

академия
больших
данных

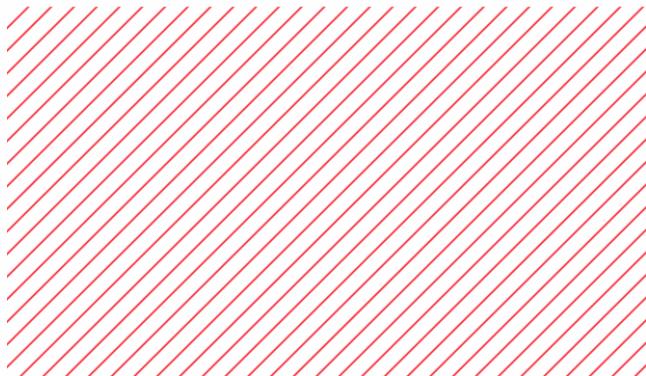


mail.ru
group

Сопоставление изображений и локальные дескрипторы

Андрей Савченко

Профессор НИУ ВШЭ-Нижний Новгород





Опрос

<https://forms.gle/FTuMJuSsP8AP2ecY8>

Некоторые задачи

Robust tracking/detection

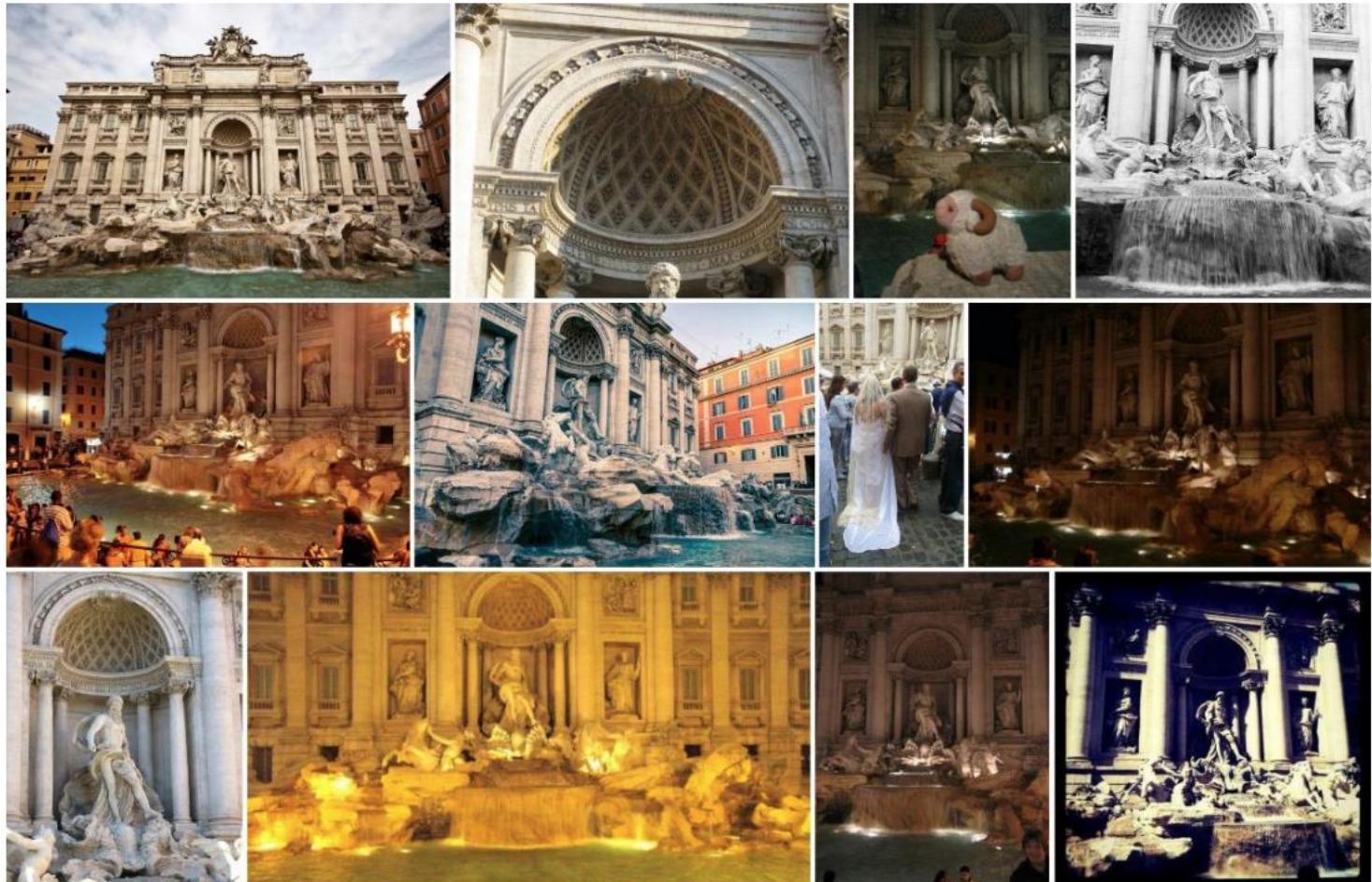


Lepetit et al «Randomized Trees for Real-Time Keypoint Recognition», CVPR 2005

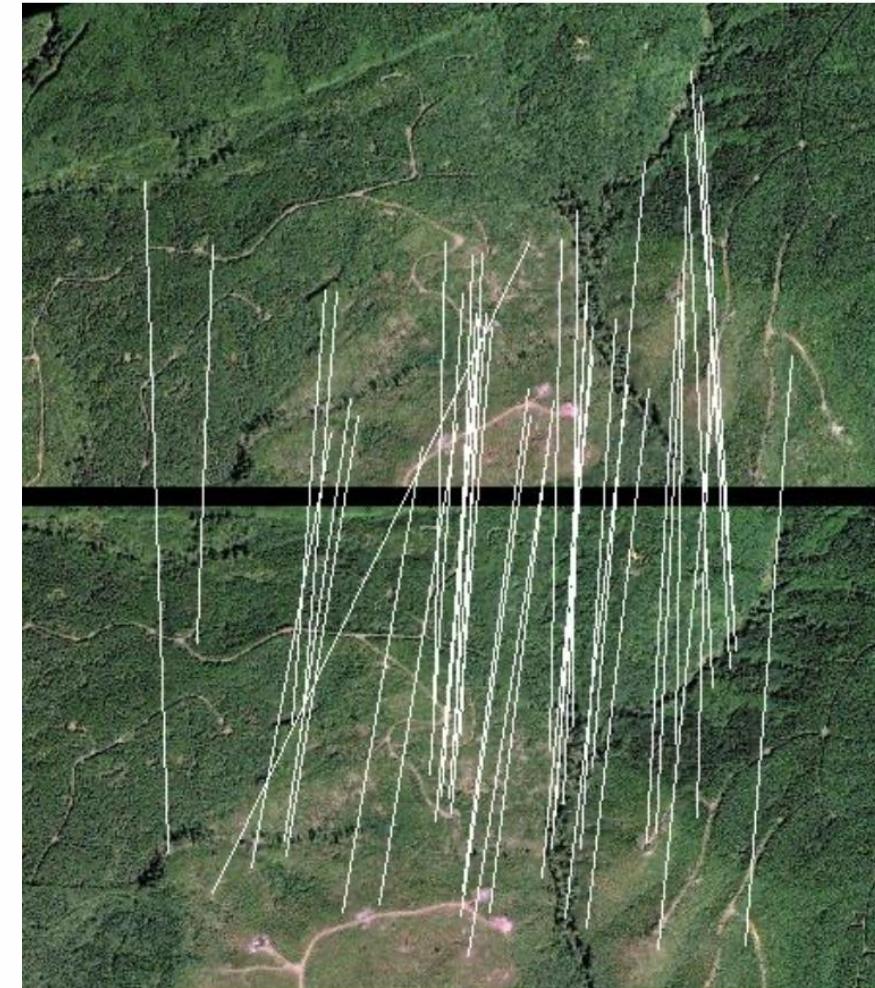
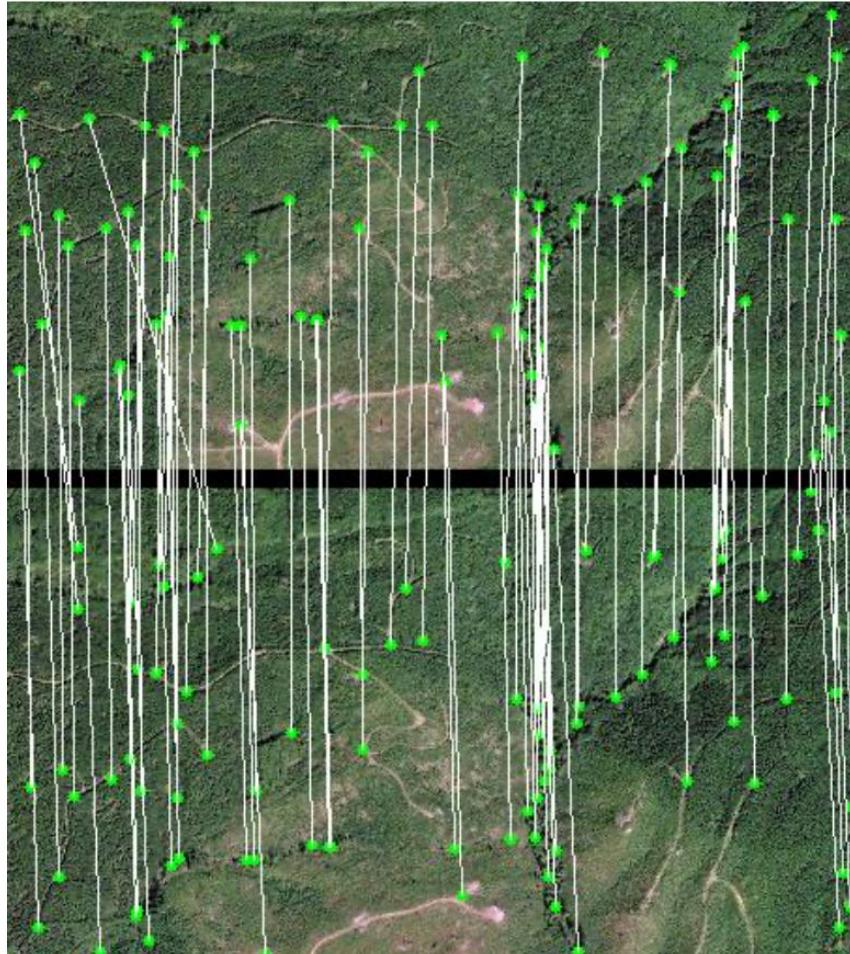
Создание панорам



Photo-tourism dataset

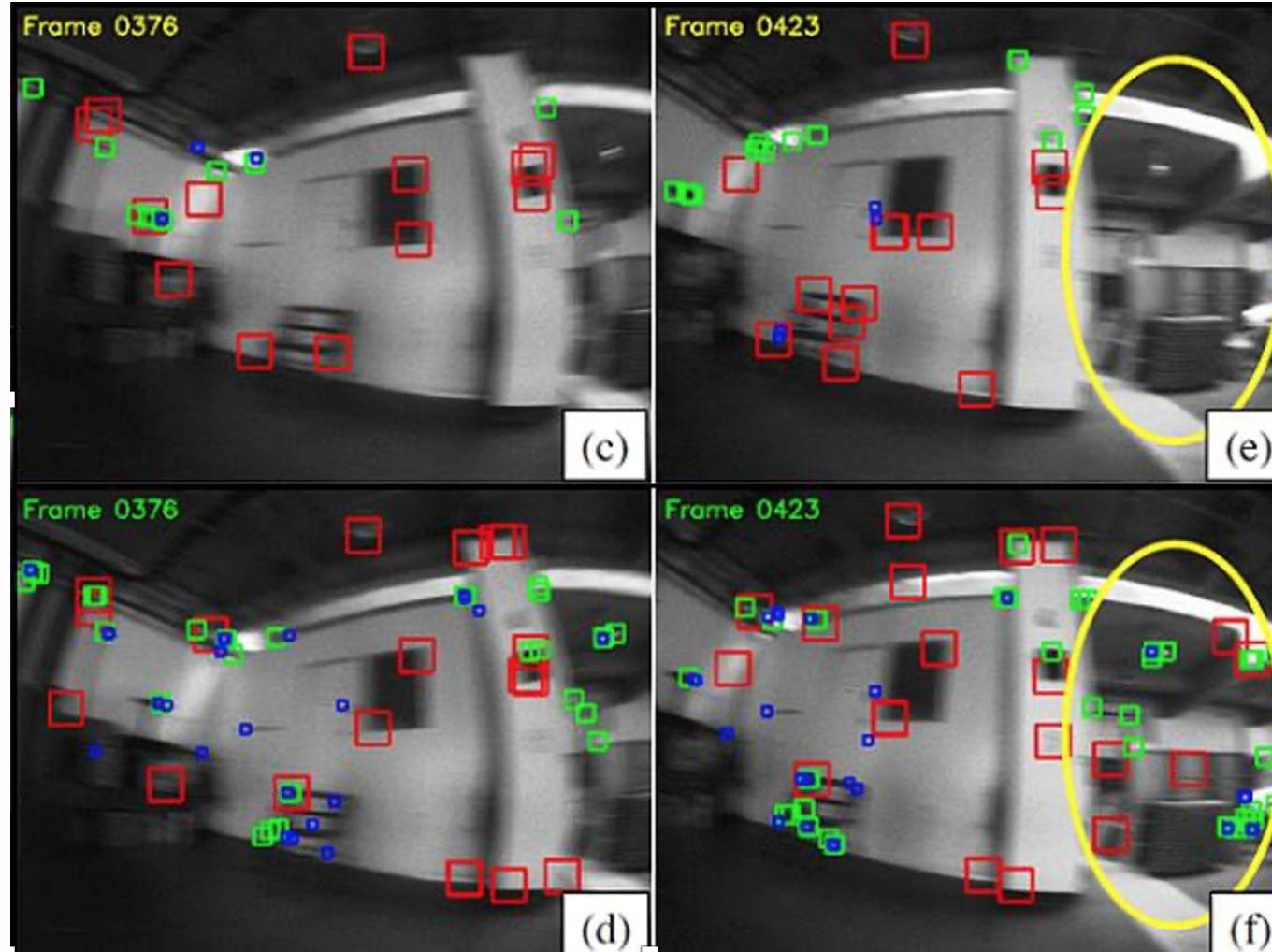


Навигация беспилотных летательных аппаратов

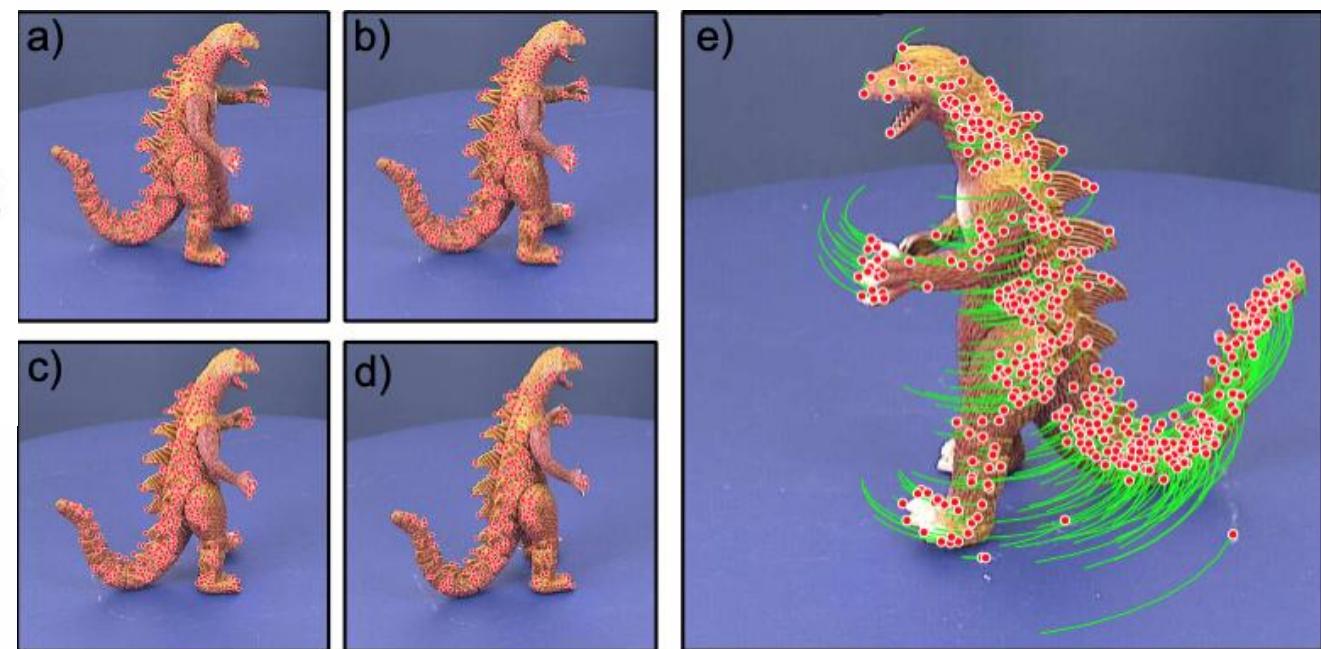
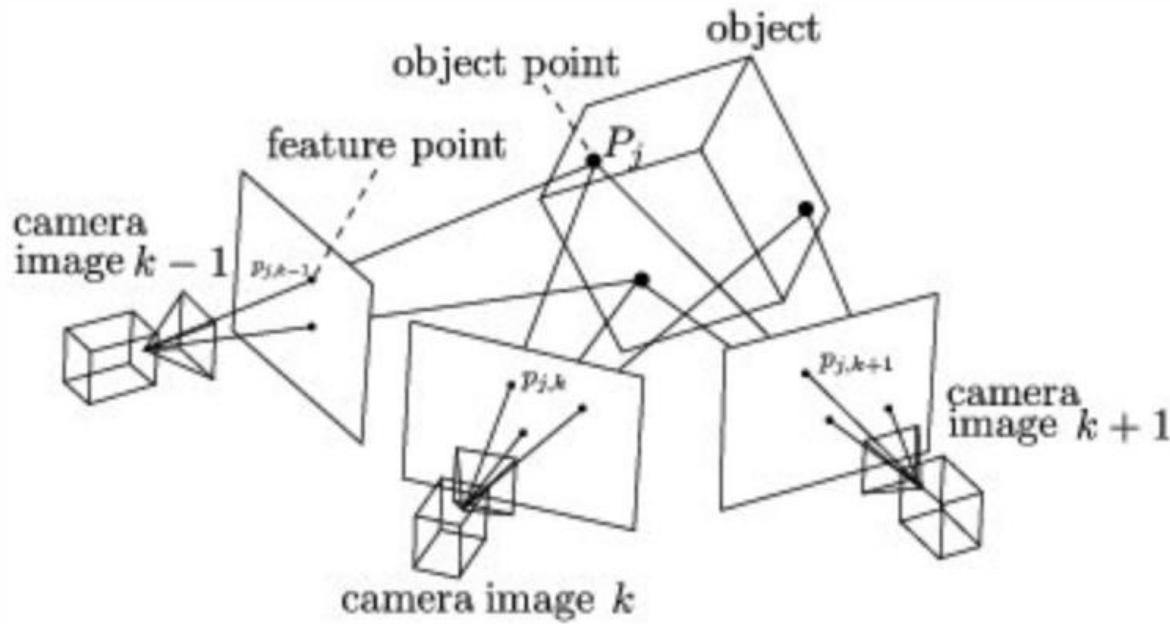


Степанов дисс. к.т.н. 2016

Simultaneous Localization and Mapping (SLAM)



Structure from motion (SoM)



Moulon et al «Adaptive Structure from Motion with a Contrario Model Estimation», ACCV 2012

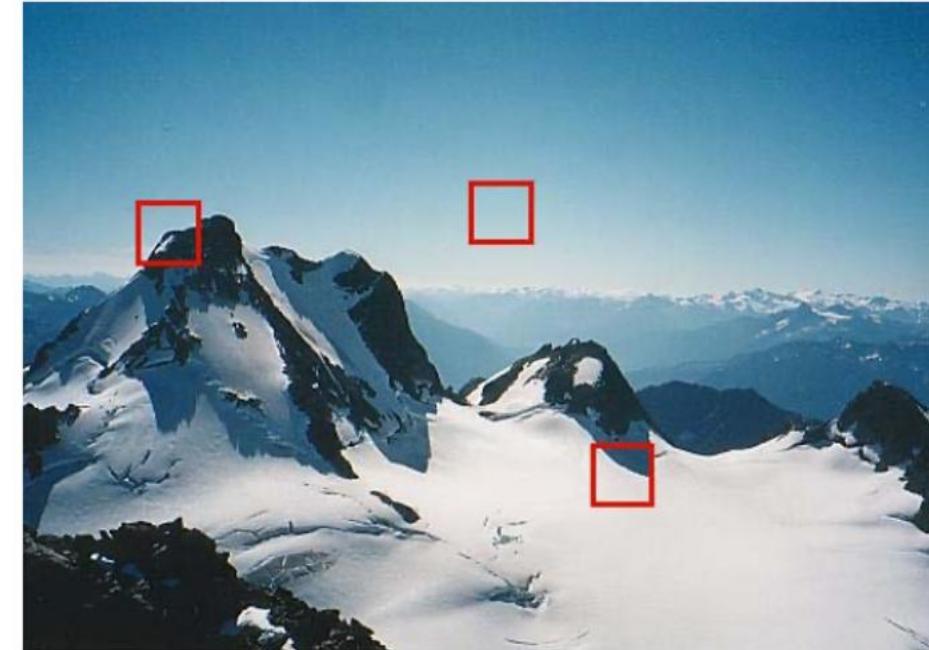
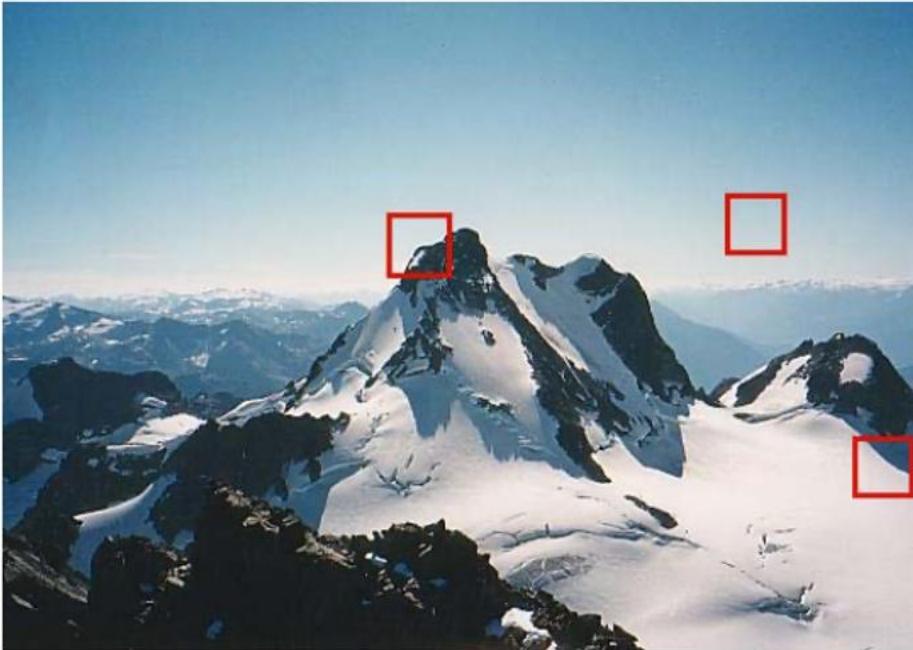
Prince «Computer vision: models, learning and inference»

Общая задача: image matching

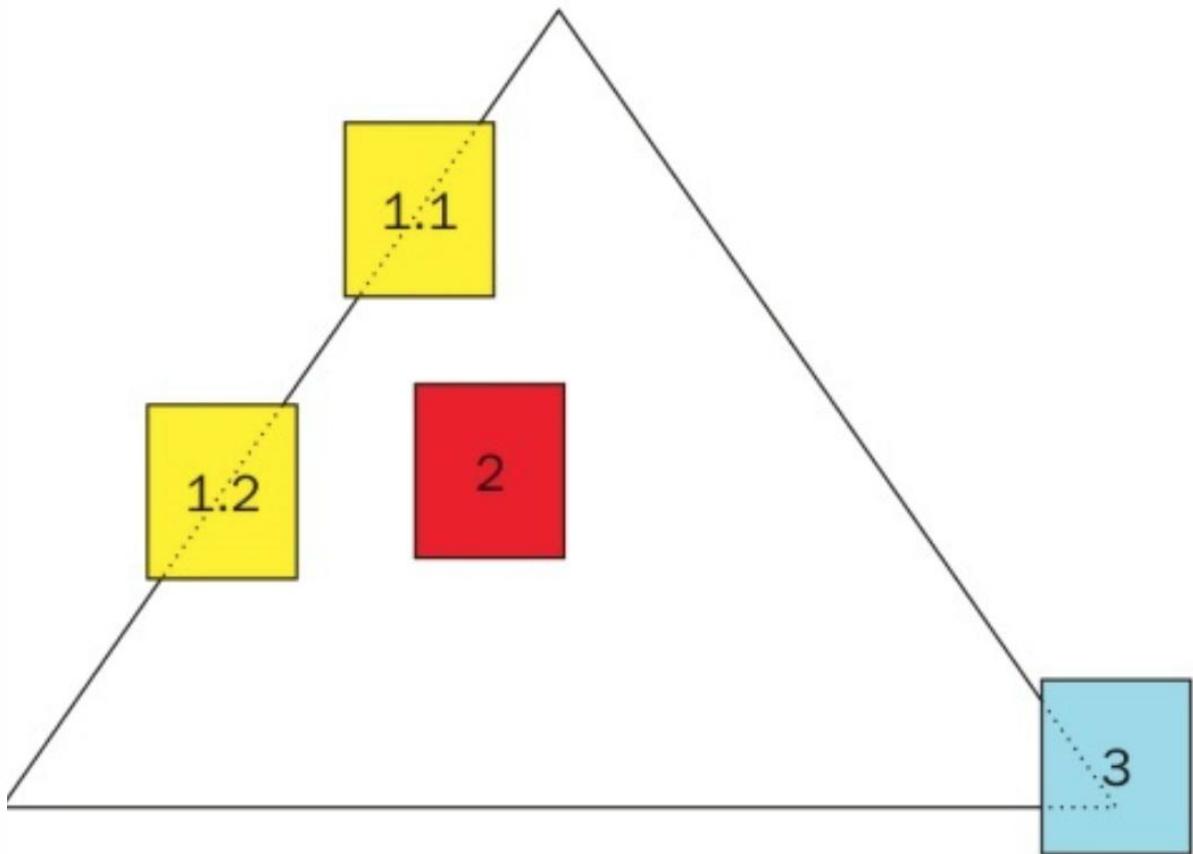


Особые (ключевые) точки

Окрестности (patches) локальных особенностей



Interest/(local) feature/key points



1 – край (edge)

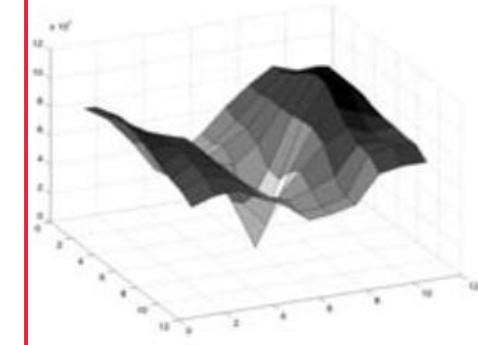
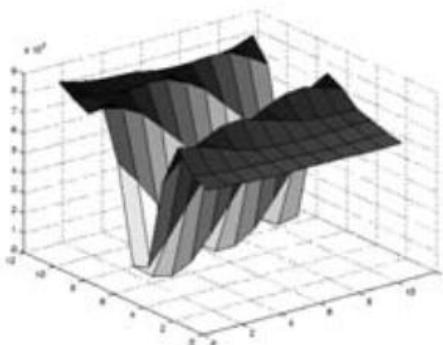
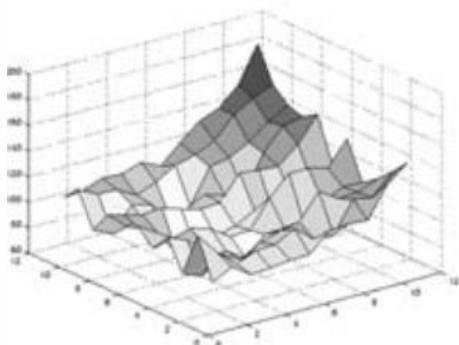
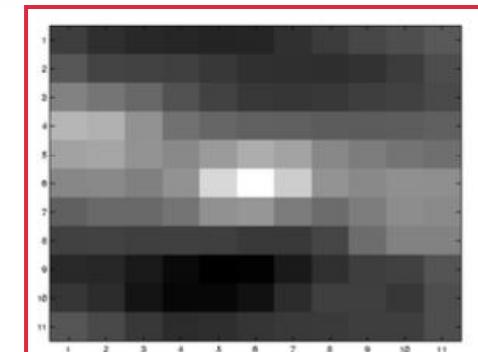
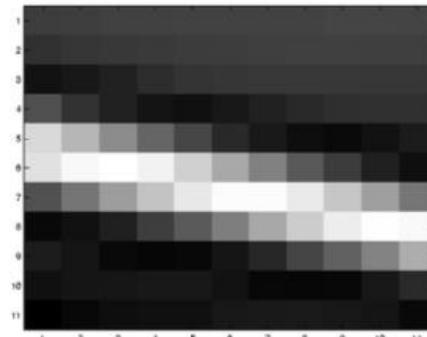
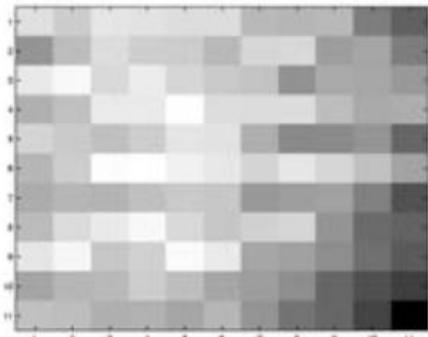
2 – монотонная область

3 – угол (corner)

Детектор углов Харриса (1)

Ищутся точки с резко меняющейся автокорреляционной поверхностью

$$E_{AC}(\Delta \mathbf{u}) = \sum_i w(\mathbf{x}_i) [I_0(\mathbf{x}_i + \Delta \mathbf{u}) - I_0(\mathbf{x}_i)]^2$$



Детектор углов Харриса (2)

Разложение в ряд Тейлора

$$[I_0(\mathbf{x}_i + \Delta\mathbf{u}) - I_0(\mathbf{x}_i)]^2 \approx [I_0(\mathbf{x}_i) + \nabla I_0(\mathbf{x}_i) \cdot \Delta\mathbf{u} - I_0(\mathbf{x}_i)]^2$$

$$= [I_x u + I_y v]^2 = I_x^2 u^2 + 2I_x I_y u v + I_y^2 v^2$$

Отклики фильтров
горизонтальных/ вертикальных
частных производных

$$= (\mathbf{u}, \mathbf{v}) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} (\mathbf{u}, \mathbf{v})^T$$

$$\nabla I_0(\mathbf{x}_i) = \left(\frac{\partial I_0}{\partial x}, \frac{\partial I_0}{\partial y} \right)(\mathbf{x}_i)$$

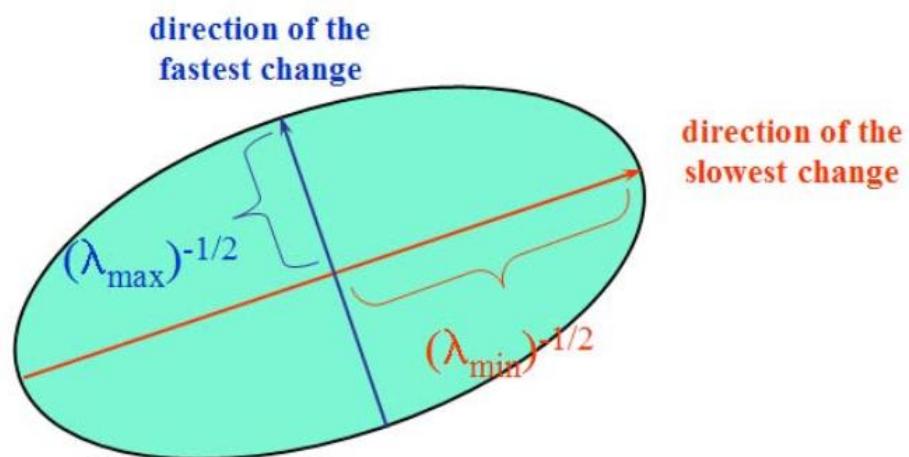
Детектор углов Харриса (3)

Автокорреляционная матрица
(Image structure tensor)

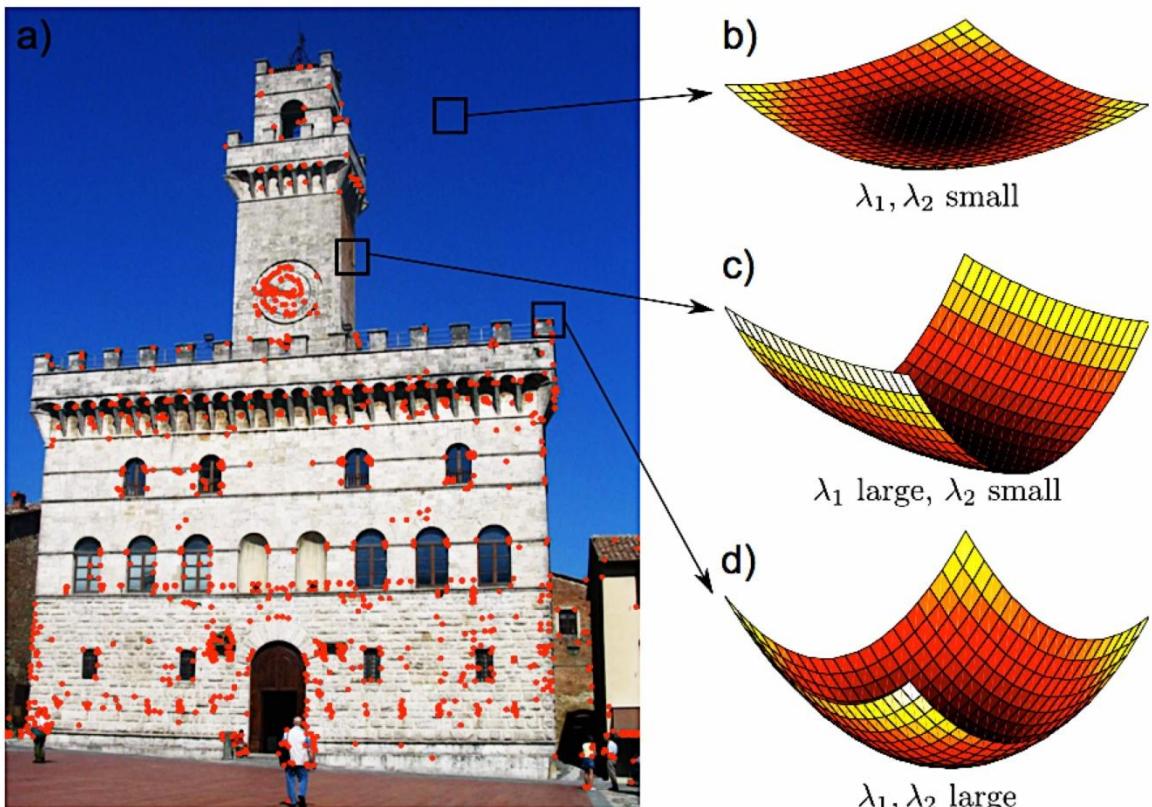
$$E_{AC}(\Delta u) = \Delta u^T A \Delta u$$

$$A = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Симметричную матрицу А можно разложить на собственные векторы



Детектор углов Харриса (4)



OpenCV: `cornerHarris()`

1. Ищется максимум минимального собственного значения [Shi and Tomasi, 1994].

2. [Harris and Stephens, 1988] ($k=0.06$):

$$\det(\mathbf{A}) - \alpha \operatorname{trace}(\mathbf{A})^2 = \lambda_1 \lambda_2 - \kappa(\lambda_1^2 + \lambda_2^2)$$

3. [Triggs, 2004] ($\alpha=0.05$): $\lambda_1 - \alpha \lambda_2$

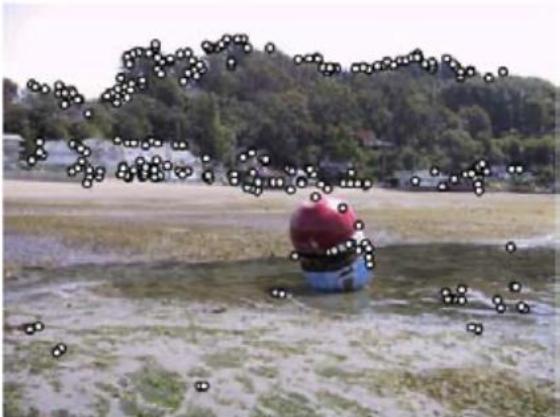
Алгоритм

1. Фильтр краев (первые производные)
2. Вычислить 3 изображения - произведения компонент градиента
3. Сгладить все 3 изображения фильтром Гаусса с большим разбросом σ
4. Выбрать точки с локальным максимумом скалярной меры «интереса»

Adaptive non-maximal suppression

Проблема – много близко расположенных особых точек

[Brown, Szeliski, and Winder, 2005]: среди локальных максимумов выбираются те, для которых мера «особенности» значительно (более чем на 10%) превышает меру «особенности» всех ее соседних точек в окрестности заданного радиуса r



(a) Strongest 250

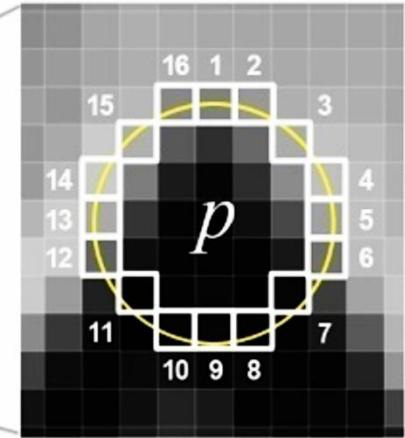


(c) ANMS 250, $r = 24$

Детектор FAST (Features from Accelerated Segment Test)

Тестирование 16 пикселей в окрестности, детектирование угла, если 12 идущих подряд из них ярче/темнее центрального пикселя

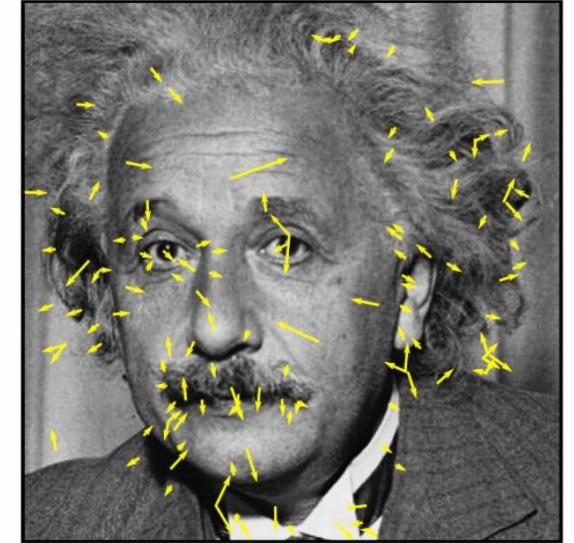
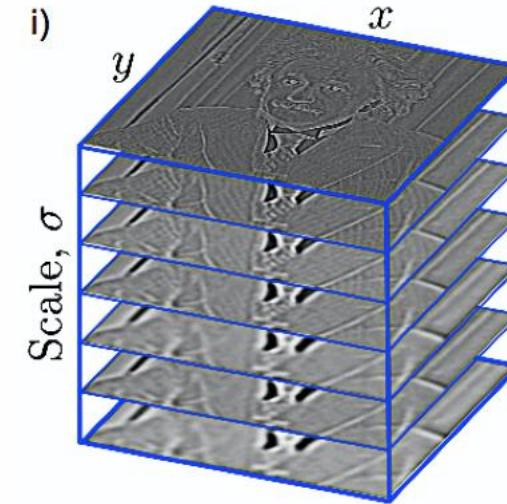
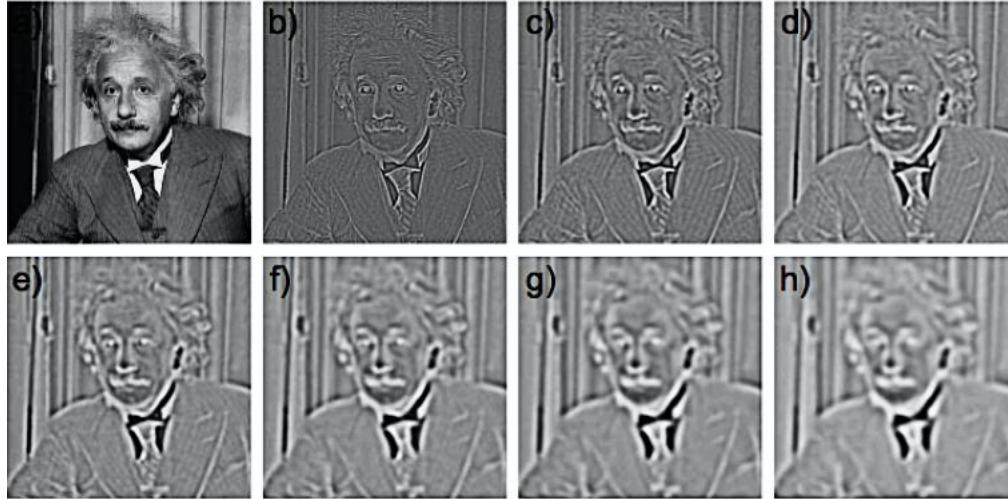
В начале проверяются только 4 пикселя
(1, 9, 5, 13)



Инвариантность к масштабу. Пирамида изображений



Детектор SIFT (Scale-Invariant Feature Transform)



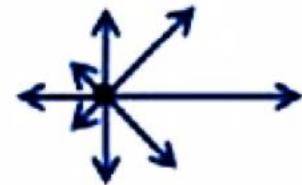
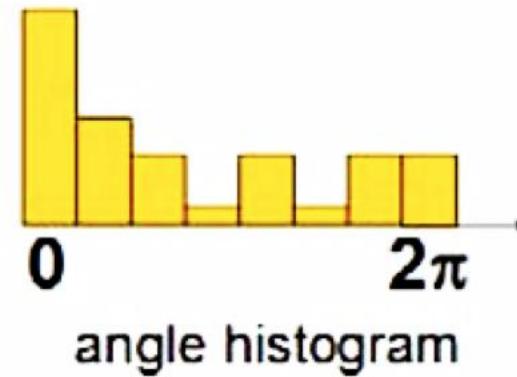
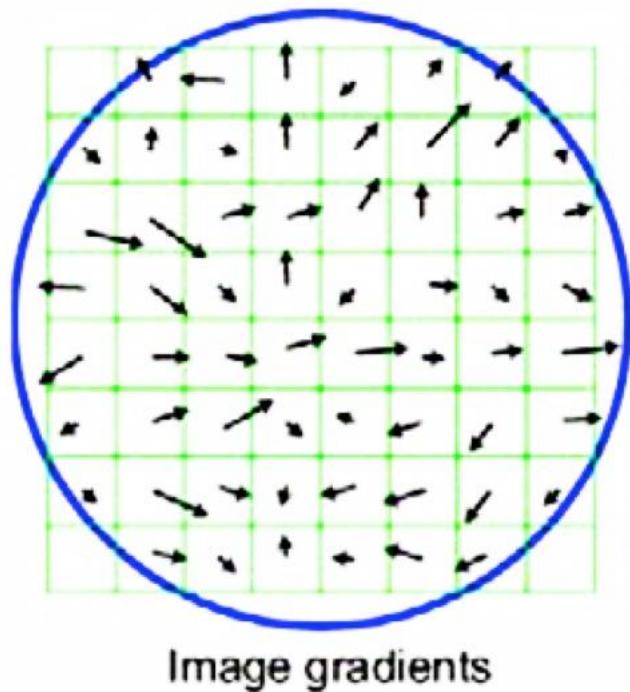
b-h) DoG-фильтрация с уменьшающейся дисперсией

i) Пирамида - 3D volume. Поиск в пирамиде экстремальных точек – интенсивность больше/меньше всех ($3 \times 3 \times 3 - 1 = 26$ соседей)

- Проверка условий детектора Харриса
- Построение гистограммы ориентации градиента в локальной окрестности (36 блоков на 360 градусов) и выбор одного или двух пиков в гистограмме

Дескрипторы

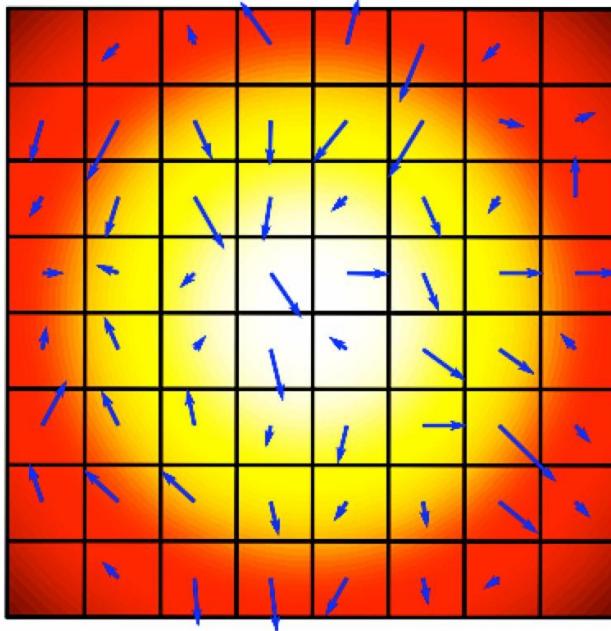
Гистограмма ориентаций градиента



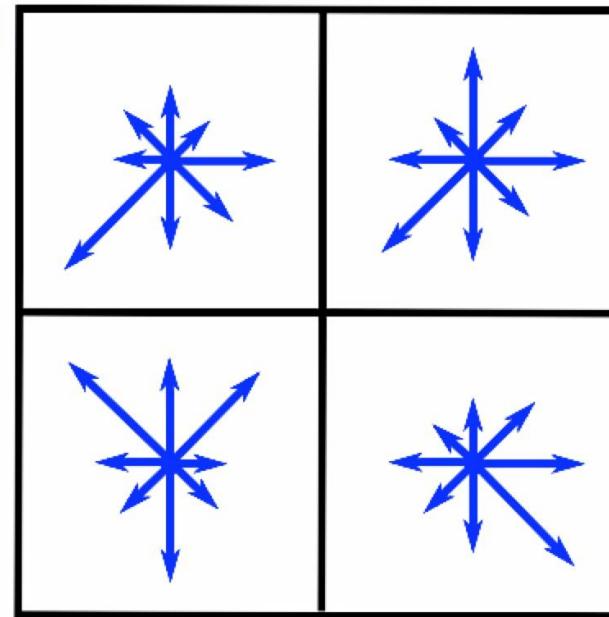
[Lowe, 2004]

Дескриптор SIFT

a)

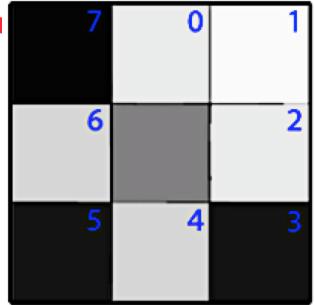


b)



1. 16x16 окрестность точки разбивается сеткой 4x4 ячейки
2. В каждой ячейке вычисляется взвешенная гистограмма ориентации градиента (8 блоков)
3. Гистограммы всех ячеек нормируются по ориентации ключевой точки и объединяются в единый SIFT дескриптор (размерность $4 \times 4 \times 8 = 128$)

Локальные бинарные шаблоны (Local Binary Patterns, LBP)



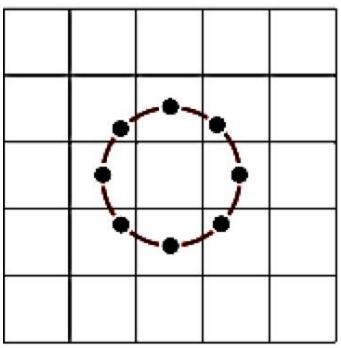
42	7	199	0	234	1
177	6	129	2	199	2
65	5	177	4	65	3

1	7	0	0	1
0	6	X	0	2
1	5	0	4	3

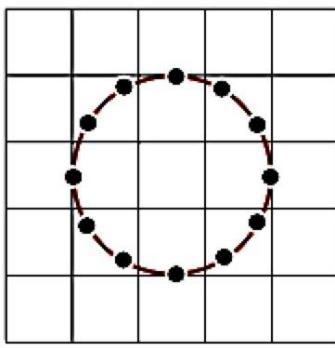
1	7	0	6	1	5	0	4	1	3	0	2	0	1	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

$$128 + 32 + 8 = 168$$

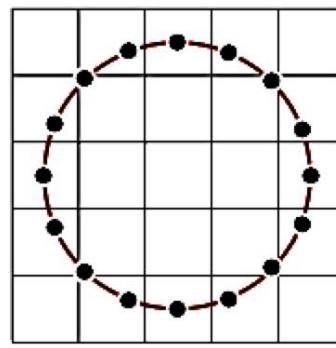
$LBP_{P,R}$



$(P = 8, R = 1.0)$



$(P = 12, R = 1.5)$



$(P = 16, R = 2.0)$

Uniform pattern – число с 0, 1 или 2 переходами 0-1 или 1-0. Примеры: «00111100», «11111111» и «01111111».

Non-uniforms patterns объединяются в один:
«10100000» и «01010101»

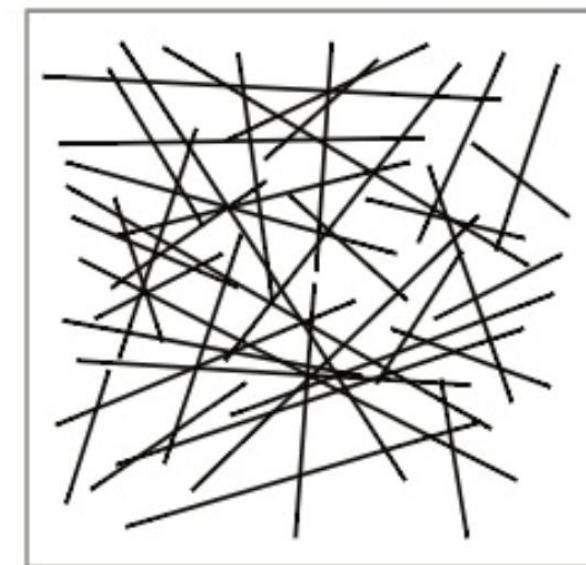
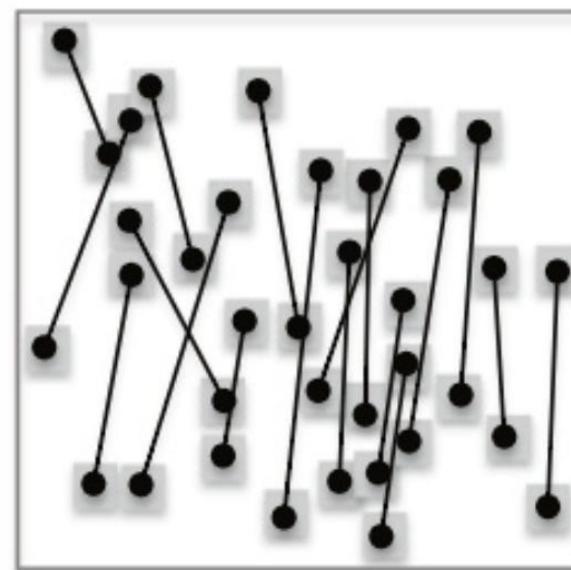
BRIEF (Binary Robust Independent Elementary Features)

-
1. Выполняется сглаживание изображения (чаще всего, фильтр Гаусса)
 2. Рассматриваются квадратные блоки (patch). В каждом выбирается наугад Nd (128,256 или 512) пар точек. Обычно для сэмплирования используется нормальное распределение. Последовательность точек фиксирована!
 3. Для каждой пары точек вычисляется показатель различия

$$\tau(p; x, y) := \begin{cases} 1 & \text{if } p(x) < p(y) \\ 0 & \text{otherwise} \end{cases}$$

4. Итоговый бинарный дескриптор

$$f_{n_d}(p) := \sum_{1 \leq i \leq n_d} 2^{i-1} \tau(p; x_i, y_i)$$



BRIEF не устойчив к повороту

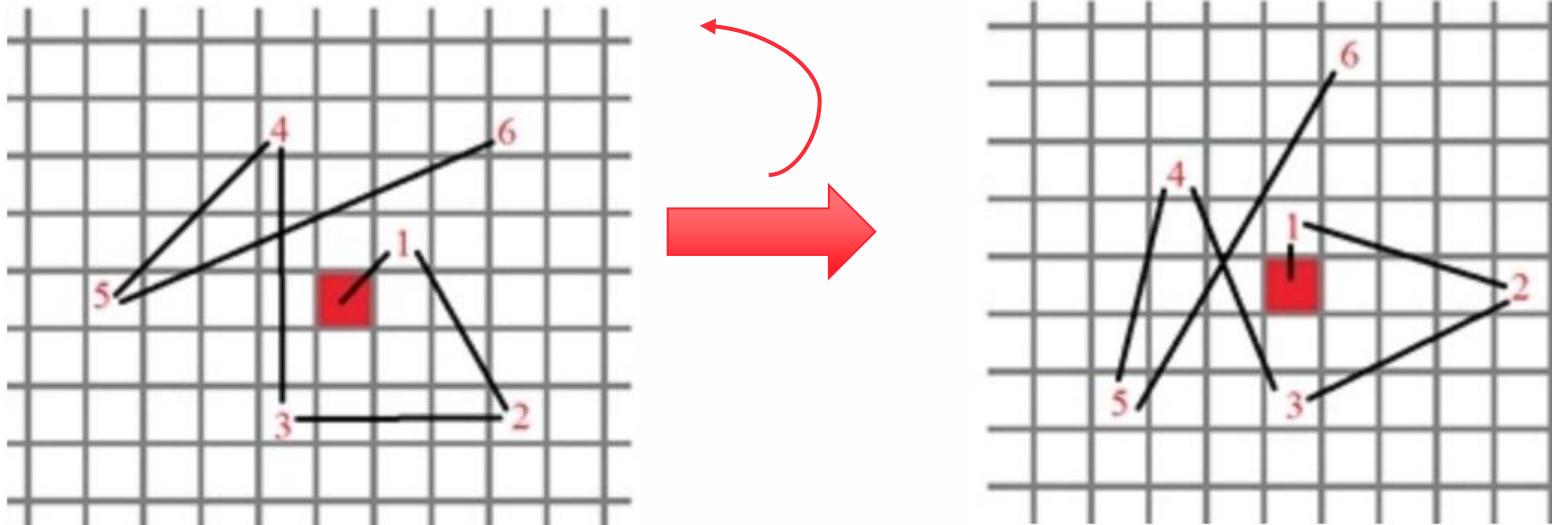
OpenCV: BriefDescriptorExtractor class

Szeliski «Computer Vision»

Muhammad «OpenCV Android Programming By Example»

ORB (Oriented FAST and rotated BRIEF)

1. Особые точки обнаруживаются при помощи быстрого древовидного FAST на исходном изображении и на нескольких изображениях из пирамиды уменьшенных изображений.
2. Вычислим ориентацию градиента в центре и выполним поворот в BRIEF



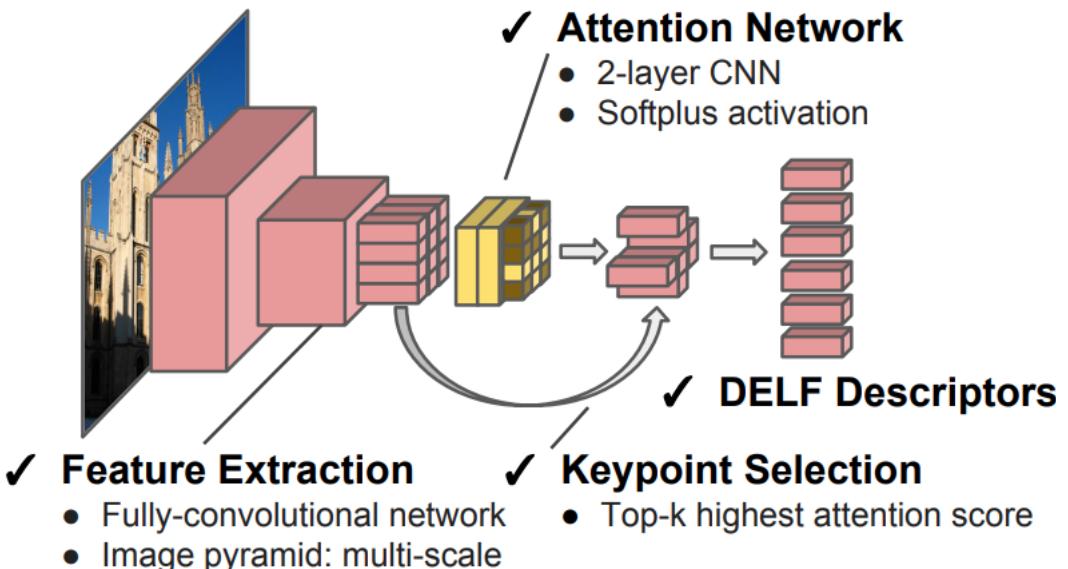
$$\mathbf{S} = \begin{pmatrix} \mathbf{x}_1, \dots, \mathbf{x}_n \\ \mathbf{y}_1, \dots, \mathbf{y}_n \end{pmatrix}$$

$$\mathbf{S}_\theta = \mathbf{R}_\theta \mathbf{S}$$

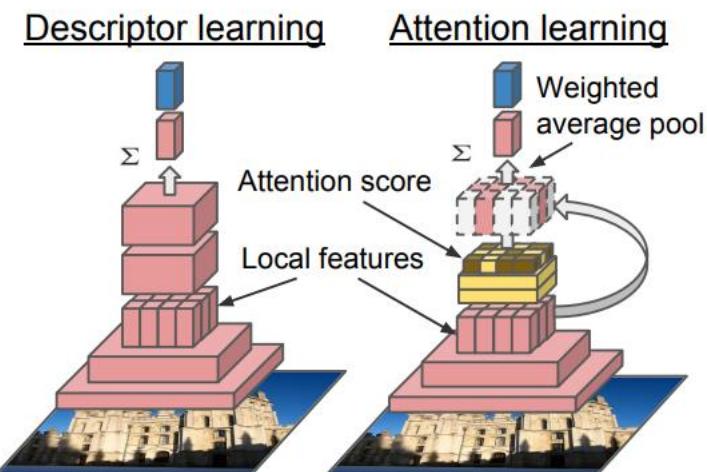
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.370.4395&rep=rep1&type=pdf>

<https://habr.com/ru/post/414459/>

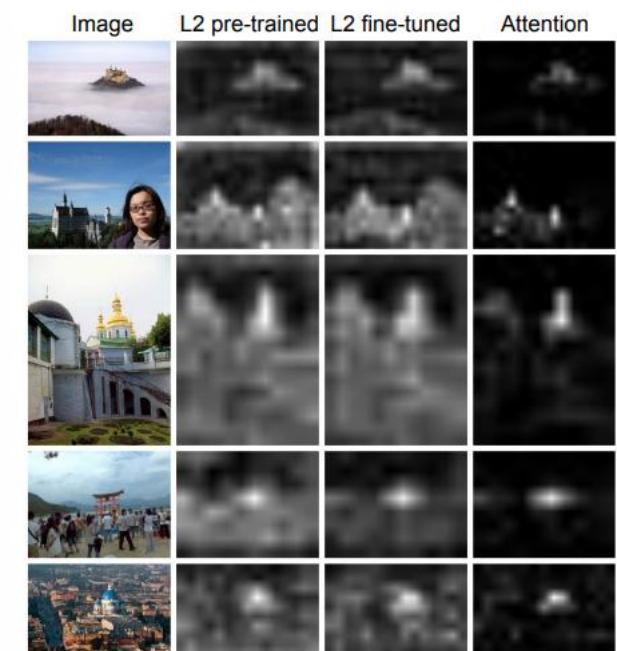
DELF (Deep Local Features)



Два этапа обучения



Выделяются ключевые точки из сетки (grid) с максимальным весом внимания

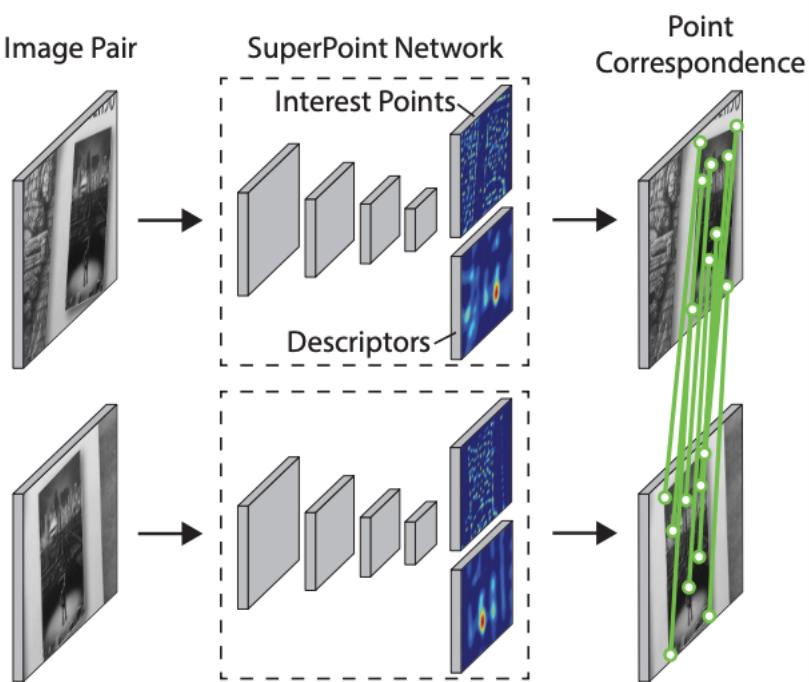


https://openaccess.thecvf.com/content_ICCV_2017/papers/Noh_Large-Scale_Image_Retrieval_ICCV_2017_paper.pdf

<https://andrefaraujo.github.io/files/posters/2017-10-22-iccv-delf-poster.pdf>

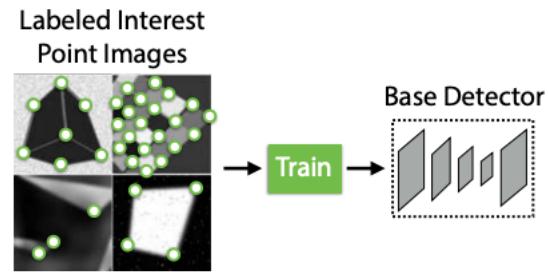
"Which Is Which? Evaluation of Local Descriptors for Image Matching in Real-World Scenarios", CAIP 2019

SuperPoint

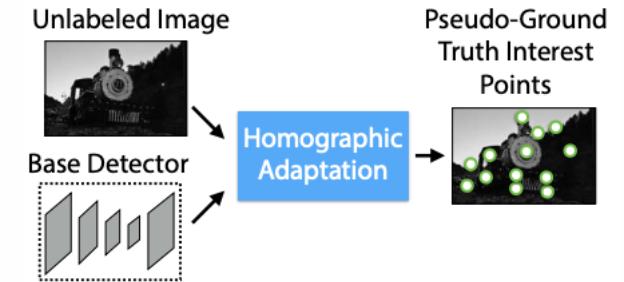


Три этапа самообучения

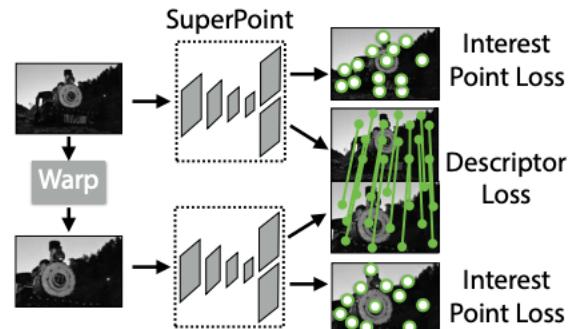
(a) Interest Point Pre-Training



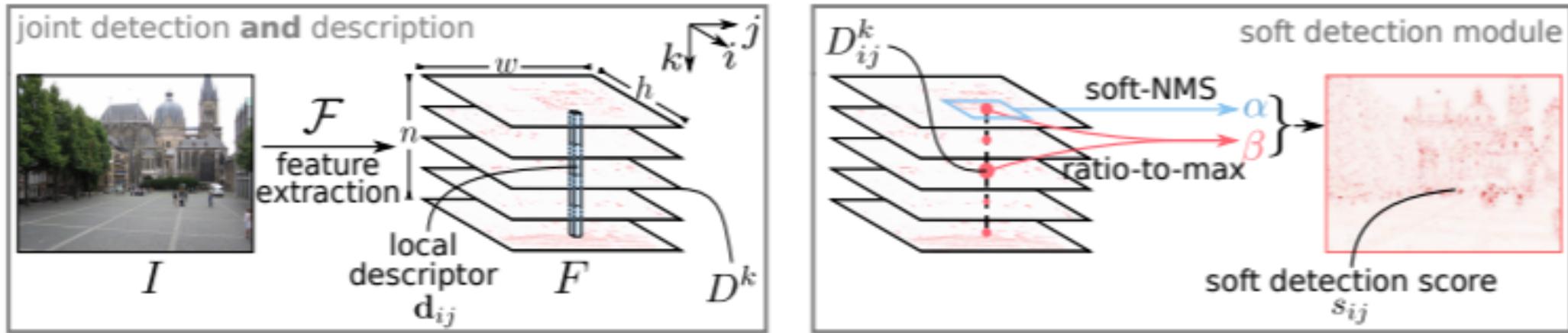
(b) Interest Point Self-Labeling



(c) Joint Training



D2Net: detect-and-describe (D2) network

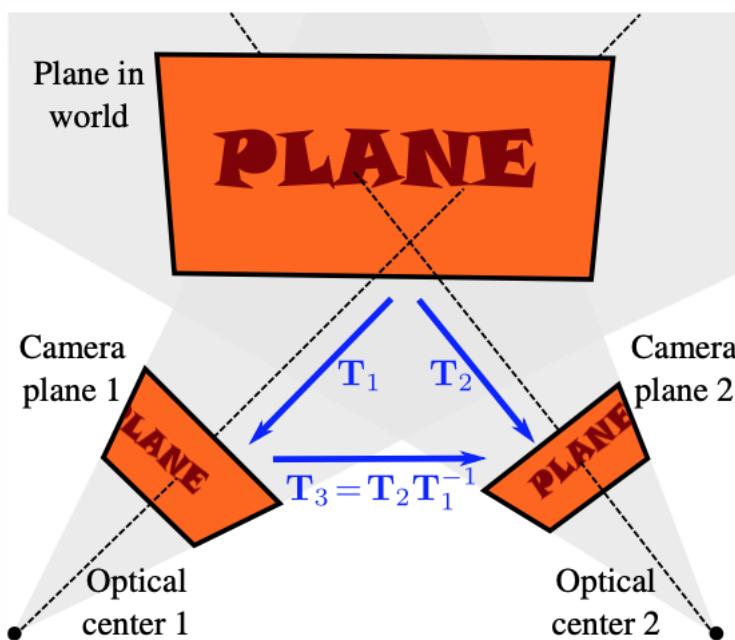


“A feature extraction CNN F is used to extract feature maps that play a dual role:

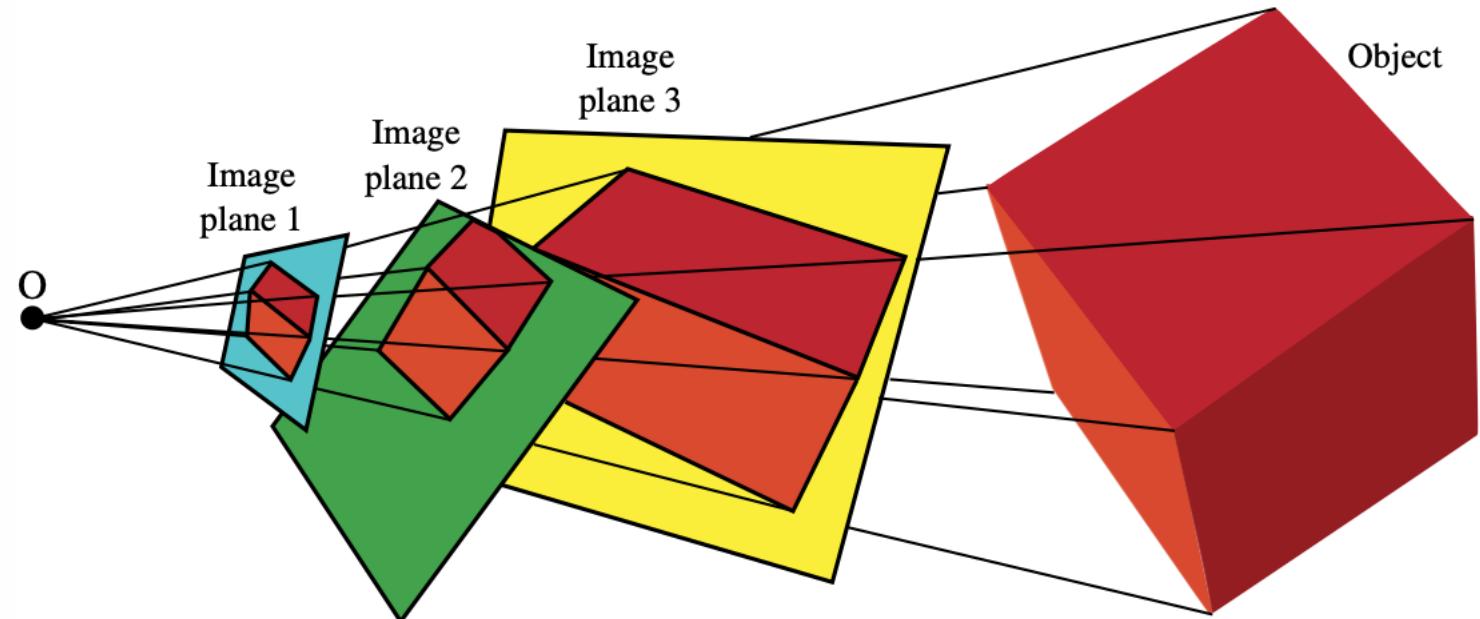
- (i) local descriptors d_{ij} are simply obtained by traversing all the n feature maps D^k at a spatial position (i, j) ;
- (ii) detections are obtained by performing a non-local-maximum suppression on a feature map followed by a non-maximum suppression across each descriptor - during training, keypoint detection scores s_{ij} are computed from a soft local-maximum score α and a ratio-to-maximum score per descriptor β .

Поиск соответствий (matching)

Гомография (Homography)



Геометрическая интерпретация

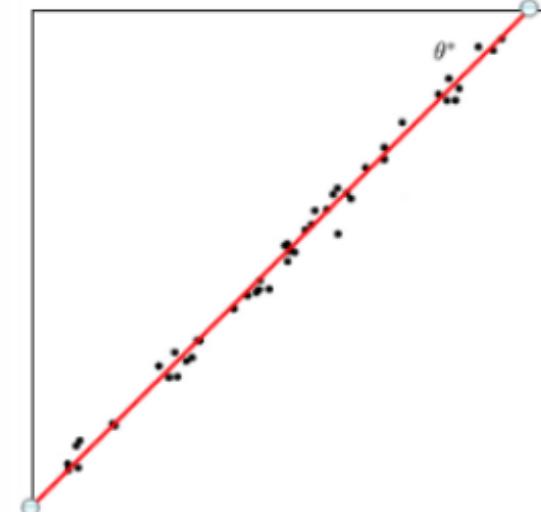
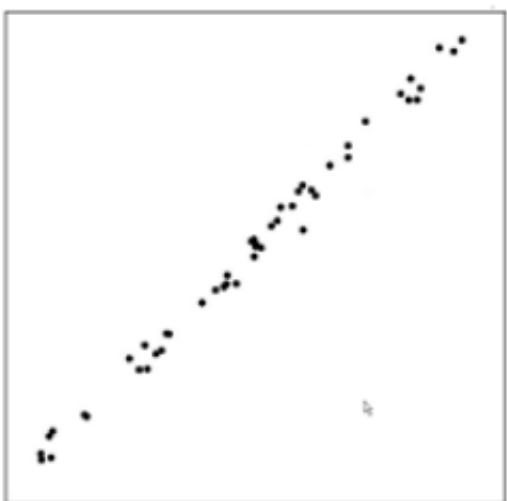


Перспективное преобразование

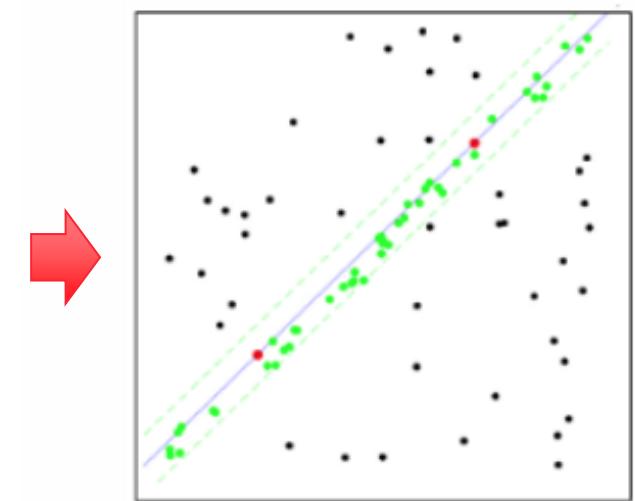
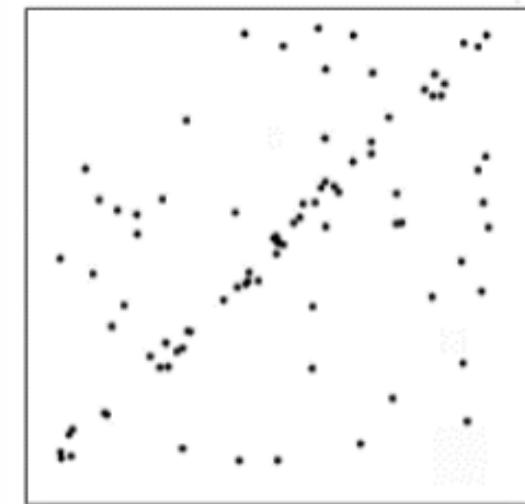
$$\lambda \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} \\ \phi_{21} & \phi_{22} & \phi_{23} \\ \phi_{31} & \phi_{32} & \phi_{33} \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Geometric Estimation

Standard Single Class Single
Instance Fitting Problem (SCSI)

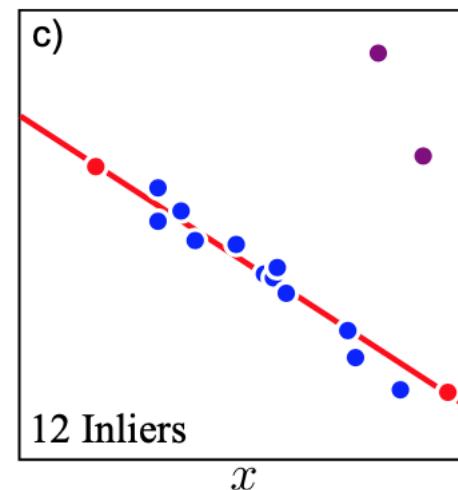
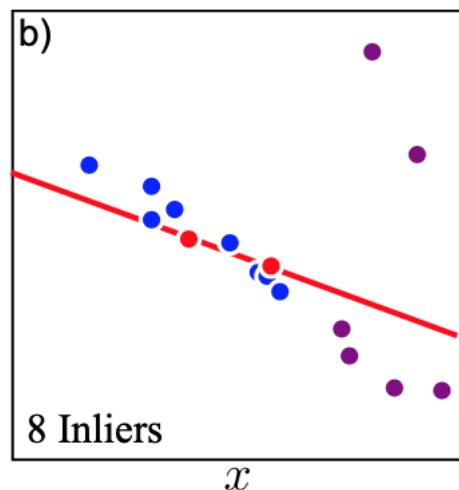
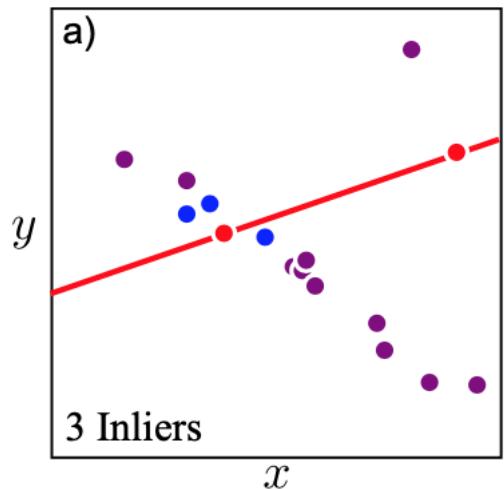


Robust Single Class Single
Instance Fitting Problem (R-SCSI)



RANSAC (Random Sampling consensus)

Многократно (S раз) выбирается **минимальное** подмножество из k точек и по ним строится модель



p – вероятность выбора inlier'a
 P – вероятность успеха после S шагов

$$1 - P = (1 - p^k)^S$$



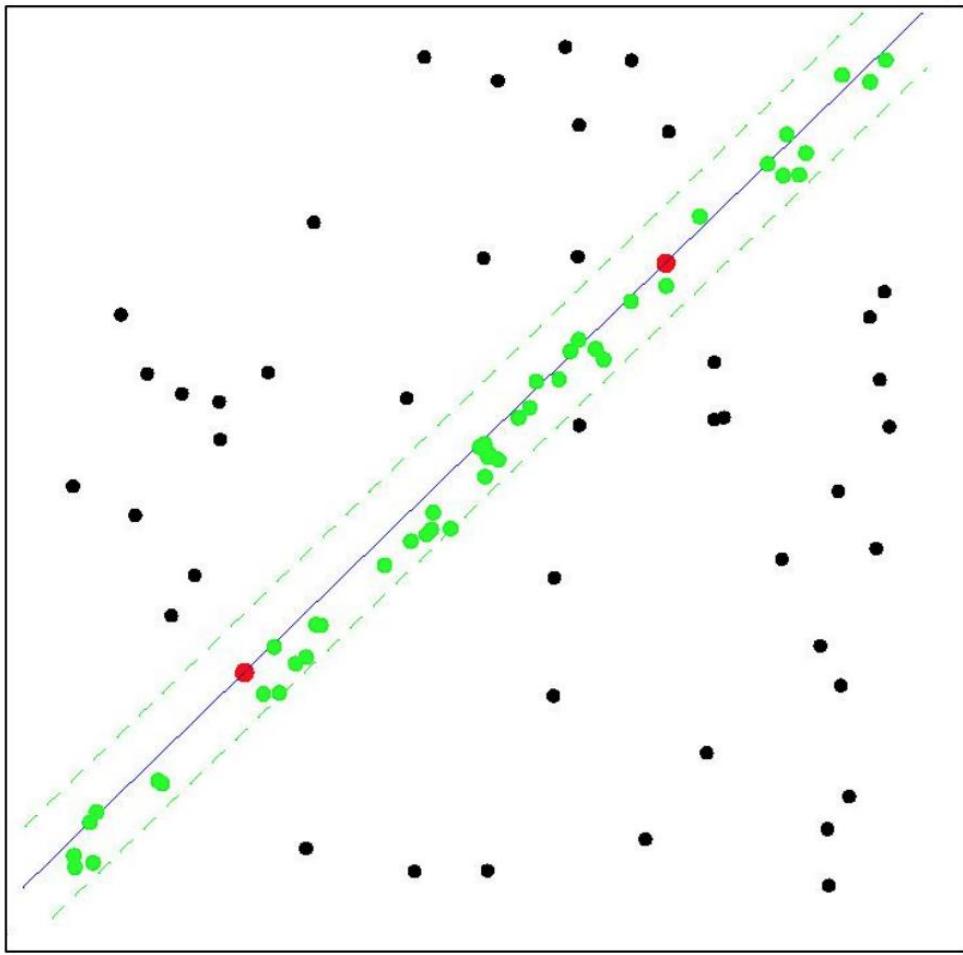
$$S = \frac{\log(1 - P)}{\log(1 - p^k)}.$$

В конце удаляются все выбросы и модель строится по всем оставшимся точкам на основе метода наименьших квадратов

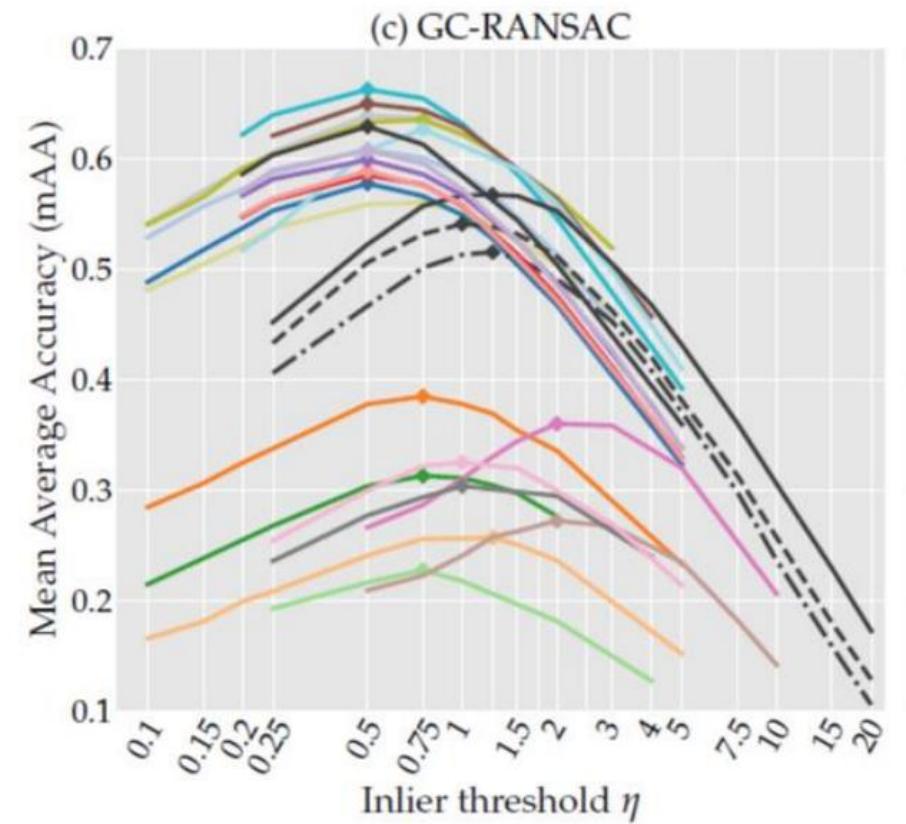
M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.

Prince «Computer vision: models, learning and inference»

RANSAC. Подбор порога



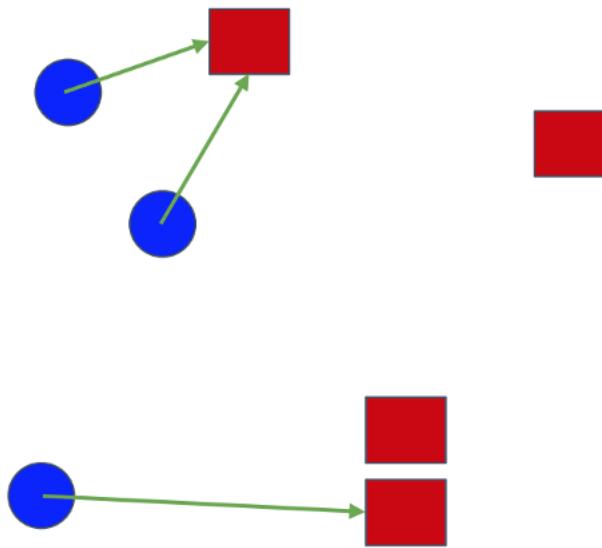
Нужно подбирать порог для каждой задачи и дескриптора



<http://cmp.felk.cvut.cz/cvpr2020-ransac-tutorial/presentations/RANSAC-CVPR20-Matas.pdf>

https://local-features-tutorial.github.io/pdfs/Local_features_from_paper_to_practice.pdf

Ближайший сосед (NN)



Дескрипторам одного изображения ● ставятся в соответствие дескрипторы другого изображения ■

- Нет симметрии
- Несколько точек могут быть «заматчены» на одну

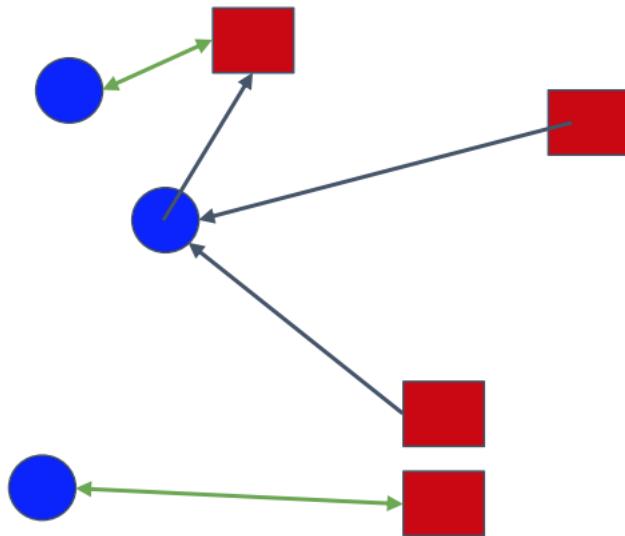


Плохо!

https://local-features-tutorial.github.io/pdfs/Local_features_from_paper_to_practice.pdf

<https://github.com/ducha-aiki/matching-strategies-comparison/blob/master/matching-strategies-comparison.ipynb>

Mutual nearest neighbor (MNN)



1. Дескрипторам одного изображения ● ставятся в соответствие ближайшие дескрипторы другого изображения ■
2. Дескрипторам второго изображения ставятся в соответствие ближайшие дескрипторы первого
3. Сохраняются только общие пары



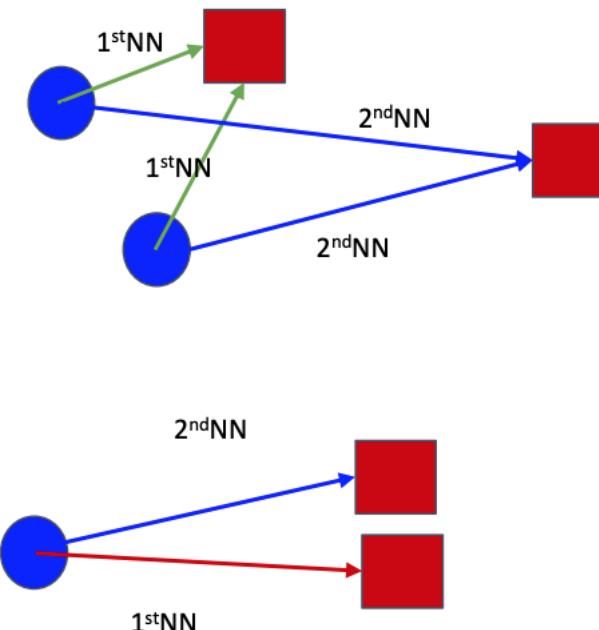
Чуть лучше, но все равно плохо!

https://local-features-tutorial.github.io/pdfs/Local_features_from_paper_to_practice.pdf

<https://github.com/ducha-aiki/matching-strategies-comparison/blob/master/matching-strategies-comparison.ipynb>

Second nearest neighbor ratio (SNN)

$1^{\text{st}}\text{NN}/2^{\text{nd}}\text{NN} < 0.8$, keep



$1^{\text{st}}\text{NN}/2^{\text{nd}}\text{NN} > 0.8$, drop

Расстояние до ближайшего соседа сравнивается с расстоянием до второго ближайшего соседа. Остаются только надежные соответствия



Намного лучше!

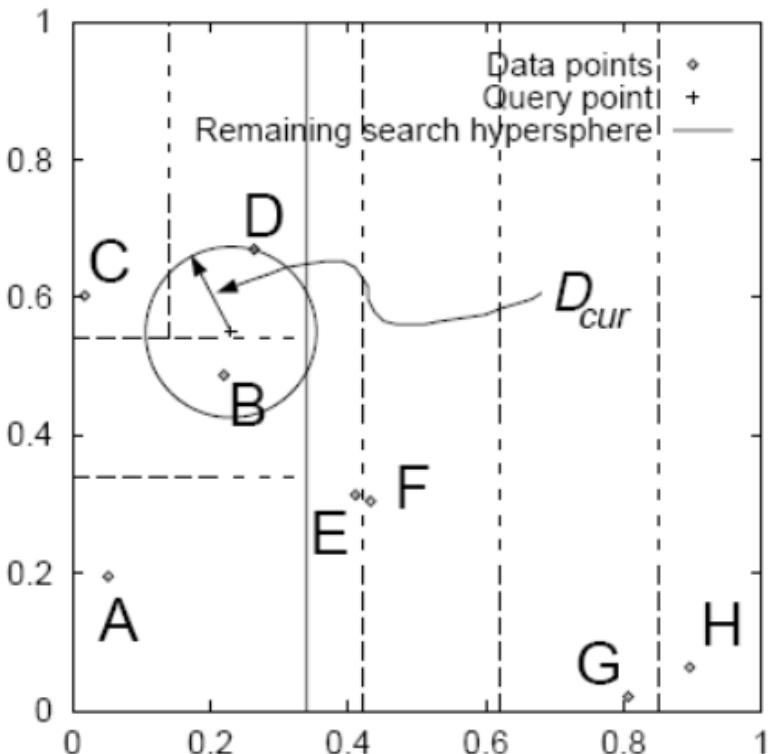
Симметричные/близкие точки на одном изображении?
FGINN: «look for 2nd nearest neighbor, which is far enough from 1st nearest»

https://local-features-tutorial.github.io/pdfs/Local_features_from_paper_to_practice.pdf

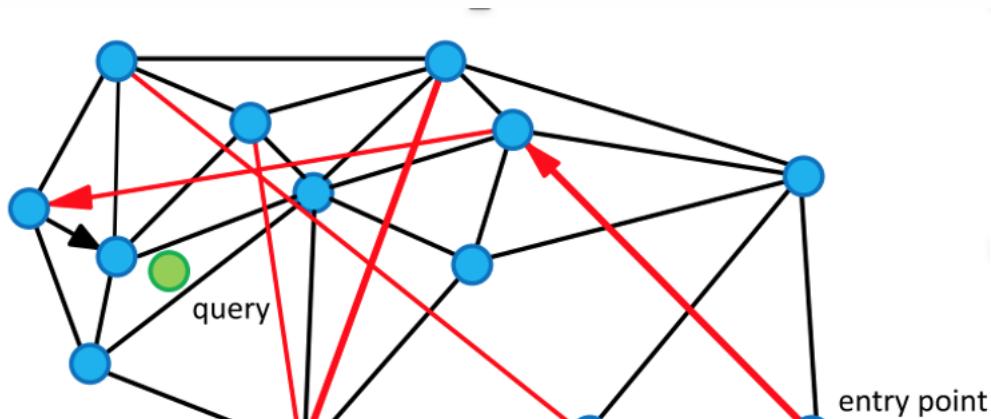
[Lowe, 2004]

Приближенный поиск ближайшего соседа

Randomized k-d tree (FLANN)



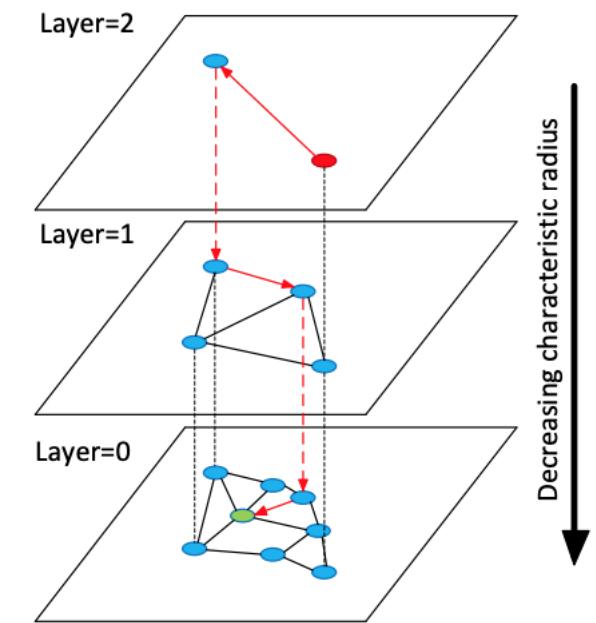
Hierarchical navigable small world,
HNSW (NonMetricSpaceLib, FAISS)



Szeliski «Computer Vision»

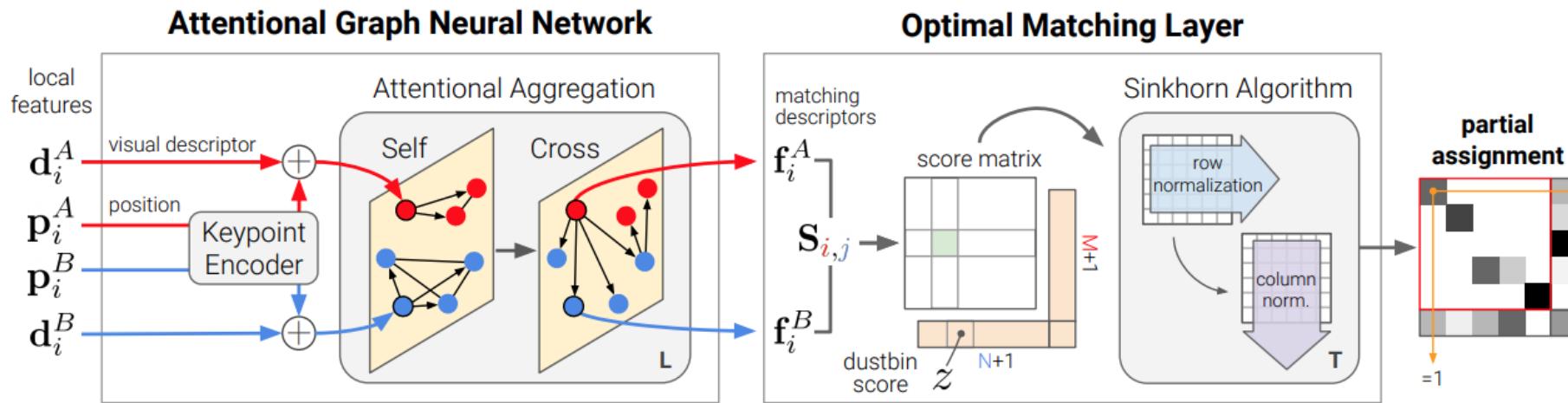
<https://upload.wikimedia.org/wikipedia/commons/4/48/кнрееоogg.ogg>

Malkov & Yashunin, IEEE TPAMI



SuperGlue

Используются **все** ключевые точки и дескрипторы обоих изображений



"SuperGlue is made up of two major components: the attentional graph neural network, and the optimal matching layer. The first component uses a keypoint encoder to map keypoint positions p and their visual descriptors d into a single vector, and then uses alternating self- and cross-attention layers (repeated L times) to create more powerful representations f . The optimal matching layer creates an M by N score matrix, augments it with dustbins, then finds the optimal partial assignment using the Sinkhorn algorithm (for T iterations).»



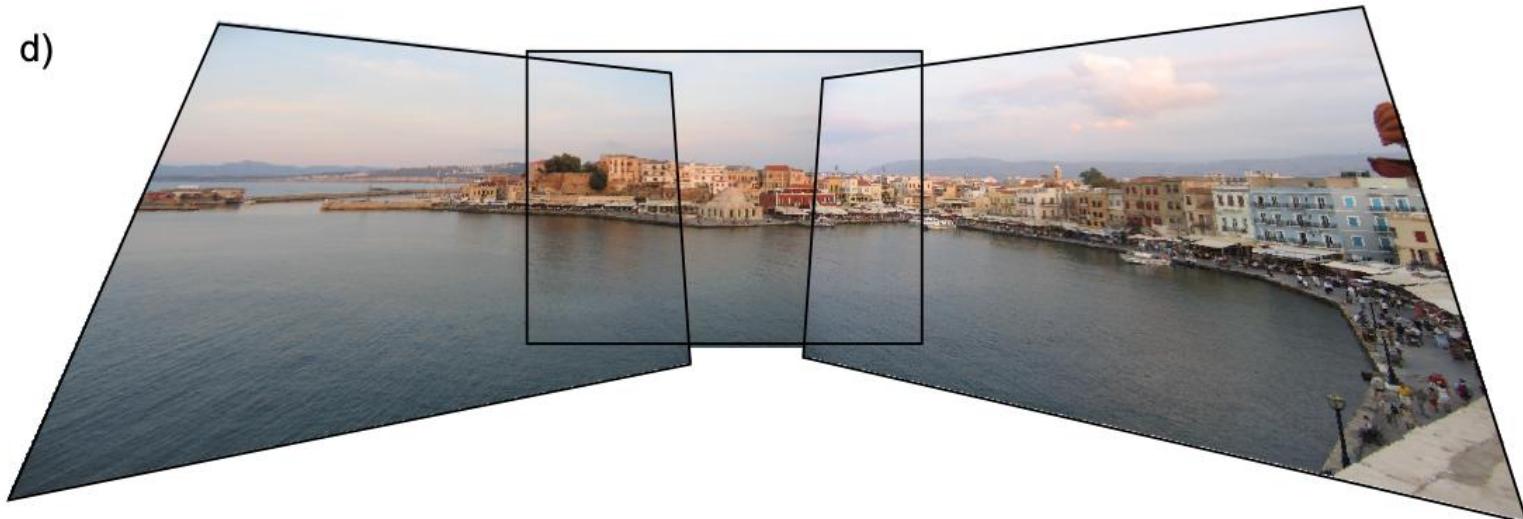
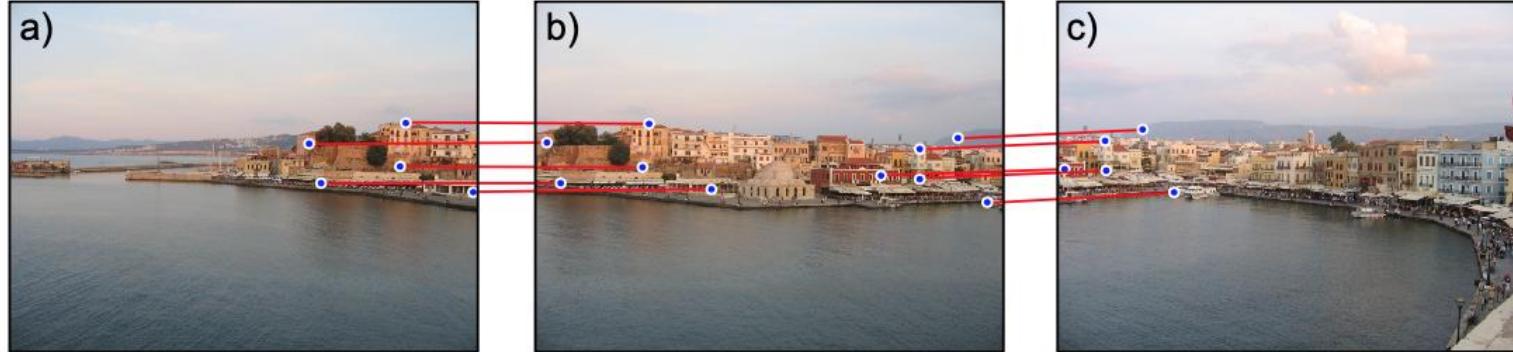
Рекомендации (CVPR2020 Tutorial "Local Features: From SIFT to Differentiable Methods")

- If you DO NOT need correspondences & camera pose → DO NOT use local features.

Use global descriptor (ResNet101 GeM) + fast search (faiss)

- Step 0: try OpenCV (root)**SIFT**
- Use proper RANSAC
- Matching → use FGNN in two-way mode (или хотя бы SNN)
- Need to be faster → ORB/SuperPoint.
- Custom data → train on your own dataset
- Landmark data → DELF
- Try SuperGlue

Пример: image stitching (панорамы)





Перейдем к примерам

<https://github.com/HSE-asavchenko/MADE-mobile-image-processing/tree/master/lesson4/src>



Домашнее задание

Разработать мобильное приложение-сканер документов:

1. Обязательно обработка фотографии из фотоальбома
2. Желательна поддержка съемки с видеокамеры
3. Реализовать поворот с автоматическим и ручным выбором углов листа, различные варианты бинаризации, фильтрации, удаление шума, повышение контраста
4. Опционально (+1 балл) реализовать совмещение (stitching) нескольких фотографий большого документа

Для того, чтобы сдать задание, необходимо исходный код выложить на `github` (публичный или приватный с доступом для `HSE-asavchenko`), сделать `pull request` из бранчи в мастер, ссылку на `PR` прислать мне.

Сдать задание нужно до 23.10.2020 (19:00)