

Школа Анализа  
Данных Яндекса

Курс «Анализ изображений и видео, ч.2»

Лекция №5  
«Оптический поток и вычитание фона»

Антон Конушин

Заведующий лабораторией компьютерной графики и мультимедиа  
ВМК МГУ

17 марта 2017 года

# Видеокамеры

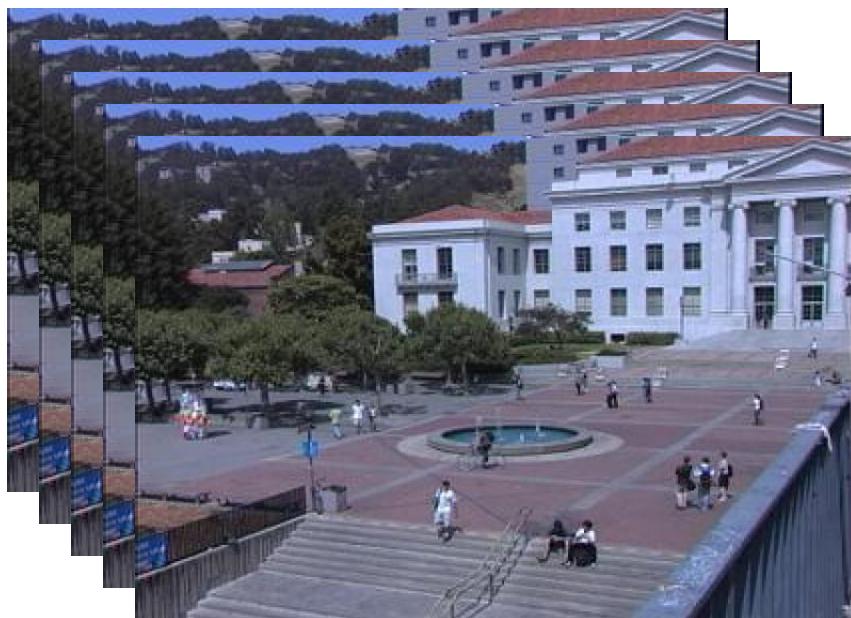
---



- Наиболее близкий аналог «человеческого глаза» из доступных компьютеру
- Широкое распространение:
  - Видеокамеры в смартфонах, планшетах, ноутбуках
  - 96 тыс. подъездов и 20 тыс. дворов оборудованы камерами в Москве (и все доступны через сайт!)



# Видеопоследовательность



Видеопоток – упорядоченная последовательность изображений, полученных с одной камеры через небольшие промежутки времени

Видеопоток подразумевает обработку на лету.

Видеопоследовательность же конечна, можно обрабатывать целиком.

- Пользовательское видео – от 3-5 кадров/сек до 30-50 кадров/сек
- Разрешение – от 320x240 до 1920\*1080 (HD) (сейчас ещё 4K)
- В градациях серого (одноканальное) или цветное (3х канальное)
- Поток данных – 2Мб (один канал HD) x 3 (RGB) x 50 кадр/с = 300 Мб/с
  - Больше, чем пропускная способность 1 гигабитной Ethernet сетки!

# Состояние и перспективы

---



- Сейчас большинство видеокамер работают в режиме «регистрации» с ограниченной автоматизацией
  - Видеорегистраторы в автомобилях
  - Видеокамеры для любительской съёмки
  - Видеозапись с камер наблюдения
- Количество информации огромное; управлять ей без анализа очень сложно; хранится, в основном, на локальных носителях
- «Видеоаналитика» - все алгоритмы для извлечения информации из видео
  - Есть ряд практических работающих систем

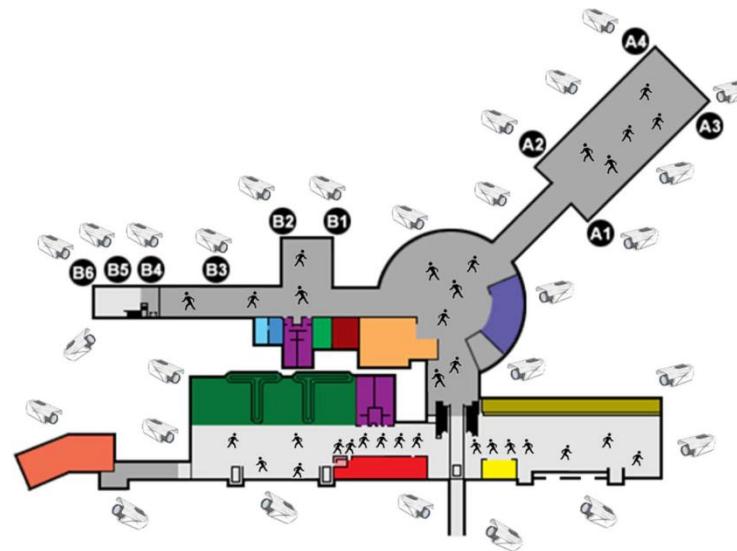
# Чего мы хотим достичь?



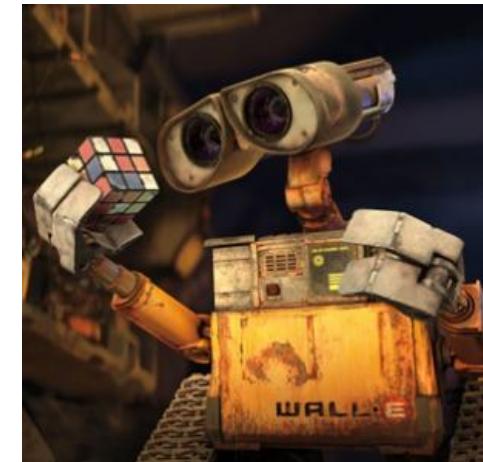
Извлечение данных из видеопотока:

- Разметить все объекты и людей
- Определить позу людей, распознать жесты
- Распознать происходящие события

# Чего мы хотим достичь?



- «Situation awareness»
- Свойство систем и класс приложений, которые позволяют извлечь из сенсорных данных знание, ведущие к действиям



# Сценарии съёмки



- Ракурс, вид наблюдаемых объектов и т.д.
- Ракурсы съёмки могут быть крайне различны
- Работающие системы удается создать, «заточившись» на определённый сценарий съёмки

# Видеопоток и изображения

---

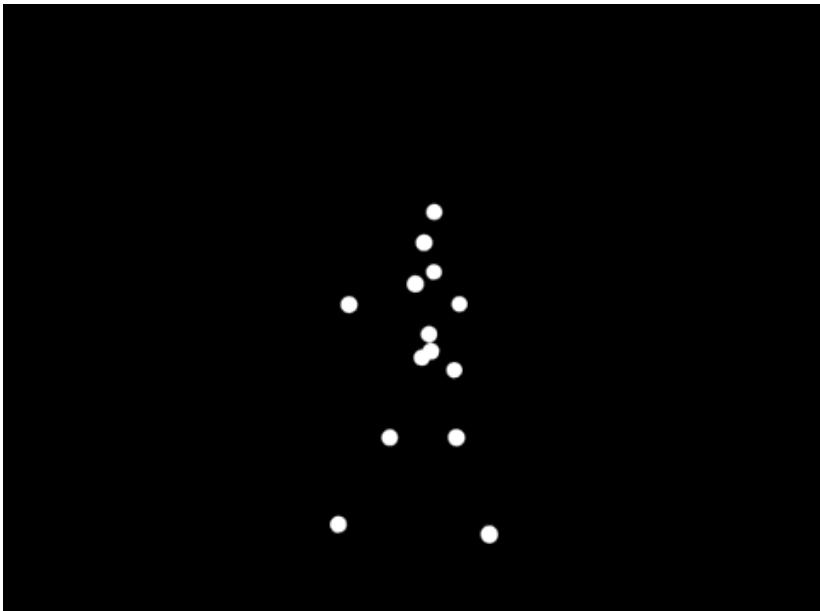


Чем принципиально отличается видеопоток от  
обычных изображений?



# Движение

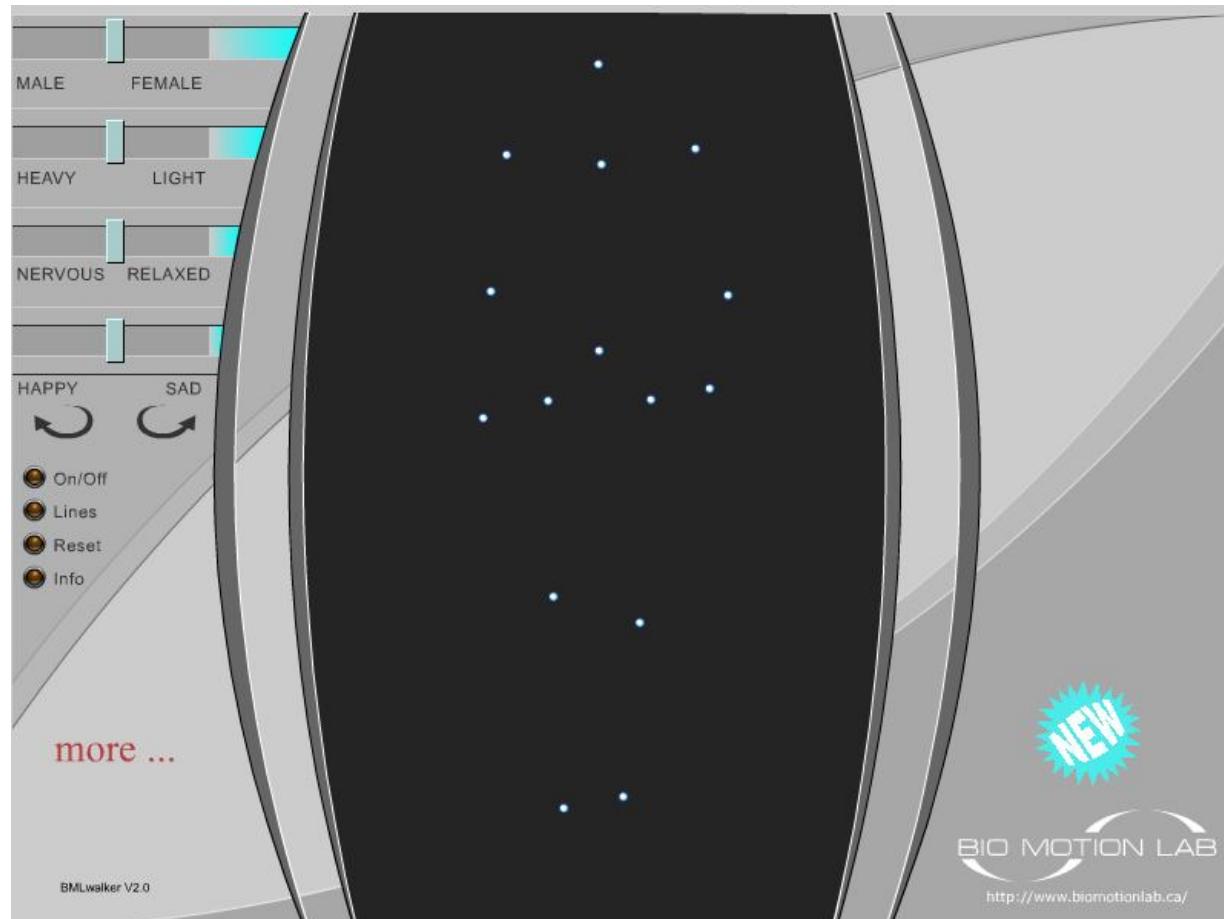
---



Что показано в видео?

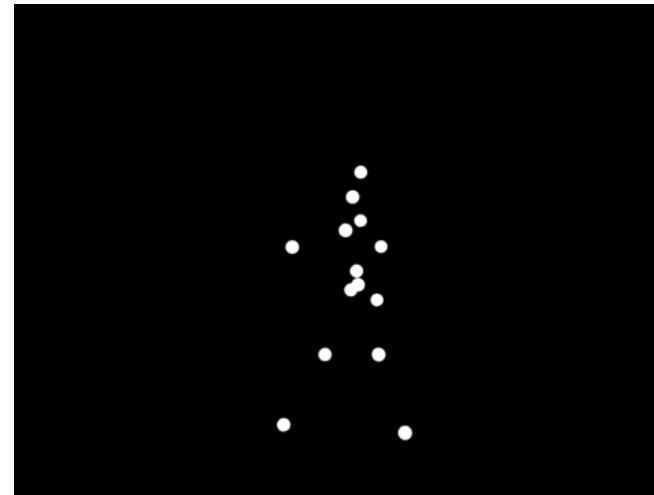
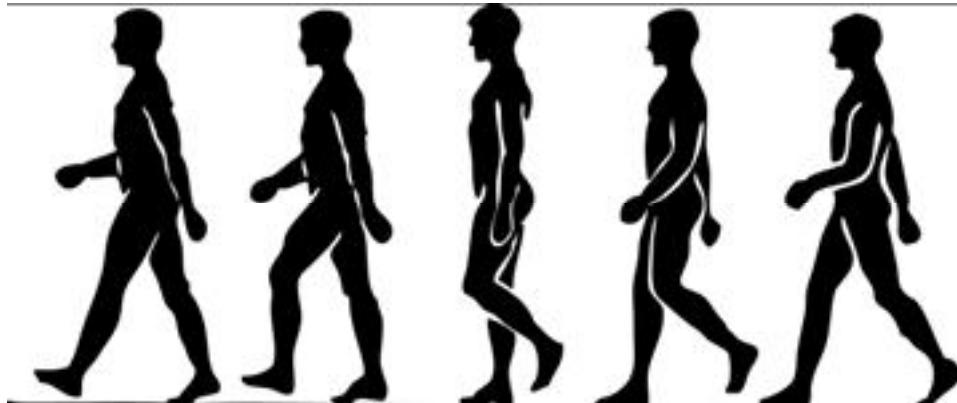
- Движение – главное отличие видео от изображений
- Движение само по себе является мощной визуальной подсказкой
- Суть многих действий именно в динамике
- Иногда достаточно отследить движение отдельных точек, чтобы распознать событие

# Распознавание по движению



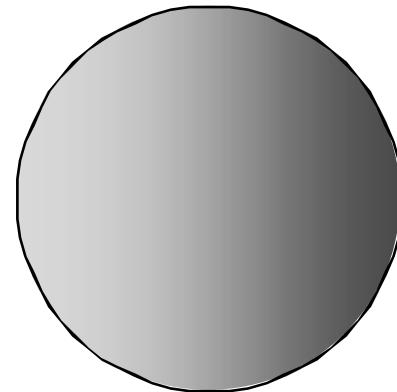
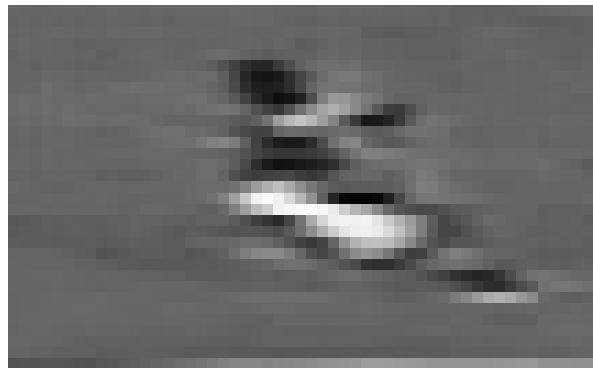
<http://www.biomotionlab.ca/Demos/BMLwalker.html>

# Описание движения



- Точки наблюдаемой сцены движутся относительно камеры / изображения
- Векторное поле движения 2D проекций на изображение 3D точек объектов сцены называется *полем движения* (*motion field*)
- Нужно это движение как-то формализовать, описывать и измерять

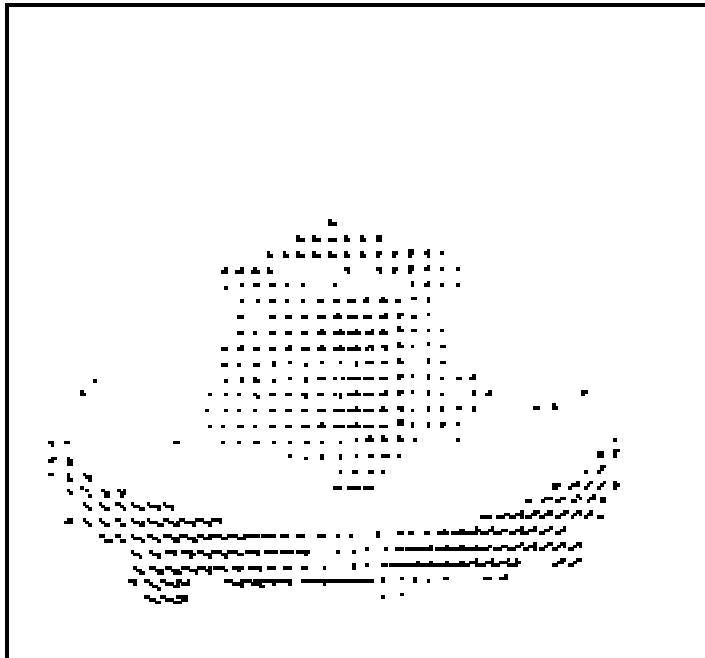
# Оптический поток



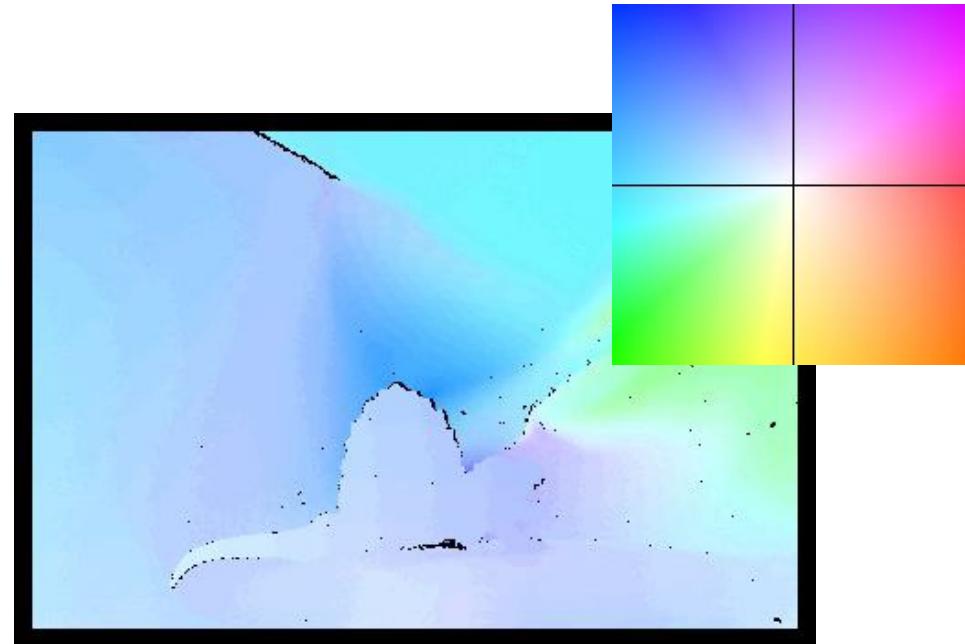
- Движение точек объектов по видео увидеть можно далеко не всегда
- Пример:
  - Серый матовый шар, освещается с одной стороны и вращается вокруг своей оси
  - Яркость всех пикселов изображения в этом случае будет постоянной
- **Оптический поток (optical flow)** – векторное поле *видимого (apparent)* движения пикселей между кадрами
  - Вычисление оптического потока – одна из базовых задач анализа видео
  - Задача похоже на задачу попиксельного сопоставления двух изображений одной и той же сцены



# Визуализация



Вектора движения для  
отдельных точек или всего  
изображения



Цветовое кодирование вектора  
движения. Каждому направлению и  
амплитуде свой цвет и яркость

Оптический поток - векторное поле  $(u_{ij}, v_{ij})$  видимого (наблюдаемого) движения пикселей между кадрами



# Рейтинг алгоритмов

Optical flow evaluation results												Statistics: Average SD R2.5 R5.0 R10.0 A50 A75 A95													
Show images: <input checked="" type="radio"/> below table <input type="radio"/> above table <input type="radio"/> in window												Error type: endpoint angle interpolation normalized interpolation													
Average angle error	avg. rank	Army (Hidden texture)			Mequon (Hidden texture)			Schefflera (Hidden texture)			Wooden (Hidden texture)			Grove (Synthetic)			Urban (Synthetic)			Yosemite (Synthetic)			Teddy (Stereo)		
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	GT	im0	im1
NNF-Local [87]	4.4	2.69 <sub>2</sub>	7.56 <sub>3</sub>	1.98 <sub>3</sub>	1.97 <sub>3</sub>	7.01 <sub>4</sub>	1.59 <sub>4</sub>	2.18 <sub>2</sub>	5.36 <sub>3</sub>	1.53 <sub>4</sub>	1.87 <sub>2</sub>	9.14 <sub>5</sub>	1.06 <sub>4</sub>	2.28 <sub>2</sub>	2.94 <sub>1</sub>	1.57 <sub>2</sub>	2.39 <sub>5</sub>	6.78 <sub>2</sub>	2.15 <sub>8</sub>	2.00 <sub>14</sub>	3.36 <sub>14</sub>	1.62 <sub>13</sub>	0.99 <sub>1</sub>	2.16 <sub>2</sub>	0.57 <sub>2</sub>
NN-field [71]	8.8	2.89 <sub>7</sub>	8.13 <sub>15</sub>	2.11 <sub>5</sub>	2.10 <sub>5</sub>	7.15 <sub>8</sub>	1.77 <sub>13</sub>	2.27 <sub>4</sub>	5.59 <sub>5</sub>	1.61 <sub>8</sub>	1.58 <sub>1</sub>	8.52 <sub>4</sub>	0.79 <sub>1</sub>	2.35 <sub>4</sub>	3.05 <sub>5</sub>	1.60 <sub>3</sub>	1.89 <sub>1</sub>	5.20 <sub>1</sub>	1.37 <sub>1</sub>	2.43 <sub>40</sub>	3.70 <sub>43</sub>	1.95 <sub>30</sub>	1.01 <sub>2</sub>	2.25 <sub>3</sub>	0.53 <sub>1</sub>
OFLAF [77]	11.1	3.04 <sub>14</sub>	7.80 <sub>9</sub>	2.40 <sub>12</sub>	2.14 <sub>6</sub>	7.02 <sub>5</sub>	1.72 <sub>8</sub>	2.25 <sub>3</sub>	5.32 <sub>2</sub>	1.56 <sub>5</sub>	2.62 <sub>15</sub>	13.7 <sub>21</sub>	1.37 <sub>18</sub>	2.35 <sub>4</sub>	3.13 <sub>6</sub>	1.62 <sub>4</sub>	2.98 <sub>17</sub>	7.73 <sub>7</sub>	2.57 <sub>16</sub>	2.08 <sub>19</sub>	3.27 <sub>9</sub>	2.05 <sub>33</sub>	1.33 <sub>12</sub>	2.43 <sub>6</sub>	1.40 <sub>15</sub>
PMMST [116]	12.5	3.42 <sub>38</sub>	7.60 <sub>4</sub>	2.65 <sub>27</sub>	2.32 <sub>10</sub>	6.39 <sub>1</sub>	2.20 <sub>29</sub>	2.63 <sub>11</sub>	6.08 <sub>8</sub>	2.03 <sub>22</sub>	2.06 <sub>4</sub>	6.07 <sub>1</sub>	1.44 <sub>23</sub>	2.60 <sub>10</sub>	3.27 <sub>8</sub>	1.91 <sub>10</sub>	2.56 <sub>6</sub>	6.78 <sub>2</sub>	2.09 <sub>4</sub>	2.06 <sub>16</sub>	3.53 <sub>32</sub>	1.63 <sub>14</sub>	1.27 <sub>9</sub>	2.29 <sub>4</sub>	1.02 <sub>6</sub>
nLayers [57]	14.6	2.80 <sub>5</sub>	7.42 <sub>2</sub>	2.20 <sub>7</sub>	2.71 <sub>28</sub>	7.24 <sub>9</sub>	2.55 <sub>54</sub>	2.61 <sub>9</sub>	6.24 <sub>9</sub>	2.45 <sub>47</sub>	2.30 <sub>9</sub>	12.7 <sub>11</sub>	1.16 <sub>7</sub>	2.30 <sub>3</sub>	3.02 <sub>3</sub>	1.70 <sub>5</sub>	2.62 <sub>9</sub>	6.95 <sub>4</sub>	2.09 <sub>4</sub>	2.29 <sub>34</sub>	3.46 <sub>22</sub>	1.89 <sub>27</sub>	1.38 <sub>14</sub>	3.06 <sub>17</sub>	1.29 <sub>13</sub>
MDP-Flow2 [68]	16.6	3.23 <sub>29</sub>	7.93 <sub>12</sub>	2.60 <sub>19</sub>	1.92 <sub>1</sub>	6.64 <sub>2</sub>	1.52 <sub>1</sub>	2.46 <sub>7</sub>	5.91 <sub>7</sub>	1.56 <sub>5</sub>	3.05 <sub>38</sub>	15.8 <sub>47</sub>	1.51 <sub>33</sub>	2.77 <sub>22</sub>	3.50 <sub>17</sub>	2.16 <sub>23</sub>	2.86 <sub>13</sub>	8.58 <sub>18</sub>	2.70 <sub>26</sub>	2.00 <sub>14</sub>	3.50 <sub>29</sub>	1.59 <sub>11</sub>	1.28 <sub>10</sub>	2.67 <sub>11</sub>	0.89 <sub>4</sub>

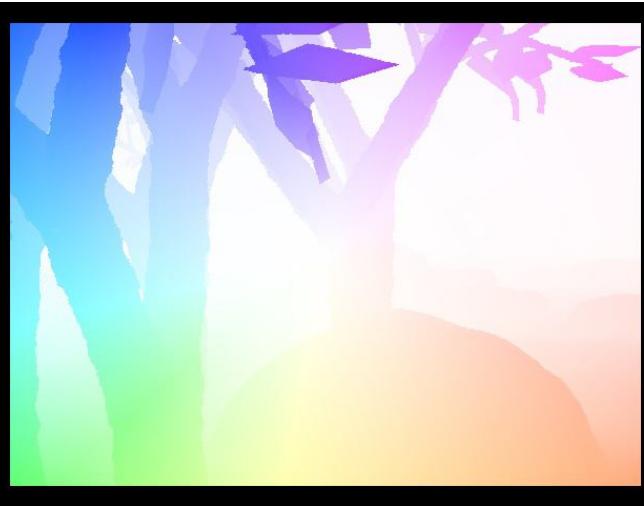
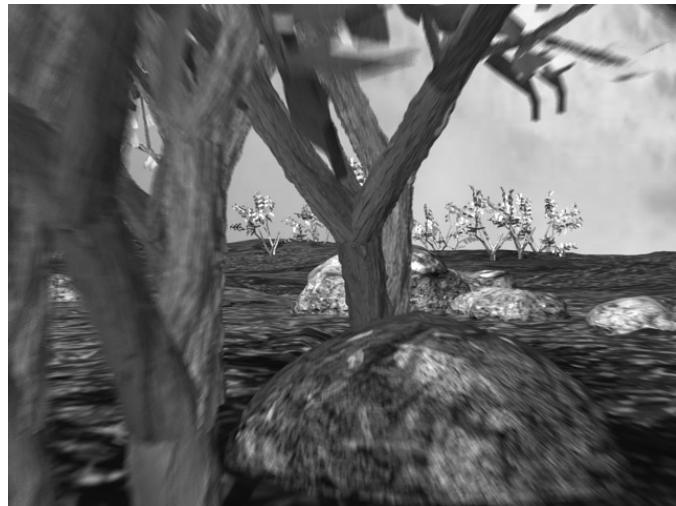
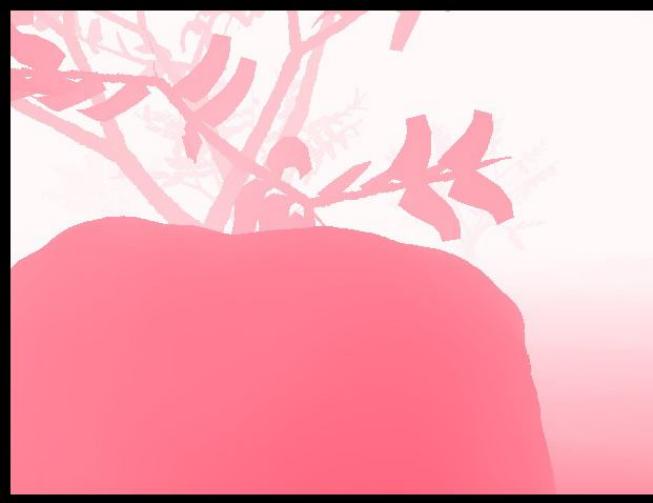
- Общий рейтинг алгоритмов вычисления оптического потока и набор сложных тестовых данных
- Как всегда, такой рейтинг и сложные данные стимулируют развитие алгоритмов
- <http://vision.middlebury.edu/flow/>

# Создание сложных данных



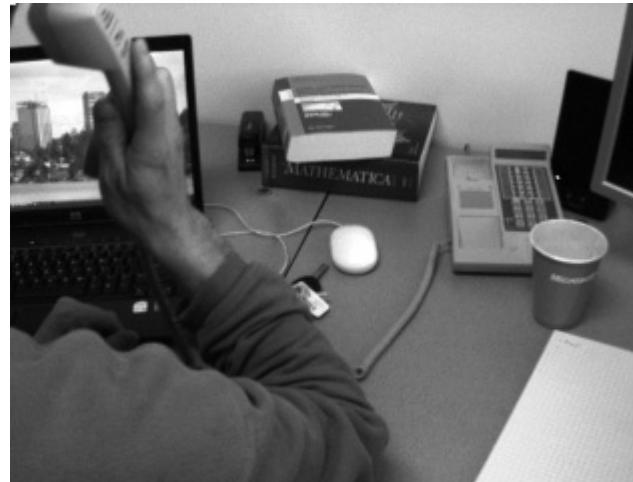
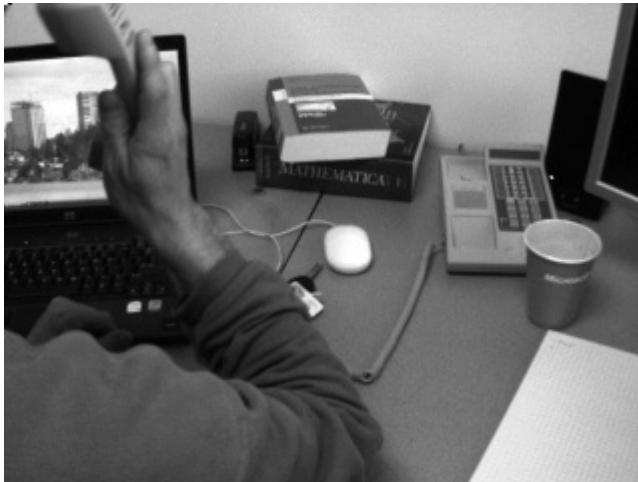
- Для численной оценки алгоритмов нужны эталонные данные.
- Вопрос, как их можно получить для реальных данных?
- Подход:
  - Съёмка в двух диапазонах
  - Флюоресцентная краска создает высокочастотную текстуру, позволяющую оценить поле движения
- Плюс сложные синтетические данные

# Синтетические данные

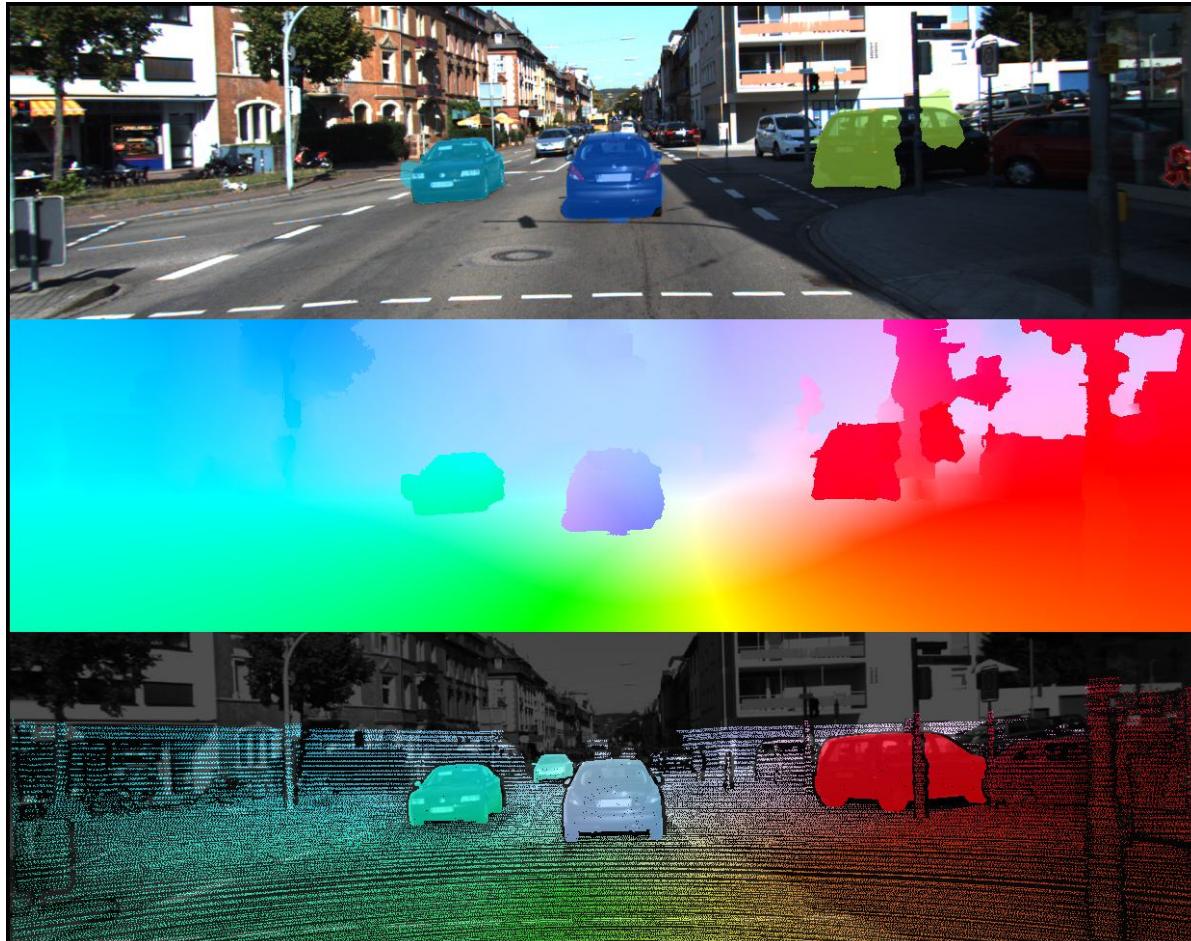




# Интерполяция



Снимаем данные камерой 100Hz, оставляем каждый 4ый кадр.  
Задача – по оптическому потоку проинтерполировать  
промежуточные



- 3D сканирование
- Отдельно фон и движущиеся объекты
- Вписывание 3D моделей объектов в движущиеся объекты

[http://www.cvlibs.net/datasets/kitti/eval\\_fow.php](http://www.cvlibs.net/datasets/kitti/eval_fow.php)



# Метрики

---

- Angular Error (AE)
  - Пусть  $(u_0, v_0)$  и  $(u_1, v_1)$  – два оптических потока
  - Тогда AE равно углу между  $(u_0, v_0, 1)$  и  $(u_1, v_1, 1)$  в 3D пространстве
  - Считается как скалярное произведение и  $\arccos$
  - Смысл – относительная ошибка, работающая и при нулевых оптических потоках
- Absolut endpoint error (EP)
  - Расстояние между концами векторов оптических потоков:  
$$\sqrt{(u_0 - u_1)^2 + (v_0 - v_1)^2}$$
  - SSD в интерполированных изображениях

$$\left[ \sum_{(x,y)} \frac{(I(x,y) - I_{\text{GT}}(x,y))^2}{\|\nabla I_{\text{GT}}(x,y)\|^2 + \epsilon} \right]^{\frac{1}{2}}$$

Нормализованное по градиенту

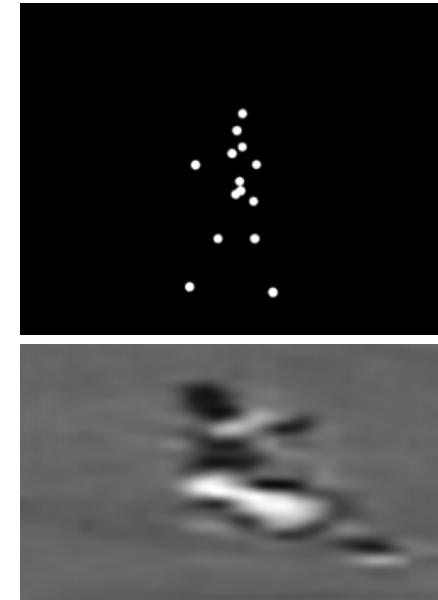
# Ключевое предположение №1



- Как оценить движение пикселей от  $I_1$  в изображение  $I_2$ ?
- Нам нужно всем пикселям из  $I_1$  найти соответствующие пиксели в  $I_2$
- Предположение – пиксель при перемещении не меняет цвет!
- Запишем это условие для всего изображения:

$$E(\mathbf{u}, \mathbf{v}) = \sum_{i,j} \{ \rho_D(I_1(i, j) - I_2(i + u_{i,j}, j + v_{i,j})) \}$$

$E(\mathbf{u}, \mathbf{v})$  – целевая функция,  $\rho_D$  – функция вычисления ошибки



- Соответствует ли минимум функции нашей цели?



# Минимизация функции

---

$$E(\mathbf{u}, \mathbf{v}) = \sum_{i,j} \{ \rho_D(I_1(i, j) - I_2(i + u_{i,j}, j + v_{i,j})) \}$$

- Глобальный минимум может не соответствовать оптическому потоку, т.к. мы можем найти похожие по яркости пиксели где угодно по изображению.
- Нужны ещё предположения и ограничения!



# Посмотрим пример

---



- Что можно сказать про смещение пикселей?
  - Пиксели смещаются на небольшое расстояние (предположение N2)
  - Можно наложить порог на длину вектора ( $u_{ij}, v_{ij}$ )



# Посмотрим пример

---



- Что можно ещё сказать про движение пикселей?
  - Соседние пиксели чаще всего двигаются похоже!
  - Т.е. вектора оптического потока близки у близких пикселей!

## Ключевое предположение №3



Добавим в нашу целевую функцию штрафы за изменение векторов оптического потока:

$$E(\mathbf{u}, \mathbf{v}) = \sum_{i,j} \{ \rho_D(I_1(i, j) - I_2(i + u_{i,j}, j + v_{i,j})) \quad (1) \\ + \lambda [\rho_S(u_{i,j} - u_{i+1,j}) + \rho_S(u_{i,j} - u_{i,j+1}) \\ + \rho_S(v_{i,j} - v_{i+1,j}) + \rho_S(v_{i,j} - v_{i,j+1})] \},$$

где  $\rho_S(x)$  – штрафная функция, например, квадратичная

$$\rho(x) = x^2$$

Получили глобальную целевую функцию для вычисления оптического потока между двумя кадрами



# Глобальный подход

---

$$E(\mathbf{u}, \mathbf{v}) = \sum_{i,j} \{ \rho_D(I_1(i, j) - I_2(i + u_{i,j}, j + v_{i,j})) \quad (1) \\ + \lambda [\rho_S(u_{i,j} - u_{i+1,j}) + \rho_S(u_{i,j} - u_{i,j+1}) \\ + \rho_S(v_{i,j} - v_{i+1,j}) + \rho_S(v_{i,j} - v_{i,j+1})] \},$$

- Это первая формулировка задачи оптического потока для всего изображения в целом (Horn, 1981)
- Будем оптимизировать её каким-нибудь методом оптимизации
- Такой подход для оценки оптического потока называют **глобальным**

# Развитие глобальных методов

---



- Глобальные методы обеспечивают сейчас наилучшее качество оптического потока
- Отличаются друг от друга:
  - Функциями штрафов за изменение вида пикселей (яркость, цвет, градиенты и т.д.)
  - Функциями штрафов за разницу в векторах потока у соседних пикселей
  - Методами оптимизации (непрерывные, дискретные, стохастические)
  - Дополнительными эвристиками и ограничениями, например, регулярным применением медианной фильтрации

# Посмотрим пример ещё раз



- Что можно ещё сказать про движение пикселей?
  - В некоторых случаях разница между векторами движения очень большая и оправданная (разные объекты)
  - В других областях вектора движения почти одинаковые



# Робастные штрафы

- Штрафы должны быть «устойчивы» к разрывами в карте оптического потока на границах объектов
- Штраф должен ещё не «портить» функцию
  - Чтобы мы могли её хорошо оптимизировать



$$\rho(x) = \sqrt{x^2 + \epsilon^2}$$

Charbonnier  
(дифференцируемый L1)

$$\rho(x) = \log(1 + \frac{x^2}{2\sigma^2})$$

Lorenzian

# Резюме глобальных методов

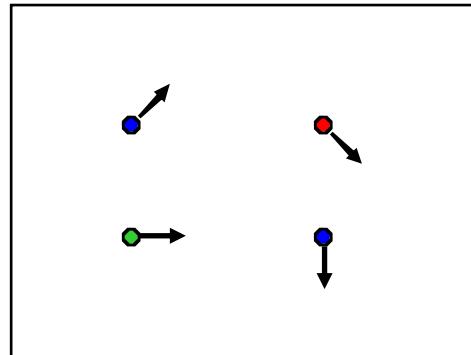


$$E(\mathbf{u}, \mathbf{v}) = \sum_{i,j} \{ \rho_D(I_1(i,j) - I_2(i + u_{i,j}, j + v_{i,j})) \quad (1) \\ + \lambda [\rho_S(u_{i,j} - u_{i+1,j}) + \rho_S(u_{i,j} - u_{i,j+1}) \\ + \rho_S(v_{i,j} - v_{i+1,j}) + \rho_S(v_{i,j} - v_{i,j+1})] \},$$

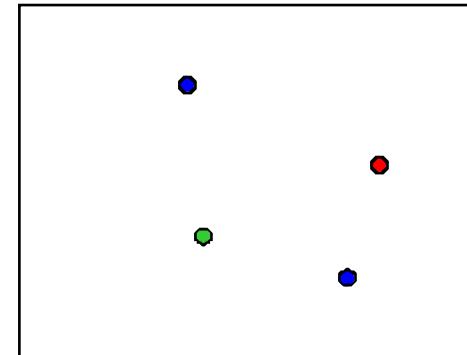
- Оптимизируем одну целевую функцию для всего векторного поля
- Обеспечивают наибольшую точность
- Многие методы крайне медленные (до нескольких часов на пару кадров!)



# Локальный подход



$H(x, y)$



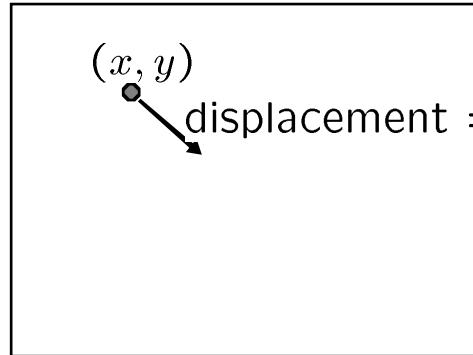
$I(x, y)$

- Попробуем вычислить вектор движения для одного пикселя  $(x, y)$  из изображения  $I$  в изображение  $H$
- Используем ограничения для формализации задачи
  - Постоянная яркость
  - Малое смещение

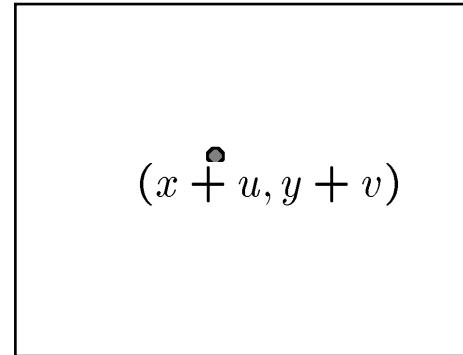


# Локальный подход

---



$$H(x, y)$$



$$I(x, y)$$

С учётом малого смещения, разложим  $I(x+u, y+v)$  в ряд Тейлора:

$$\begin{aligned} I(x+u, y+v) &= I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms} \\ &\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v \end{aligned}$$

# Уравнение оптического потока



- Объединим с условием постоянной яркости:

$$\begin{aligned} 0 &= I(x + u, y + v) - H(x, y) & I_x &= \frac{\partial I}{\partial x} \\ &\approx I(x, y) + I_x u + I_y v - H(x, y) \\ &\approx (I(x, y) - H(x, y)) + I_x u + I_y v \\ &\approx I_t + I_x u + I_y v \\ &\approx I_t + \nabla I \cdot [u \ v] \end{aligned}$$

В пределе  $u$  и  $v$  стремятся к нулю, и получаем равенство:

$$0 = I_t + \nabla I \cdot \left[ \frac{\partial x}{\partial t} \frac{\partial y}{\partial t} \right]$$

# Уравнение оптического потока

---



- Элементарное уравнение оптического потока:

$$0 = I_t + \nabla I \cdot [u \ v]$$

- Вопрос: сколько неизвестных и уравнений для каждого пикселя?
- 1 уравнение, 2 неизвестных ( $u, v$ )

# Дополнительные уравнения



- Как можно получить больше уравнений?
- Воспользуемся предположением о близости векторов движения для соседних пикселей
- Пусть для всех пикселей  $p$  из окрестности  $(x, y)$  смещение  $(u, v)$  одинаковое!!
  - Для окна  $5 \times 5$  получаем 25 уравнений для каждого пикселя!

$$0 = I_t(p_i) + \nabla I(p_i) \cdot [u \ v]$$

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix}$$

$A$   
 $25 \times 2$

$d$   
 $2 \times 1$

$b$   
 $25 \times 1$



# Алгоритм Лукаса-Канаде

- Проблема: больше уравнений, чем неизвестных!

$$\begin{matrix} A & d = b \\ 25 \times 2 & 2 \times 1 & 25 \times 1 \end{matrix} \longrightarrow \text{minimize } \|Ad - b\|^2$$

- Получаем задачу наименьших квадратов
- Можем решить её через нормальные уравнения

$$\begin{matrix} (A^T A) & d = A^T b \\ 2 \times 2 & 2 \times 1 & 2 \times 1 \end{matrix} \longrightarrow d = (A^T A)^{-1} A^T b$$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$$A^T A \qquad \qquad \qquad A^T b$$

- Суммируем по всем пикселям в окне  $K \times K$

B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA Image Understanding Workshop, April 1981*.



# Условия на разрешимость

---

- Решение задачи оптического потока  $d = (u, v)$  может быть найдено в виде

$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

$$d = (A^T A)^{-1} A^T b \quad A^T b = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

- Когда задача разрешима?
  - $A^T A$  должна быть обратимой
  - $A^T A$  не должна быть слишком близка к нулю (вырожденной)
  - $A^T A$  должна быть хорошо определена



# Что это напоминает?

$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}$$

Матрица для  
нахождения локальных  
особенностей  
детектором Харриса!

$$A^T A = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

$A^T A$  хорошо определена в  
локальных особенностях  
изображения!

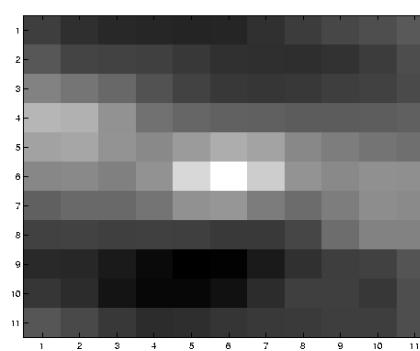
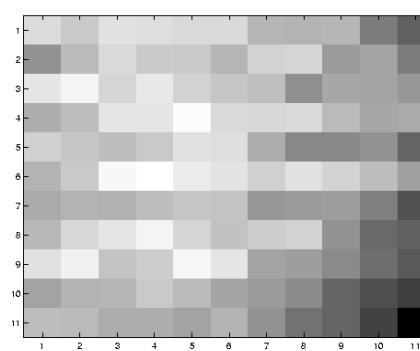
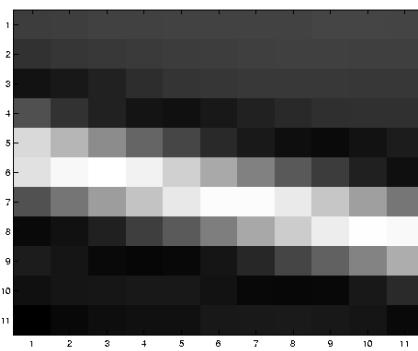
$\lambda_1$  и  $\lambda_2$  малы;  
 $A^T A$  плохо  
обусловлена



# Локальный оптический поток



- Локально оценить оптический поток может только в некоторых точках, соответствующих локальным особенностям
- Как следствие, хотя вычисляется поток по 2м изображениям, но определить, где он будет считаться хорошо мы можем по 1 изображению



Jianbo Shi and Carlo Tomasi, "Good Features to Track," CVPR 1994

# Итеративное уточнение

---

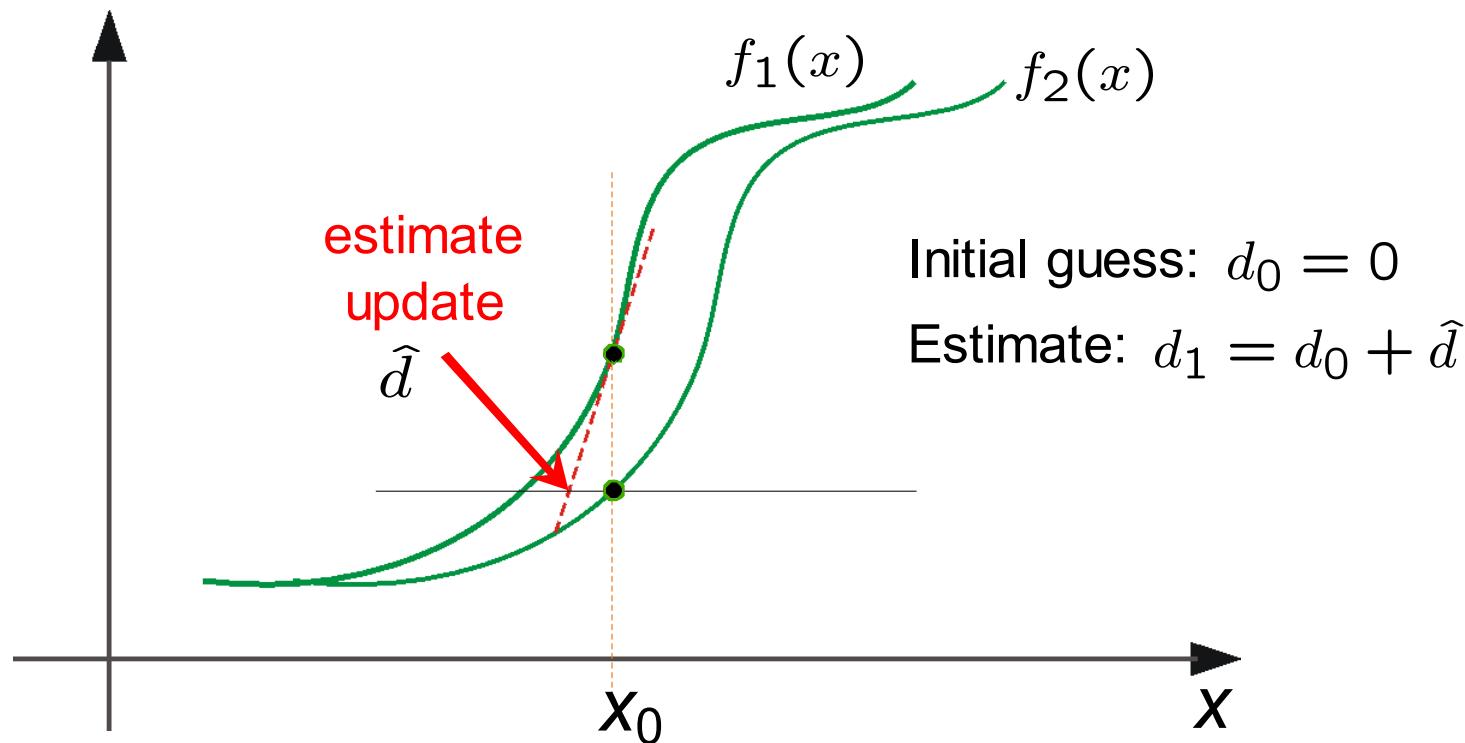


- Вспомним первый шаг

$$\begin{aligned}I(x+u, y+v) &= I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms} \\&\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v\end{aligned}$$

- Поскольку мы с самого начала использовали приближение, то получаем только приближенное решение
- Можем уточнить за счёт итеративной процедуры:
  1. Оценить движение в каждом пикселе, решив уравнения Лукаса-Канаде
  2. Преобразовать изображение  $\mathbf{H}$  используя вычисленное движение
  3. Повторить 1-2 до сходимости

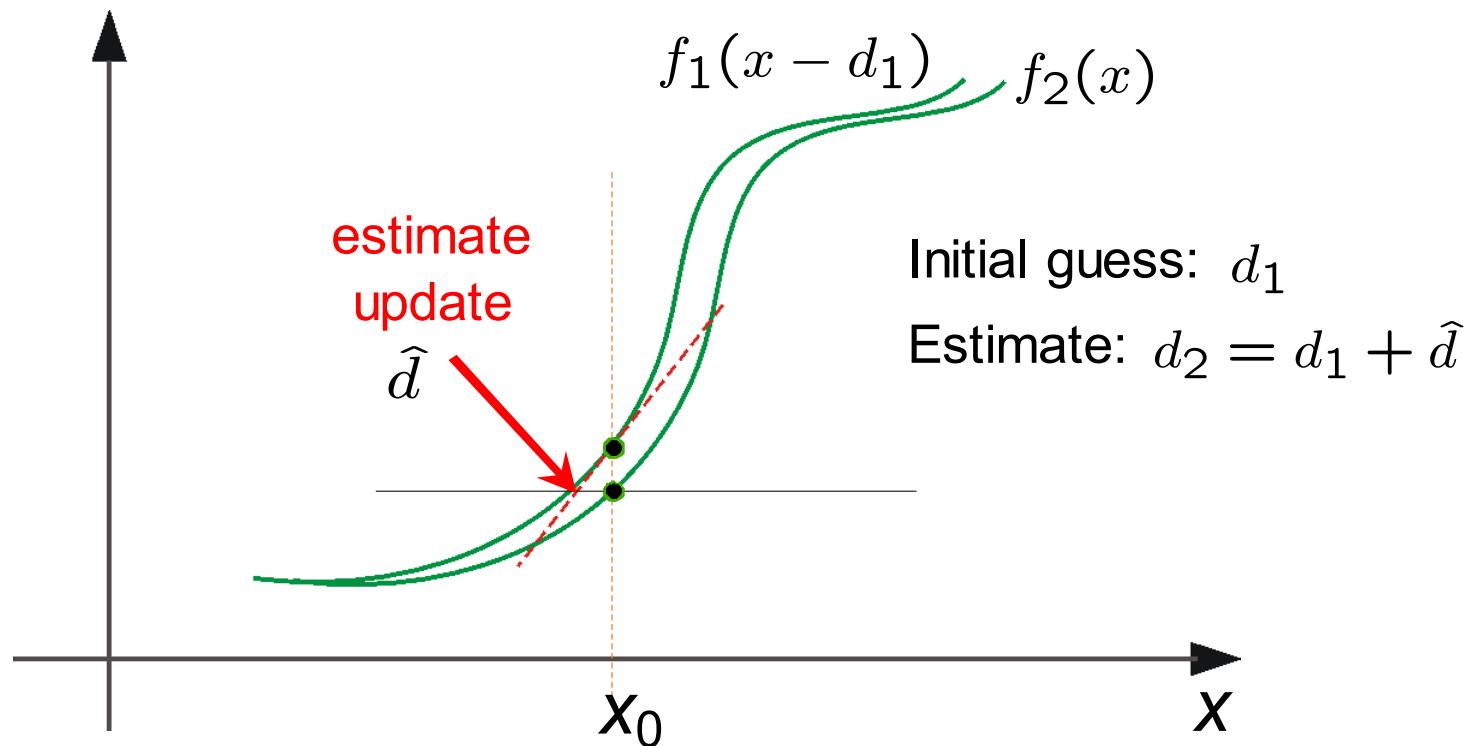
# Пример



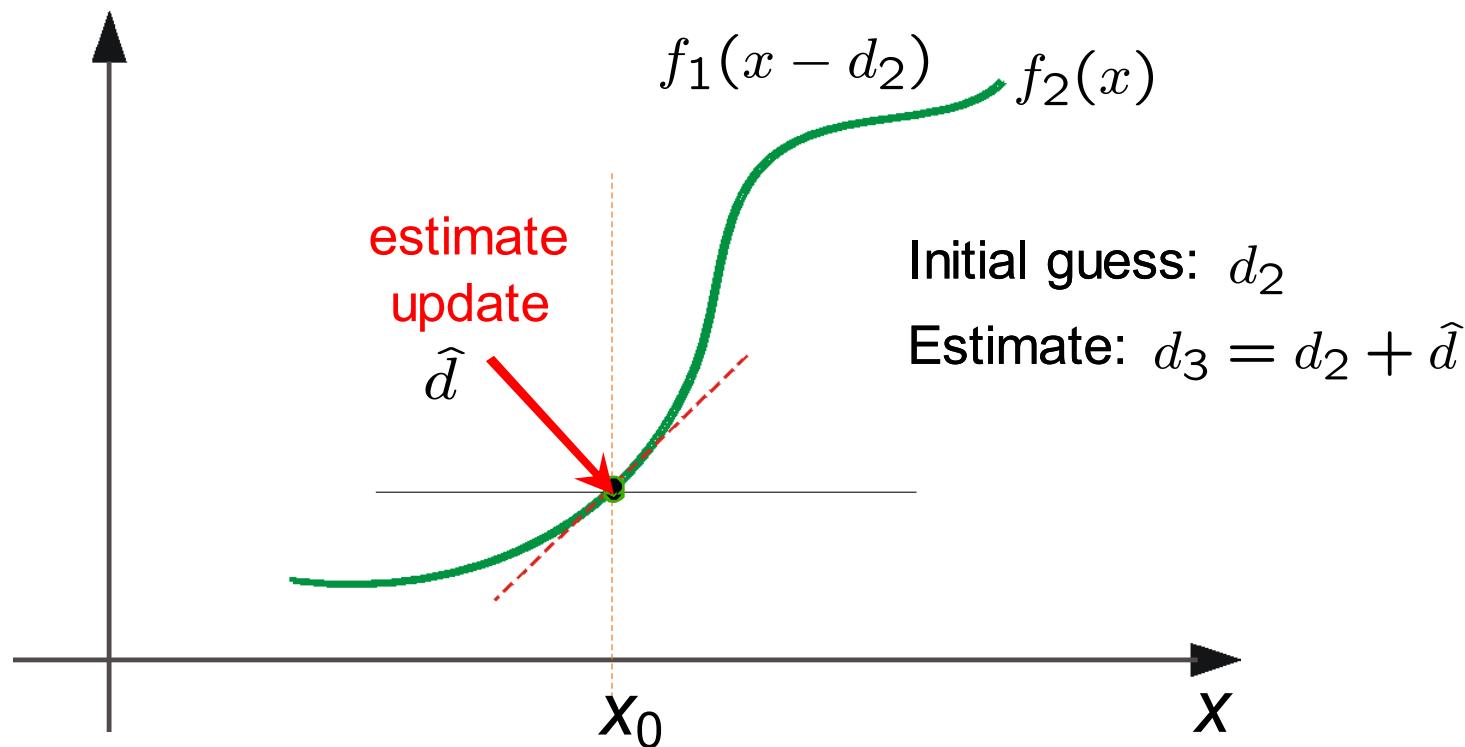
- Рассмотрим одномерный случай (d-смещение)
- Сдвиг по градиенту должен быть равен изменению значения (яркости).

$$0 = I_t + I_x d$$

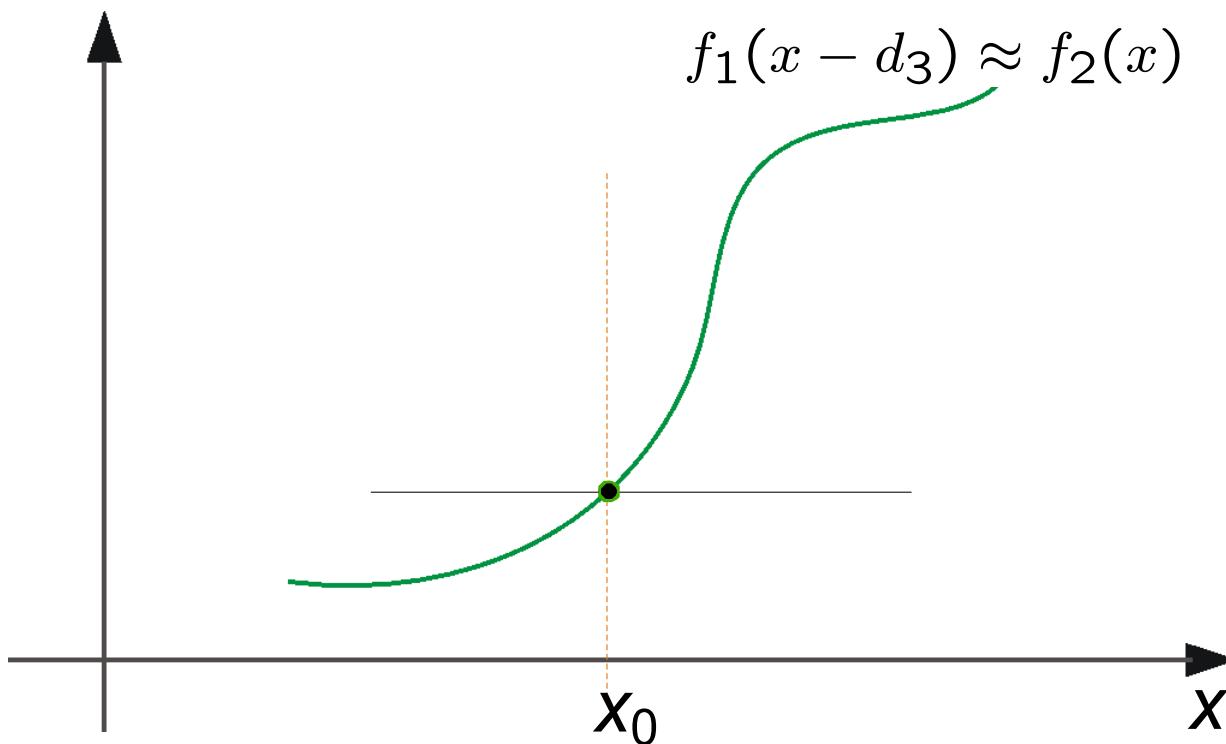
# Пример



# Пример



# Пример



# Проблема большого смещения

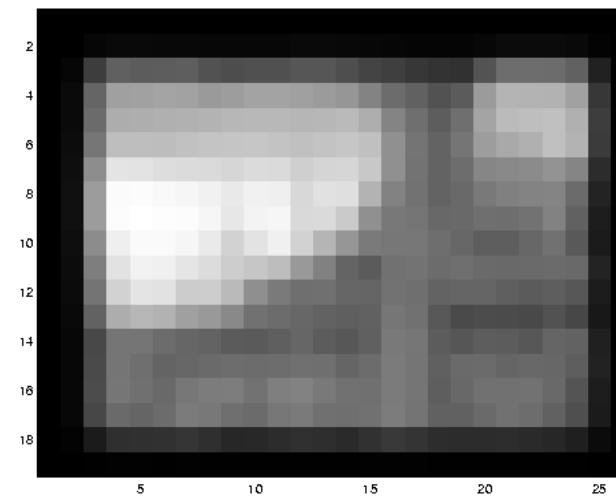
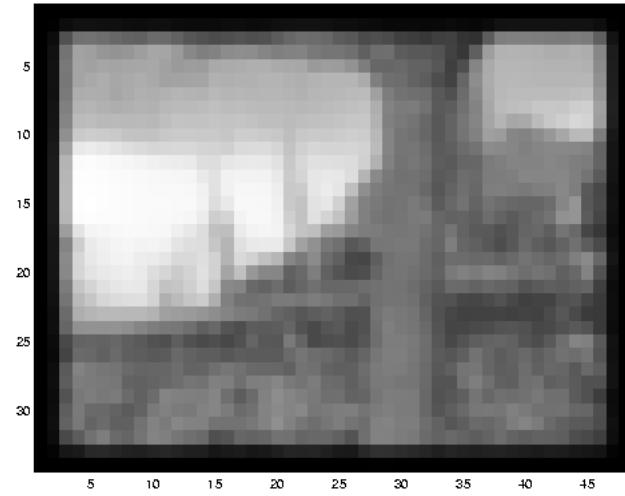
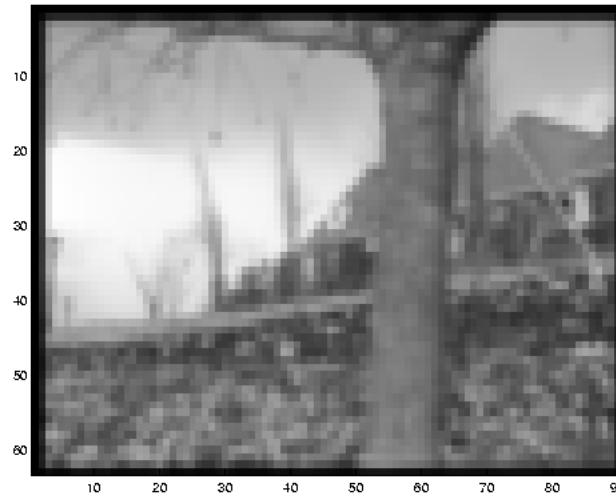


Насколько мало движение в изображении?

- Существенно больше 1-го пикселя
- Как можно решить эту проблему?

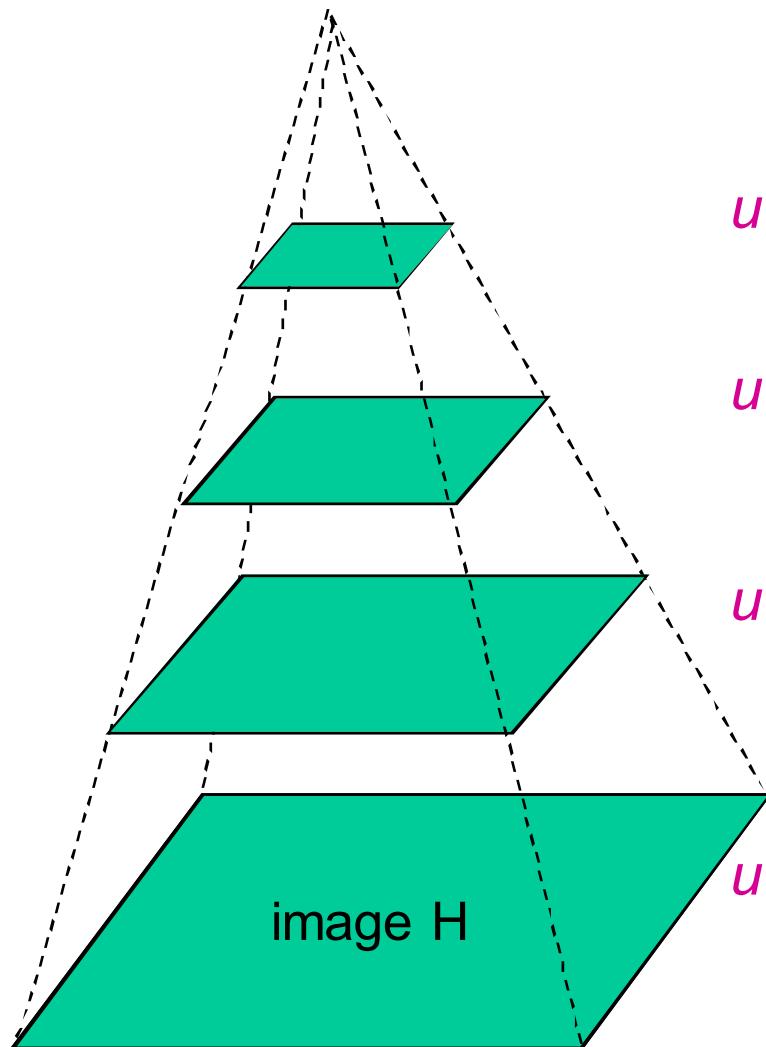


# Пирамида разрешений

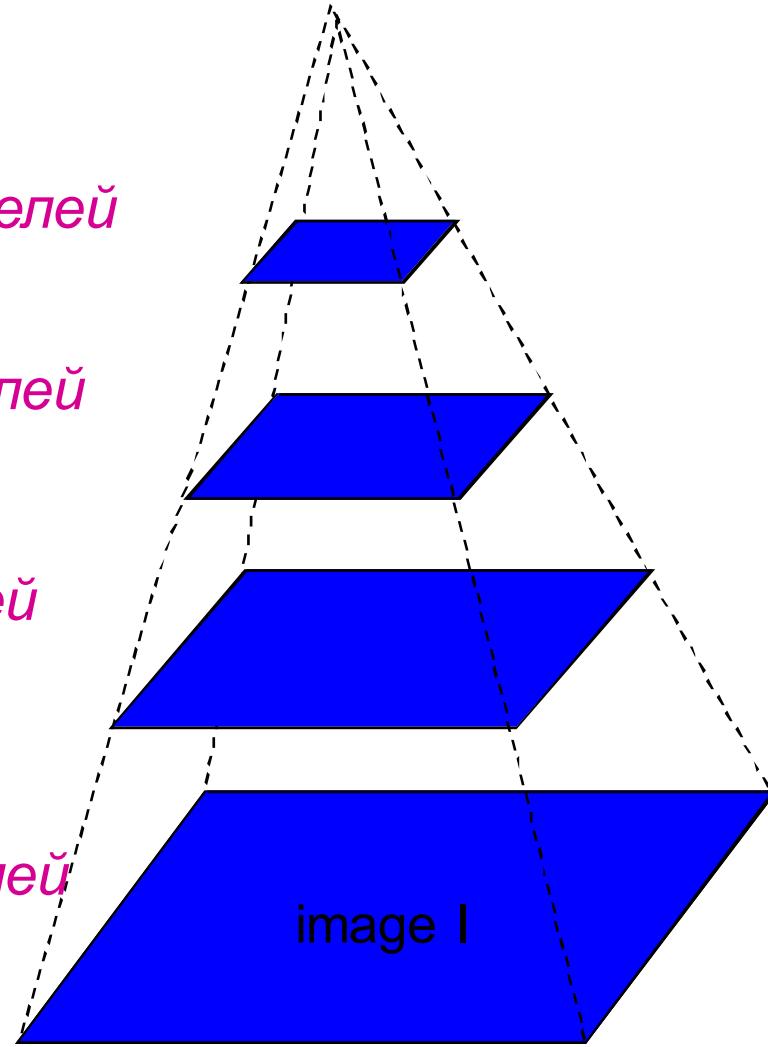


Насколько мало движение в изображении?

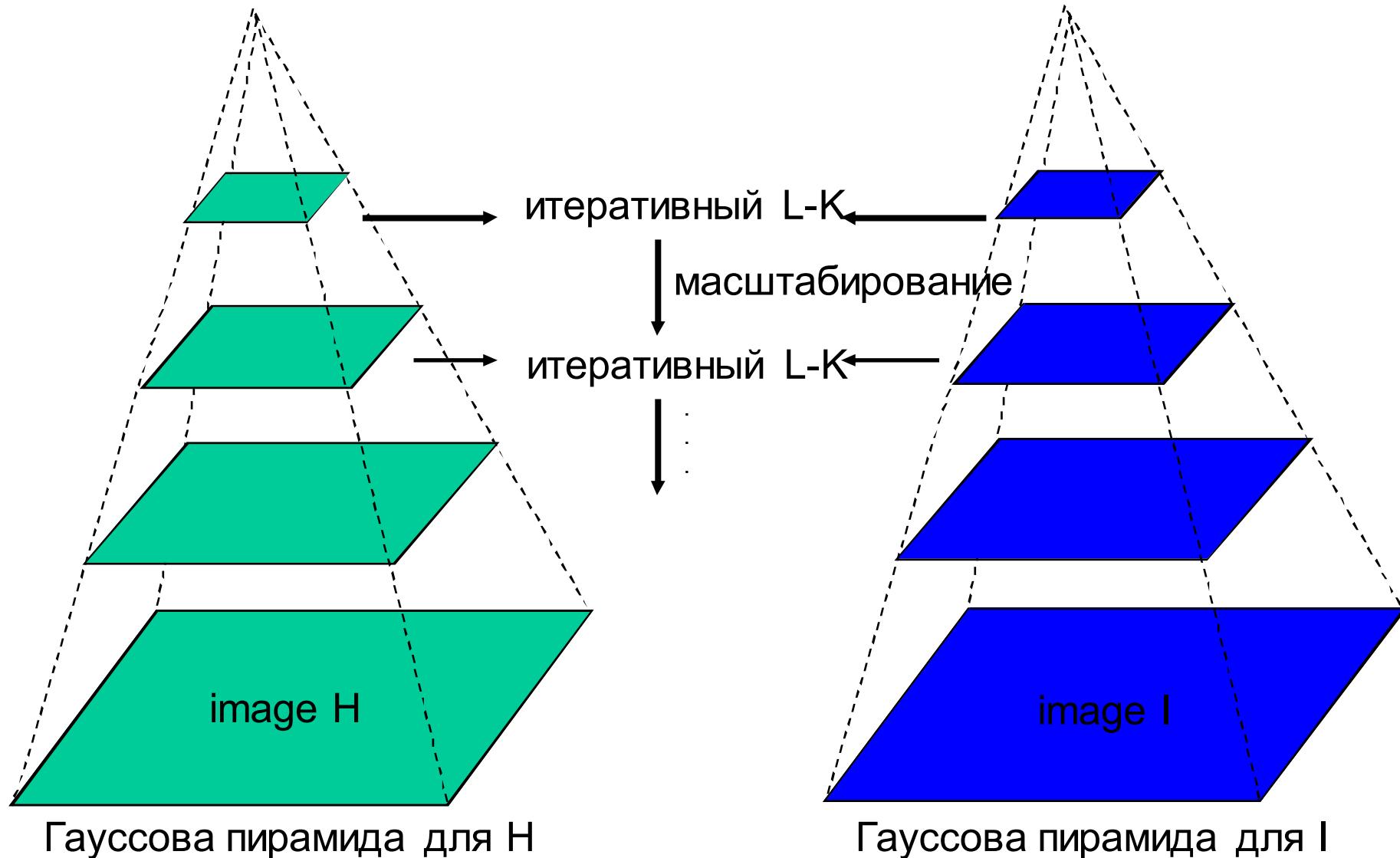
# Иерархический метод



$u=1.25$  пикселей  
 $u=2.5$  пикселей  
 $u=5$  пикселей  
 $u=10$  пикселей

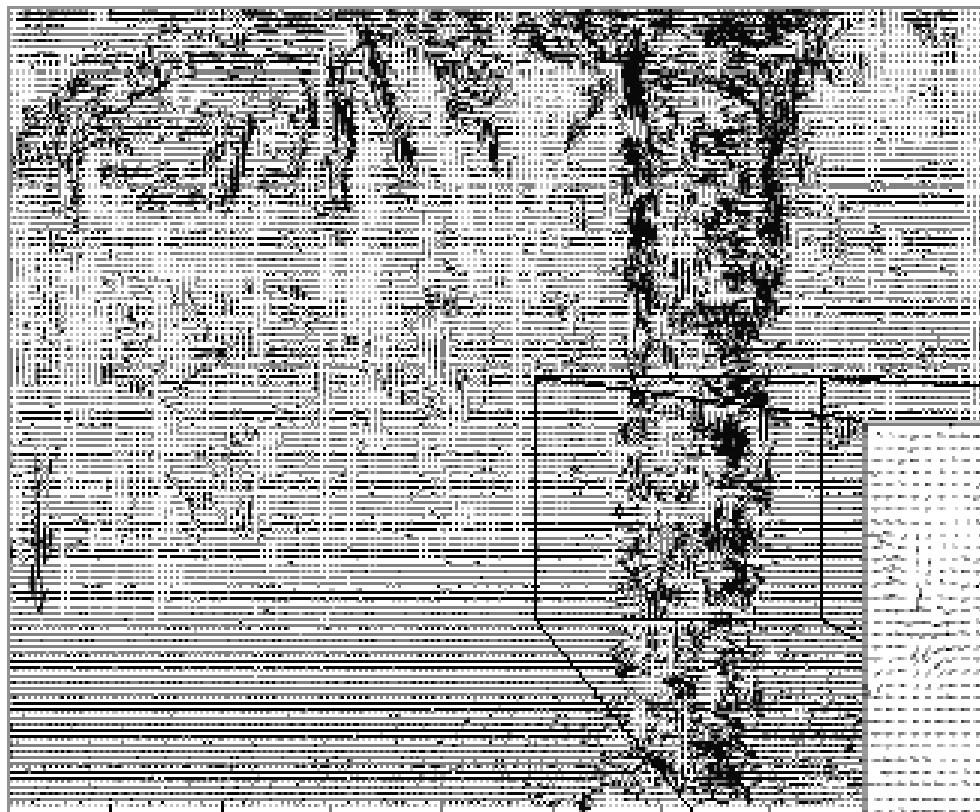


# Иерархический метод



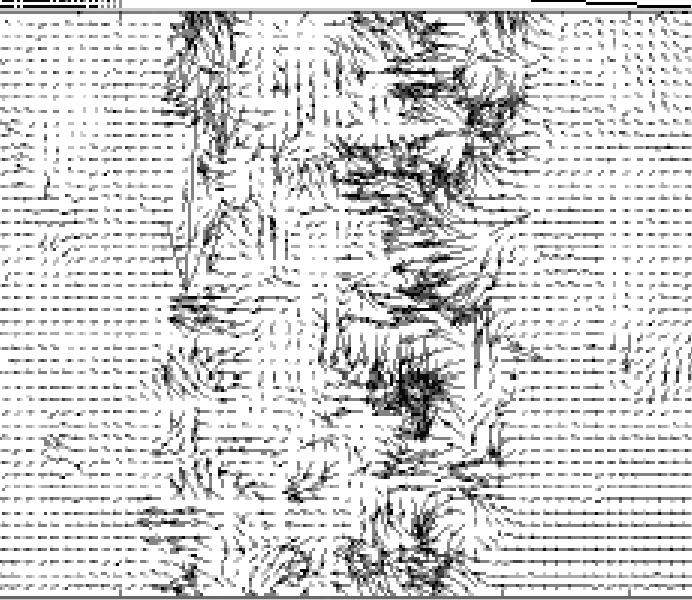


# Optical Flow Results



Lucas-Kanade  
without pyramids

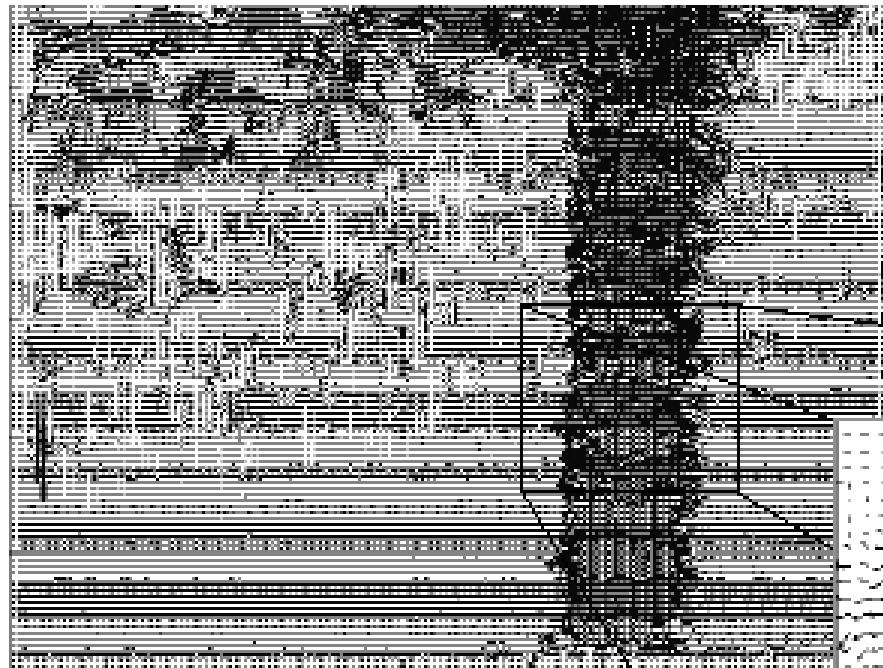
Fails in areas of large motion



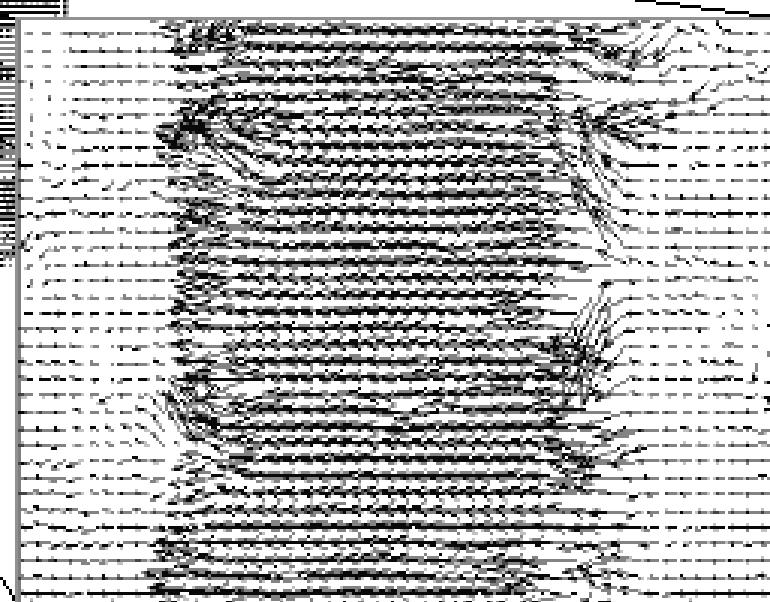


# Optical Flow Results

---



Lucas-Kanade with Pyramids



# Резюме метода Лукаса-Канаде

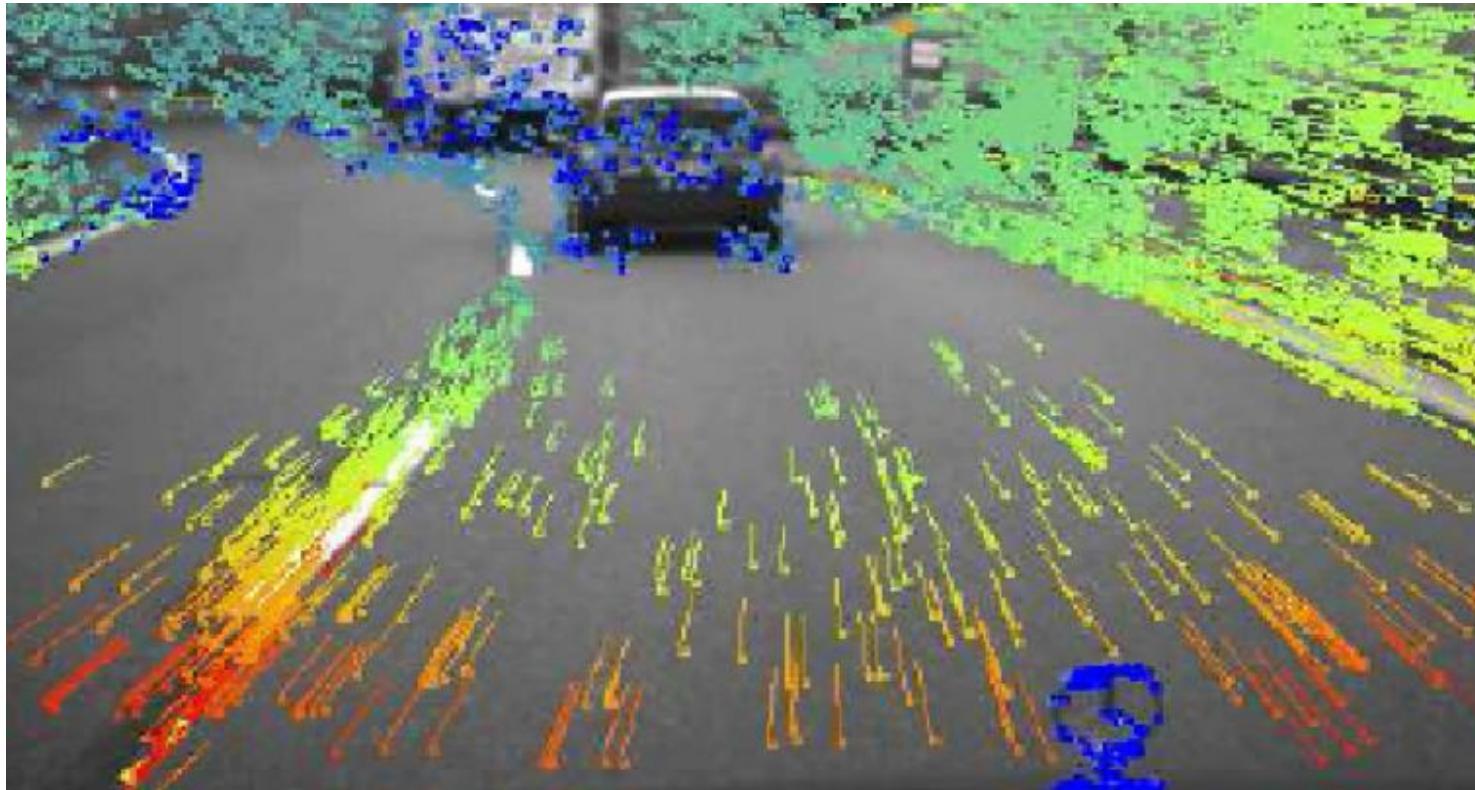
---



- Локальный метод Лукаса-Канаде позволяет надежно оценивать поток для отдельных пикселей, но только в окрестности особых точек
- Для повышения качества для больших смещений используется многомасштабная версия алгоритма
  - Многомасштабный подход активно используется и для глобальных методов



# Локальные особенности

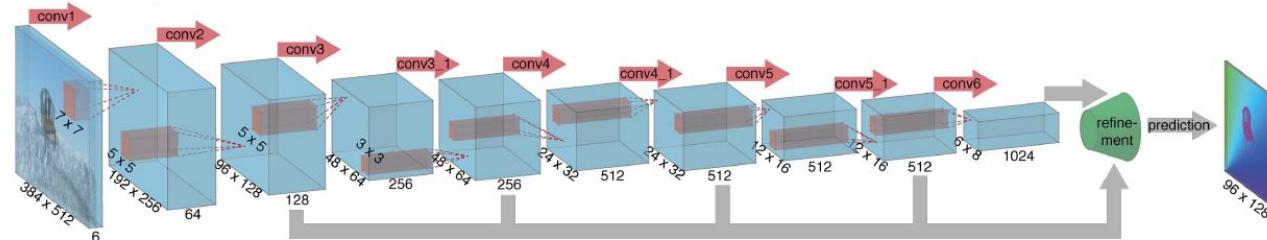


- Сопоставление локальных особенностей даёт хорошее разреженное приближение
- Подходит для больших смещений

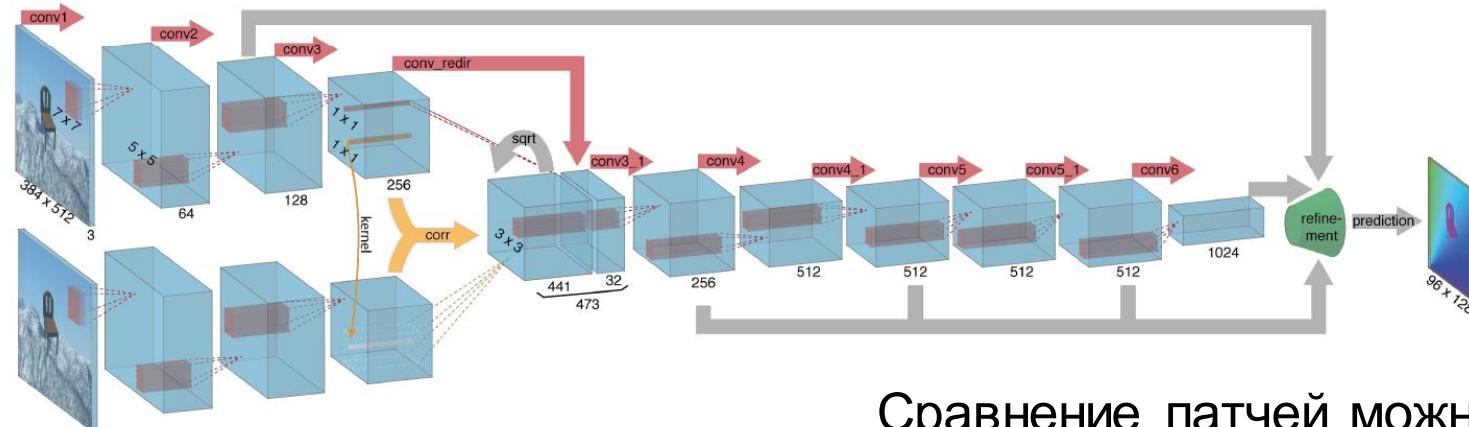
# Нейросетевые модели (FlowNet)



FlowNetSimple



FlowNetCorr



$$c(\mathbf{x}_1, \mathbf{x}_2) = \sum_{\mathbf{o} \in [-k, k] \times [-k, k]} \langle \mathbf{f}_1(\mathbf{x}_1 + \mathbf{o}), \mathbf{f}_2(\mathbf{x}_2 + \mathbf{o}) \rangle$$

Сравнение патчей можно реализовать наподобие свёртки, но без обучаемых параметров. Возможные сдвиги ограничены  $[-d, d]$

A. Dosovitskiy et. al. FlowNet: Learning Optical Flow with Convolutional Networks. 2015

# Обучение



	Frame pairs	Frames with ground truth	Ground truth density per frame
Middlebury	72	8	100%
KITTI	194	194	~50%
Sintel	1,041	1,041	100%
Flying Chairs	22,872	22,872	100%

**Большой объём  
синтетических данных –  
летеющие стулья**

Table 1. Size of already available datasets and the proposed Flying Chair dataset.

# Постобработка

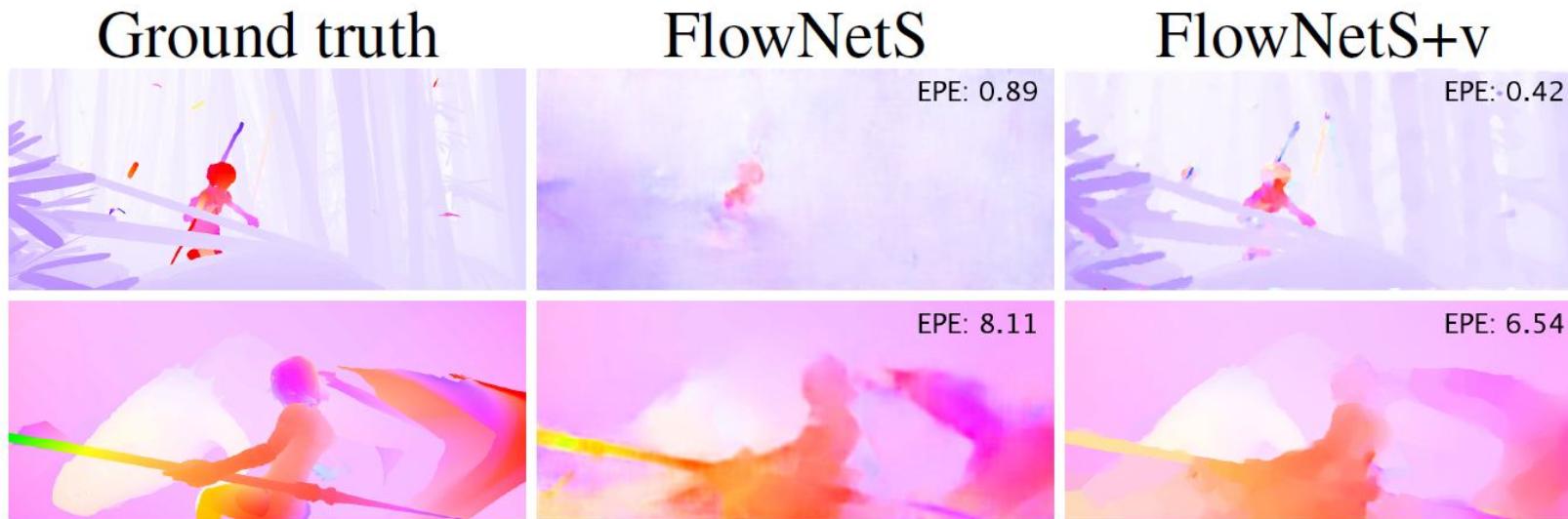


Figure 4. The effect of variational refinement. In case of small motions (first row) the predicted flow is changed dramatically. For larger motions (second row), big errors are not corrected, but the flow field is smoothed, resulting in lower EPE.

# Оценка точности и сравнение



Method	Sintel Clean		Sintel Final		KITTI		Middlebury train		Middlebury test		Chairs	Time (sec)	
	train	test	train	test	train	test	AEE	AAE	AEE	AAE	test	CPU	GPU
EpicFlow [30]	2.27	4.12	3.57	6.29	3.47	3.8	0.31	3.24	0.39	3.55	2.94	16	-
DeepFlow [35]	3.19	5.38	4.40	7.21	4.58	5.8	0.21	3.04	0.42	4.22	3.53	17	-
EPPM [3]	-	6.49	-	8.38	-	9.2	-	-	0.33	3.36	-	-	0.2
LDOF [6]	4.19	7.56	6.28	9.12	13.73	12.4	0.45	4.97	0.56	4.55	3.47	65	2.5
FlowNetS	4.50	7.42	5.45	8.43	8.26	-	1.09	13.28	-	-	2.71	-	0.08
FlowNetS+v	3.66	6.45	4.76	7.67	6.50	-	0.33	3.87	-	-	2.86	-	1.05
FlowNetS+ft	(3.66)	6.96	(4.44)	7.76	7.52	9.1	0.98	15.20	-	-	3.04	-	0.08
FlowNetS+ft+v	(2.97)	6.16	(4.07)	7.22	6.07	7.6	0.32	3.84	0.47	4.58	3.03	-	1.05
FlowNetC	4.31	7.28	5.87	8.81	9.35	-	1.15	15.64	-	-	2.19	-	0.15
FlowNetC+v	3.57	6.27	5.25	8.01	7.45	-	0.34	3.92	-	-	2.61	-	1.12
FlowNetC+ft	(3.78)	6.85	(5.28)	8.51	8.79	-	0.93	12.33	-	-	2.27	-	0.15
FlowNetC+ft+v	(3.20)	6.08	(4.83)	7.88	7.31	-	0.33	3.81	0.50	4.52	2.67	-	1.12

Table 2. Average endpoint errors (in pixels) of our networks compared to several well-performing methods on different datasets. The numbers in parentheses are the results of the networks on data they were trained on, and hence are not directly comparable to other results.

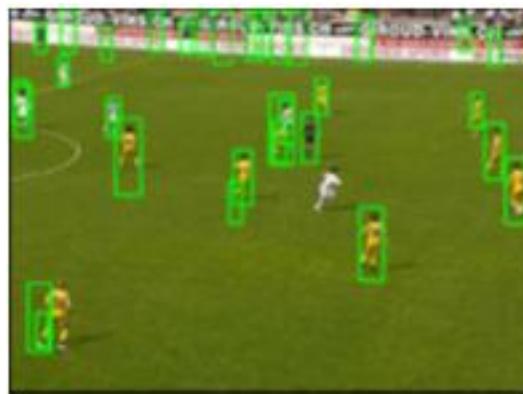


# Резюме

---

- «Оптический поток» формализует понятие движения в видео
- Для оценки методов используются несколько коллекций – Middlebury, KITTI и другие
- Мы посмотрели подходы:
  - Локальный метод Лукаса-Канаде
  - Глобальный на основе оптимизации
  - Нейросетевая модель FlowNet

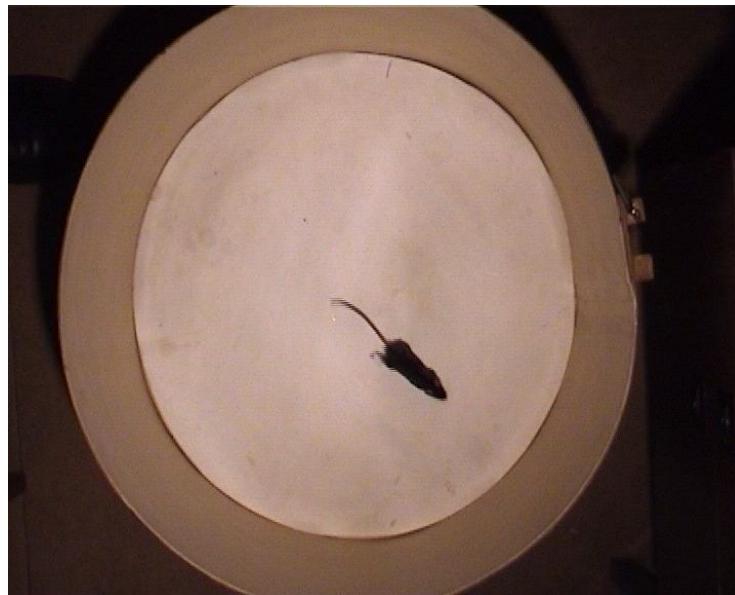
# Система видеонаблюдения



- Камера, наблюдающая некоторую сцену (в закрытом или открытом пространстве)
- Необходимо выделить «объекты интереса» на каждом кадре
- Результат:
  - Ограничивающий прямоугольник
  - Попиксельная маска объекта интереса

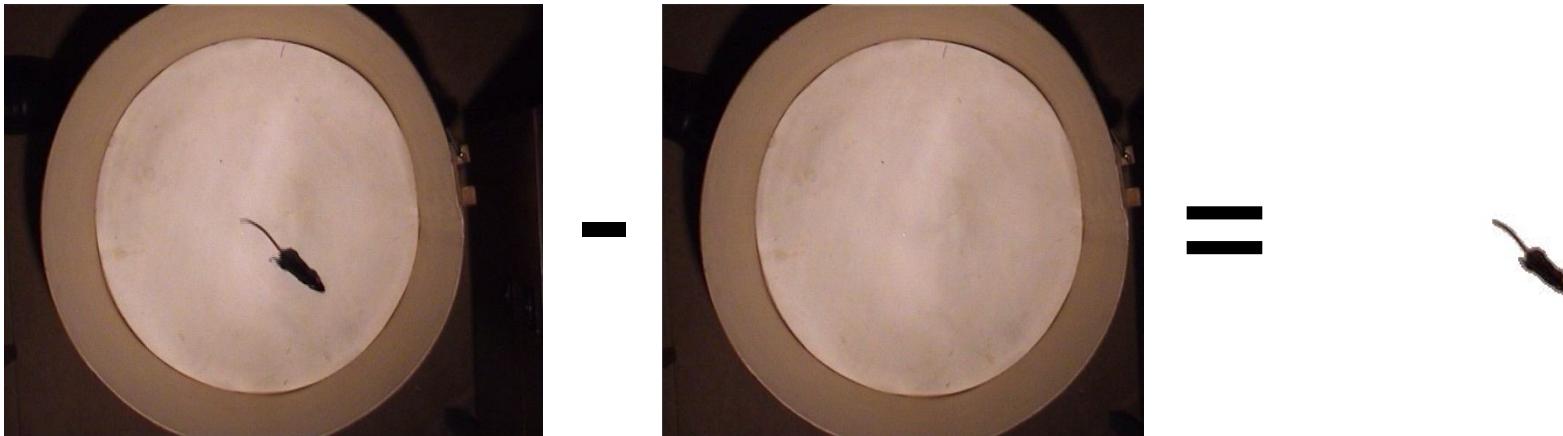


# Свойства задачи



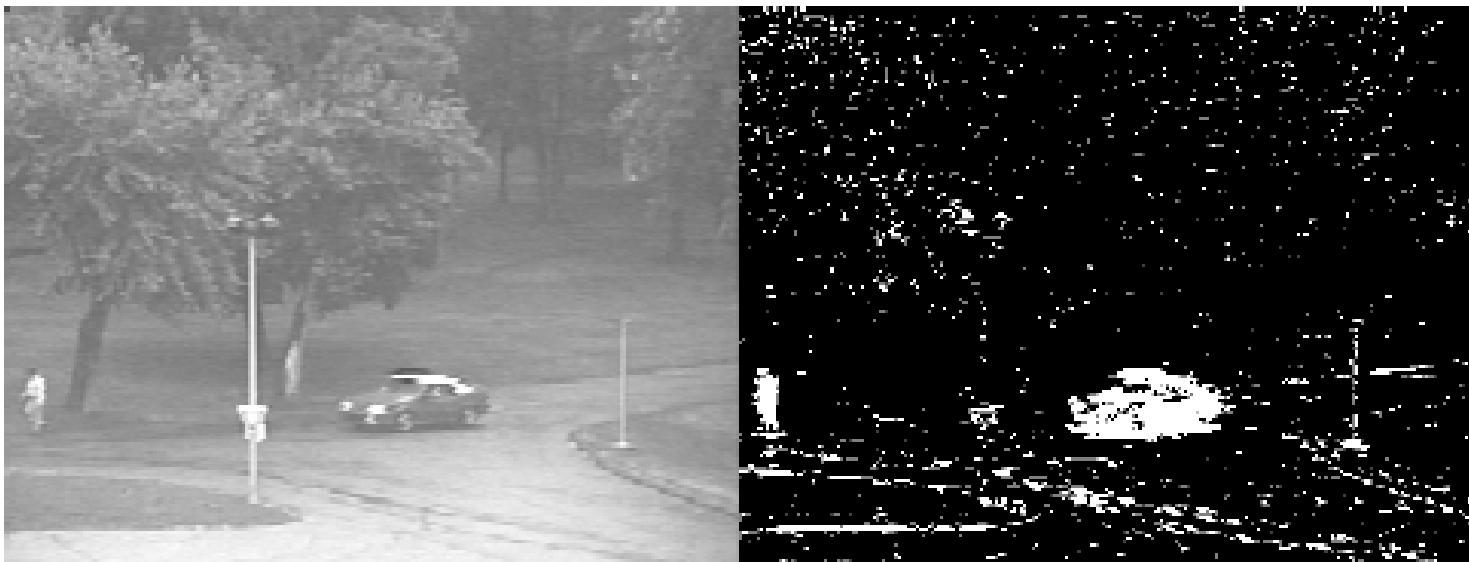
- Камеру считаем стационарной, не меняющей ракурса
- Фон считаем «стабильным», малоподвижным, незначительно меняющимся между кадрами
- Вывод – объекты должны отличаться от фона
- Ещё системы называют «детектор движения»

# Вычитание фона



- Метод «Background subtraction»:
  - Возьмем «чистое» изображение без объектов – «фон» (*background*)
  - Вычтем фон из новых изображений с объектами
  - Сравним разницу для каждого пикселя с порогом
    - Порог – параметр алгоритма
  - Если разница больше порога - то считаем пиксель принадлежащим «переднему плану» (*foreground*)
  - Получаем бинарную маску «переднего плана»
    - Фон (0), передний план (1)

# Обработка переднего плана



- Бинарная маска переднего плана обычно шумная и пиксели не связаны между собой
- Нам нужно:
  - Уменьшить шум в изображении (фильтрация)
  - Выделить области, потенциально соответствующие объектам (связанные компоненты)
- «Блоб» («капля», *blob*) - связанная компоненты маски переднего плана, потенциальный объект

# Наблюдение за мышами

---



- Если объект в сцене может быть только один, то самый крупную связанные компоненту (блоб) будем считать объектом
- Получили «базовый» алгоритм вычитания фона, который работает в ряде лабораторных задач (отслеживание мышей)
- Схема:
  - Попиксельное вычитание изображения фона из текущего кадра
  - Сравнение попиксельной разницы с порогом
  - Фильтрация маски
  - Выделение связанных компонент и выбор самой большой

# Если объектов много?

---



- Случай множества связанных компонент – это уже задача отслеживания объекта
- Некоторые подходы к ней рассмотрим на следующей лекции



# Проблемы вычитания фона

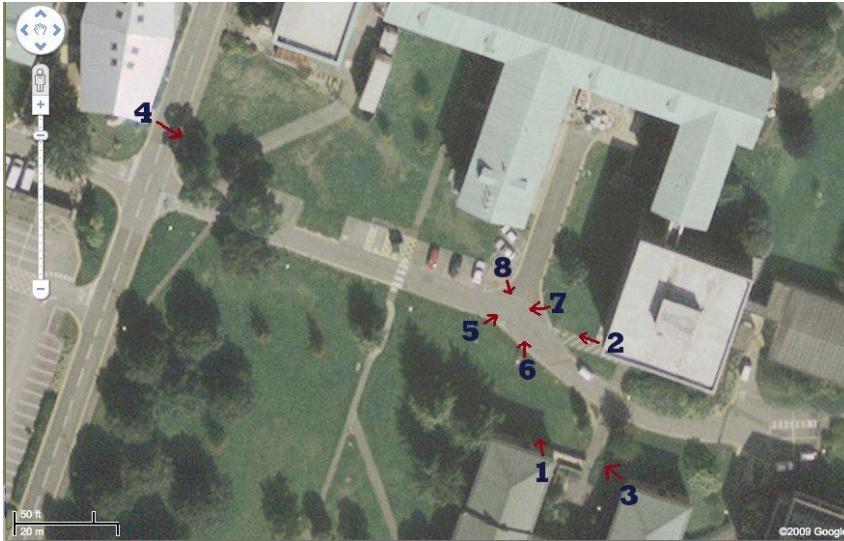
	Moved Object	Time of Day	Light Switch	Waving Trees	Camouflage	Bootstrapping	Foreground Aperture
Test Image							
	Chair moved	Light gradually brightened	Light just switched on	Tree Waving	Foreground covers monitor pattern	No clean background training	Interior motion undetectable
Ideal Foreground							
Mean & Threshold							

- Шум камеры
- Стационарные объекты

from K. Toyama et al.



# Данные



- На пути создания общедоступных тестовых баз два препятствия:
  - Приватность
  - Сложность разметки данных
- PETs Performance Evaluation of Tracking and Surveillance
- <http://www.cvg.rdg.ac.uk/PETS2009/a.html>



# Пример работы



Пример слежения с реальной  
системы видеонаблюдения



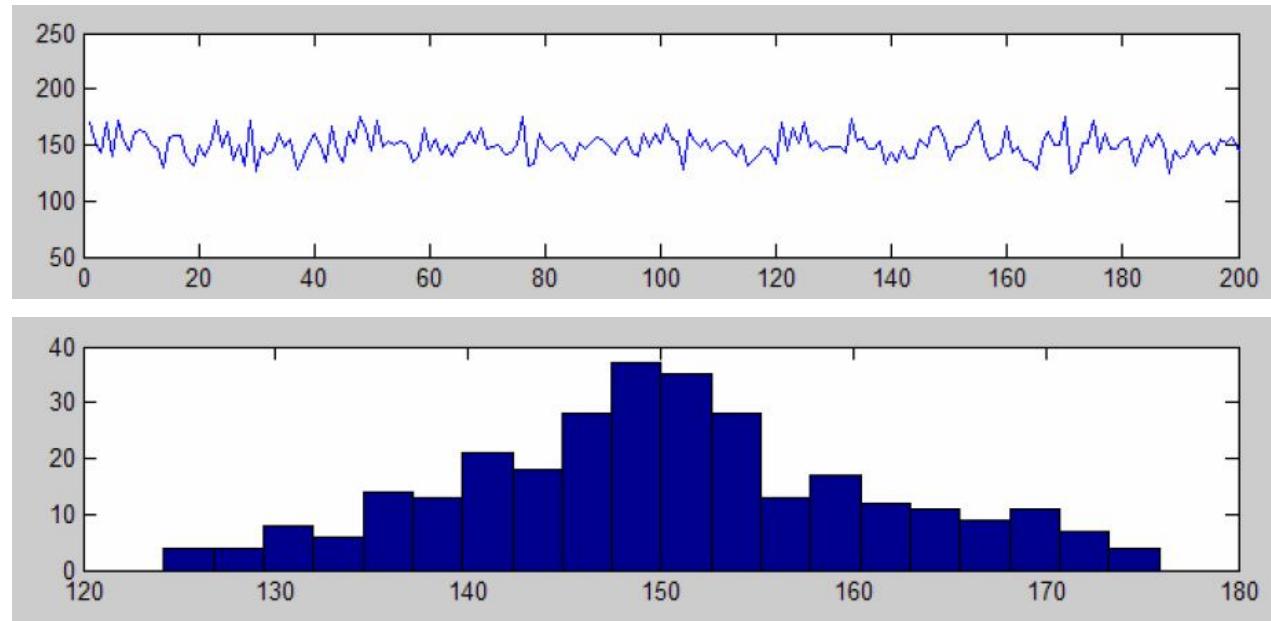
Открытая коллекция размеченных данных разного типа.  
Сейчас оптимальный подход к экспериментальной  
оценке методов.

<http://changedetection.net>



# Моделирование фона

- Реальная статичная сцена:
  - «Шум камеры»
  - Интенсивность меняется в небольших пределах вокруг какого-то значения
  - Нормальное (Гауссово) распределение



Wren, Christopher R., Ali Azarbajayani, Trevor Darrell, and Alex Pentland. “Pfinder: Real-Time Tracking of the Human Body,” IEEE PAMI, 1997

# Оценка параметров модели

---



- По чистому видеоролику для каждого пикселя вычислим параметры фона для модели – нормального распределения (случай одного канала):

- Вычисляем среднее

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i$$

- Вычисляем дисперсию

$$\sigma^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \mu)^2$$



# Вычитание фона

---

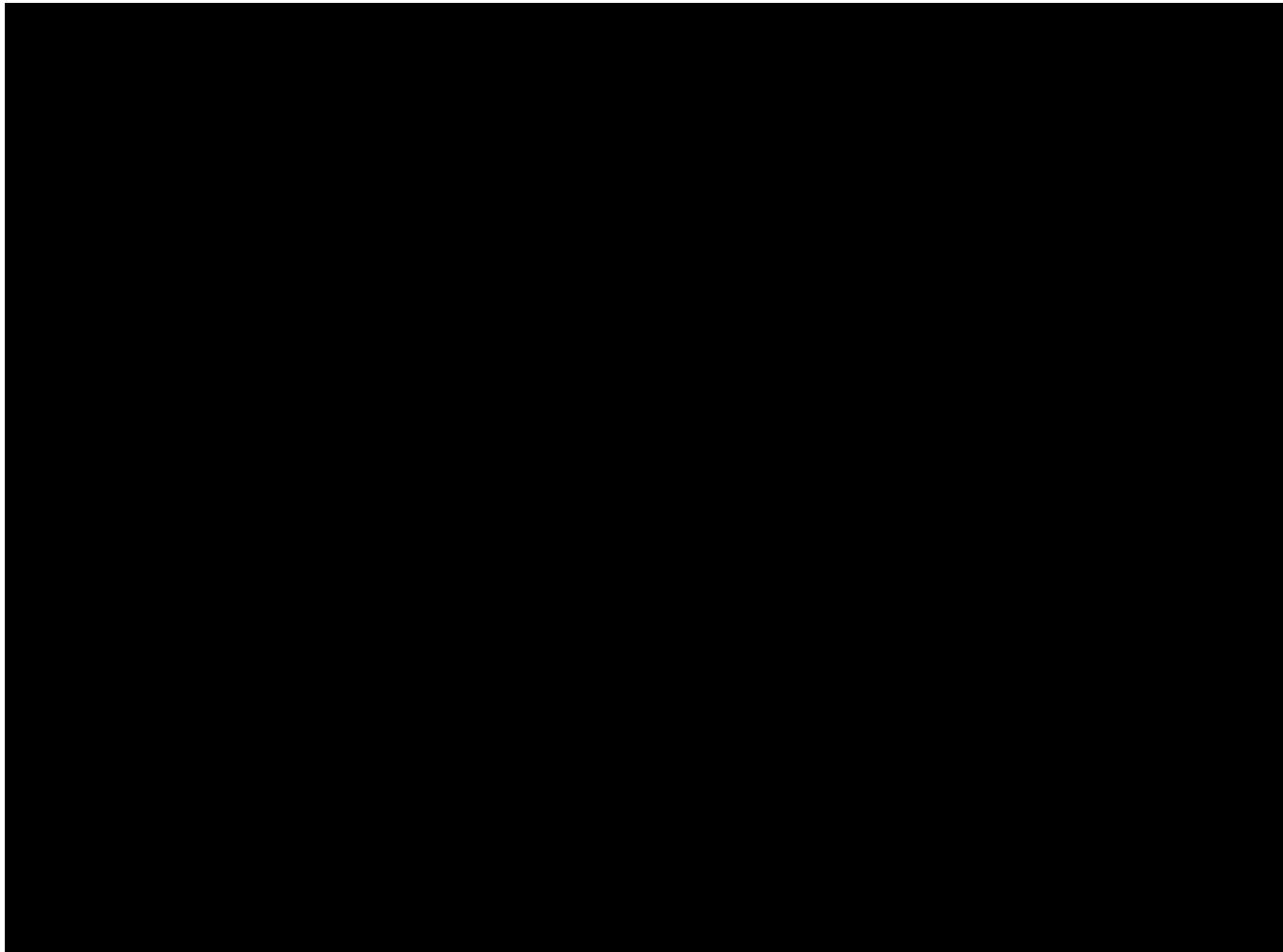
- Вероятность  $x$  при вычисленных параметрах модели фона:

$$\rho(x | \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Проще использовать сравнение с порогом
- Считаем, что все пиксели фона должны попасть в интервал  $(\mu-3\sigma, \mu+3\sigma)$
- Если  $|x-\mu| > 3\sigma$ , тогда  $x$  – пиксель переднего плана

# Пример работы

---





# Получение фона

---



- Как быть, если невозможно получить обучающий ролик без объектов?
- Задача весьма сложная, есть много методов
- Простейший вариант – взять медиану всех кадров

# Пример медианного фона

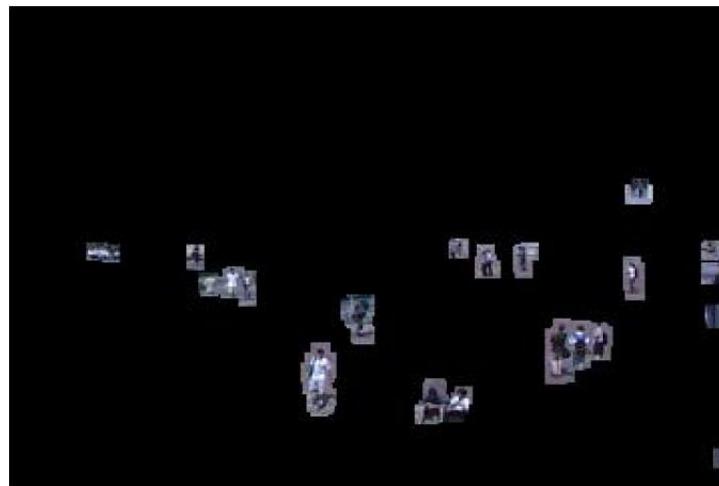
---



=



=





# Получение фона

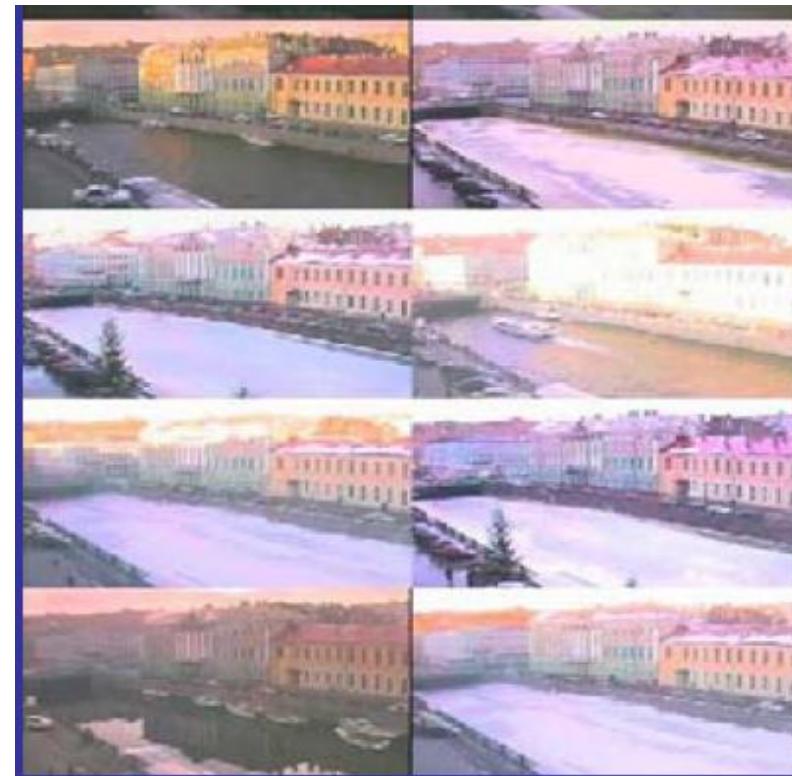


- В сложных случаях медиана не помогает
- Предлагаются разные идеи – поиск наиболее стабильных кластеров, пространственная связанность и т.д.

# Длительное время работы



- Как меняется фон на длинных периодах времени?
- Может плавно меняться освещенность сцены
- Что это скажется на работе алгоритма?
- Что мы можем сделать для того, чтобы алгоритм мог работать?
  
- Нам нужно дополнить алгоритм этапом обновления модели фона



# Общая схема вычитания фона

---



- *Initialize\_background\_model()*
- *For*  $t = 1:N$ 
  - *Compute\_frame\_difference()*
  - *Threshold\_frame\_difference()*
  - *Noise\_removal()*
  - ***Update\_background\_model()***
- *end*

Как обновить модель фона для случая одной гауссианы?

# Обновление параметров модели

---



Одноканальное изображение (серое)

- Обновление матожидания

$$\mu_{t+1} = \alpha \mu_t + (1 - \alpha) x_{t+1}$$

- Обновление дисперсии

$$\sigma_{t+1}^2 = \alpha(\sigma_{t+1}^2 + (\mu_{t+1} - \mu_t)^2) + (1 - \alpha)(x_{t+1} - \mu_{t+1})^2$$

$\alpha$  – скорость обновления (обучения)

# Многоканальное изображение

---

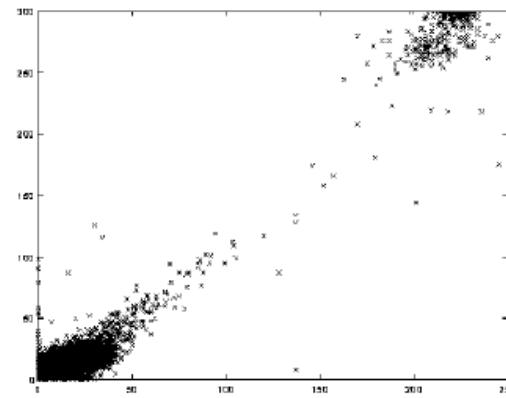


- Можно рассчитать полную матрицу ковариации К
- Обновлять полную матрицу сложно
- Обновление только среднего, с сохранением ковариации

$$\mu_{t+1} = \alpha \mu_t + (1 - \alpha) x_{t+1}$$



# Более сложные случаи



Распределение  
интенсивности красного  
и зеленого каналов

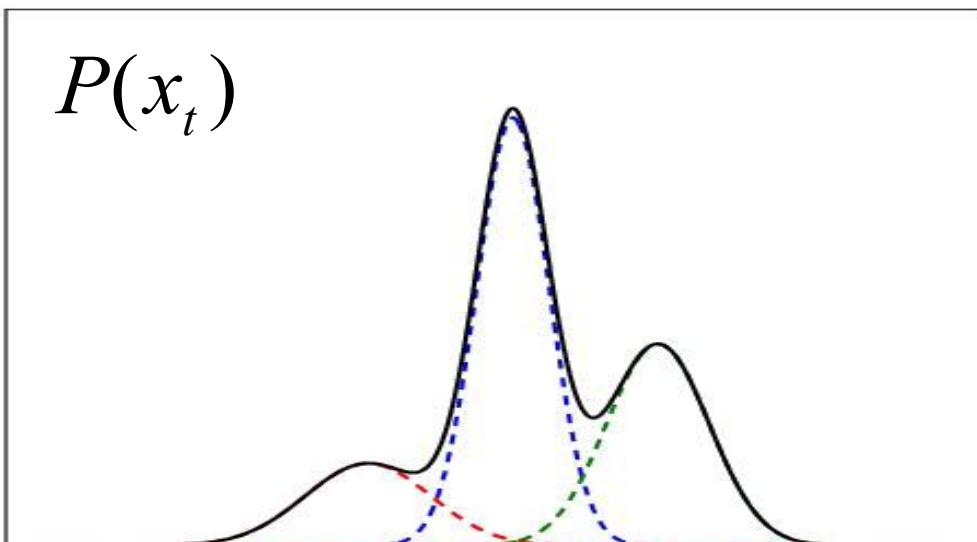
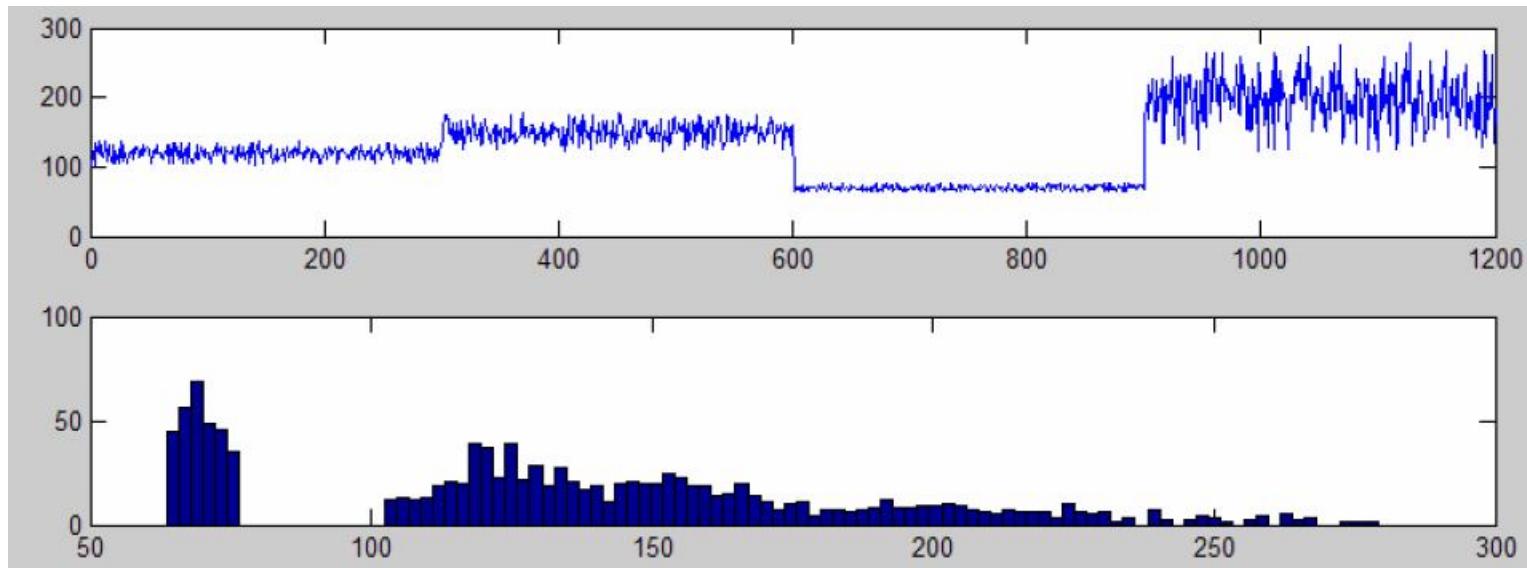


Как моделировать фон в  
таких, более сложных,  
случаях?

Стационарные объекты  
(машины)



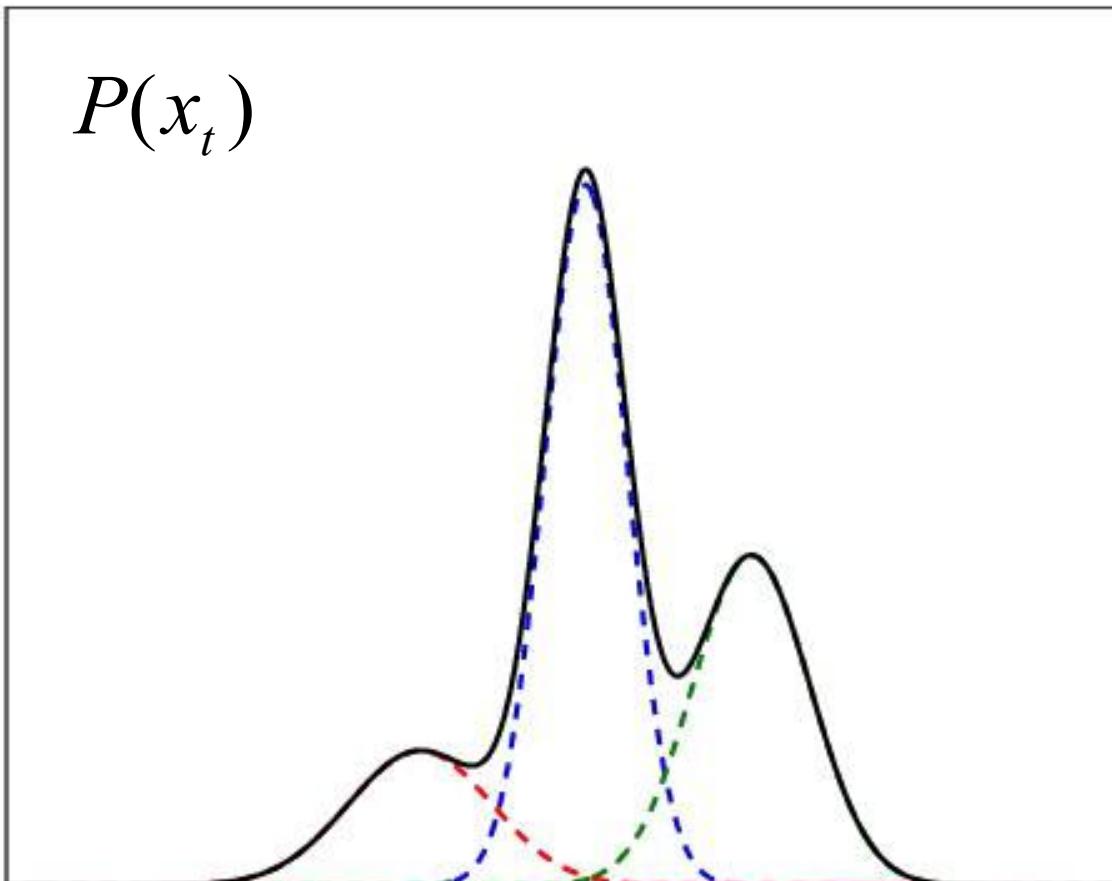
# Более сложные модели



«Многомодальное распределение» - значения могут группироваться в несколько кластеров



# Смесь Гауссиан



$$P(x_t) = \sum_{i=1}^K \omega_{i,t} N(x_t, \mu_{i,t}, \sigma_{i,t})$$

Мы полагаем, что  
сигнал порождён  
несколькими  
независимыми  
моделями

Пример №1:

- Вода
- Блик на воде

Пример №2:

- Небо
- Листья на дереве



# Смесь Гауссиан

---

- Плотность вероятности яркости пикселя  $\mathbf{x}$  при модели смеси из  $K$  нормальных распределений

$$P(x_t) = \sum_{i=1}^K \omega_{i,t} N(x_t, \mu_{i,t}, \sigma_{i,t})$$

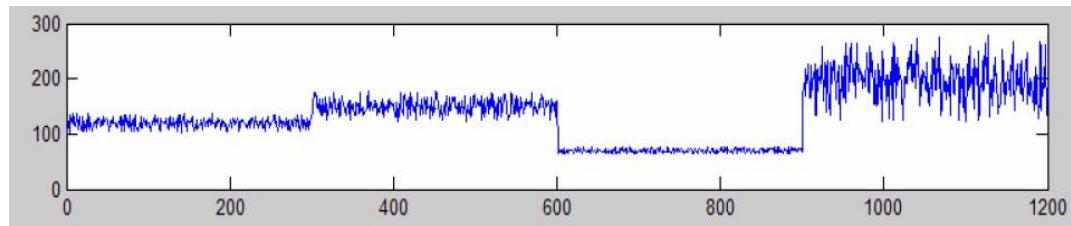
$\omega_{i,t}$  - вес компоненты  $i$  в момент  $t$

- Вычисление смеси требует сложного EM (Expectation-Maximization) алгоритма (не в реальном времени), поэтому был предложен приближенный алгоритм

# Обучение смеси на лету



- Пусть  $N$  – количество компонент в смеси
  - Инициализируем 1ую компоненту по первому изображению, вес = 1, вес остальных – 0
  - Сравниваем пиксель с каждой компонентой, пока не найдем совпадения
  - Обновляем матожидание и дисперсию совпавшей компоненты
  - Если совпадения не найдено, то заменяем компоненту с наименьшим весом
  - Обновляем веса



$$\sum_{i=1}^K \omega_{i,t} N(x_t, \mu_{i,t}, \sigma_{i,t})$$



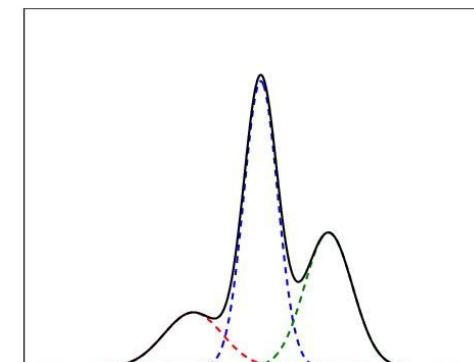
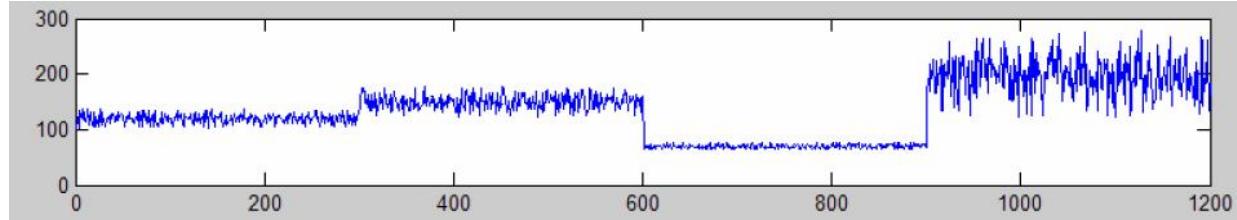
# Обновление весов

Обновление весов:  $P(x_t) = \sum_{i=1}^K \omega_{i,t} N(x_t, \mu_{i,t}, \sigma_{i,t})$

$$\omega_{i,t} = (1 - a)\omega_{i,t-1} + aM_{i,t}$$

$M_{i,t} = 1$ , если интенсивность пикселя  
удовлетворяет  $i$ -ой компоненте

После обновления всех весов, они нормализуются



# Моделирование фона

---



- Упорядочим все компоненты по критерию  $\omega_i / \sigma$ 
  - Чем «стабильнее» компонента, тем выше
  - Чем больше вес (чаще встречается, тем тоже выше)
- Определим порог  $T$  - какая доля выборки для каждого пикселя должна соответствовать фону
- Тогда фон для каждого пикселя:

$$B = \arg \min_b \left( \sum_{i=1}^b \omega_i > T \right)$$

- Все остальные компоненты - объекты

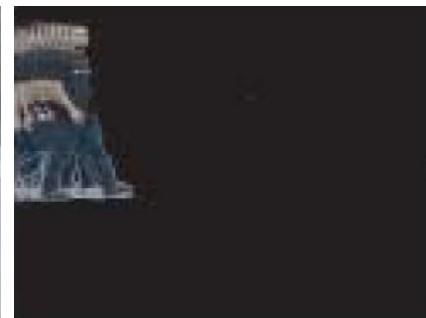
# Иллюстрация работы



(1)



(2)



(3)

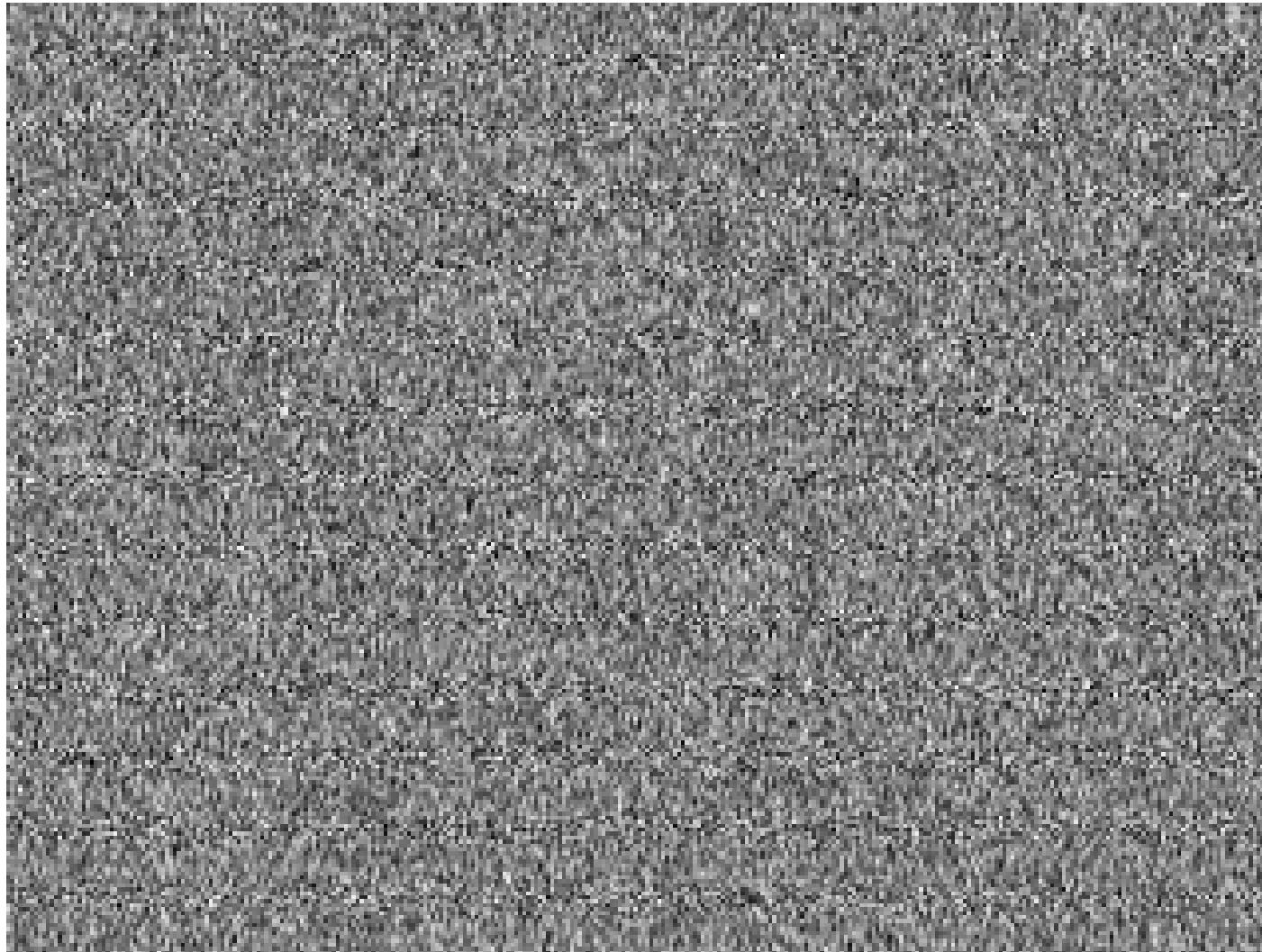


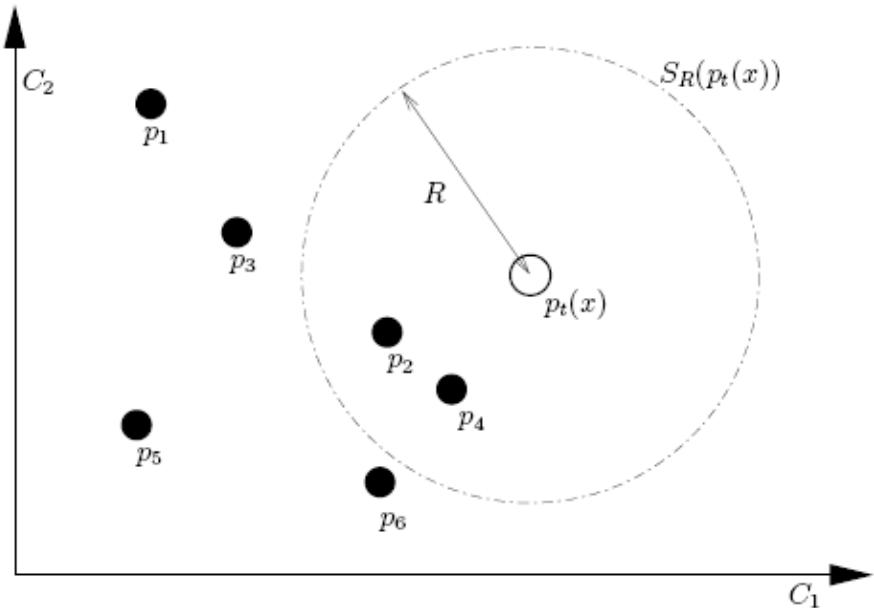
(4)

1. Текущий кадр видео (момент  $t$ )
2. Старшая гауссиана для каждого пикселя (модель фона)
3. Вторая гауссиана для каждого пикселя (модель движущегося объекта)
4. Маска переднего плана для текущего кадра

# Пример визуализации фона

---





- Модель фона состоит из  $N$  примеров фона ( $N=20$ )
- Новый пиксель считаем фоном, если он ближе порога к  $K$  примерам модели
- Модель обновляется случайно – новый пример фона с некоторой вероятностью заменяет один из примеров в модели

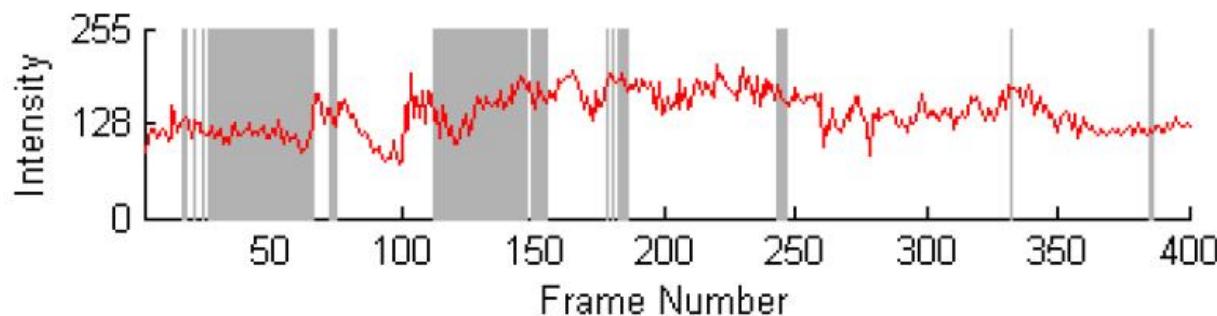


# Ещё посложнее пример

Камера наблюдения в лесу у кормушки для птиц



График яркости одного из пикселов



Как сработают алгоритмы и почему?

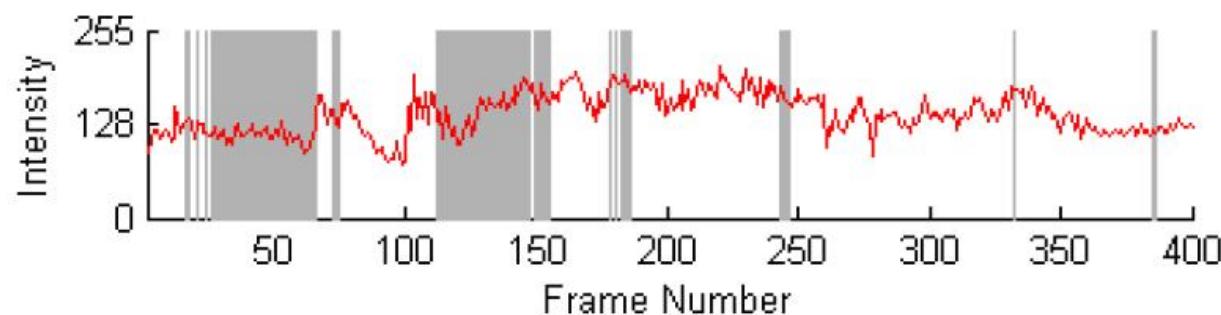
# Как быть?



Будем анализировать окрестность каждого пикселя



# Статистики по окрестностям



[Ko, T.\[Teresa\]](#), [Soatto, S.\[Stefano\]](#), [Estrin, D.\[Deborah\]](#), Background Subtraction on Distributions, *ECCV 2008*



# Схема алгоритма

- Посчитаем статистику (гистограмму) по 3D окрестности пикселя (фон):

$$p_{ij}(x) = \frac{1}{|S|} \sum_{s \in S} \delta(s - x)$$

$$S = \{x_t(a, b) \mid |a - i| < c, |b - j| < c, 0 \leq t < T\}$$

- Посчитаем статистику по 2д окрестности пикселя на текущем кадре:

$$q_{ij,\tau}(x) = \frac{1}{|S_\tau|} \sum_{s \in S_\tau} \delta(s - x)$$

$$S_\tau = \{x_\tau(a, b) \mid |a - i| < c, |b - j| < c\}$$

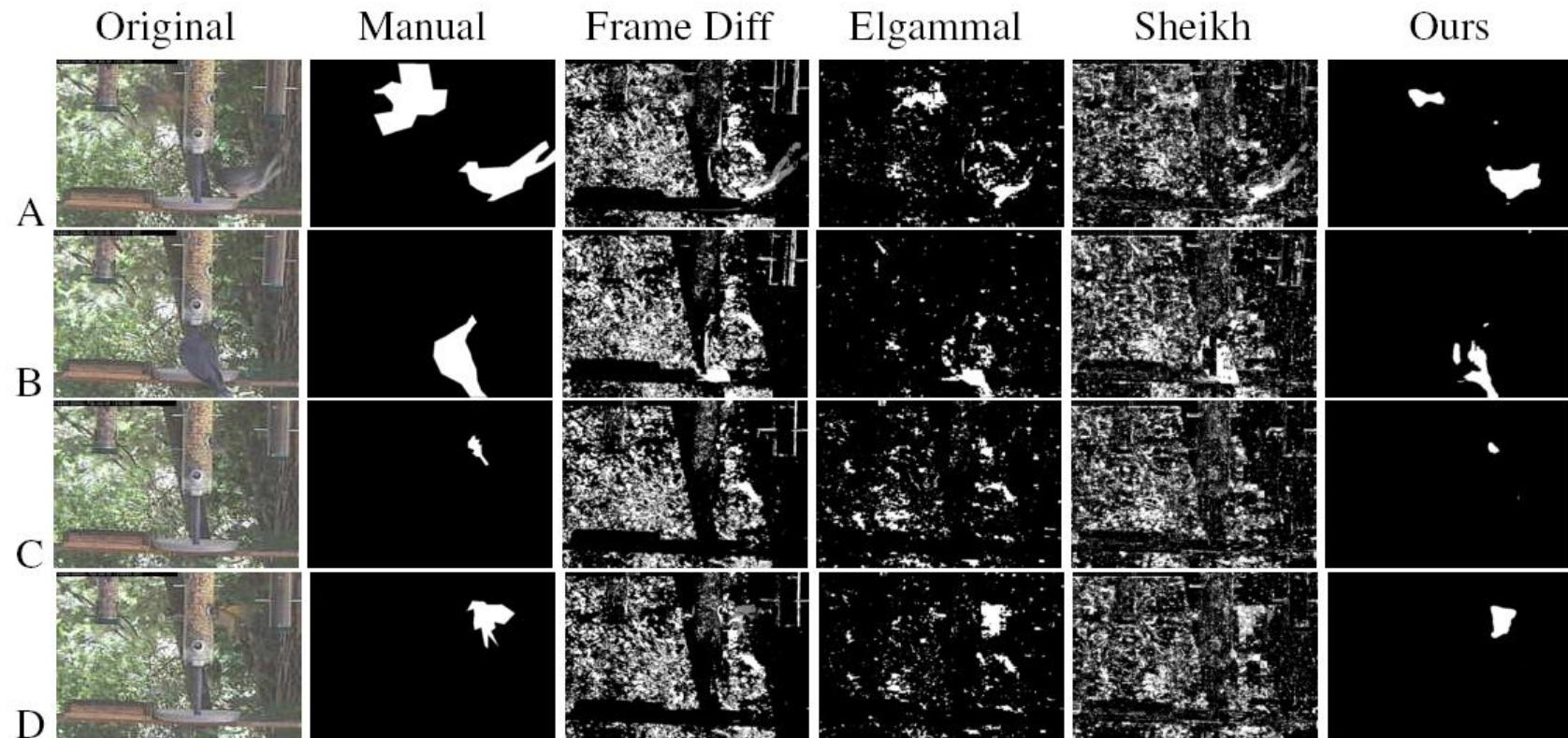
- Вычислим разницу:

$$d = \int_X \sqrt{p_{ij,\tau-1}(x)q_{ij,\tau}(x)} dx$$

- Обновление модели:

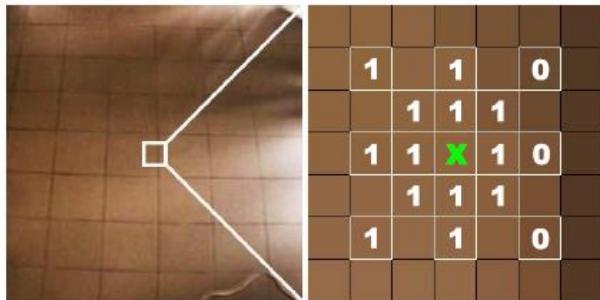
$$p_{ij,t}(x) = (1 - \alpha)p_{ij,\tau-1}(x) + \alpha q_{ij,\tau}(x)$$

# Результаты работы





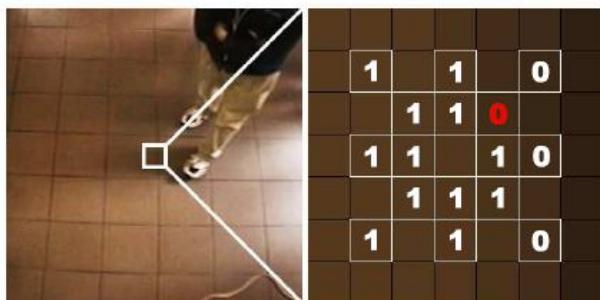
# Local Binary Similarity Patterns



binary string:  
[1, 1, 0, ... 1, 1, 0]

used as  
background  
reference

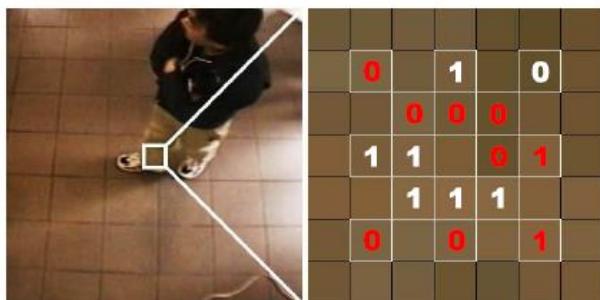
(a) background, intra-LBSP



binary string:  
[1, 1, 0, ... 1, 1, 0]

15/16 matches  
=  
“background”

(b) soft shadow, inter-LBSP



binary string:  
[0, 1, 0, ... 0, 0, 1]

7/16 matches  
=  
“foreground”

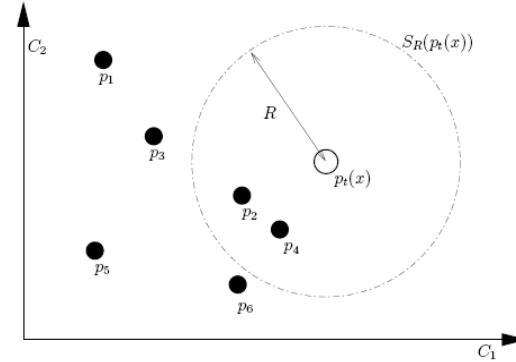
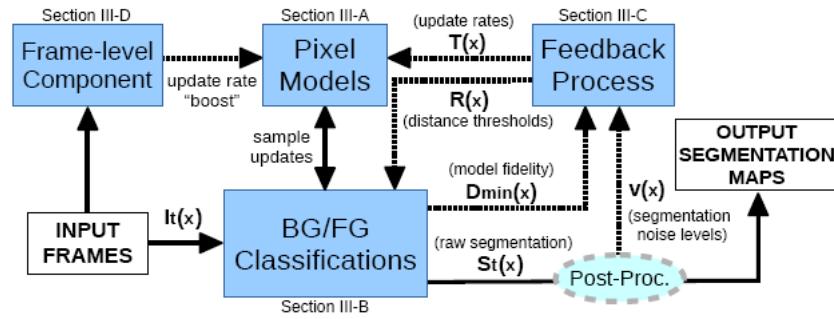
(c) foreground, inter-LBSP

O	O	O	O	O
O	O	O	O	O
O	O	X	O	O
O	O	O	O	O
O		O		O

- Текстурный шаблон на основе 16 сравнений яркости
- Порог вычисляем как  $a^*$  яркость пикселя (адаптивный)
- 16 битный код

St-Charles, P.-L., Bilodeau, G.-A., Bergevin, R., "SuBSENSE : A Universal Change Detection Method with Local Adaptive Sensitivity". Accepted for IEEE Transactions on Image Processing, Nov. 2014

# SuBSENSE



- Цвет и текстура
  - Храним цвет (RGB) и текстуру (LBSP)
  - Если цвета похожи, тогда сравниваем по текстуре.
  - Считаем фоном, только если по обоим параметрам похоже
- ViBe схема
  - Храним N примеров фонов в каждом пикселе
  - Заменяем каждый с вероятностью 1/T на новый фоновый
  - Заменяем одного из соседей с вероятностью
- Адаптивная настройка параметров

# Развитие методов

---



- Другие признаки для моделирования фона
- Более сложные модели модели фона
- Обучение специальных классификаторов фон / объект
- Подавление теней и отражений
- Обработка внештатных ситуаций (резкое изменение освещения)
- Иерархические методы для ускорения вычислений
- Ускорение вычислений (параллельные реализации, GPU и т.д.)

# Резюме вычитания фона

---



- Вычитание фона – очень эффективный инструмент выделения объектов при условии стационарности камеры
- Почти вся видеоаналитика в современных системах основана на вычитании фона
- Есть множество методов вычитания фона, четыре из которых мы рассмотрели
- Точное вычитание фона в реальном времени для HD разрешений сделать крайне сложно