

ВИДЕО СЛЕЖЕНИЕ

Лекция 10.

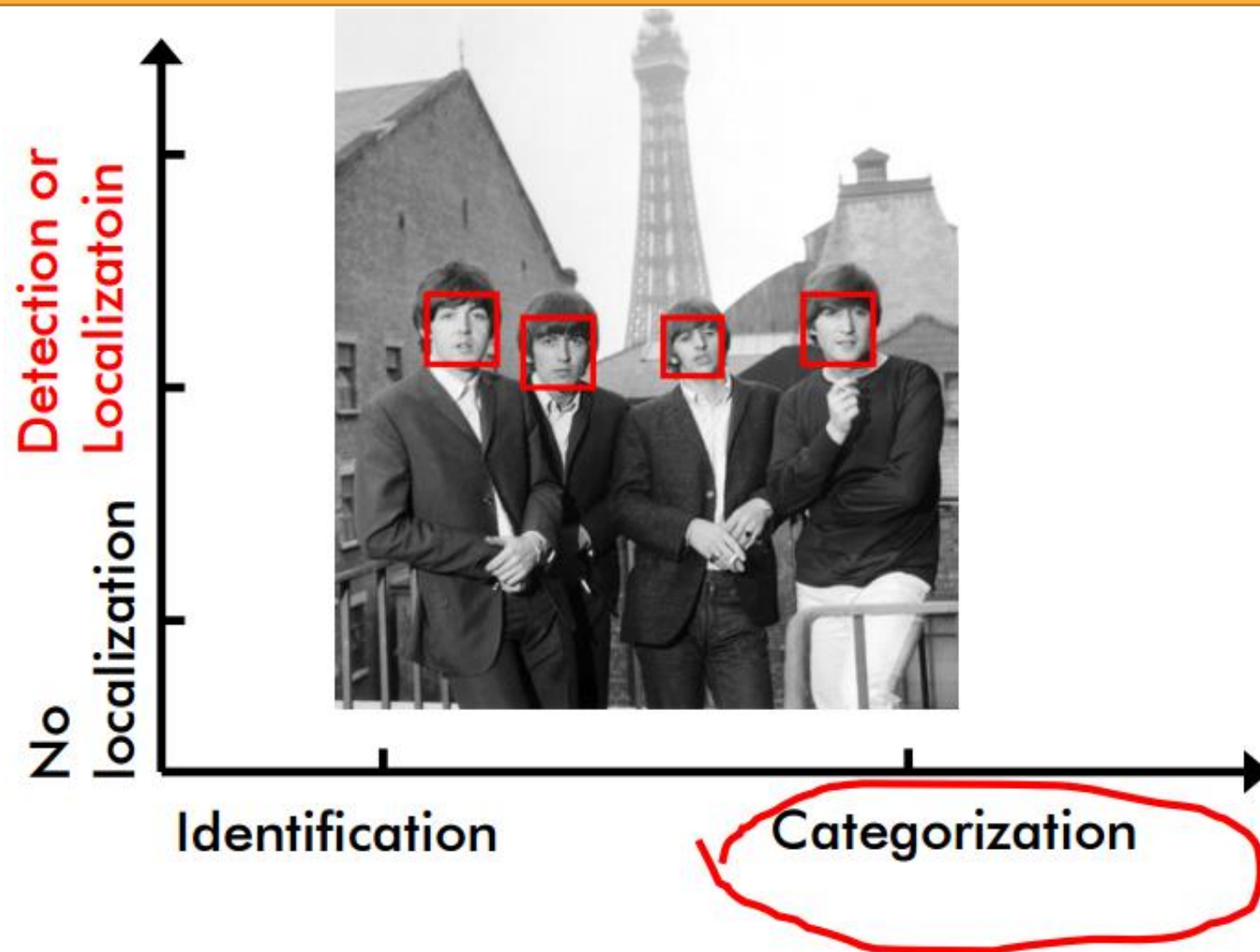
Преподаватель: Сибирцева Елена
elsibirtseva@gmail.com

Домашнее задание

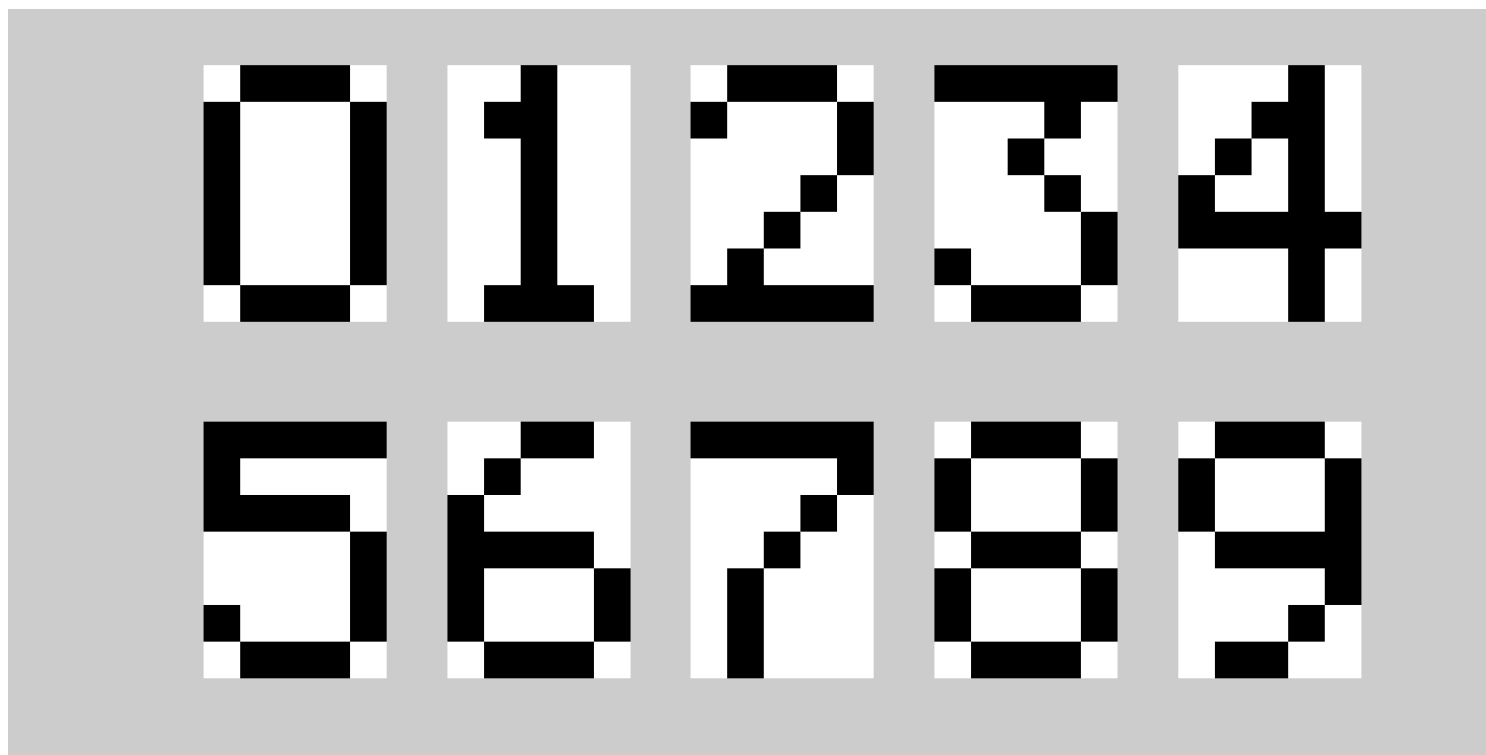
- Дедлайн **5 мая**
- Есть 4 варианта на выбор:
- 1. С помощью OpenCV реализовать распознавание и трекинг пешехода (их может быть несколько). Но такое уже есть в сорсах. Поэтому, чтобы не было скучно, дополнительно к трекингу рассчитывать расстояние до целей.
- 2. Вырезать человека из фона и подставить свою картинку.
- 3*. Распознать на видео взрыв (желательно вертолета)
- 4*. Распознавать кошачьи морды на фото и видео.

В предыдущих сериях...

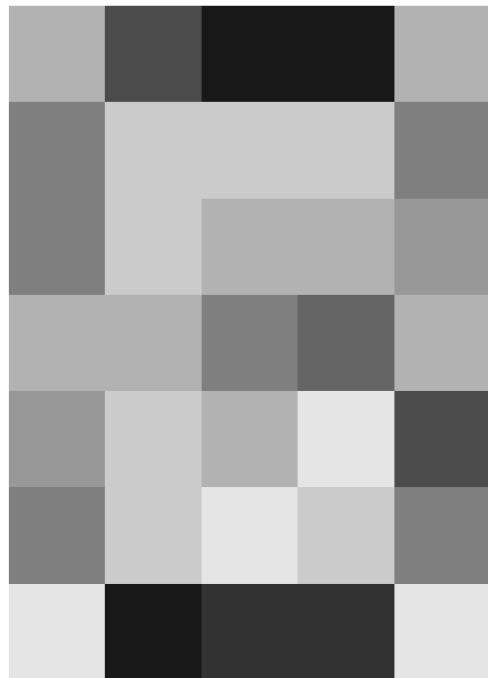
Что такое “распознавание”



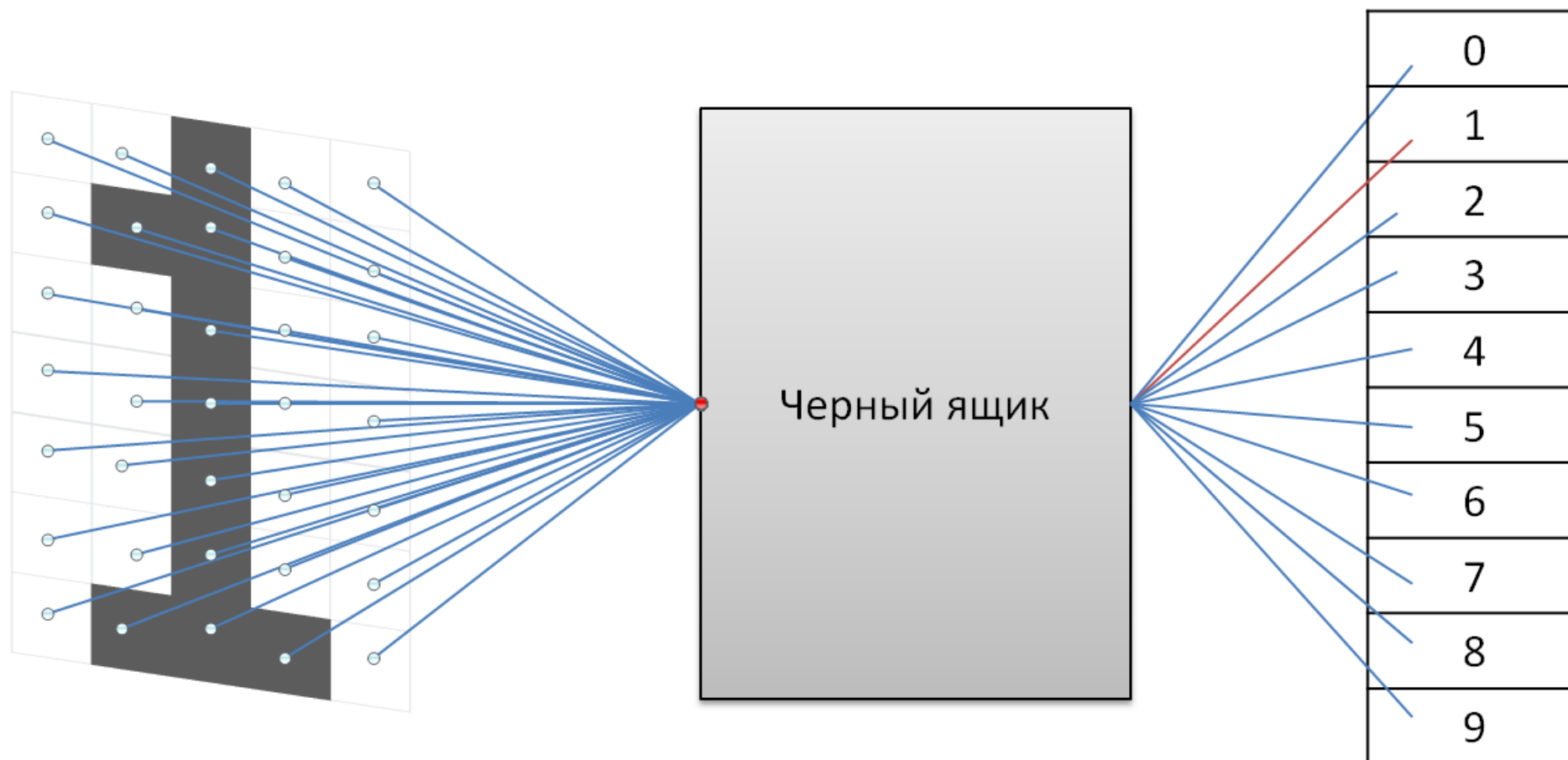
Easy as pie



Средняя цифра



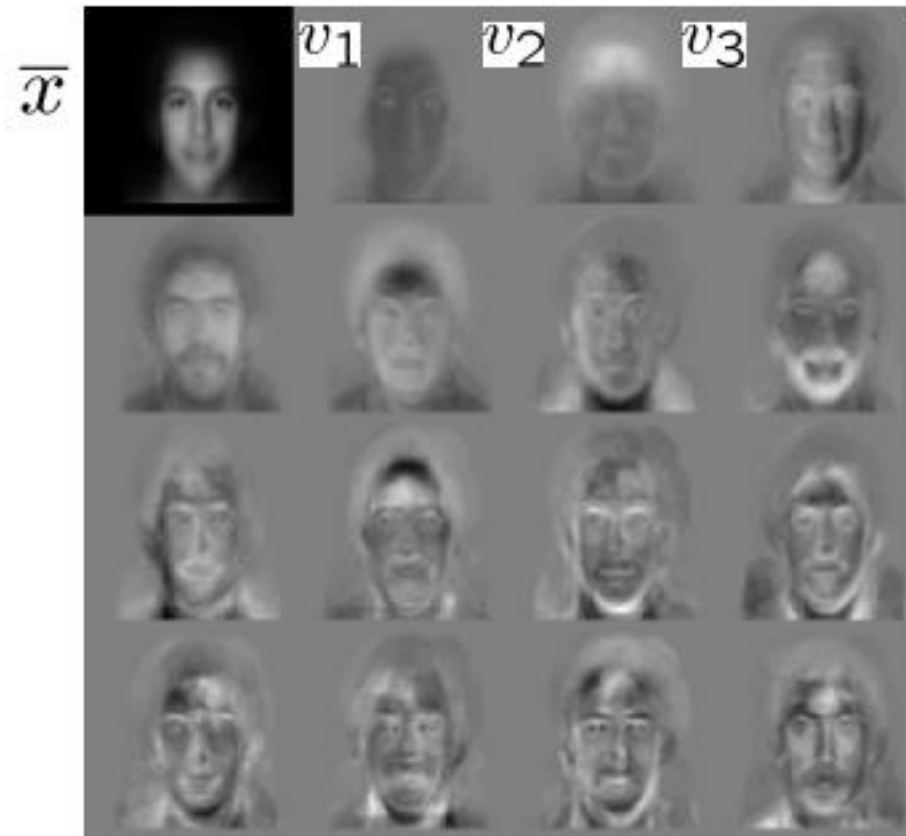
0.3	0.7	0.9	0.9	0.3
0.5	0.2	0.2	0.2	0.5
0.5	0.2	0.3	0.3	0.4
0.3	0.3	0.5	0.6	0.3
0.4	0.2	0.3	0.1	0.7
0.5	0.2	0.1	0.2	0.5
0.1	0.9	0.8	0.8	0.1



○ Вместо 35 сигналов подается только 4. Ура.

Eigenfaces

- PCA extracts the eigenvectors of S_T
 - Gives a set of vectors v_1, v_2, v_3, \dots
 - Each one of these vectors is a direction in face space:



Algorithm

Training

1. Align training images x_1, x_2, \dots, x_N



Note that each image is formulated into a long vector!

2. Compute average face $u = 1/N \sum x_i$



3. Compute the difference image $\varphi_i = x_i - u$

Algorithm

4. Compute the covariance matrix (total scatter matrix)

$$S_T = (1/N) \sum \varphi_i \varphi_i^T = BB^T, \quad B = [\varphi_1, \varphi_2 \dots \varphi_N]$$

5. Compute the eigenvectors of the covariance matrix, W

Testing

1. Projection in Eigenface

$$\text{Projection } \omega_i = W (X - u), \quad W = \{\text{eigenfaces}\}$$

2. Compare projections

Illustration of Eigenfaces

- The visualization of eigenvectors:

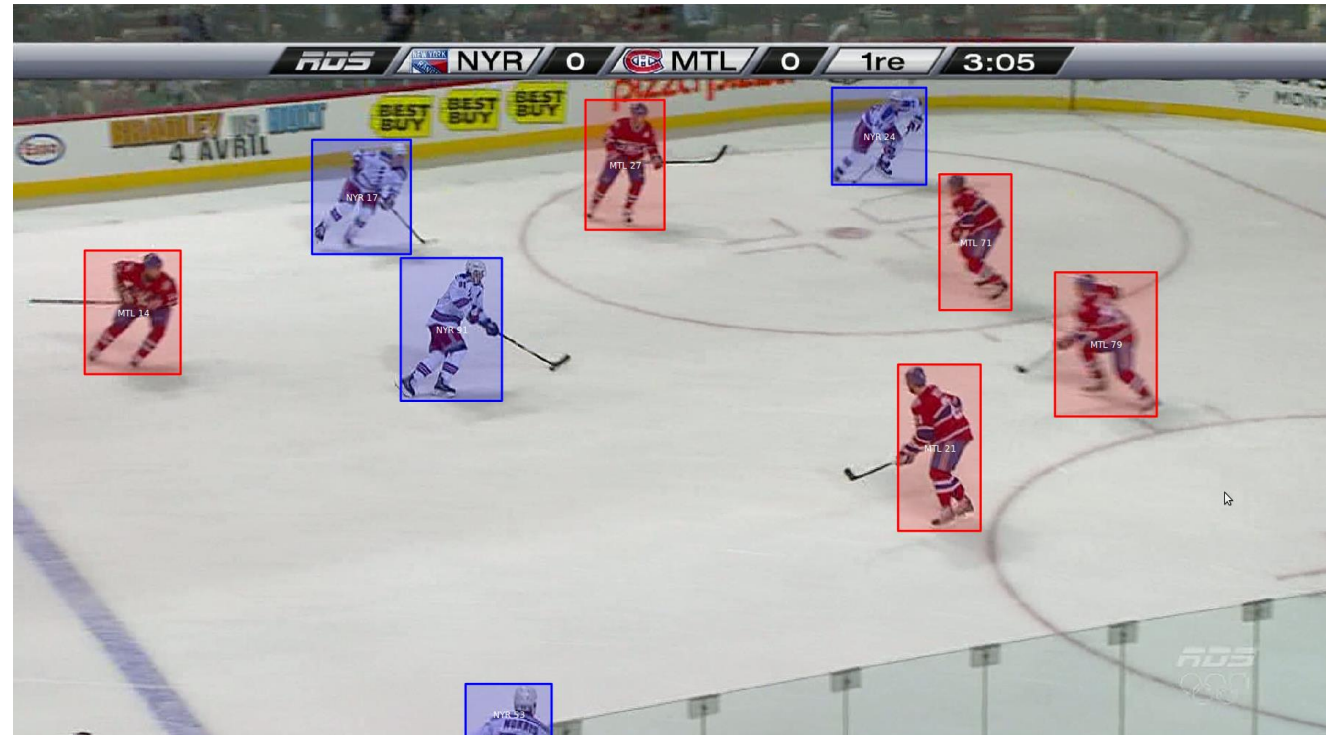


These are the first 4 eigenvectors from a training set of 400 images (ORL Face Database). They look like faces, hence called Eigenface.

ВИДЕО СЛЕЖЕНИЕ

Основы слежения

- В отличие от изображения, на видео мы видим объект слежения во времени. Это дает дополнительную информацию о его форме, объеме, движениях и относительном положении.
- Основная задача: выделить объект и сопровождать его из кадра в кадр.
- Используется
 - Отслеживание объектов
 - Стабилизация видео
 - Генерация панорам
 - 3x-мерная реконструкция
 - Спецэффекты



**Существуют два варианта: либо
мы знаем, что ищем, либо мы
понятия не имеем.**

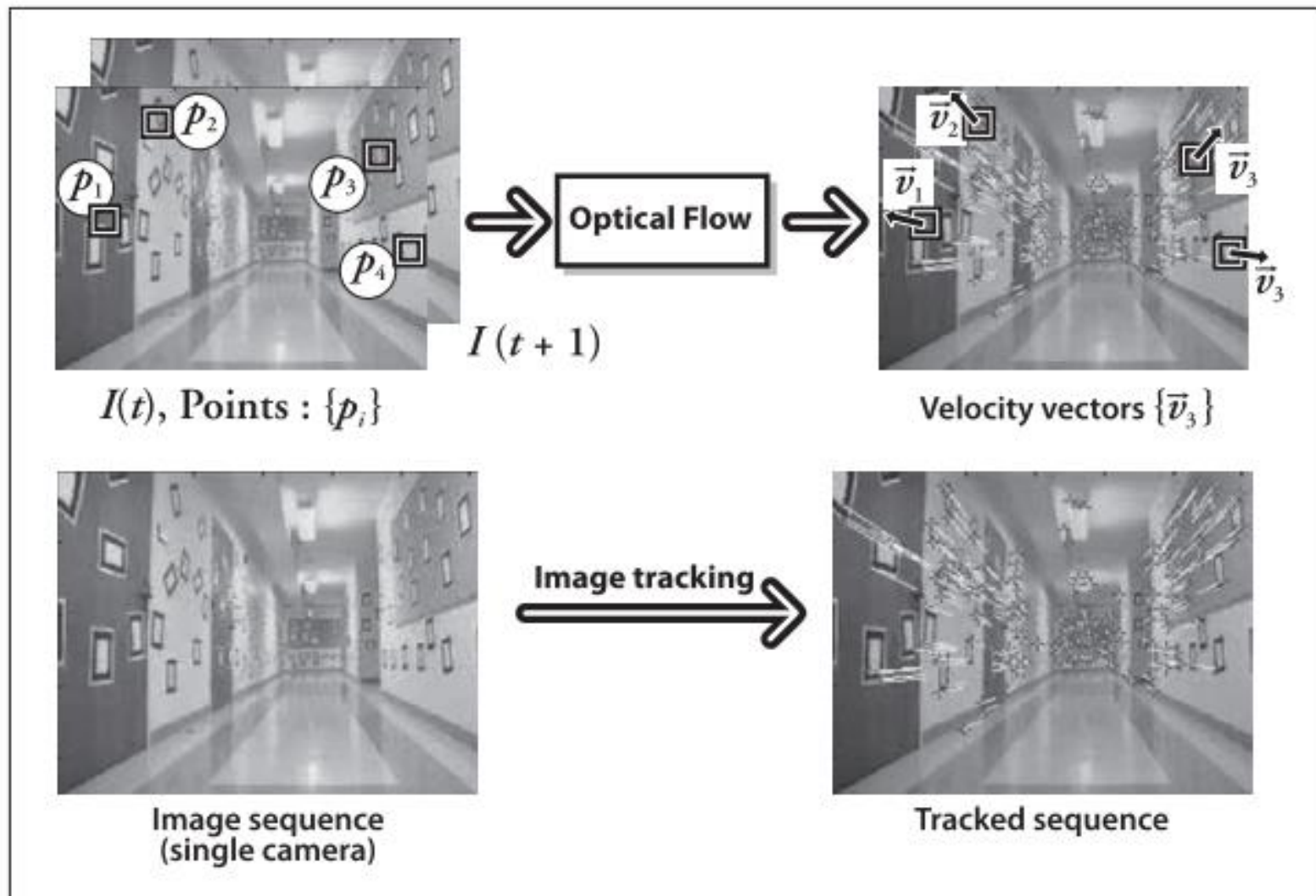
Вариант 1. Мы знаем.

- Постановка задачи: дана область на видео, за которой необходимо следить.
- Это может быть выделенная пользователем область, результат пробега другого алгоритма, к примеру EigenFaces. Либо мы знаем цвет объекта.

Вариант 2. Мы не знаем.

- Зачем это вообще нужно?
- Слежение не пойми за чем важно, когда мы пытаемся найти что-то интересное, основываясь на его движениях или когда именно движения – это объект нашего интереса.
- Обычно техники слежения на неопознанном объектом включают в себя поиск характеристических точек (каким алгоритмом?).

Оптический поток



Методы слежения

- Sparse optical flow – следим только за точками интереса
 - Lucas-Kanade
- Dense optical flow – следим за изменениями в каждом пикселе изображения
 - Horn-Schunck
 - MeanShift
 - CamShift



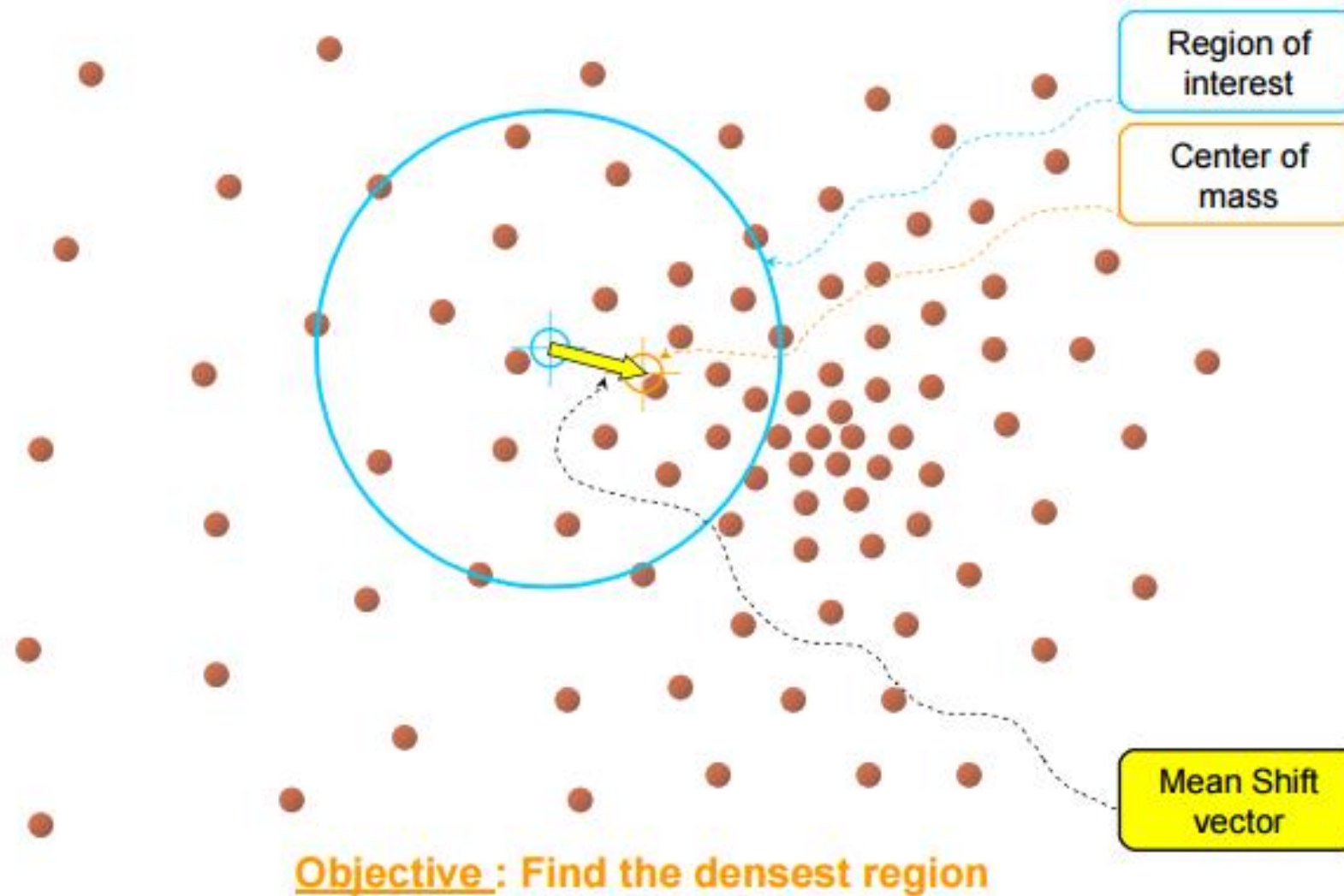
**Какой вариант больше
подходит для системы
родительского контроля,
блокирующей видео 18+?**

MEANSHIFT

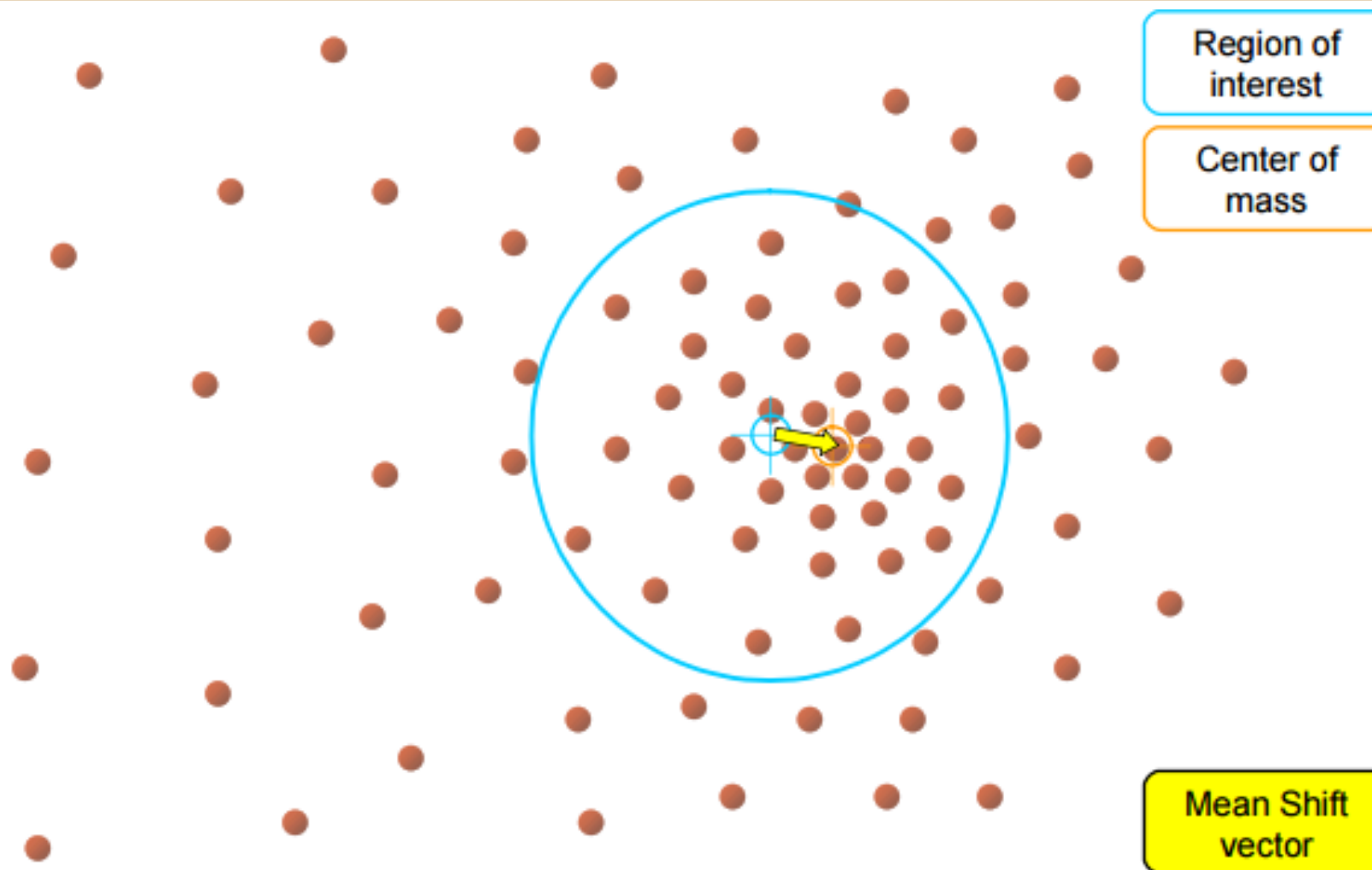
MeanShift

- MeanShift алгоритм – верный выбор, если объект трекинга может быть выделен из всего видеопотока по его цвету.
- Однако, MeanShift не ограничивается только цветом, также можно использовать ориентацию углов, текстуру и движение.
- Часто называют градиентным анализом или градиентным спуском.

MeanShift на пальцах

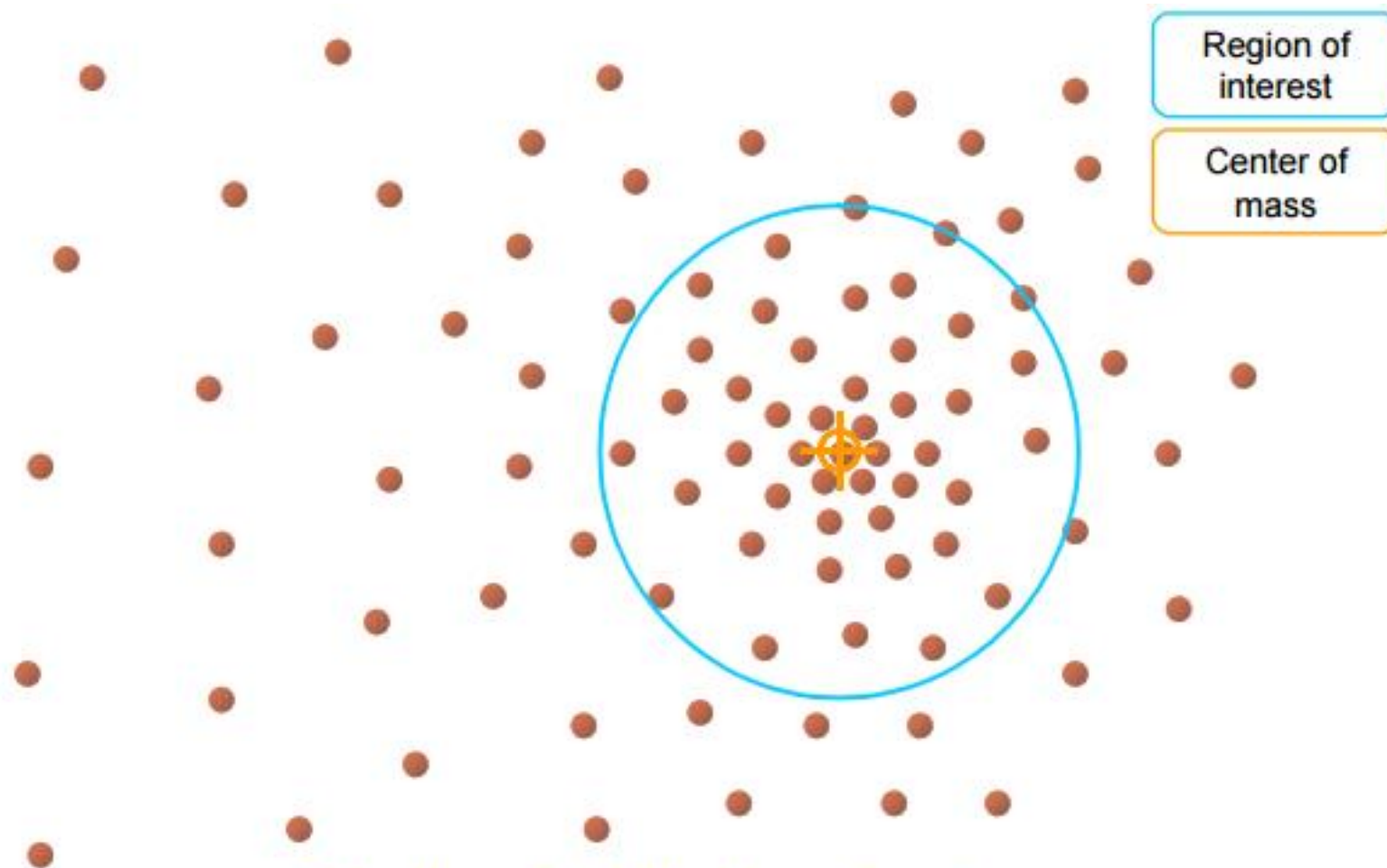


MeanShift на пальцах



Objective : Find the densest region

MeanShift на пальцах



Objective : Find the densest region

MeanShift простая математика

Compute zero moment:

$$M_{00} = \sum_x \sum_y I(x, y)$$

Compute 1st moment for (x,y)

$$M_{01} = \sum_x \sum_y xI(x, y) \quad M_{10} = \sum_x \sum_y yI(x, y)$$

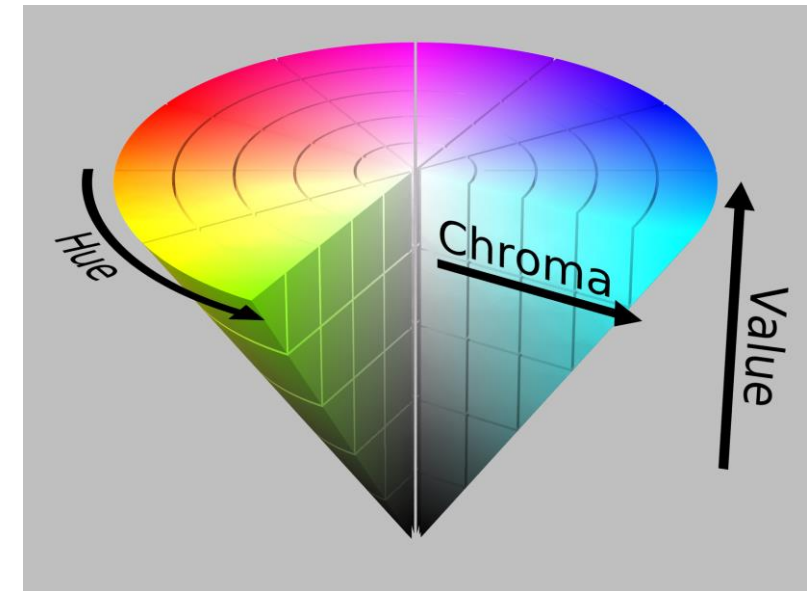
Compute the new center of the feature window of the of the next frame.

$$x_{centroid} = \frac{M_{01}}{M_{00}} \quad y_{centroid} = \frac{M_{10}}{M_{00}}$$

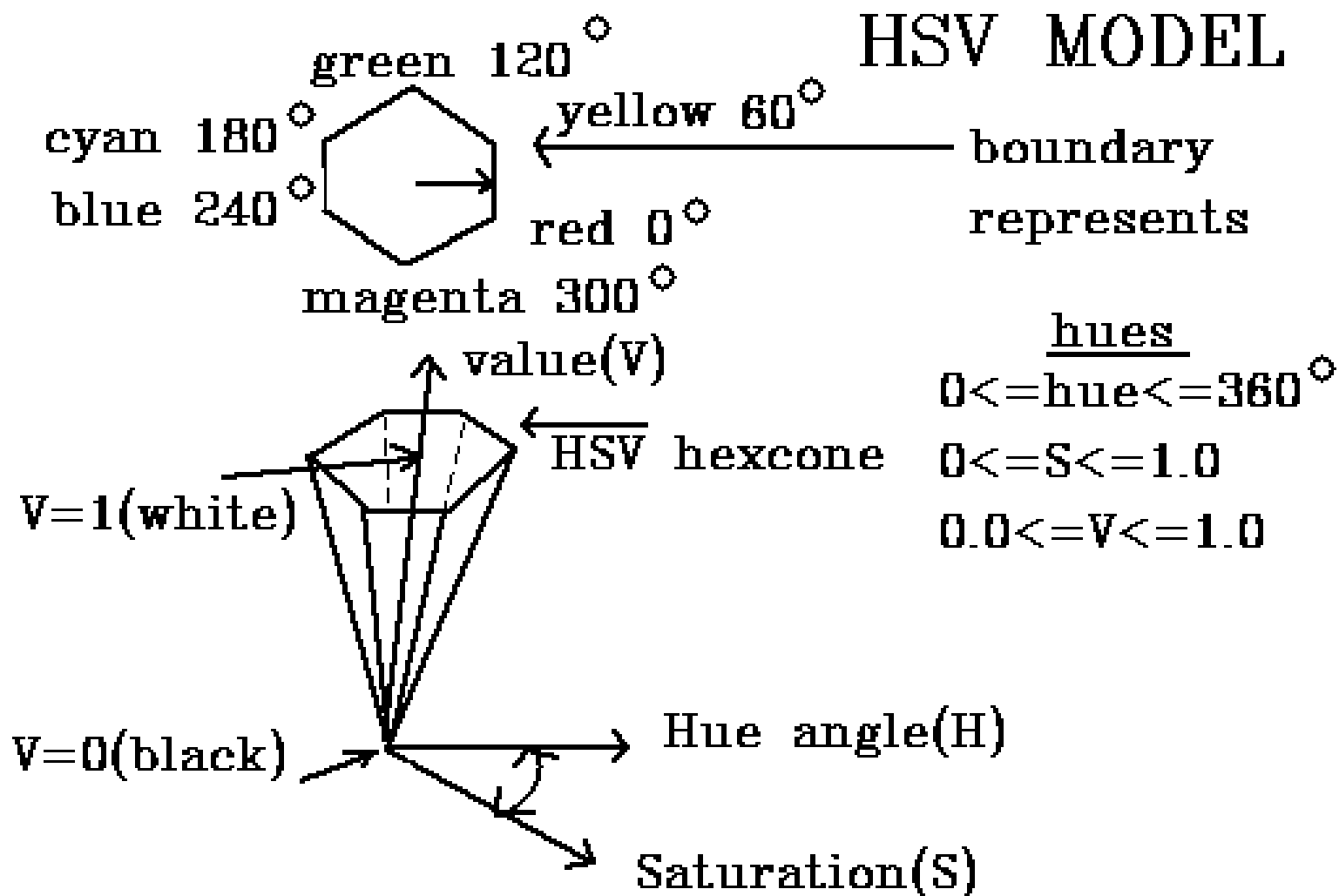
CAMSHIFT

CamShift

- CamShift – Continuously Adaptive (window) Mean Shift algorithm
- Также основан на трекинге по цвету (текстуре, ориентации углов и т.п.)
- Использует окно поиска, чтобы следить за движущимся объектом трекинга, игнорируя все, что находится вне окна поиска
- Изменяет масштаб окна поиска в зависимости от размера объекта, таким образом позволяет следить за объектом на разном расстоянии от камеры
- Переводит изображение из цветового пространства из RGB в HSV, которое менее чувствительно к изменению освещения
- Цветовая модель строится на основании гистограммы Hue (цветности), что позволяет избавиться от большого количества шума на изображении



HSV на всякий случай



CamShift простая математика

Compute zero moment:

$$M_{00} = \sum_x \sum_y I(x, y)$$

Compute 1st moment for (x,y)

$$M_{01} = \sum_x \sum_y xI(x, y) \quad M_{10} = \sum_x \sum_y yI(x, y)$$

Compute the new center of the feature window of the of the next frame.

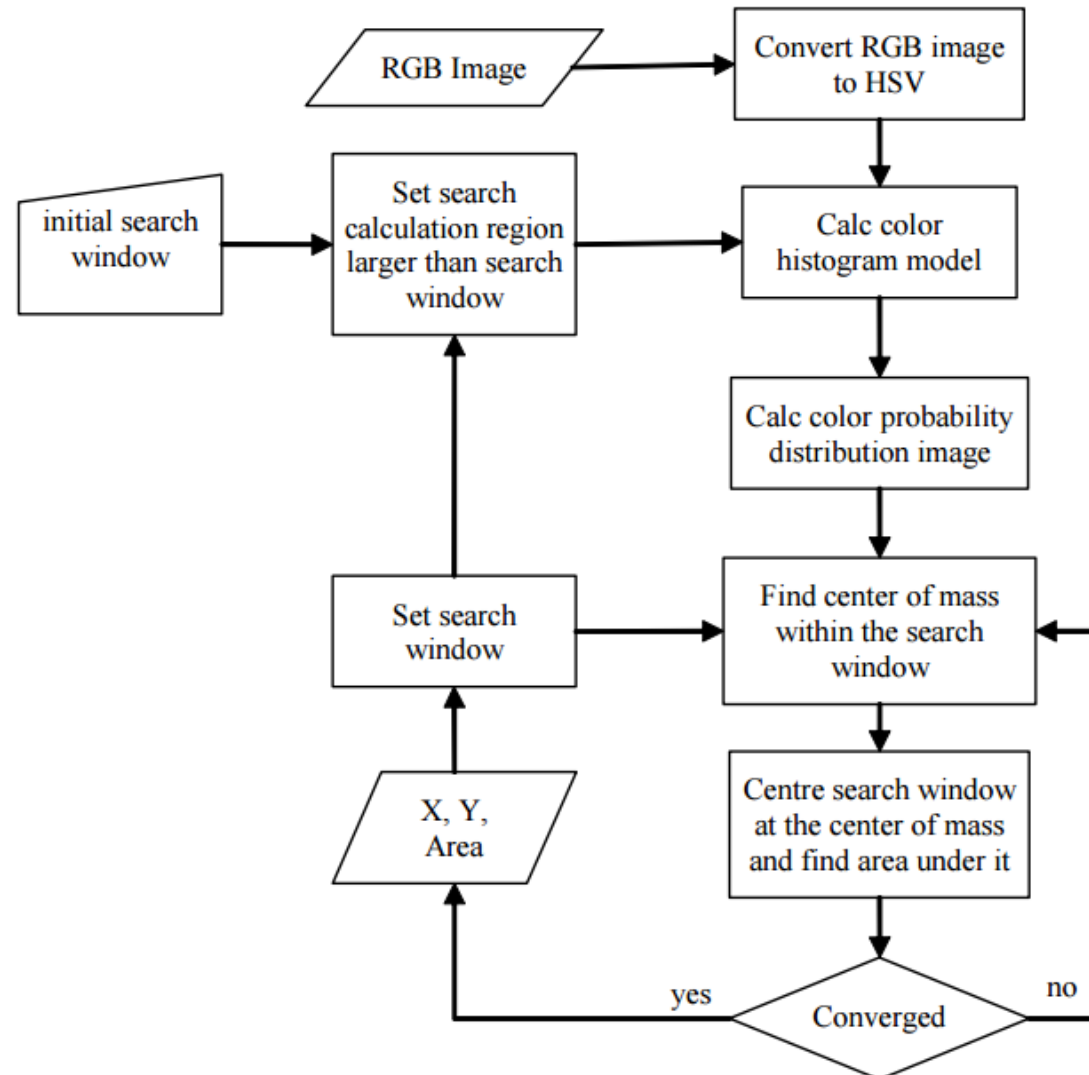
$$x_{centroid} = \frac{M_{01}}{M_{00}} \quad y_{centroid} = \frac{M_{10}}{M_{00}}$$

Calculate new window size

$$w = r_1 \sqrt{M_{00}}; \quad l = r_2 \sqrt{M_{00}}$$

CamShift алгоритм

- Каждый кадр переводится в изображение распределения цветовой вероятности, которая рассчитывается по цветовой модели гистограмм того объекта, за которым мы следим.
- Центр и размер объекта данного цвета ищется CAMSHIFT по тому же изображению распределения цветовой вероятности
- Текущие размер и позиция объекта запоминаются и используются для задания размера поискового окна на следующем кадре.
- Повторять, пока не надоеет.



CamShift проблемы

- Проблемы подстерегают в HSV пространстве

При низкой яркости (V около 0), насыщенность тоже не блещет (S около 0). Тогда цветность становится очень зашумленной, так как в такой маленькой шестигранной пирамидке содержится соответственно невеликое множество дискретных значений цветности, поэтому адекватно отобразить малейшие изменения, как в RGB, не могут.

Это приводит к значительным изменениям цветности и головной боли программиста

- Как с этим бороться:

Игнорировать цветность пикселей, которые имеют низкий и высокий уровень яркости

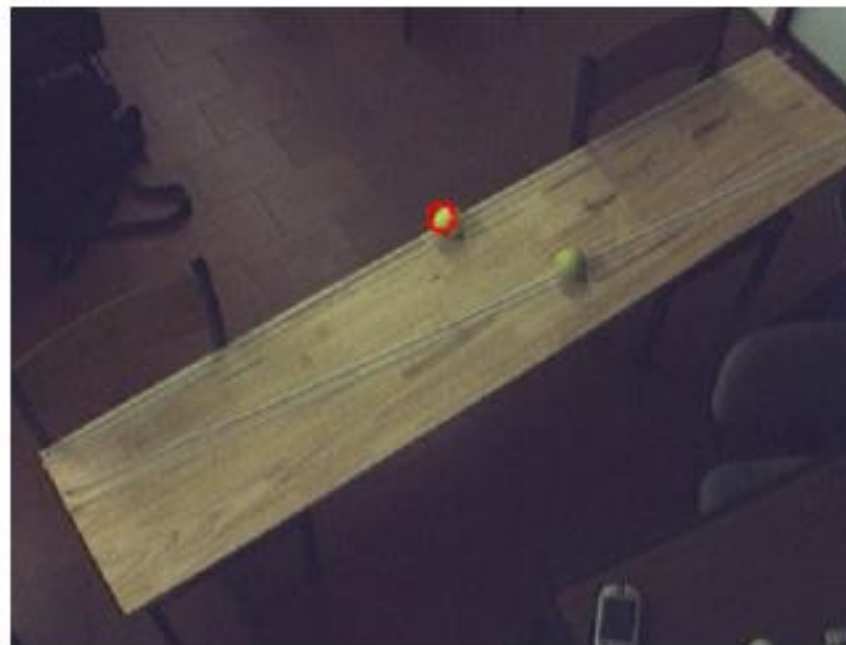
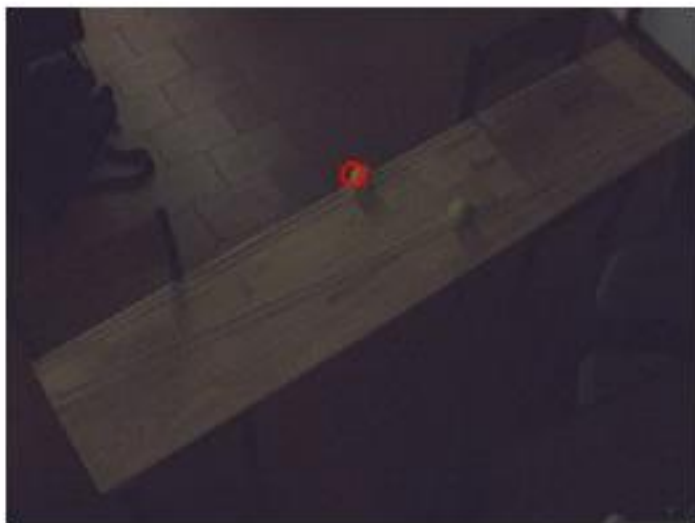
Слишком темное изображение для нас ничем не хуже засвеченного

При очень низкой насыщенности цветность так же неадекватна, так что их тоже стоит игнорировать

- Чтобы не перейти в тотальное игнорирование всего изображения, логично предположить, что необходимо сначала выровнять уровень яркости, а уже потом применять CamShift

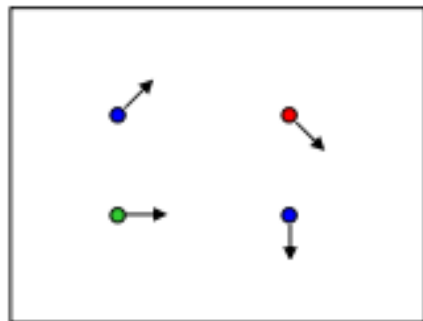
CamShift пример

- CAMSHIFT Algorithm
 - Illumination Drift
 - Presence of other object

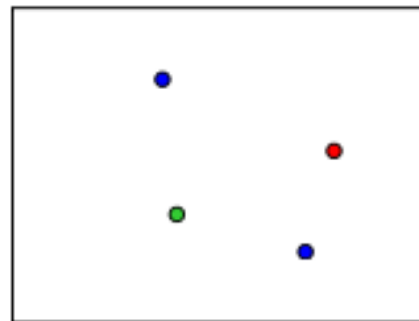


LUCAS-KANADE

Постановка задачи



$H(x, y)$



$I(x, y)$

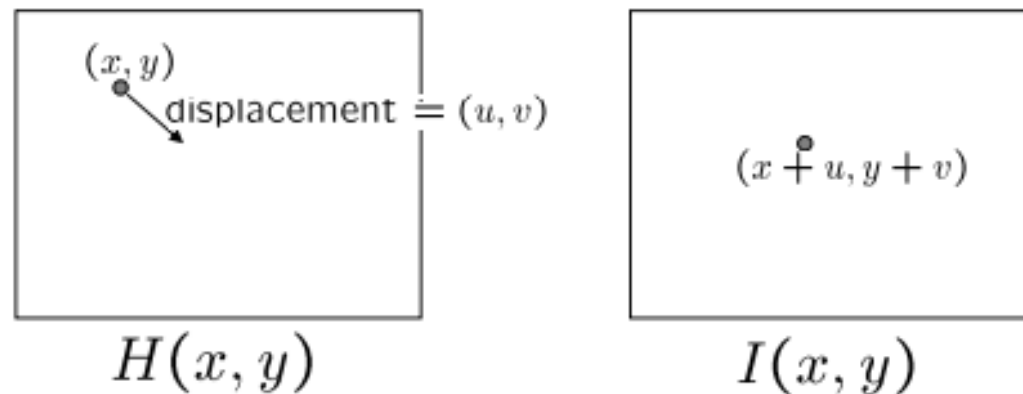
- Как оценить движение пикселей от H в изображение I ? Проблема соответствия пикселей!
 - Пусть дан пиксель H , найти **близкие** пиксели **того же цвета** в I

Ключевые предположения

- **Константный цвет**: точка в H выглядит также, как и в I
 - Для изображения в градациях серого, это постоянная яркость
- **Малое движение**: точки не уезжают далеко между кадрами

Эта задача называется поиск «оптического потока»

Ограничения



- Используем ограничения для формализации задачи

- Постоянная яркость
- Малое смещение: (u и v меньше 1-го пикселя)
 - Разложим функцию картинки в ряд тейлора I:

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

Уравнение оптического потока

- Объединим два ограничения

$$0 = I(x + u, y + v) - H(x, y)$$

$$\approx I(x, y) + I_x u + I_y v - H(x, y)$$

$$\approx (I(x, y) - H(x, y)) + I_x u + I_y v$$

$$\approx I_t + I_x u + I_y v$$

$$\approx I_t + \nabla I \cdot [u \ v]$$

$$I_x = \frac{\partial I}{\partial x}$$

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

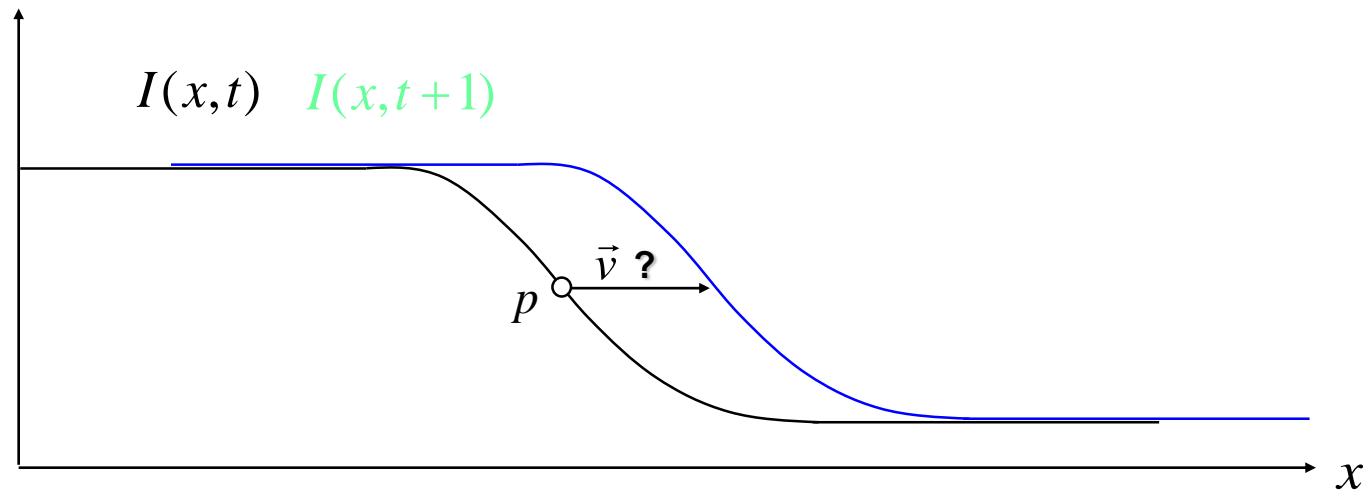
$$0 = I_t + \nabla I \cdot \left[\frac{\partial x}{\partial t} \ \frac{\partial y}{\partial t} \right]$$

Уравнение оптического потока

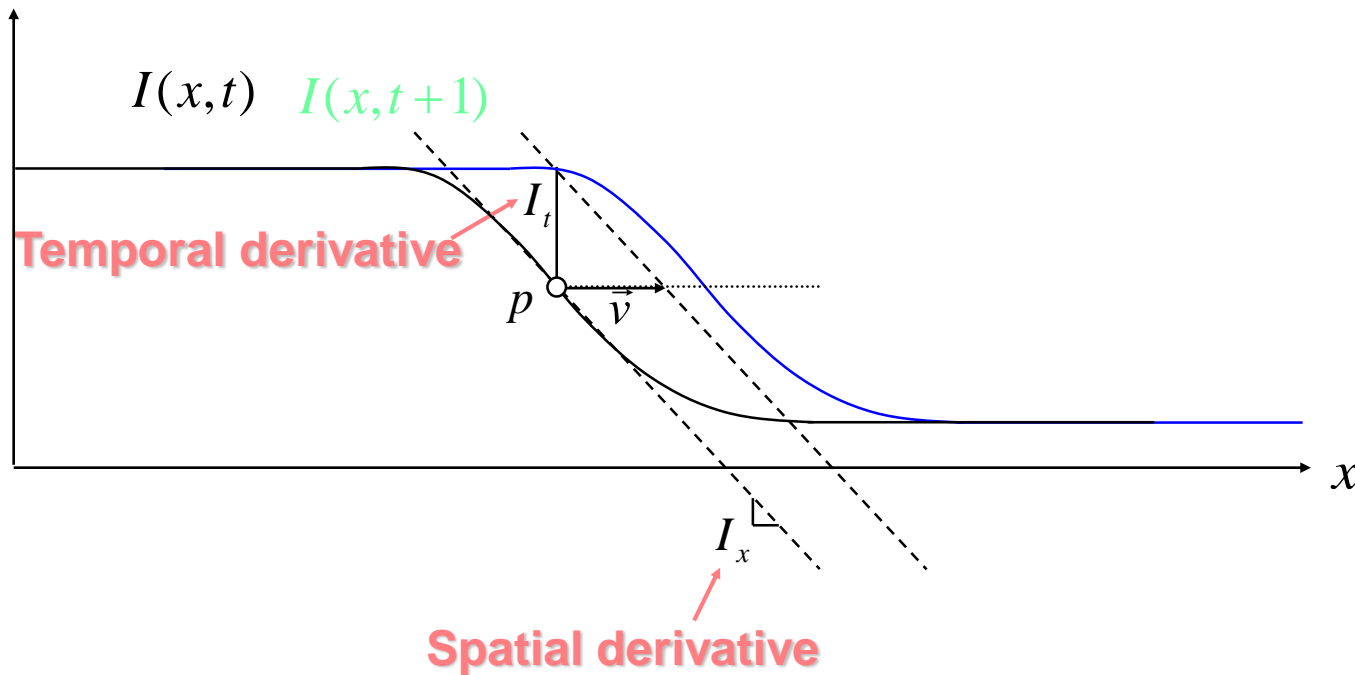
$$0 = I_t + \nabla I \cdot [u \ v]$$

- Вопрос: сколько неизвестных и уравнений для каждого пикселя?

Tracking in the 1D case:



Tracking in the 1D case:



$$I_x = \left. \frac{\partial I}{\partial x} \right|_t$$

$$I_t = \left. \frac{\partial I}{\partial t} \right|_{x=p}$$



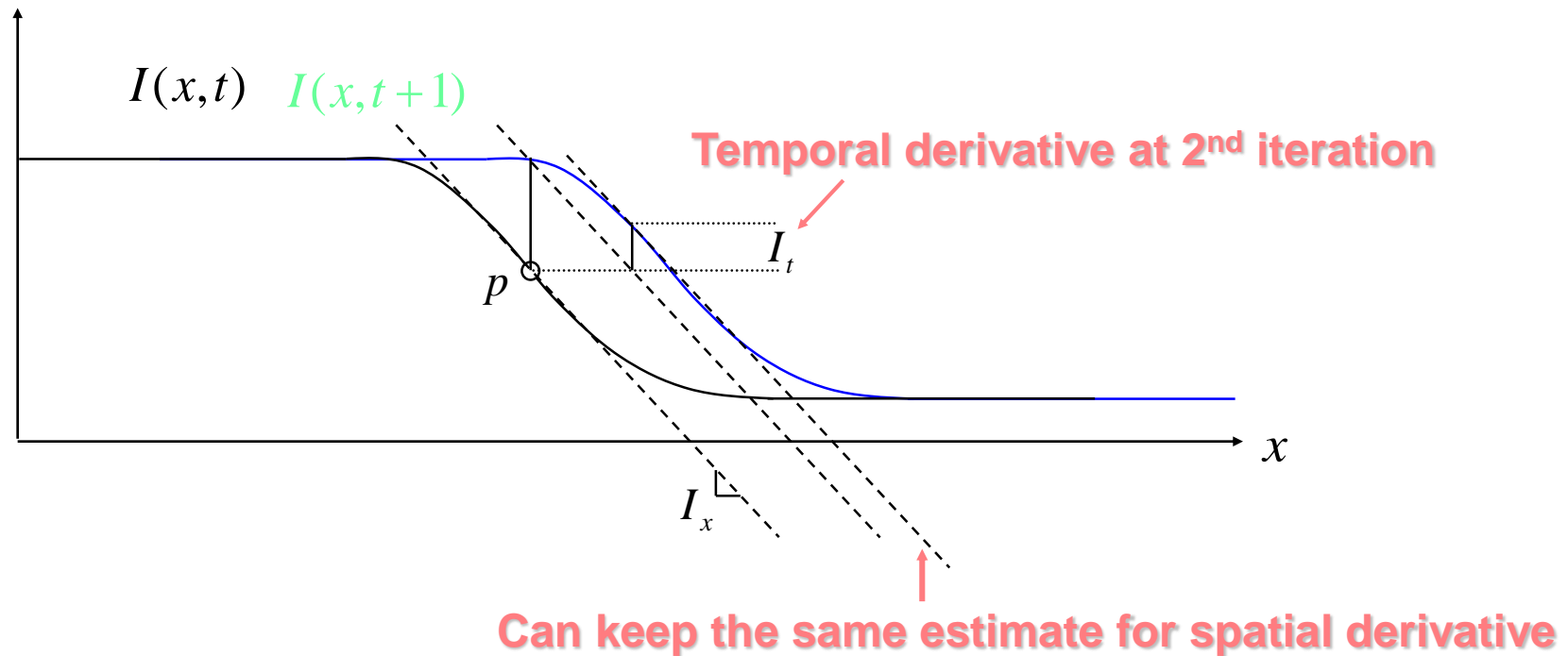
$$\vec{v} \approx -\frac{I_t}{I_x}$$

Assumptions:

- Brightness constancy
- Small motion

Tracking in the 1D case:

Iterating helps refining the velocity vector




$$\vec{v} \leftarrow \vec{v}_{previous} - \frac{I_t}{I_x}$$

Converges in about 5 iterations

From 1D to 2D tracking

$$\text{1D: } \frac{\partial I}{\partial x} \bigg|_t \left(\frac{\partial x}{\partial t} \right) + \frac{\partial I}{\partial t} \bigg|_{x(t)} = 0$$

$$\text{2D: } \frac{\partial I}{\partial x} \bigg|_t \left(\frac{\partial x}{\partial t} \right) + \frac{\partial I}{\partial y} \bigg|_t \left(\frac{\partial y}{\partial t} \right) + \frac{\partial I}{\partial t} \bigg|_{x(t)} = 0$$


$$\frac{\partial I}{\partial x} \bigg|_t u + \frac{\partial I}{\partial y} \bigg|_t v + \frac{\partial I}{\partial t} \bigg|_{x(t)} = 0$$

Shoot! One equation, two velocity (u, v) unknowns...

From 1D to 2D tracking

Notation

$$I_x u + I_y v + I_t = 0$$

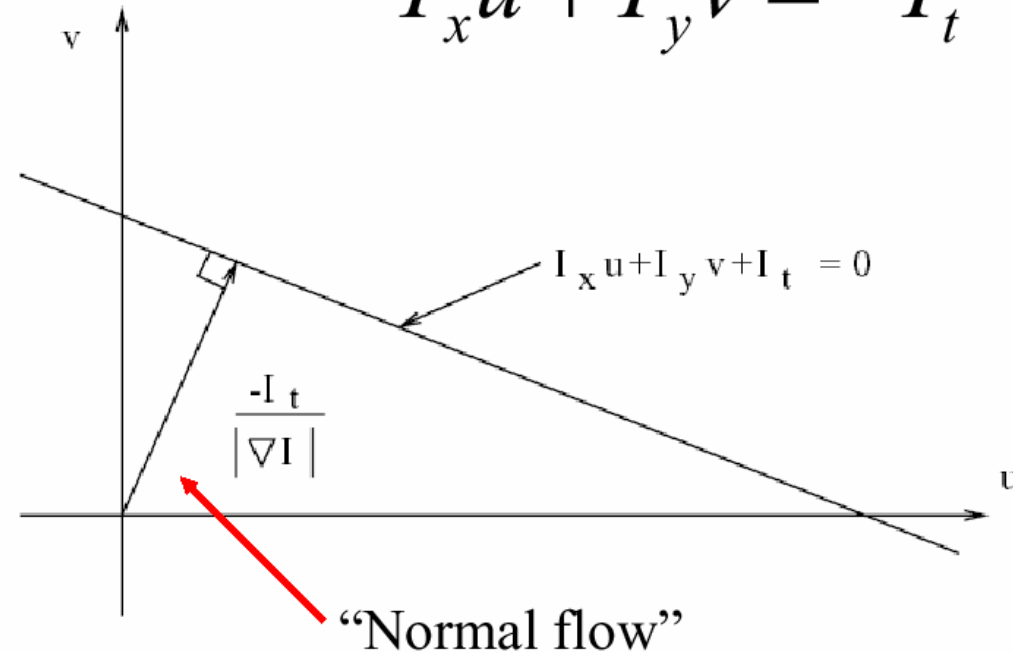
↓

$$\nabla I^T \mathbf{u} = -I_t$$

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \quad \nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix}$$

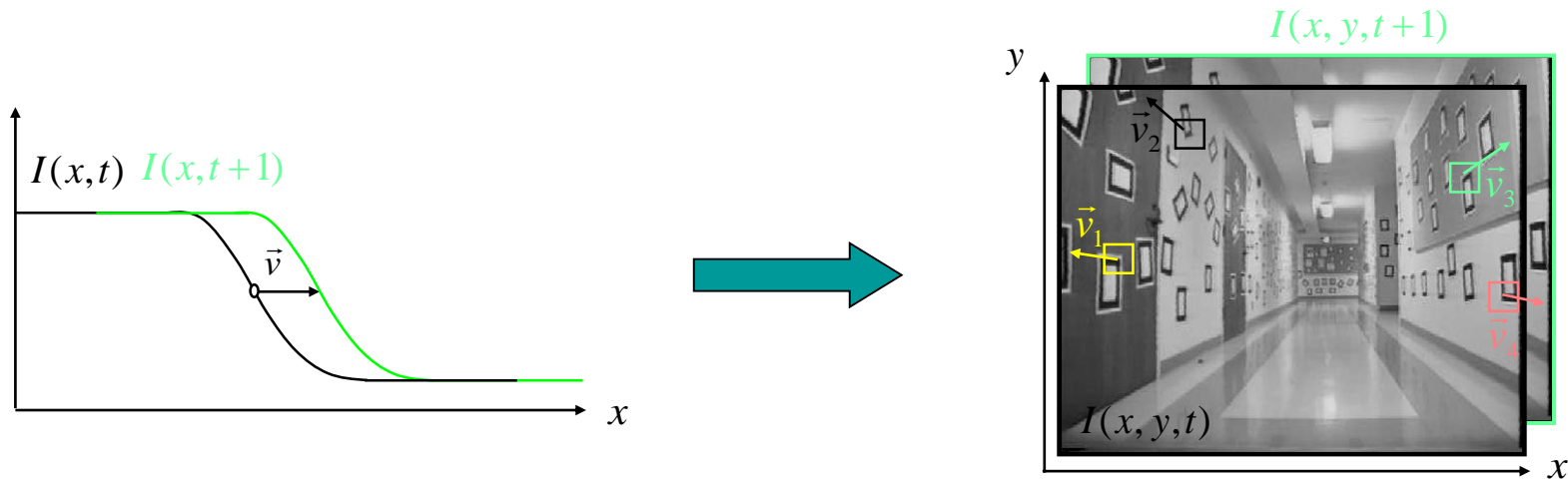
At a single image pixel, we get a line:

$$I_x u + I_y v = -I_t$$



We get at most “Normal Flow” – with one point we can only detect movement perpendicular to the brightness gradient. Solution is to take a patch of pixels Around the pixel of interest.

From 1D to 2D tracking



The Math is very similar:

$$\vec{v} \approx -\frac{I_t}{I_x}$$



Aperture problem

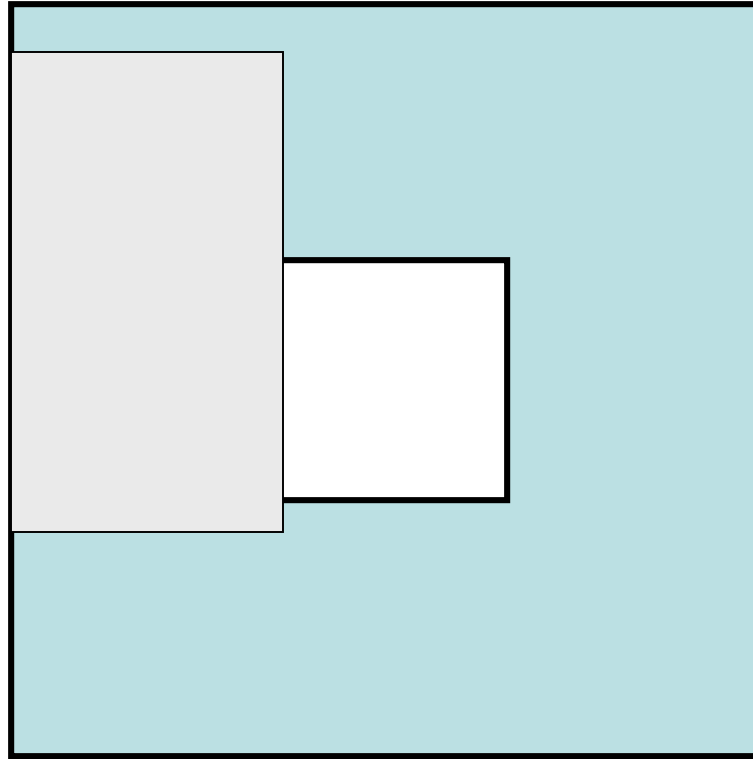
$$\left\{ \begin{array}{l} \vec{v} \approx -G^{-1}b \\ G = \sum_{\text{window around } p} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \\ b = \sum_{\text{window around } p} \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix} \end{array} \right.$$

Window size here ~ 11x11

Проблема апертуры



Aperture Problem Exposed

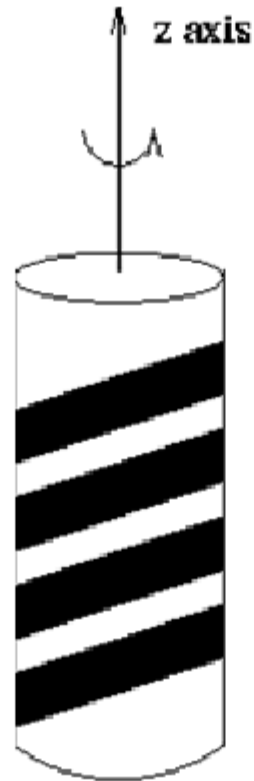


Motion along just an edge is ambiguous

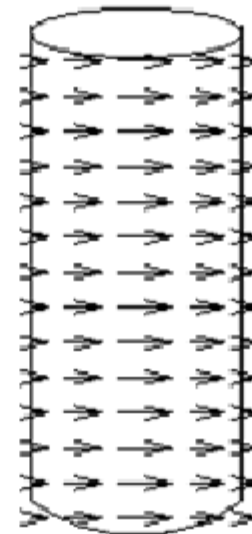
Aperture Problem in Real Life

Aperture Problem

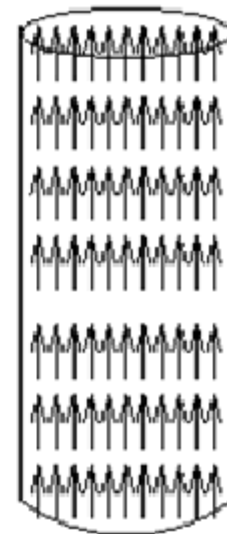
Barber pole illusion



Barber's pole



Motion field



Optical flow

Решение апертурной проблемы

- Как можно получить больше уравнений?
 - Идея: наложить дополнительные ограничения
 - Пусть оптический поток меняется плавно
 - Вариант: пусть для всех пикселей из окрестности смещение постоянно! (u, v)
 - » Для окна 5x5 получаем 25 уравнений для каждого пикселя!

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

$$\underbrace{\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix}}_{\substack{A \\ 25 \times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2 \times 1}} = - \underbrace{\begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix}}_{\substack{b \\ 25 \times 1}}$$

Цвет вместо яркости

- При использовании окна 5x5 получается 25*3 уравнений на пиксель!

$$0 = I_t(p_i)[0, 1, 2] + \nabla I(p_i)[0, 1, 2] \cdot [u \ v]$$

$$\underbrace{\begin{bmatrix} I_x(p_1)[0] & I_y(p_1)[0] \\ I_x(p_1)[1] & I_y(p_1)[1] \\ I_x(p_1)[2] & I_y(p_1)[2] \\ \vdots & \vdots \\ I_x(p_{25})[0] & I_y(p_{25})[0] \\ I_x(p_{25})[1] & I_y(p_{25})[1] \\ I_x(p_{25})[2] & I_y(p_{25})[2] \end{bmatrix}}_{\substack{A \\ 75 \times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2 \times 1}} = - \underbrace{\begin{bmatrix} I_t(p_1)[0] \\ I_t(p_1)[1] \\ I_t(p_1)[2] \\ \vdots \\ I_t(p_{25})[0] \\ I_t(p_{25})[1] \\ I_t(p_{25})[2] \end{bmatrix}}_{\substack{b \\ 75 \times 1}}$$

Метод Лукаса-Канаде

- Проблема: больше уравнений, чем неизвестных!

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

Решение: получаем задачу наименьших квадратов

- Значение d можно получить как:

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

$$\begin{matrix} \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} & \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \\ A^T A & A^T b \end{matrix}$$

- Суммируем по всем пикселям в окне $K \times K$
- Это метод был предложен Лукасом и Канаде в 1981 году

Условия на разрешимость

- Оптимальные (u, v) удовлетворяют уравнению

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

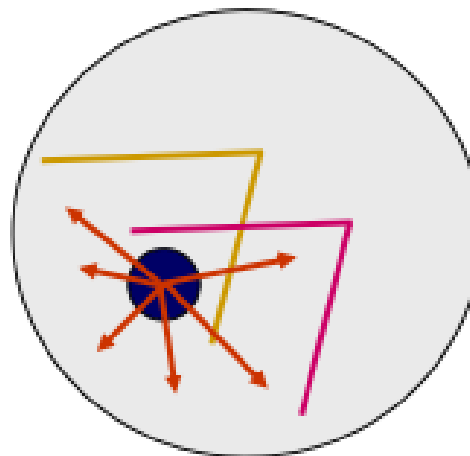
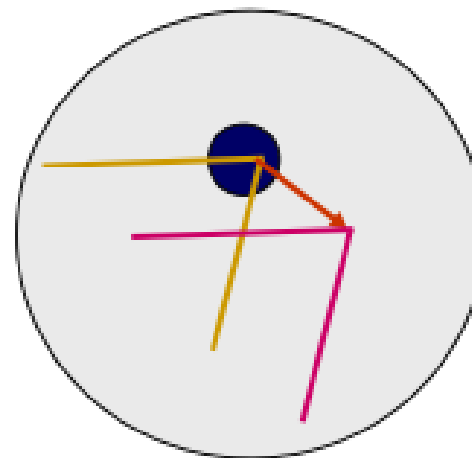
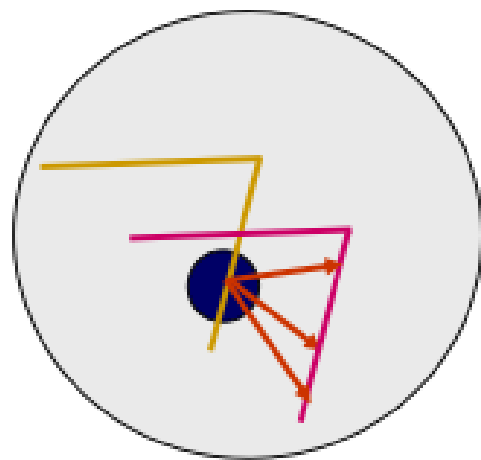
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

Когда задача разрешима?

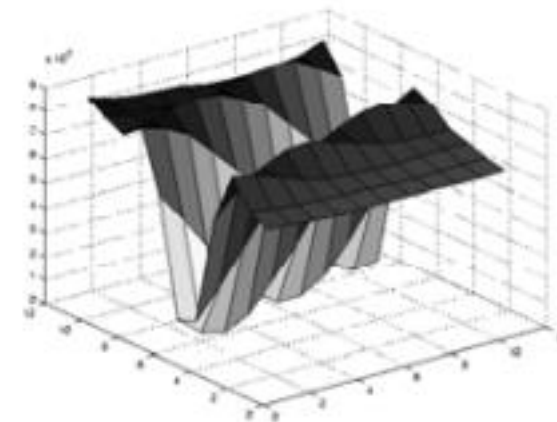
- $A^T A$ должна быть обратимой
- $A^T A$ не должна быть слишком близка к нулю
 - С.значения λ_1 и λ_2 матрицы $A^T A$ не должны быть малы
- $A^T A$ должна быть хорошо определима
 - λ_1 / λ_2 не должно быть слишком велико
 - ($\lambda_1 =$ наибольшее с.значение)

$A^T A$ разрешима, когда нет апертурной проблемы

Анализ участка изображения



Края

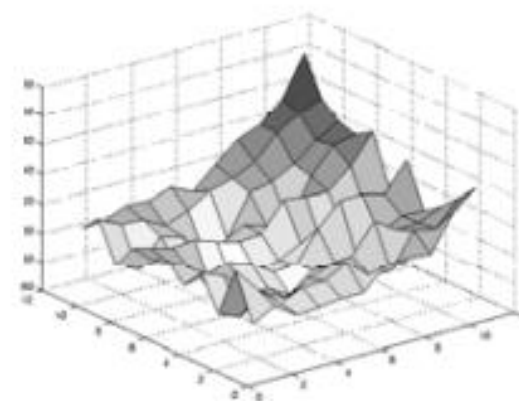
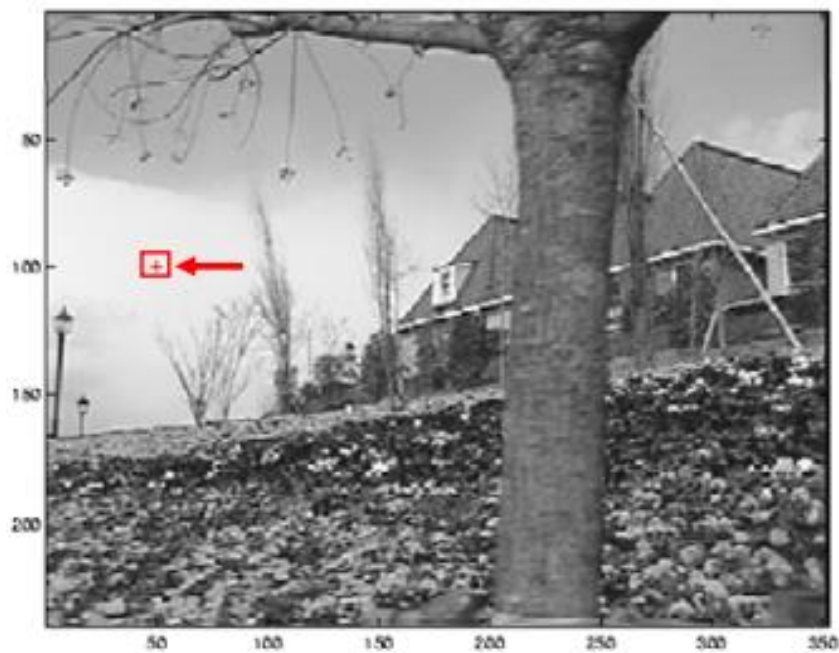


$$\sum \nabla I (\nabla I)^T$$

— большие градиенты

— большое λ_1 , маленькое λ_2

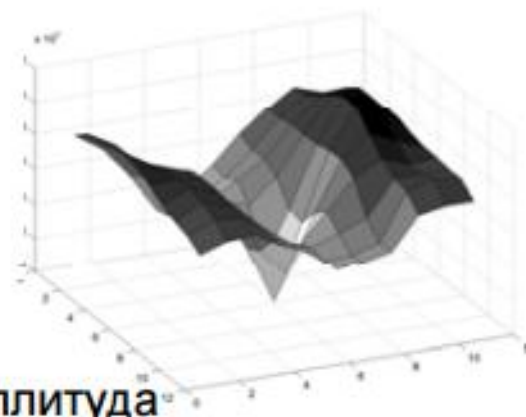
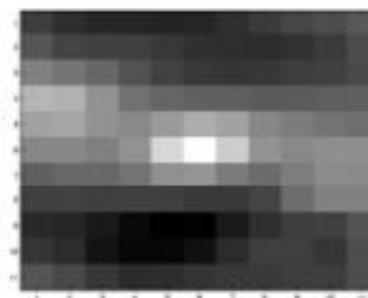
Слабоконтрастная текстура



$$\sum \nabla I (\nabla I)^T$$

- величина градиента мала
- малое λ_1 , малое λ_2

Текстурированная область



$$\sum \nabla I (\nabla I)^T$$

- градиенты разные, большая амплитуда
- большое λ_1 , большое λ_2

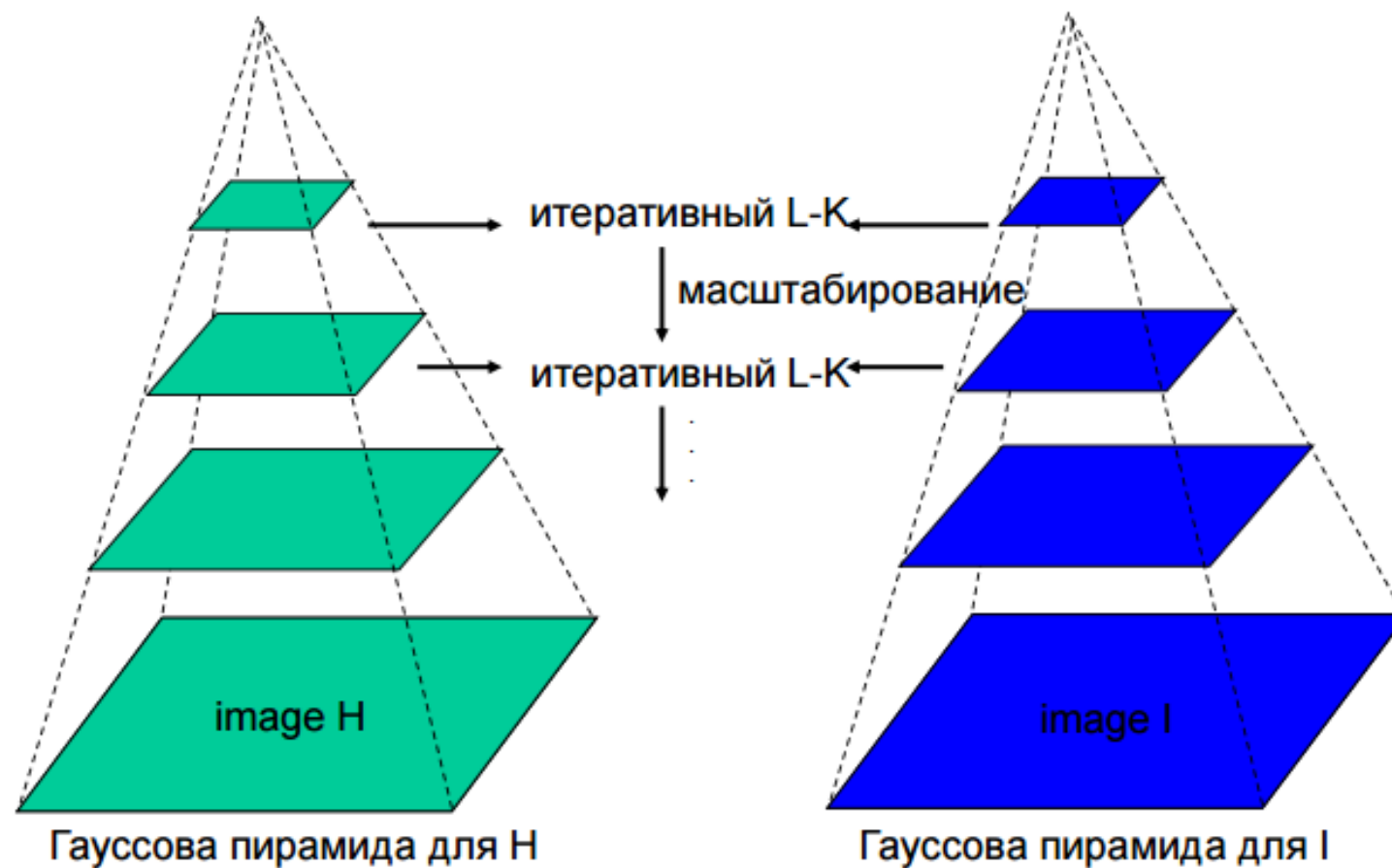
Погрешности метода Лукаса-Канаде

- Каковы потенциальные источники ошибок?
 - Предполагаем, что АТА обратима
 - Предполагаем, что в изображении мало шума

Когда эти предположения нарушаются:

- Яркость точки не постоянная
- Движение между кадрами большое
- Точка не двигается также, как ее соседи
 - Окно поиска слишком большое
 - Какой наилучший размер окна поиска?

Бонус: пирамидальный Лукас-Канаде



В следующих сериях...



$\sin(x)$



$\cos(x)$



$\tan(x)$



$\cot(x)$



$|x|$



x



x^2



$x^2 + y^2$



\sqrt{x}



$\sqrt{-x}$



$\frac{1}{x}$



crap.