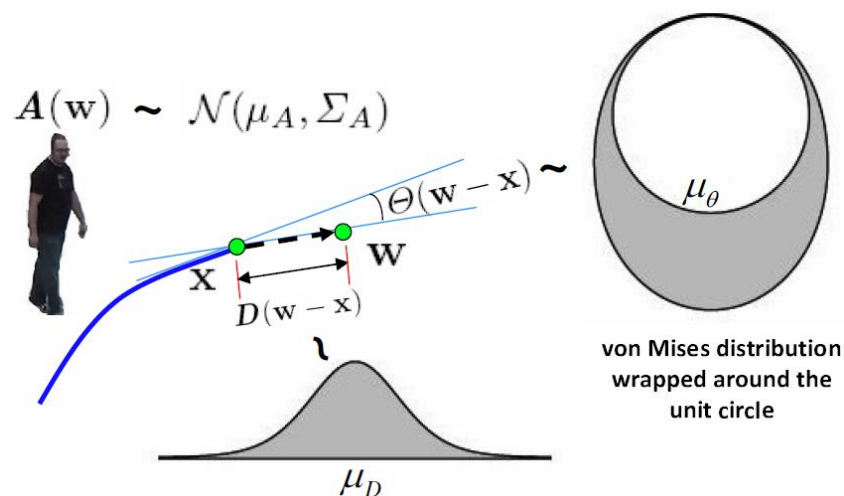


Иллюстрация функций вероятности

$$q_{tl}(w|x) \propto v(\Theta(w-x)|\mu_\theta, \kappa) \mathcal{N}(D(w-x)|\mu_D, \sigma_D) \mathcal{N}(A(w)|\mu_A, \Sigma_A)$$

$v(\mu_\theta, \kappa)$ – функция распределения von Mises с параметрами μ_θ, κ
 \mathcal{N} – функция нормального распределения с заданными средним и дисперсией

Взвешивание треклетов

Начальная точка треклета

Для треклета вводится функция вероятности $p_{tl}(z)$, определяющая, является ли z новым началом треклета:

$$p_{tl}(z) = e^{-d_{bh}(z)}$$

d_{bh} – расстояние Бхаттачарии до усредненной модели объекта

$$d_{bh}(p, q) = -\ln \left[\sum_{x \in X} \sqrt{p(x)q(x)} \right]$$

Взвешивание треклетов

Построение графа треклетов

- Финальная весовая функция для конечной точки \mathbf{x} одного треклета и начальной точки \mathbf{w} другого треклета:

$$\rho_{tl}(\mathbf{x}, \mathbf{w}) = \min \left\{ \frac{p_{tl}(\mathbf{w})q_{tl}(\mathbf{x}|\mathbf{w})}{p_{tl}(\mathbf{x})q_{tl}(\mathbf{w}|\mathbf{x})}, 1 \right\}$$

- Вычисляется для каждой пары треклетов
- Определяет вес ребер в графе сопоставления треклетов

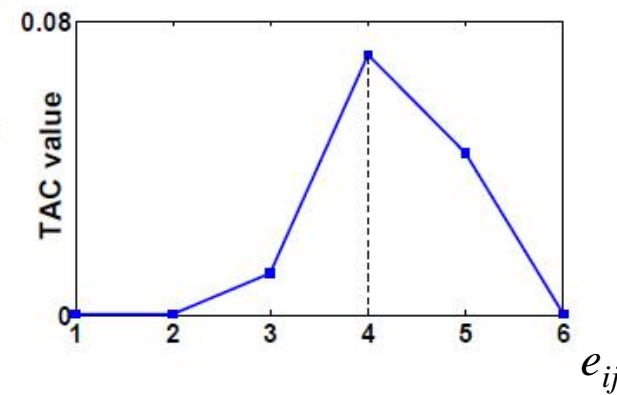
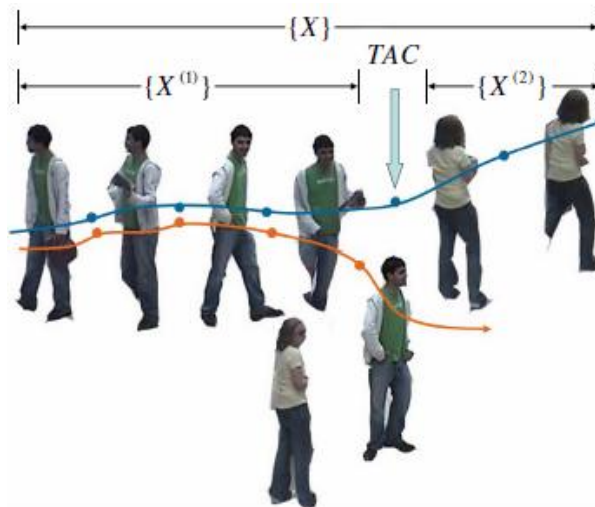
Оптимизация графа

- Применяется Венгерский алгоритм оптимизации
- Инициализация траекторий – набор первых (по времени) треклетов
- Результат работы – множество $\{\lambda_g\}$ траекторий (последовательностей найденных объектов e_{ij})

Найденные траектории

Оценка оптимальности (1)

Для каждой точки e_{ij} траектории λ_q вычисляется



$$TAC(e_{ij}) = \frac{|S_T - S_W|}{|S_W|}$$

- S_T – дисперсия X (множества всех точек траектории)
- S_W – сумма дисперсий $X^{(1)}$ и $X^{(2)}$ (подразбиений траектории относительно e_{ij})

Найденные траектории

Оценка оптимальности (2)

- Для всех траекторий λ_q вычисляется:

$$L(\lambda_q) = \sum_{\lambda_q} (TAC(e_{ij} \in \lambda_q^{(n)}))$$

- Если $L(\lambda_q)$ больше заданного числа, то:
 - У точек с высоким TAC случайным образом изменяется вес в графе треклетов
 - Оптимизационная задача пересчитывается

GE-Based Tracking

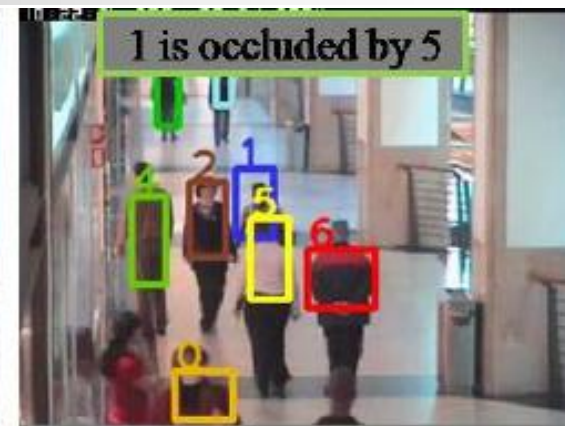
Результаты



Frame 230



Frame 257



Frame 313



Frame 1761



Frame 2480



Frame 3680

GE-Based Tracking

Выводы

- Достоинства:
 - Необычный подход
 - Высокое качество трекинга
- Недостатки:
 - Обработка всей последовательности целиком (не real-time метод)
 - Неизвестное время работы (вероятно, большое)



Содержание

- Введение
- Predator
- Particle Filtering
- Graph Evolution-Based Tracking
- **Заключение**

Заключение

Текущая задача (1)

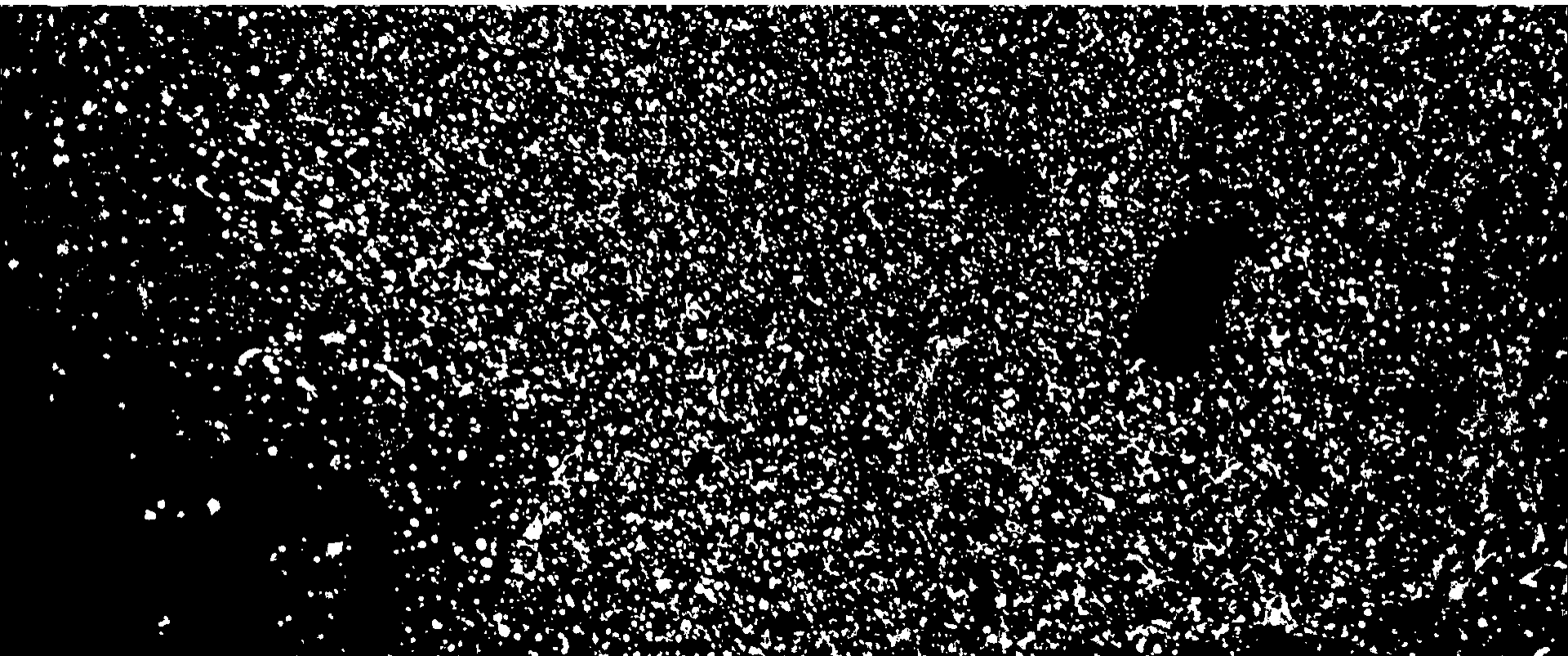


Дана видеопоследовательность с большим количеством движущихся частиц



Заключение

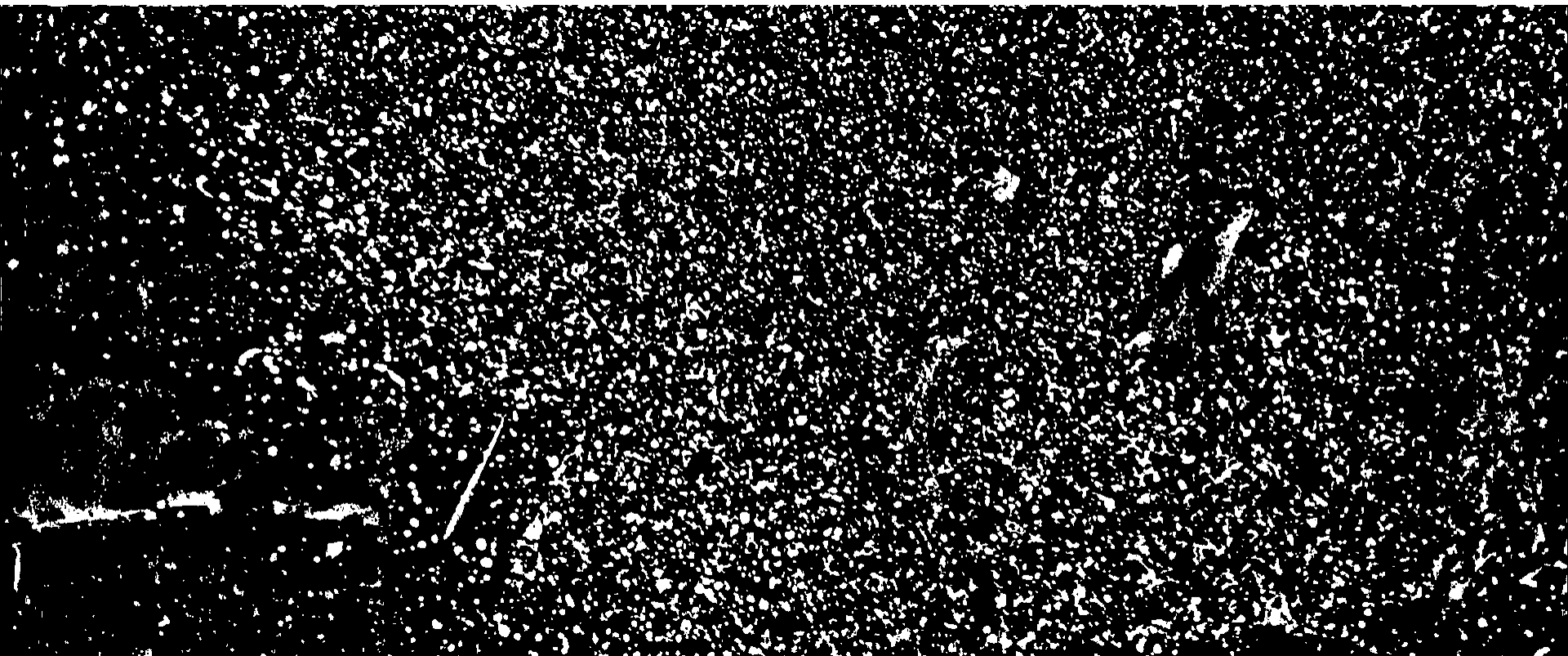
Текущая задача (2)



Требуется сформировать стабильную во времени
маску объектов переднего плана

Решенные этапы

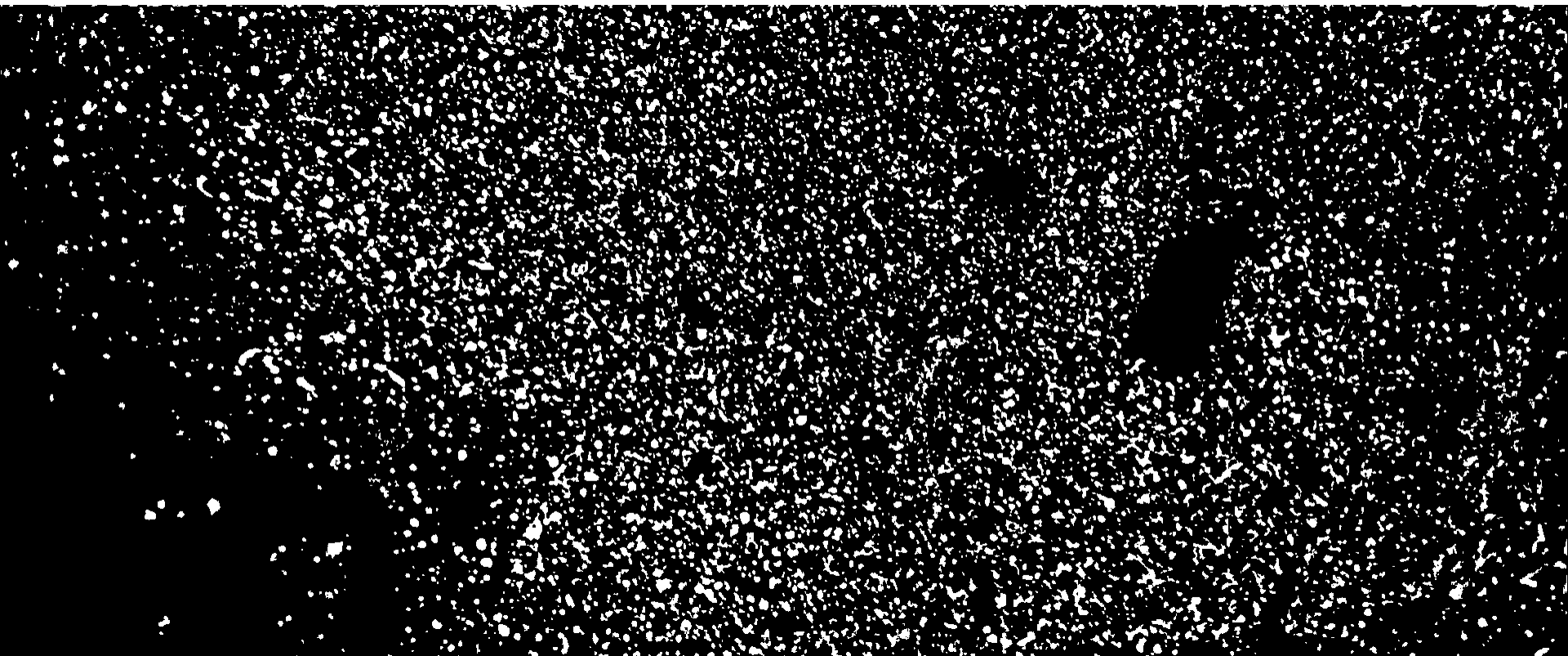
Стабильная маска кандидатов



Присутствует большое количество ложных срабатываний

Решенные этапы

Пороговая фильтрация по признакам



Объекты, чьи признаки находятся на границе порога,
становятся нестабильны во времени

Текущая задача

Предполагаемый алгоритм

1. Инициализация объектов на первом кадре
2. Вычисление движения в сцене
 - Motion Estimation
 - Feature Tracking
 - Particle Filtering
3. Сопоставление объектов по признакам
4. Ограничения на появление новых объектов

Литература

1. C. Yang, R. Duraiswami, A. Elgammal, and L. Davis, "On-Line Kernel-Based Tracking in Joint Feature-Spatial Spaces," *DEMO on IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.
2. Антон Конушин. Слежение за точечными особенностями сцены (Point feature tracking). *Компьютерная графика и мультимедиа*. Выпуск №1(5)/2003. Available <http://cgm.computergraphics.ru/content/view/54>
3. Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints," *Conference on Computer Vision and Pattern Recognition*, 2010.
4. Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-Backward Error: Automatic Detection of Tracking Failures," *International Conference on Pattern Recognition*, 2010, pp. 23-26.

Литература

5. Z. Kalal, J. Matas, and K. Mikolajczyk, "Online learning of robust object detectors during unstable tracking," *On-line Learning for Computer Vision Workshop*, 2009.
6. M. D. Breitenstein, F. Reichlin, B. Leibe, E. K. Meier, and L. V. Gool, "Robust Tracking-by-Detection using a Detector Confidence Particle Filter," *IEEE International Conference on Computer Vision (ICCV'09)*, Kyoto, JP, 2009.
7. B. Song, T. Jeng, E. Staudt, and A.K.R. Chowdhury, "A Stochastic Graph Evolution Framework for Robust Multi-target Tracking," in *Proceedings European Conference on Computer Vision*, 2010, pp. 605-619.

Лаборатория компьютерной графики и мультимедиа



Видеогруппа — это:

- Выпускники в аспирантурах Англии, Франции, Швейцарии (в России в МГУ и ИПМ им. Келдыша)
- Выпускниками защищены 5 диссертаций
- Наиболее популярные в мире сравнения видеокодеков
- Более 3 миллионов скачанных фильтров обработки видео