Введение в компьютерное зрение

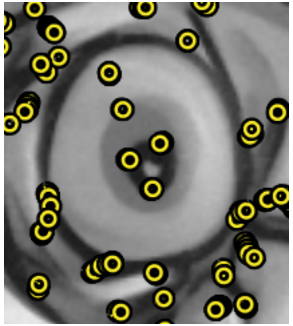
Особые точки, оптический поток, оценка движения в 2D

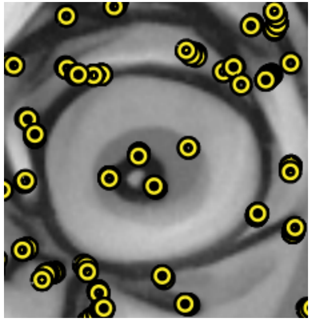
Алексей Спижевой
Itseez-UNN Summer School 2015

Содержание

1. Особые точки
    a. Угловые точки Харриса
    b. FAST
2. Оптический поток: алгоритм Лукаса-Канаде
3. Робастная оценка движения: RANSAC
4. Приложение: видеостабилизация

Особые точки





Особая (ключевая) точка -- точка, обладающая высокой локальной информативностью.

Детектор -- это метод извлечения особых точек из изображения.

Характеристики детектора особых точкек:

- количество выдаваемых точек,
- точность их локализации,
- повторяемость:
    - инвариантность к поворотам, сдвигам, изменениям яркости, ...
    - устойчивость к шумам, размытиям, ...
- информативность окрестности,
- эффективность, в смысле времени обработки изображений,
- …

Дескриптор -- описатель особой точки, выделяющий её из остального множества особых точек.

Особые точки

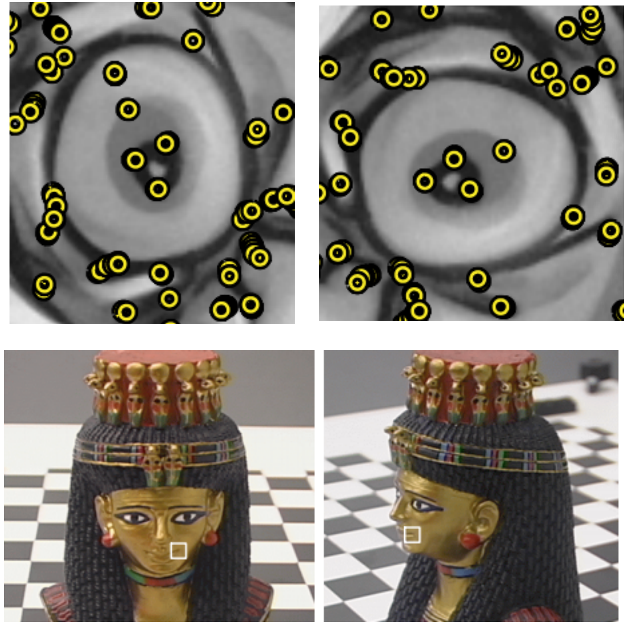
Особая (ключевая) точка -- точка, обладающая высокой локальной информативностью.

Детектор -- это метод извлечения особых точек из изображения.

Характеристики детектора особых точкек:

- количество выдаваемых точек,
- точность их локализации,
- повторяемость:
  - инвариантность к поворотам, сдвигам, изменениям яркости, ...
  - устойчивость к шумам, размытиям, ...
- информативность окрестности,
- эффективность, в смысле времени обработки изображений,
- …

Дескриптор -- описатель особой точки, выделяющий её из остального множества особых точек.



Особые точки

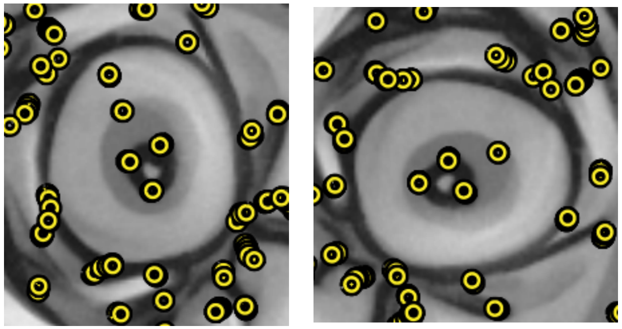
Особая (ключевая) точка -- точка, обладающая высокой локальной информативностью.

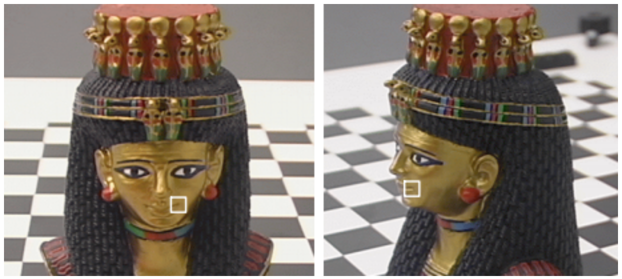
Детектор -- это метод извлечения особых точек из изображения.

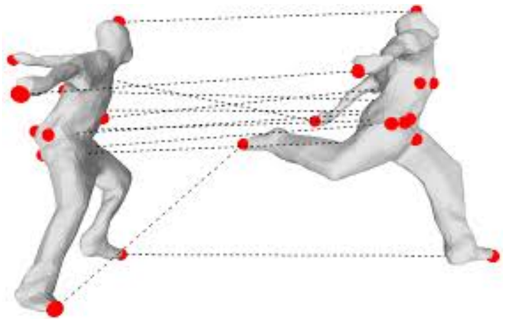
Характеристики детектора особых точкек:

- количество выдаваемых точек,
- точность их локализации,
- повторяемость:
  - инвариантность к поворотам, сдвигам, изменениям яркости, ...
  - устойчивость к шумам, размытиям, ...
- информативность окрестности,
- эффективность, в смысле времени обработки изображений,
- …

Дескриптор -- описатель особой точки, выделяющий её из остального множества особых точек.



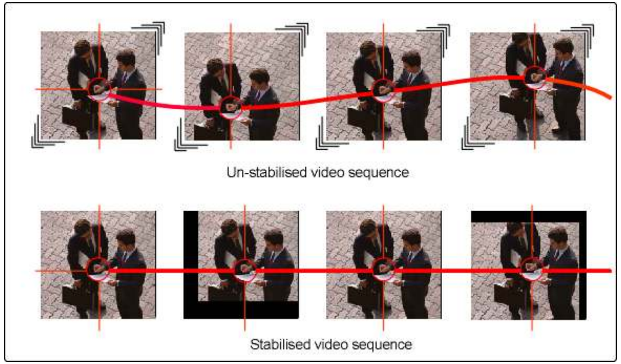




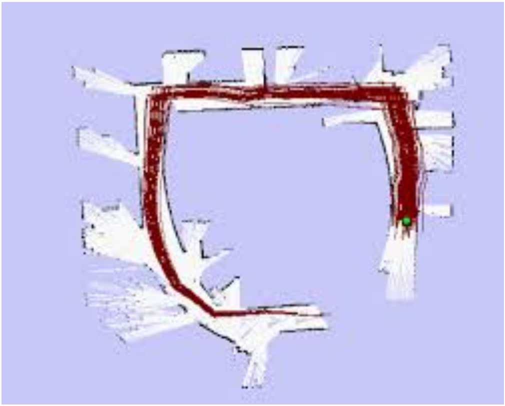
Примеры использования

image mosaicing

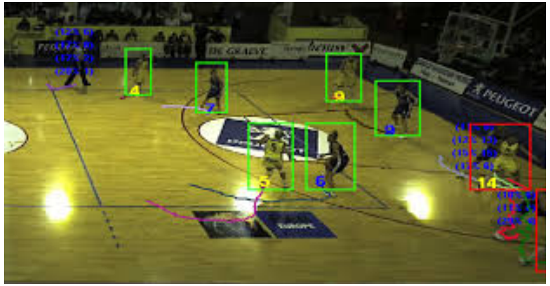
video stabilization



simultaneous localization and mapping



object tracking



3d reconstruction

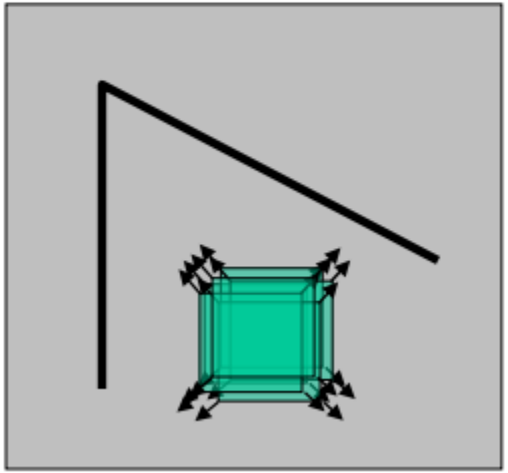
Детекторы особых точек

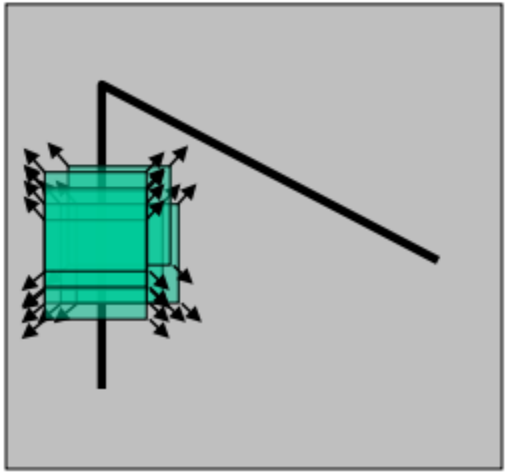
Детектор угловых точек Харриса

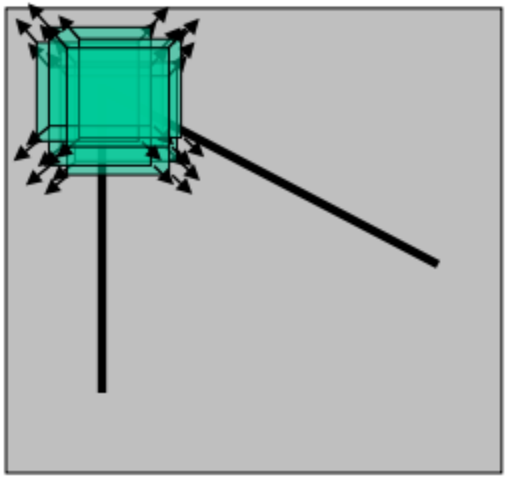
Попробуем найти угловые точки. То есть такие у которых окрестность сильно меняется, в какую бы сторону мы не сдвигались (локально).

Детектор угловых точек Харриса

Попробуем найти угловые точки. То есть такие у которых окрестность сильно меняется, в какую бы сторону мы не сдвигались (локально).



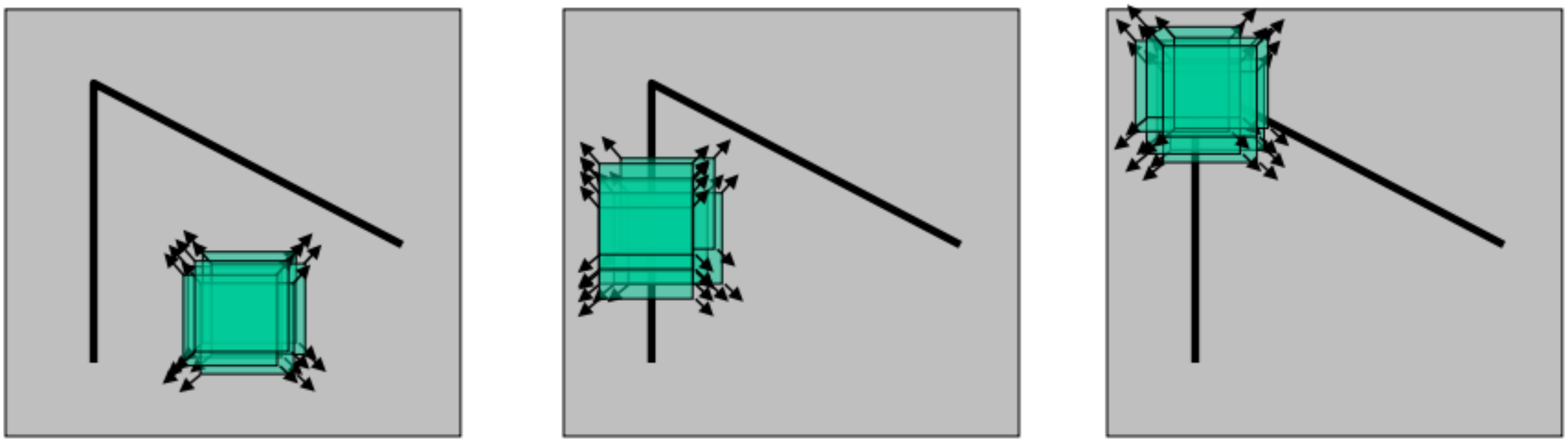




Детектор угловых точек Харриса

Попробуем найти угловые точки. То есть такие у которых окрестность сильно меняется, в какую бы сторону мы не сдвигались (локально).

нет

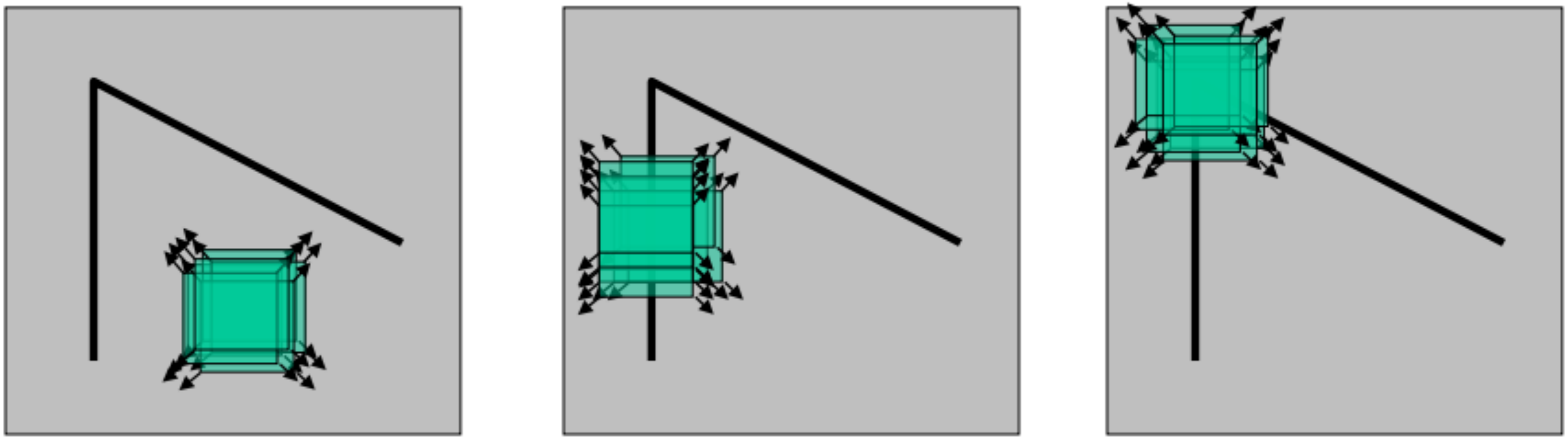


Детектор угловых точек Харриса

Попробуем найти угловые точки. То есть такие у которых окрестность сильно меняется, в какую бы сторону мы не сдвигались (локально).

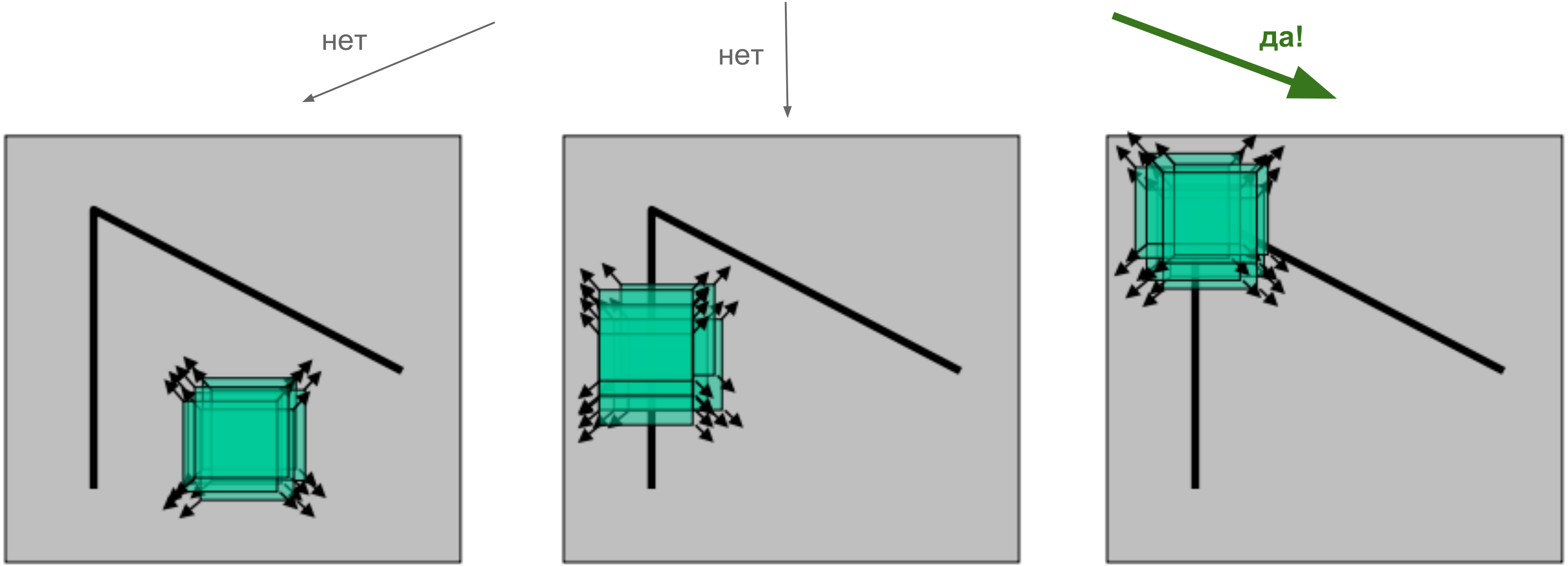
нет

нет

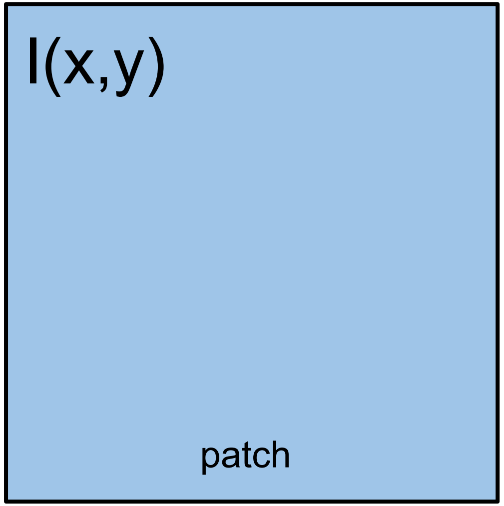


Детектор угловых точек Харриса

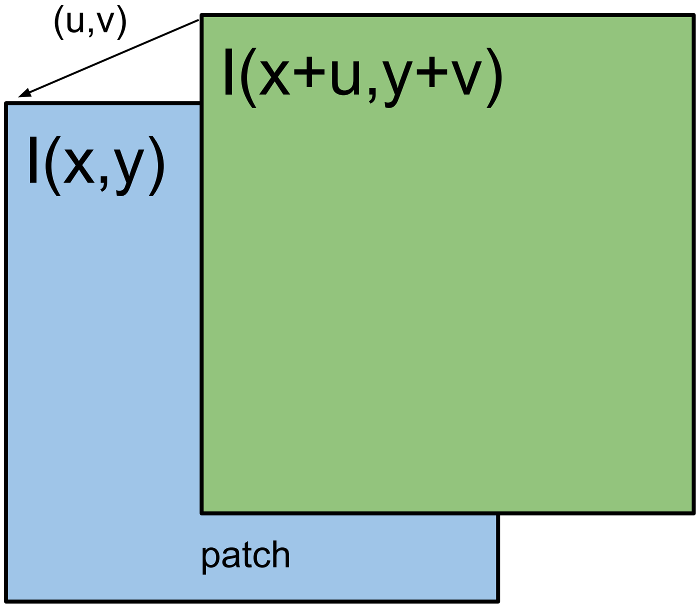
Попробуем найти угловые точки. То есть такие у которых окрестность сильно меняется, в какую бы сторону мы не сдвигались (локально).



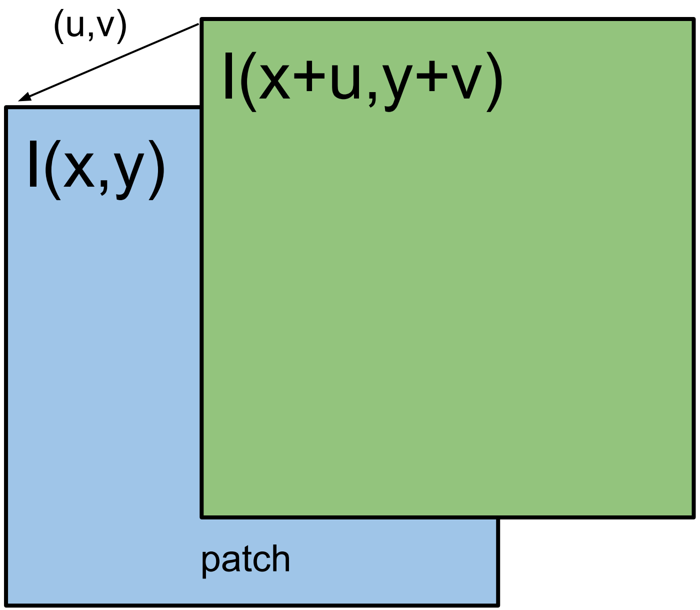
Детектор угловых точек Харриса



Детектор угловых точек Харриса



Детектор угловых точек Харриса



$$S(u,v) = \sum_{(x,y)\in patch} \left(I(x+u, y+v) - I(x,y)\right)^2$$

-- непохожесть окон при сдвиге на (u,v)

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

-- разложили в ряд Тейлора,
производные выше первой отбросили

Детектор угловых точек Харриса

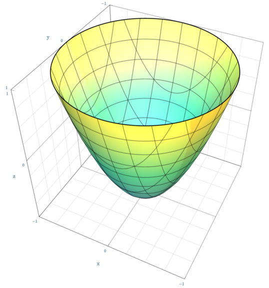
$$S(u, v) = \sum_{x,y} (I_x(x, y)u + I_y(x, y)v)^2 = \sum_{x,y} \begin{pmatrix} u \\ v \end{pmatrix}^T \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

Детектор угловых точек Харриса

$$S(u,v) = \sum_{x,y} (I_x(x,y)u + I_y(x,y)v)^2 = \sum_{x,y} \begin{pmatrix} u \\ v \end{pmatrix}^T \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}$$

$$S(u,v) = \mathbf{a}^T M \mathbf{a}$$

$$\mathbf{a} = \begin{pmatrix} u \\ v \end{pmatrix} \quad M = \sum_{x,y} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$ -- структурный тензор

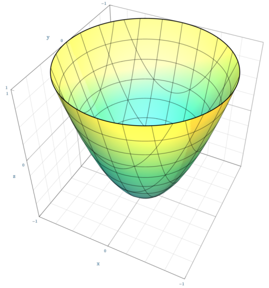


Детектор угловых точек Харриса

$$S(u,v) = \sum_{x,y}(I_x(x,y)u + I_y(x,y)v)^2 = \sum_{x,y}\begin{pmatrix}u\\v\end{pmatrix}^T\begin{pmatrix}I_x^2 & I_xI_y\\ I_xI_y & I_y^2\end{pmatrix}\begin{pmatrix}u\\v\end{pmatrix}$$

$$S(u,v) = \mathbf{a}^T M\mathbf{a}$$

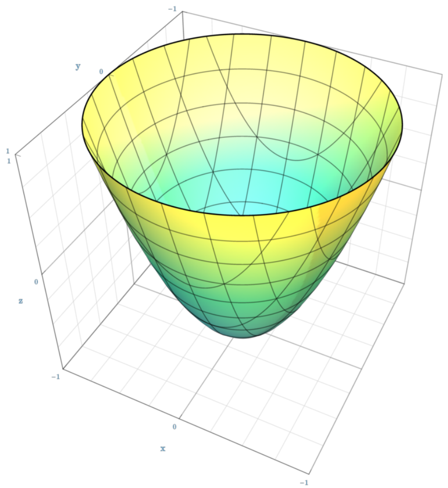
$$\mathbf{a} = \begin{pmatrix}u\\v\end{pmatrix}\quad M = \sum_{x,y}\begin{pmatrix}I_x^2 & I_xI_y\\ I_xI_y & I_y^2\end{pmatrix}$$ -- структурный тензор

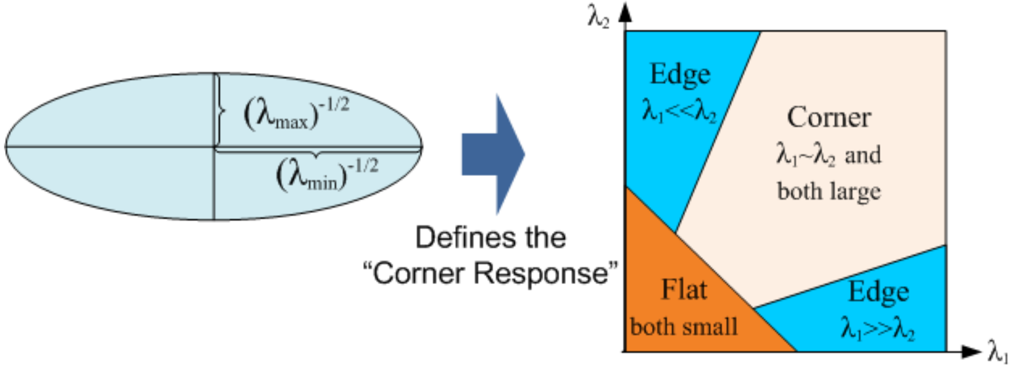


$$S(u,v) = \mathbf{a}^T M\mathbf{a} \in [\lambda_1, \lambda_2], if\,|\mathbf{a}| = 1$$

- оба собственных числа близки к нулю -- однородная область
- одно большое, другое маленькое -- ребро
- ***оба большие -- угловая точка***

Детектор угловых точек Харриса





Т.о. нужно найти точки у которых минимальное собственное число структурного тензора больше порога (параметр метода).

`cv::cornerEigenValsAndVecs, cv::cornerMinEigenVal`

Детектор угловых точек Харриса

Можно ли обойтись без нахождения собственных чисел, и определить является ли точка угловой быстрее?

Детектор угловых точек Харриса

Можно ли обойтись без нахождения собственных чисел, и определить является ли точка угловой быстрее? Да! Harris score.

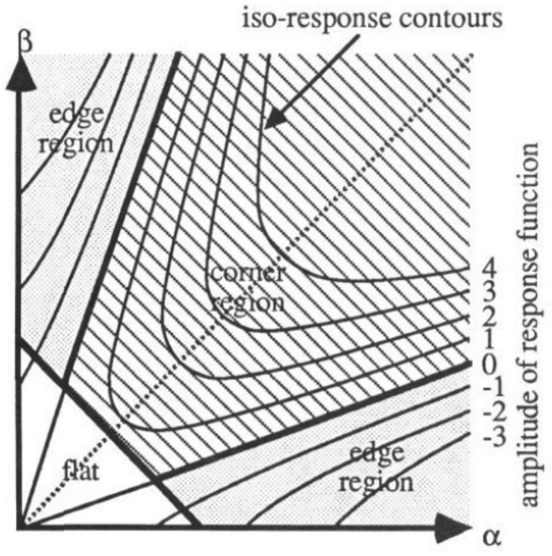
$$cornerness(x, y) = |M| - k * \mathrm{trace}(M)^2$$

Детектор угловых точек Харриса

Можно ли обойтись без нахождения собственных чисел, и определить является ли точка угловой быстрее? Да! Harris score.

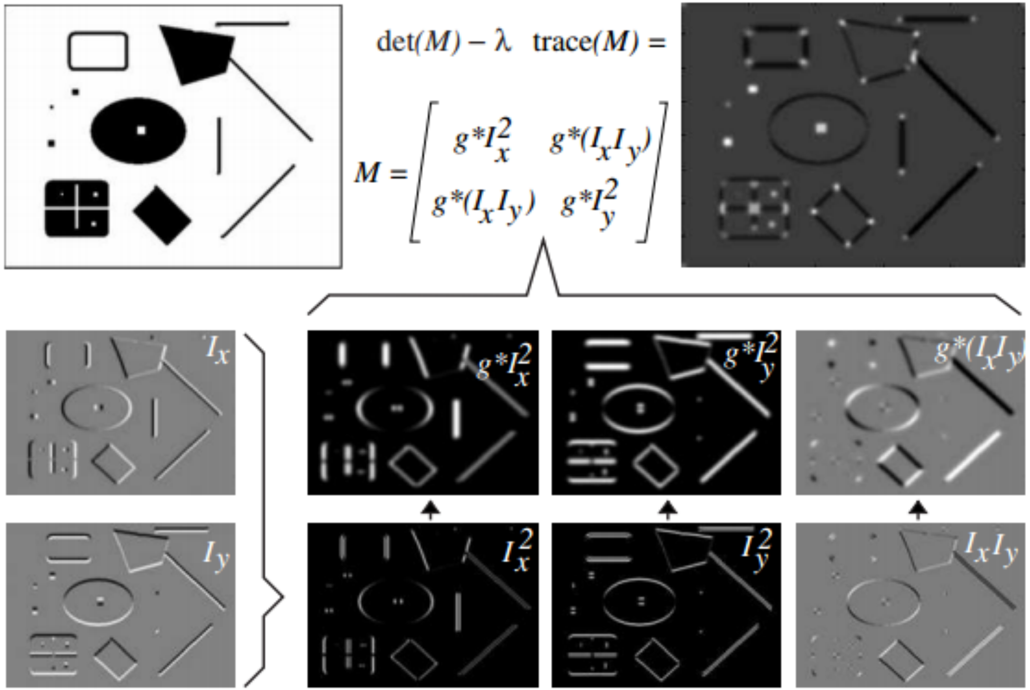
$$cornerness(x, y) = |M| - k * \mathrm{trace}(M)^2$$

$$|M| - k * \mathrm{trace}(M)^2 = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$



линии уровня

Детектор угловых точек Харриса



cv::cornerHarris

Детектор угловых точек Харриса

1. Вычислить производные (Sobel)

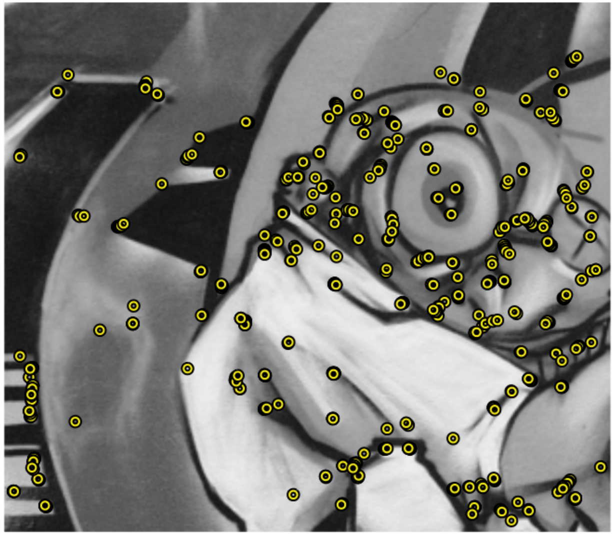
2. Подсчитать score (cornerness или мин. собств. число) для точек

3. Выбрать точки с высоким значением score

4. Non-maximum suppression: среди близко расположенных угловых точек оставить только наилучшую

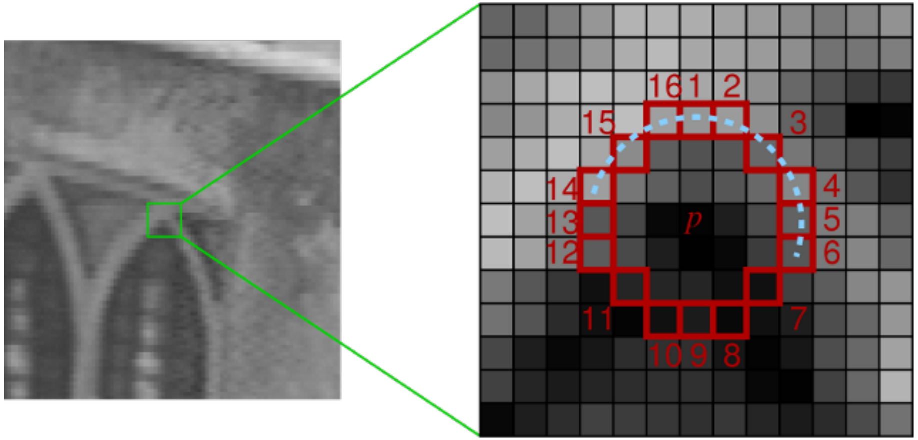
5. Удалить слишком слабые точки (по сравнению с наилучшей)

6. Есть точек слишком много, просто удалить слабейшие



`cv::GFTT, cv::goodFeaturesToTrack`

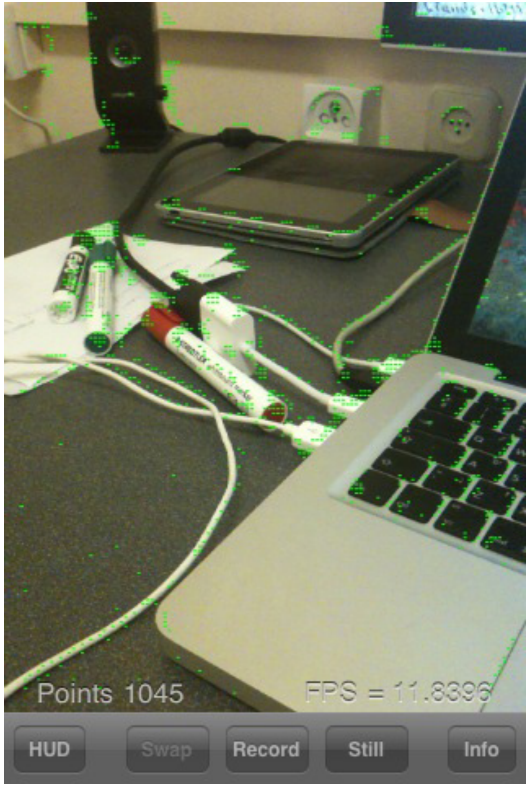
FAST детектор

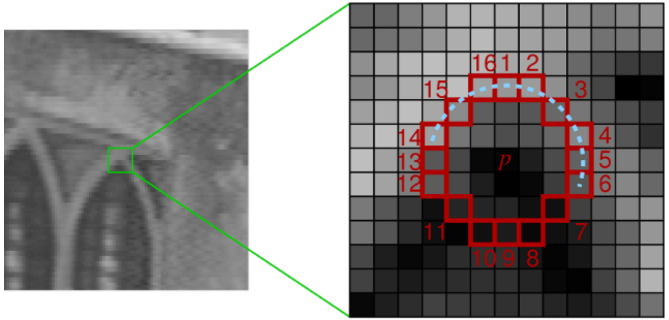


$$S_{p\to x} = \begin{cases} d, & I_{p\to x} \le I_p - t \quad \text{(darker)} \\ s, & I_p - t < I_{p\to x} < I_p + t \quad \text{(similar)} \\ b, & I_p + t \le I_{p\to x} \quad \text{(brighter)} \end{cases}$$

Если среди 16 соседей точки *p*, есть >= *n* смежных таких, что *I(x)* > *I(p)*+*t* (или *I(x)* < *I(p)*-*t*), то точка считается угловой

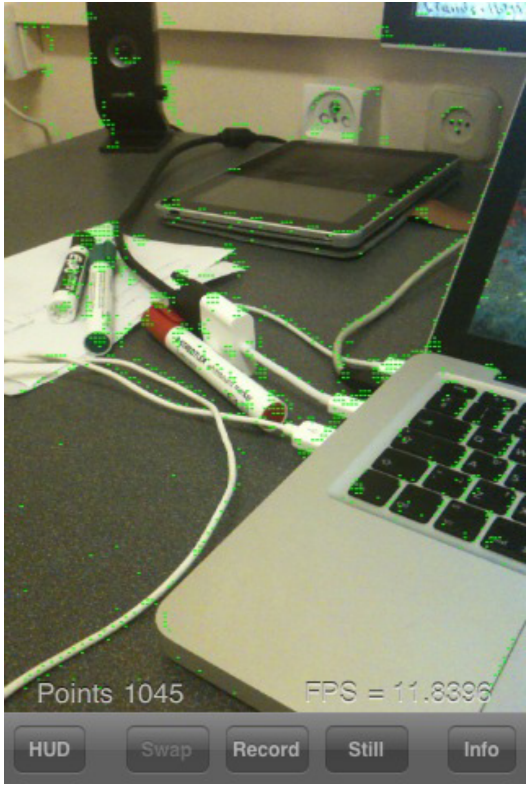
FAST детектор

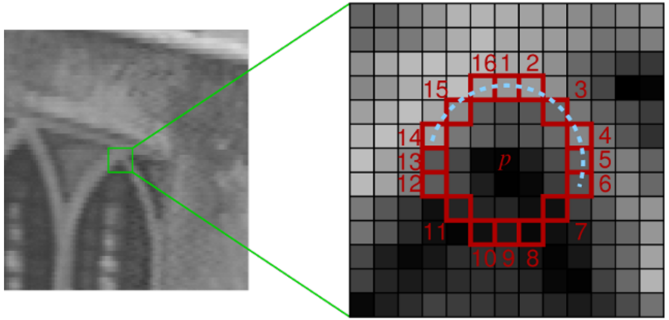




Необходимо ли проверять все >= n точек?

FAST детектор



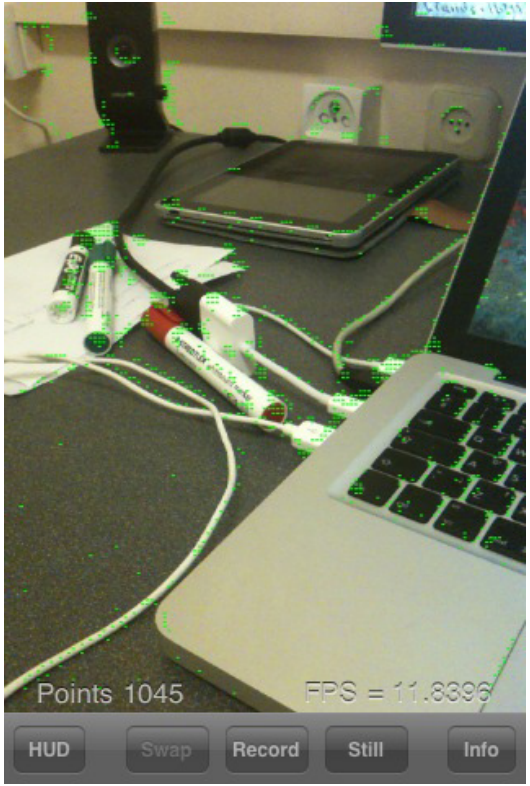


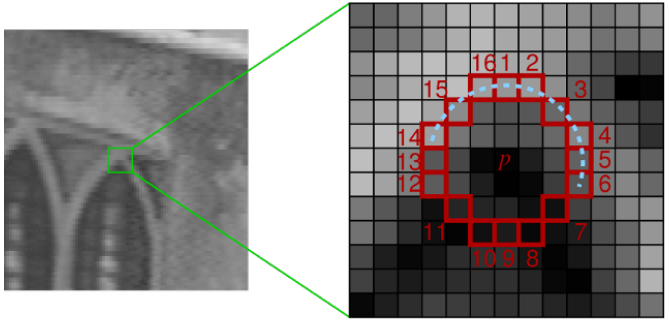
Необходимо ли проверять все >= n точек? Не всегда.

High-speed test (n>=12)

- Можно сначала проверить точки 1,5,9,13. Из них три должны быть ярче (темнее) центральной.

FAST детектор





Необходимо ли проверять все >= n точек? Не всегда.

High-speed test (n>=12)

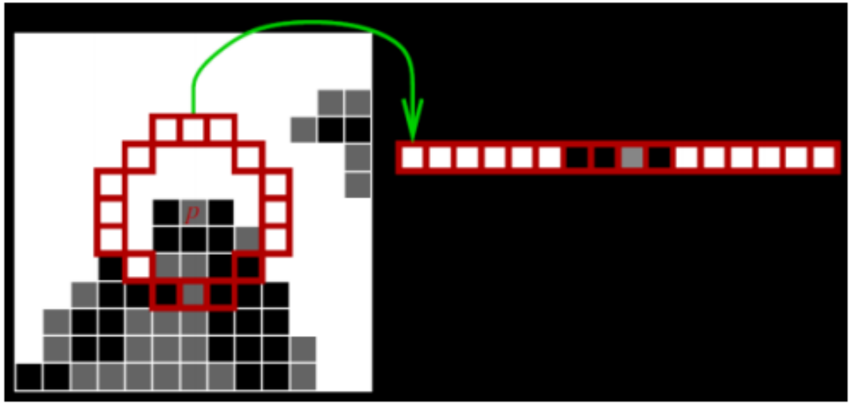
- Можно сначала проверить точки 1,5,9,13. Из них три должны быть ярче (темнее) центральной.

n влияет на повторяемость. Эксперименты показывают, что вариант с n=9 лучше.

FAST детектор

Можно попытаться найти последовательность сравнений с помощью методов машинного обучения

1. Собрать тренировочный набор изображений.
2. “Честно” найти ключевые точки.
3. Построить дерево решений (оно будет корректно классифицировать все точки тренировочной выборки).
4. Сконвертировать структуру дерева в C-код.

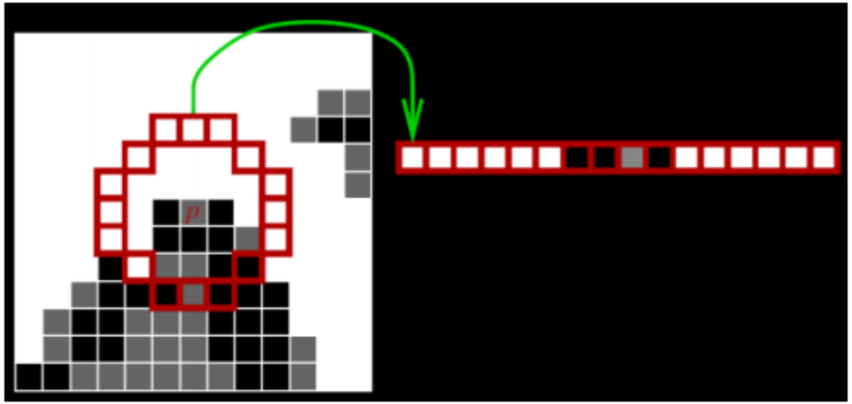


Тренировочный набор данных: все пиксели всех изображений с метками классов (ключевая точка или нет).

FAST детектор

Можно попытаться найти последовательность сравнений с помощью методов машинного обучения

1. Собрать тренировочный набор изображений.
2. “Честно” найти ключевые точки.
3. Построить дерево решений (оно будет корректно классифицировать все точки тренировочной выборки).
4. Сконвертировать структуру дерева в C-код.



Тренировочный набор данных: все пиксели всех изображений с метками классов (ключевая точка или нет).

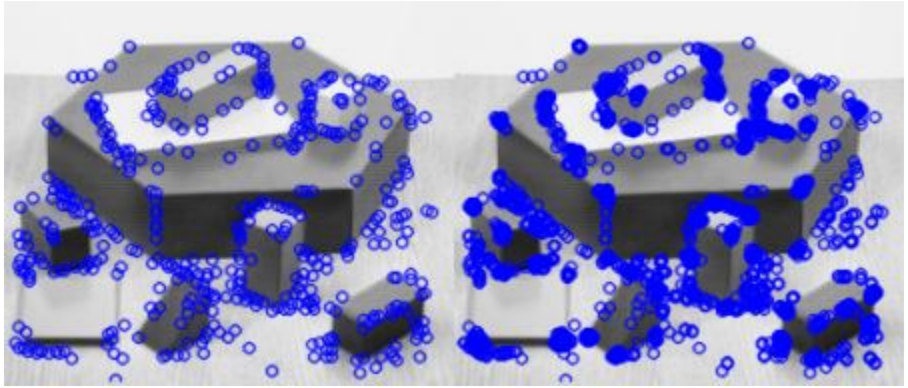
Если какое-то сравнение на пути классификации в дереве не встречется, то оно не делается. Итого в среднем <3 сравнений с соседними точками на пиксель изображения

FAST детектор

Non-maximum suppression: среди близко расположенных угловых точек оставить только наилучшую

Для этого нужно ввести способ сравнения угловых точек:

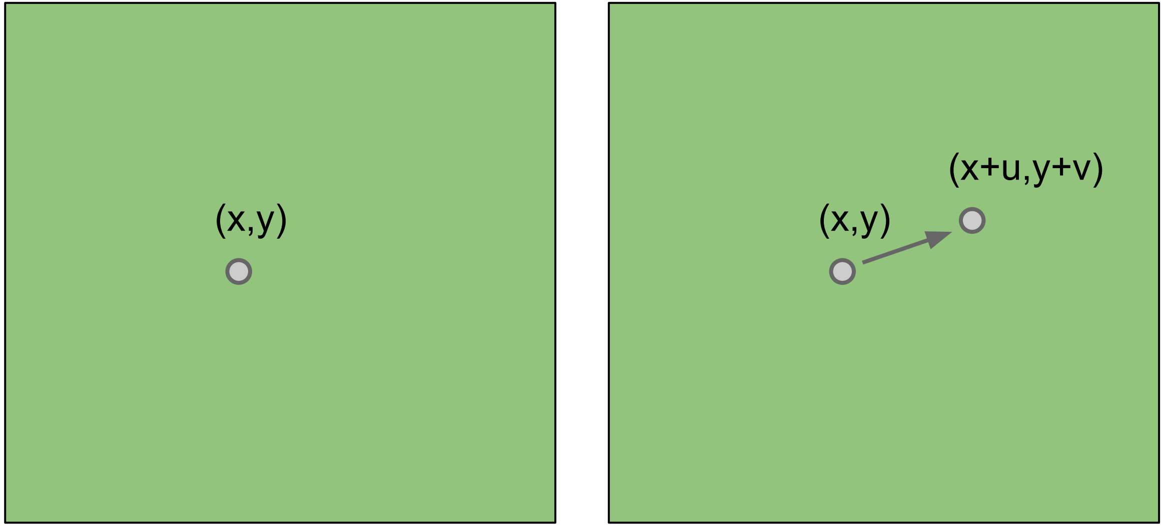
$$V = \max\left(\sum_{x\in S_{\text{bright}}} |I_{p\to x} - I_p| - t\ ,\ \sum_{x\in S_{\text{dark}}} |I_p - I_{p\to x}| - t\right)$$



`cv::FAST`

Оптический поток

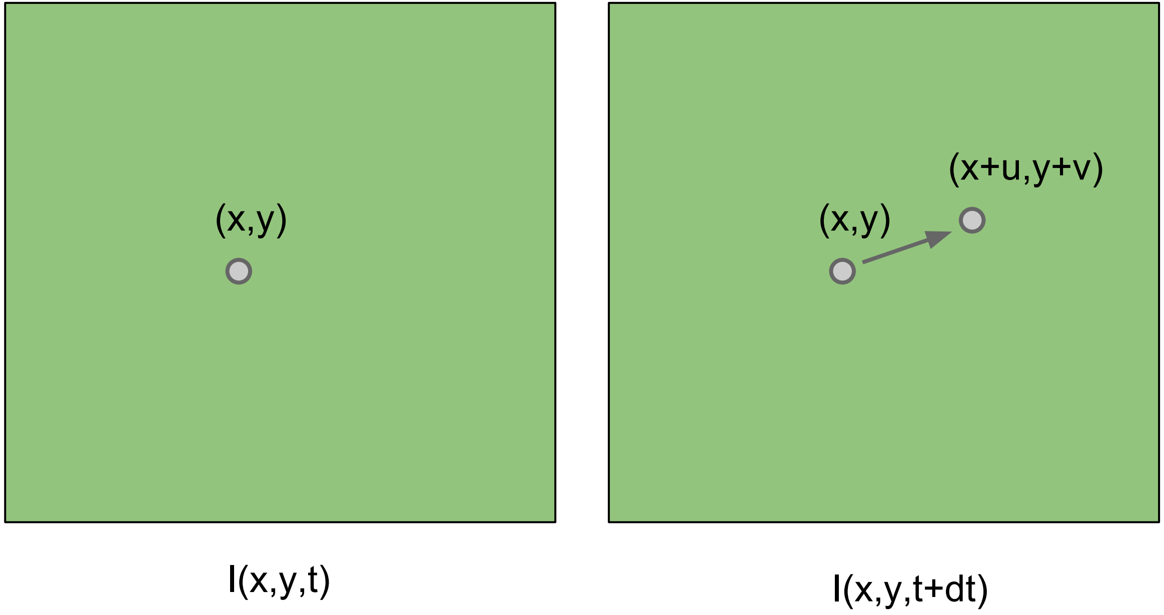
Оптический поток



$I(x,y,t)$ $\quad$ $I(x,y,t+dt)$

$(u,v) = ?$

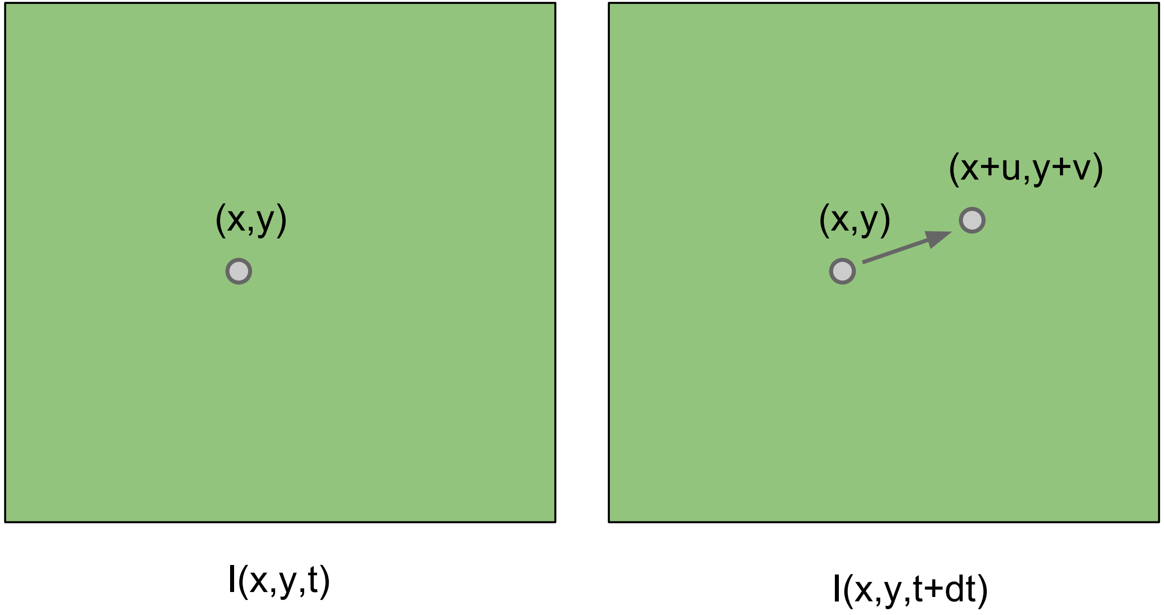
Оптический поток



Предположим, что яркость точки не меняется:

$$\frac{dI(x(t), y(t), t)}{dt} = 0$$

Оптический поток



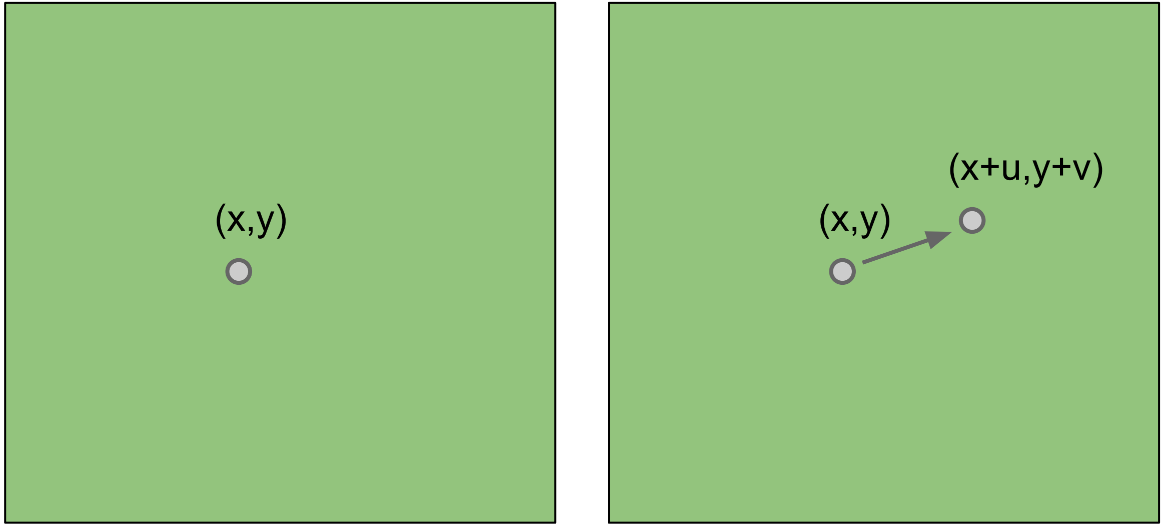
Предположим, что яркость точки не меняется:

$$\frac{dI(x(t), y(t), t)}{dt} = 0$$

Уравнение оптического потока:

$$I_x \frac{dx}{dt} + I_y \frac{dy}{dt} + I_t = 0$$

Оптический поток



I(x,y,t)

I(x,y,t+dt)

Предположим, что яркость точки не меняется:

$$\frac{dI(x(t), y(t), t)}{dt} = 0$$

Уравнение оптического потока:

$$I_x \frac{dx}{dt} + I_y \frac{dy}{dt} + I_t = 0$$

или u=dx, v=dy, dt=1 (t -- номер кадра):

$$\boxed{I_x u + I_y v = -I_t}$$

Уравнение оптического потока

$$I(x, y, t) = I(x + u, y + v, t + 1)$$

$$I(x + u, y + v, t + 1) \approx I + I_x u + I_y v + I_t$$

-- разложим в ряд Тейлора, производные выше первой отбросим

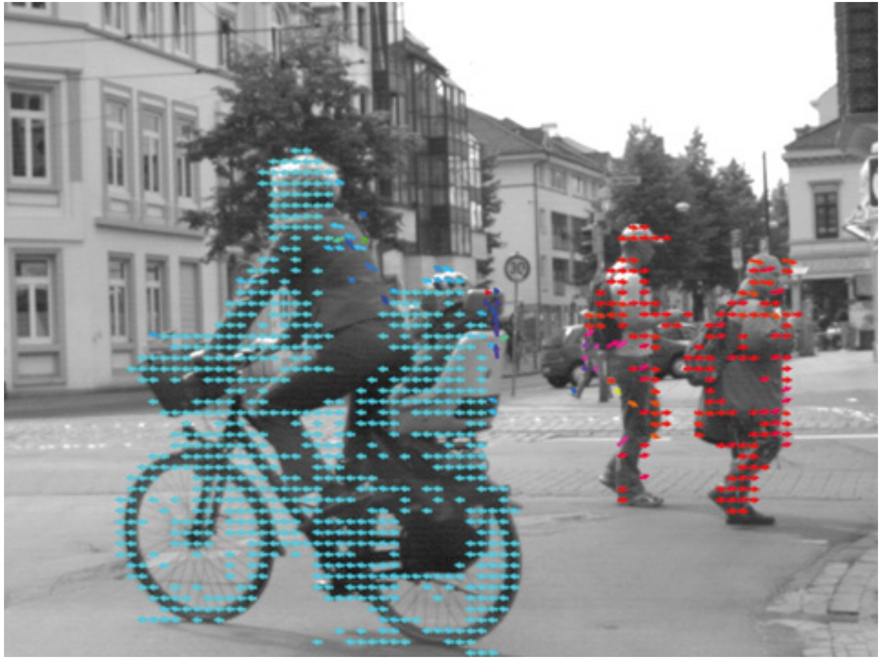
$$\boxed{I_x u + I_y v = -I_t}$$

-- уравнение оптического потока для точки (x,y)

Оптический поток

Оптический поток

Оптический поток



Метод Лукаса-Канаде

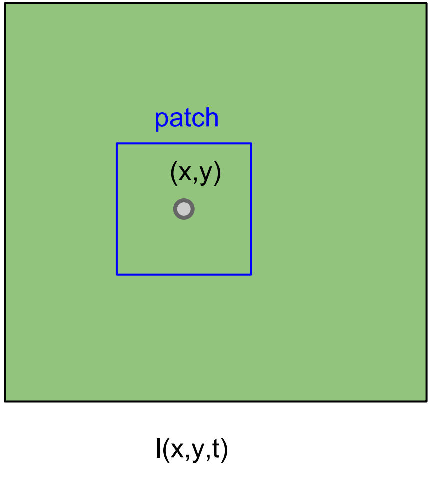
$I_x u + I_y v = -I_t$ -- уравнение оптического потока для точки (x,y)

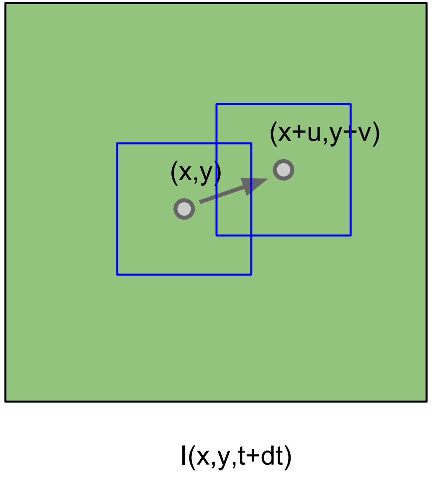
Две неизвестных на точку, одно уровнение. Что делать?

Метод Лукаса-Канаде

$I_x u + I_y v = -I_t$ -- уравнение оптического потока для точки (x,y)

Две неизвестных на точку, одно уровнение. Что делать? Рассмотреть движение точки с окрестностью.





$$\begin{pmatrix} I_x(x_1, y_1) & I_y(x_1, y_1) \\ \cdots & \cdots \\ I_x(x_n, y_n) & I_y(x_n, y_n) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -I_t(x_1, y_1) \\ \cdots \\ -I_t(x_n, y_n) \end{pmatrix}$$

-- уравнение оптического потока для окрестности точки (x,y)

Метод Лукаса-Канаде

$$\begin{pmatrix} I_x(x_1, y_1) & I_y(x_1, y_1) \\ \dots & \dots \\ I_x(x_n, y_n) & I_y(x_n, y_n) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -I_t(x_1, y_1) \\ \dots \\ -I_t(x_n, y_n) \end{pmatrix}$$

-- уравнение оптического потока для окрестности точки (x,y)

$$A \begin{pmatrix} u \\ v \end{pmatrix} = b$$

Метод Лукаса-Канаде

$$\begin{pmatrix} I_x(x_1, y_1) & I_y(x_1, y_1) \\ \dots & \dots \\ I_x(x_n, y_n) & I_y(x_n, y_n) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -I_t(x_1, y_1) \\ \dots \\ -I_t(x_n, y_n) \end{pmatrix}$$

-- уравнение оптического потока для окрестности точки (x,y)

$$A \begin{pmatrix} u \\ v \end{pmatrix} = b$$

В случае окрестности 5x5 имеем 25 уравнений и 2 неизвестные -- система переопределена

$$\sum_{x,y} (I_x u + I_y v + I_t)^2 \to min$$

-- воспользуемся методом наименьших квадратов

Метод Лукаса-Канаде

$$\begin{pmatrix} I_x(x_1, y_1) & I_y(x_1, y_1) \\ \dots & \dots \\ I_x(x_n, y_n) & I_y(x_n, y_n) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -I_t(x_1, y_1) \\ \dots \\ -I_t(x_n, y_n) \end{pmatrix}$$

-- уравнение оптического потока для окрестности точки (x,y)

$$A \begin{pmatrix} u \\ v \end{pmatrix} = b$$

В случае окрестности 5x5 имеем 25 уравнений и 2 неизвестные -- система переопределена

$$\sum_{x,y} (I_x u + I_y v + I_t)^2 \to min$$

-- воспользуемся методом наименьших квадратов

$$A^T A \begin{pmatrix} u \\ v \end{pmatrix} = A^T b$$

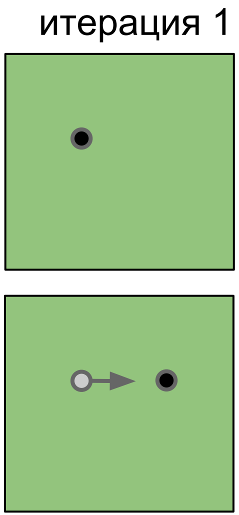
-- система 2x2, дает решение МНК

Оптический поток

- Метод ЛК применяется итеративно:
  - Вычислили поток
  - Преобразовали изображение
  - …
  - Вычислили поток
  - Преобразовали изображение
  - И так пока на сошлись или число итераций не превысит порог

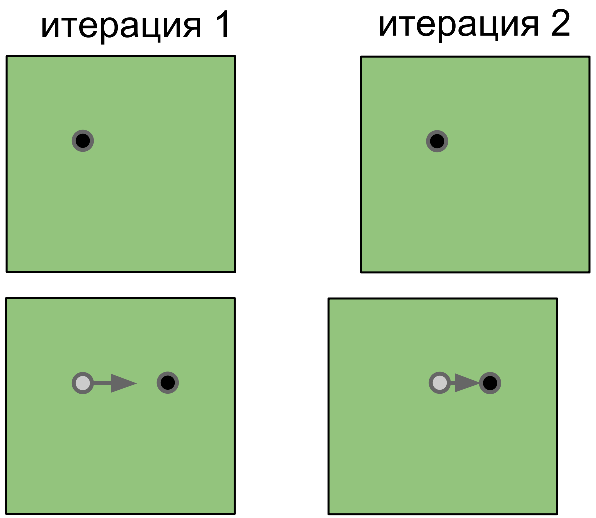
Оптический поток

- Метод ЛК применяется итеративно:
  - Вычислили поток
  - Преобразовали изображение
  - …
  - Вычислили поток
  - Преобразовали изображение
  - И так пока на сошлись или число итераций не превысит порог



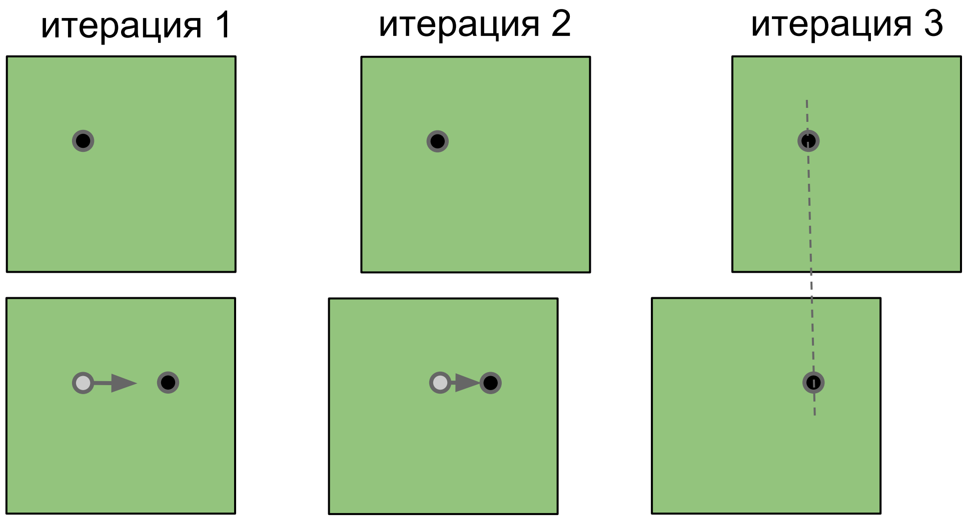
Оптический поток

- Метод ЛК применяется итеративно:
    - Вычислили поток
    - Преобразовали изображение
    - …
    - Вычислили поток
    - Преобразовали изображение
    - И так пока на сошлись или число итераций не превысит порог

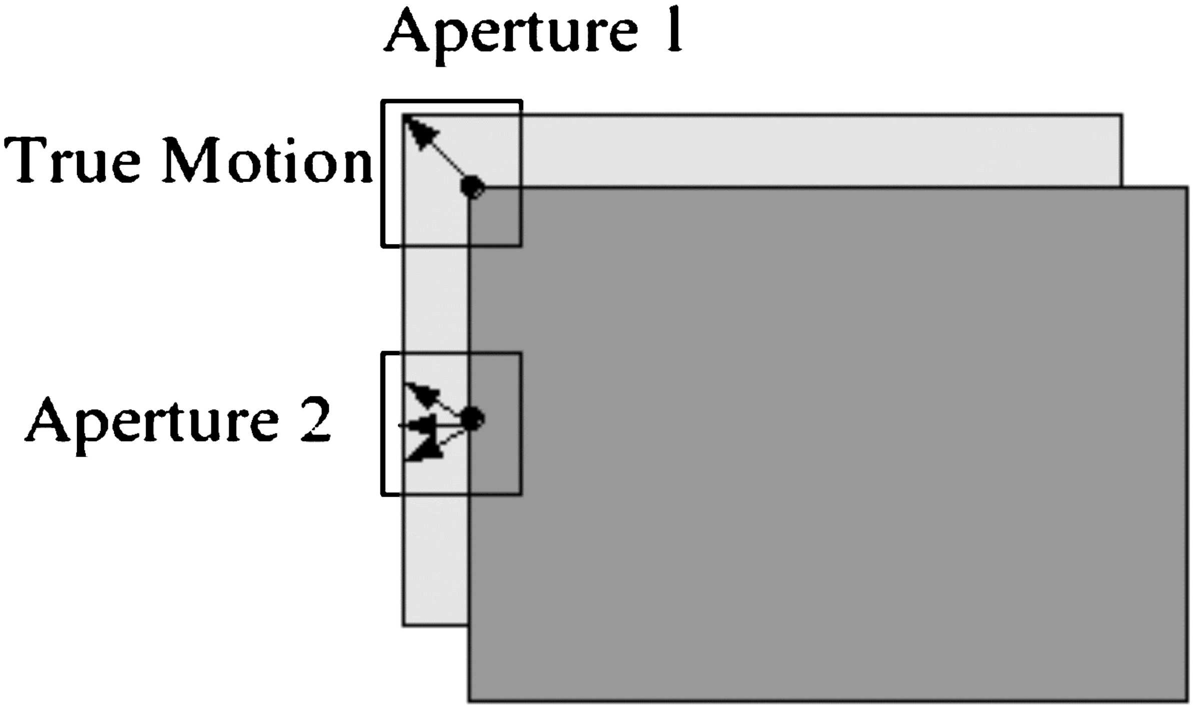


Оптический поток

- Метод ЛК применяется итеративно:
    - Вычислили поток
    - Преобразовали изображение
    - …
    - Вычислили поток
    - Преобразовали изображение
    - И так пока на сошлись или число итераций не превысит порог



Aperture problem



Не всегда по локальной окрестности пожно понять куда сдвинулась точка

Aperture problem: метод ЛК

$$A^T A \begin{pmatrix} u \\ v \end{pmatrix} = A^T b$$ -- система 2x2, дает решение МНК

$$A^T A = M = \sum_{x,y} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$ -- структурный тензор (мы уже сталкивались с ним)

Aperature problem выражается в том, что матрица M оказывается вырожденной или плохообусловленной:

$$cond(M) = \lambda_{2(max)} / \lambda_{1(min)}$$

Aperture problem: метод ЛК

$$A^T A \begin{pmatrix} u \\ v \end{pmatrix} = A^T b$$ -- система 2x2, дает решение МНК

$$A^T A = M = \sum_{x,y} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$ -- структурный тензор (мы уже сталкивались с ним)

Как бороться с проблемой?

Aperature problem выражается в том,
что матрица M оказывается вырожденной
или плохообусловленной:

$$cond(M) = \lambda_{2(max)} / \lambda_{1(min)}$$

Aperture problem: метод ЛК

$$A^T A \begin{pmatrix} u \\ v \end{pmatrix} = A^T b$$ -- система 2x2, дает решение МНК

$$A^T A = M = \sum_{x,y} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$ -- структурный тензор (мы уже сталкивались с ним)

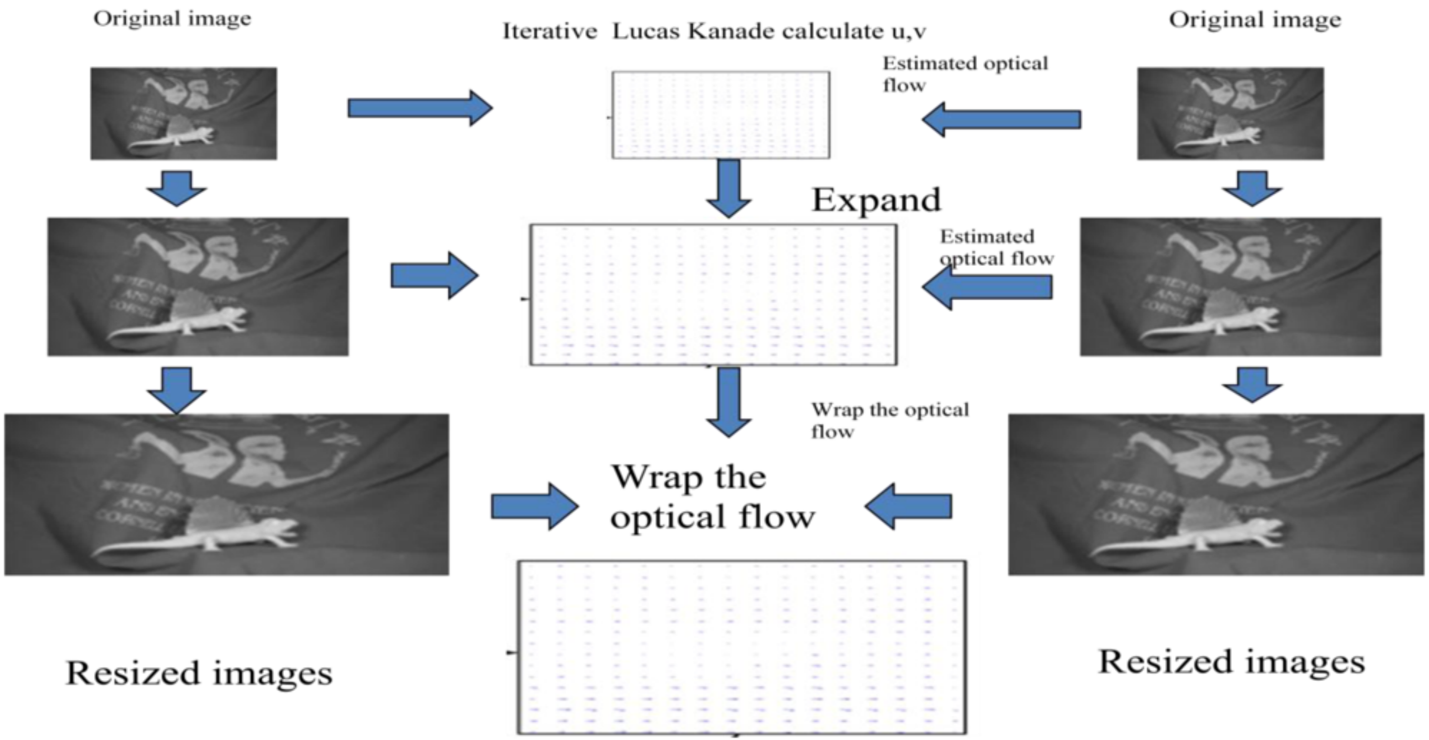
Aperature problem выражается в том, что матрица M оказывается вырожденной или плохообусловленной:

$$cond(M) = \lambda_{2(max)} / \lambda_{1(min)}$$

Как бороться с проблемой?

- Считать в хороших точках (угловых)
- Использовать нелокальные методы (которые предполагают гладкость потока)
- Постобработка (т.е. попытайтесь сделать ваше приложение робастным к встречающимся иногда ошибка оценки потока)

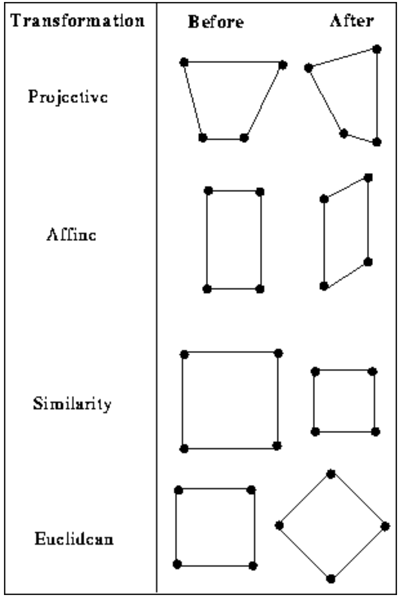
Пирамидальный метод ЛК



`cv::calcOpticalFlowPyrLK`

Оценка движения в 2D

Оценка движения объекта в 2d



Пример: модель движения поворот, сдвиг, масштабирование (изотропное)

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = s \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Параметры движения объекта: a,b,t_x,t_y
Одно соответствие: два уравнения

Оценка движения объекта в 2d

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} a & -b & t_x \\ b & a & t_y \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}, i = \overline{1, n}$$

Оценка движения объекта в 2d

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} a & -b & t_x \\ b & a & t_y \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}, i = \overline{1, n}$$

$$\downarrow$$

$$A \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} = b$$ -- (переопределенная) система 2n линейных уравнений

Оценка движения объекта в 2d

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} a & -b & t_x \\ b & a & t_y \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}, i = \overline{1, n}$$

$$A \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} = b$$ -- (переопределенная) система 2n линейных уравнений

Можно воспользоваться МНК, но в случае наличия хотя бы одного неправильного соответствия **все будет испорчено**

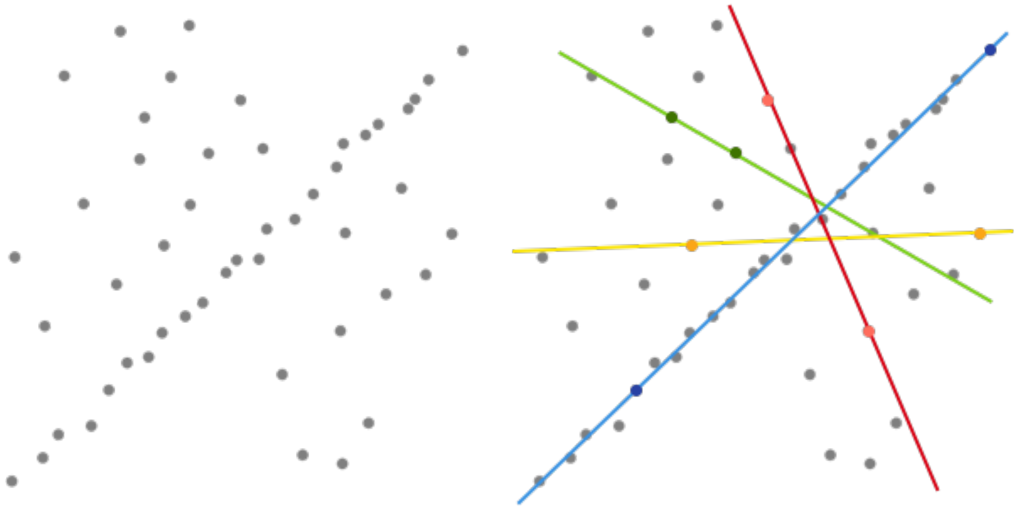
$$\left\| A \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} - b \right\|_2^2 \to min$$

RANdom SAmple Consensus (RANSAC)

Рандомизированный алгоритм оценки параметров модели, устойчивый к выбросам

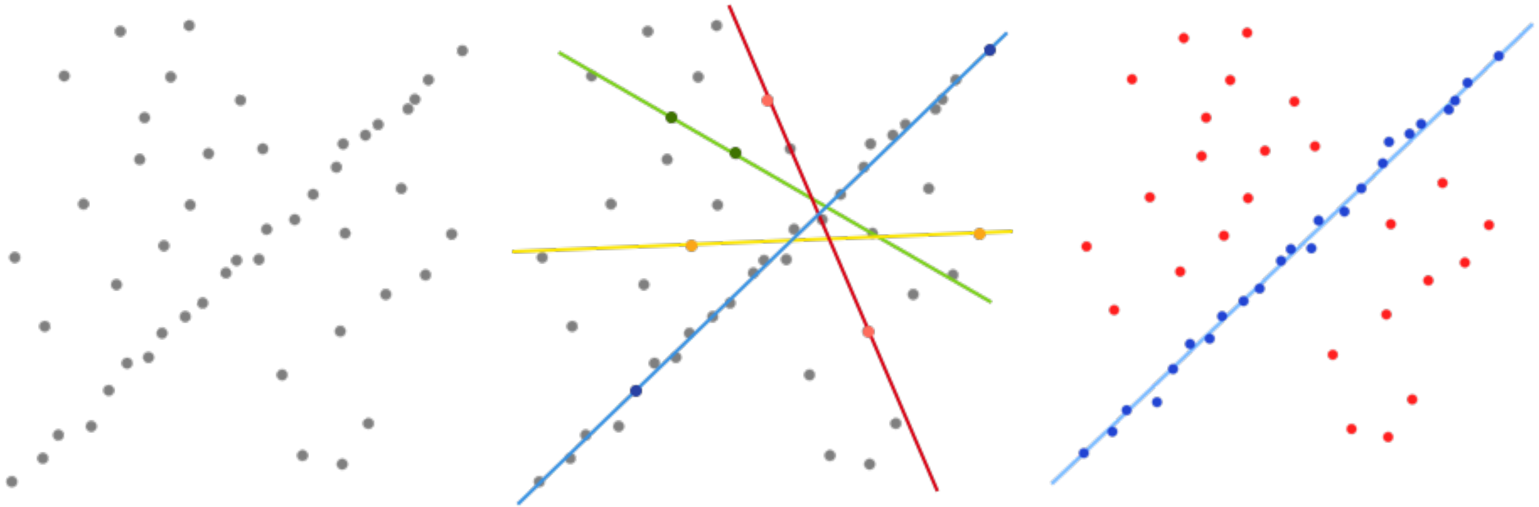
RANdom SAmple Consensus (RANSAC)

Рандомизированный алгоритм оценки параметров модели, устойчивый к выбросам



RANdom SAmple Consensus (RANSAC)

Рандомизированный алгоритм оценки параметров модели, устойчивый к выбросам



RANSAC

Необходимо иметь:

- Метод оценки параметров по подмножеству данных
- Способ оценки качества модели

1. Пока на превышен лимит по итерациям:
   a. Из n соответствий выбрать m
   b. Решить (МНК) задачу на подмножестве из m точек:

   $$\left\| A \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} - b \right\|_2^2 \rightarrow min$$

   c. Оценить полученные параметры движения (количество точек, которые подчиняются данному закону с заданной степенью точности)
   d. Запомнить лучшую модель (и точки, которые ей удовлетворяют)
2. Уточнить модель на всех лучших точках лучшей модели

RANSAC

Необходимо иметь:

- Метод оценки параметров по подмножеству данных
- Способ оценки качества модели

Как оценить число итераций? Как быть уверенным в корректности результата (ведь алгоритм рандомизированный)?

1. Пока на превышен лимит по итерациям:
   a. Из n соответствий выбрать m
   b. Решить (МНК) задачу на подмножестве из m точек:

   $$\left\| A \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} - b \right\|_2^2 \to min$$

   c. Оценить полученные параметры движения (количество точек, которые подчиняются данному закону с заданной степенью точности)
   d. Запомнить лучшую модель (и точки, которые ей удовлетворяют)
2. Уточнить модель на всех лучших точках лучшей модели. Т.е. еще раз решить:

$$\left\| A \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} - b \right\|_2^2 \to min$$

RANSAC для оценки движения

ε -- относительное число хороших матчей

RANSAC для оценки движения

ε -- относительное число хороших матчей

ε^m

RANSAC для оценки движения

ε -- относительное число хороших матчей

ε^m

$1 - \varepsilon^m$

RANSAC для оценки движения

ε -- относительное число хороших матчей

ε^m

$1 - \varepsilon^m$

$(1 - \varepsilon^m)^N$

RANSAC для оценки движения

ε -- относительное число хороших матчей

$$\varepsilon^m$$

$$1 - \varepsilon^m$$

$$(1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N$$

RANSAC для оценки движения

ε -- относительное число хороших матчей

$$\varepsilon^m$$

$$1 - \varepsilon^m$$

$$(1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N > p$$

RANSAC для оценки движения

ε -- относительное число хороших матчей

$$\varepsilon^m$$

$$1 - \varepsilon^m$$

$$(1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N > p$$

$$N > \frac{\log(1-p)}{\log(1-\varepsilon^m)}$$

-- число итераций, необходимое для нахождения правильной модели с вероятностью > p

RANSAC для оценки движения

ε -- относительное число хороших матчей

$$\varepsilon^m$$

$$1 - \varepsilon^m$$

$$(1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N$$

$$1 - (1 - \varepsilon^m)^N > p$$

$$N > \frac{log(1-p)}{log(1-\varepsilon^m)}$$

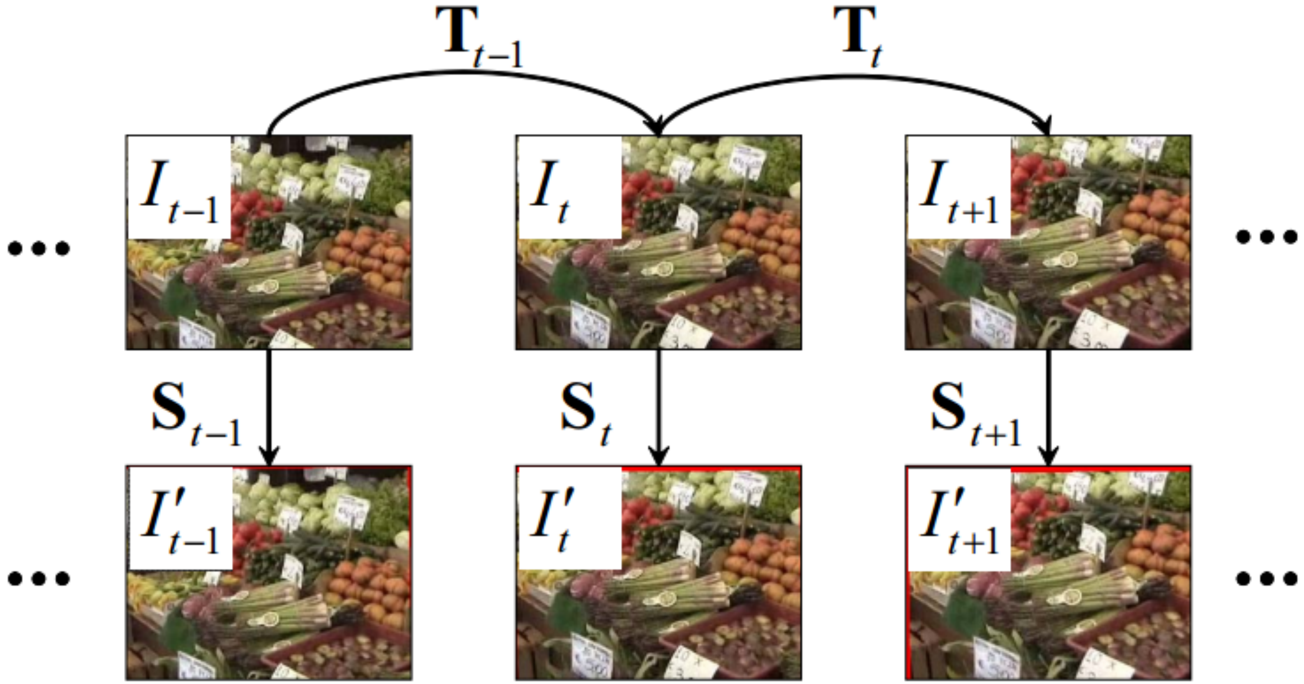
-- число итераций, необходимое для нахождения правильной модели с вероятностью > p

Пример: eps=0.3, m=2, p=0.99 => **N>49**
eps=0.1, m=2, p=0.99 => **N>459**
eps=0.3, m=3, p=0.99 => **N>169**
eps=0.3, m=2, p=0.95 => **N>32**

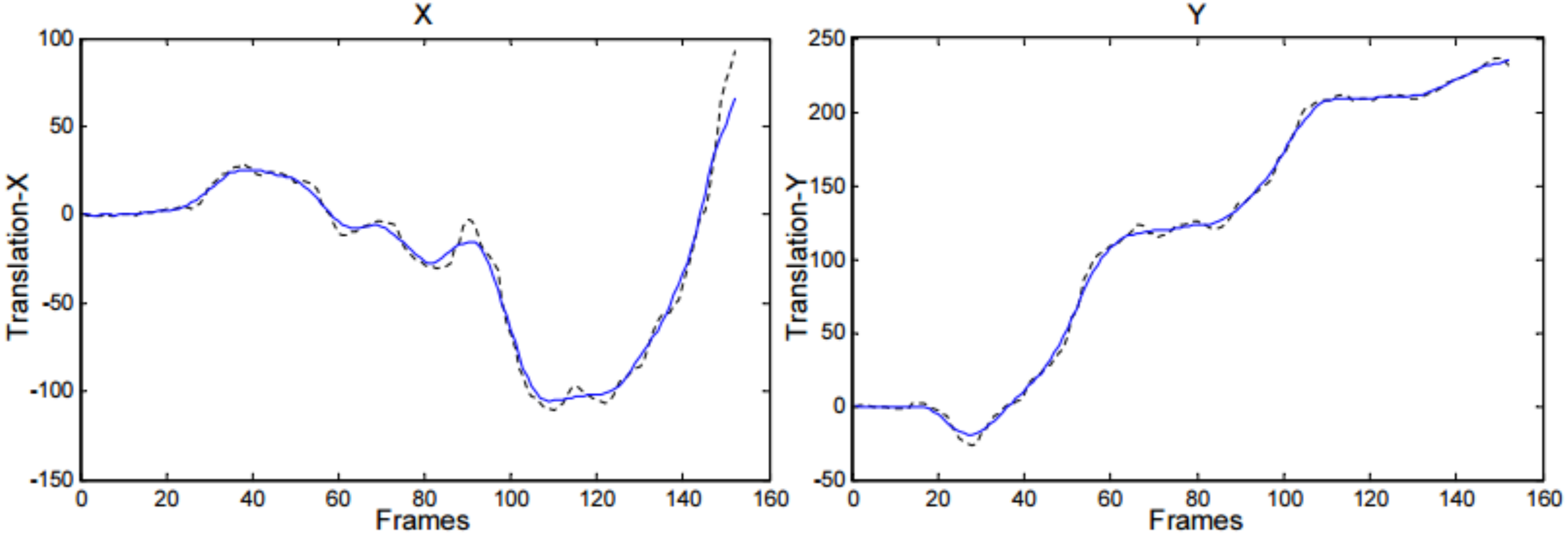
`cv::videostab::estimateGlobalMotionLeastSquares,`
`cv::videostab::estimateGlobalMotionRansac`

Видеостабилизация

Видеостабилизация



Видеостабилизация



Видеостабилизация

видео