视觉定位

吴毅红，中国科学院大学，2020春季

视觉定位综述

Yihong Wu, Fulin Tang, and Heping Li.

Image Based Camera Localization: an Overview.

Invited paper by Visual Computing for Industry, Biomedicine and Art, 2018.

下载地址：
http://vision.ia.ac.cn/Faculty/yhwu/index_Chinese.html

中文版：

《人工智能》2019年4月第9期，视觉智能大脑——从“云、边、端”，洞悉计算机视觉发展蓝图。

分类

- **3D模型已知**

  2D to 3D 匹配，PnP，SLAM重定位

- **3D模型未知**

  SLAM，同步定位与地图构建

- 重定位的必要性
  - sudden illumination change
  - full occlusion
  - extreme motion blur

- 任务
  - re-estimating the camera pose after tracking failure

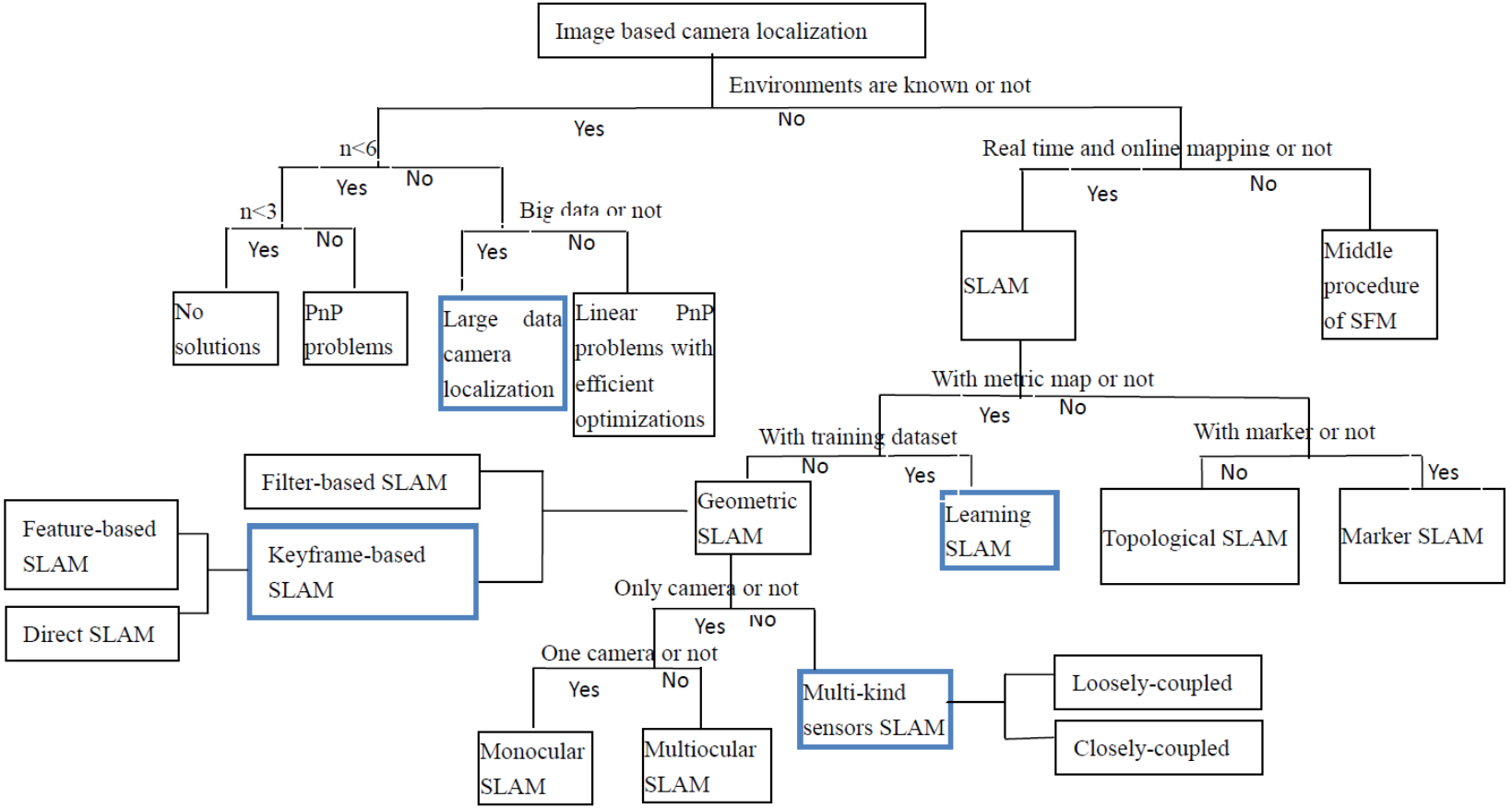


Image based camera localization
Environments are known or not
Yes
No
n<6
Yes
No
n<3
Yes
No
No solutions
PnP problems
Big data or not
Yes
No
Large data camera localization
Linear PnP problems with efficient optimizations
Real time and online mapping or not
Yes
No
SLAM
Middle procedure of SFM
With metric map or not
Yes
No
With training dataset
No
Yes
With marker or not
No
Yes
Geometric SLAM
Learning SLAM
Topological SLAM
Marker SLAM
Filter-based SLAM
Feature-based SLAM
Keyframe-based SLAM
Direct SLAM
Only camera or not
Yes
No
One camera or not
Yes
No
Monocular SLAM
Multiocular SLAM
Multi-kind sensors SLAM
Loosely-coupled
Closely-coupled

- 小数据下的PnP problem: P3P, P4P, P5P

L. Quan and Z. Lan. Linear N-point Camera Pose Determination. IEEE Trans. on PAMI, 21(8): 774-780, 1999.

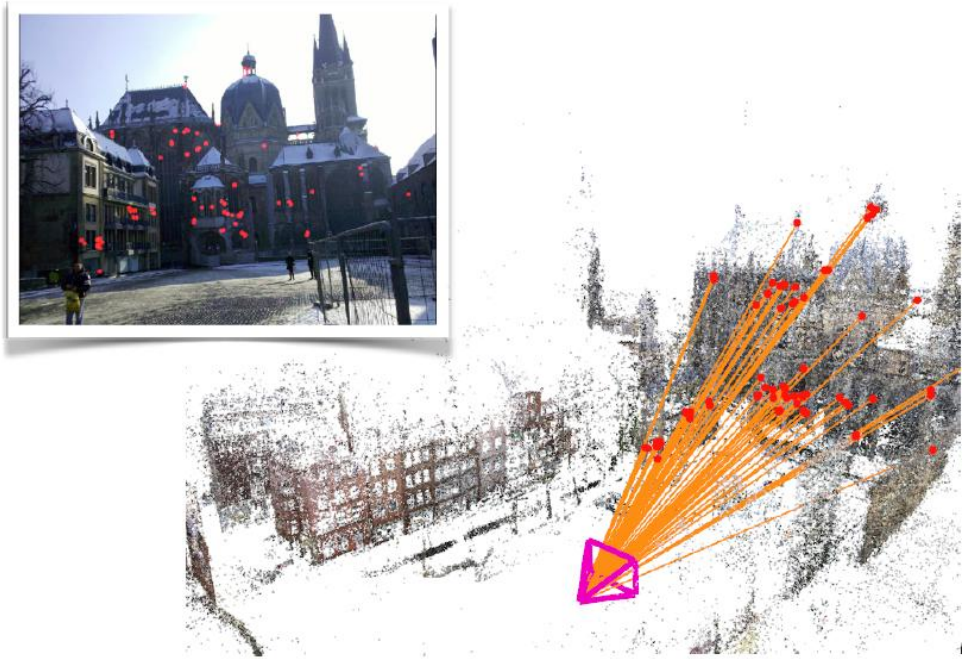
Y. Wu and Z. Hu. PnP Problem Revisited. *Journal of Mathematical Imaging and Vision*, Vol. 24, No. 1, pp. 131-141, 2006.

口 机器人视觉定位的理论和方法

- Youji Feng, Lixin Fan, and Yihong Wu. Fast Localization in Large Scale Environments Using Supervised Indexing of Binary Features. IEEE Transactions on Image Processing, Vol. 25, No. 1, pp. 343-358, 2016.
- Liu Liu, Hongdong Li, and Yuchao Dai. Efficient Global 2D-3D Matching for Camera Localization in a Large-Scale 3D Map. ICCV 2017.
- Eric Brachmann, Carsten Rother. Learning Less is More - 6D Camera Localization via 3D Surface Regression. CVPR 2018. (Deep learning, https://github.com/vislearn/LessMore)

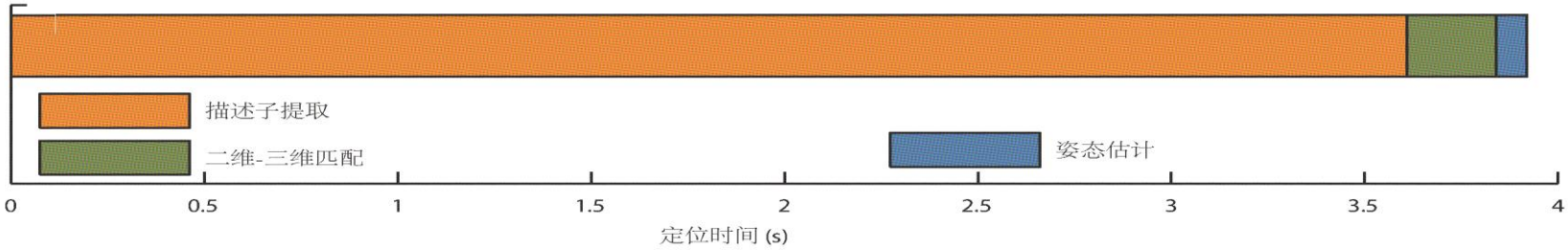
Large data camera localization

基于随机森林的城市大数据（千万级）的毫秒级定位

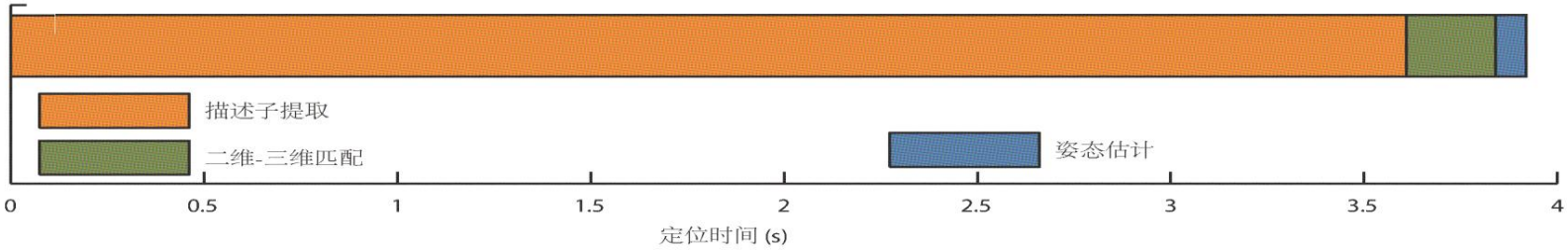


- **The problem**

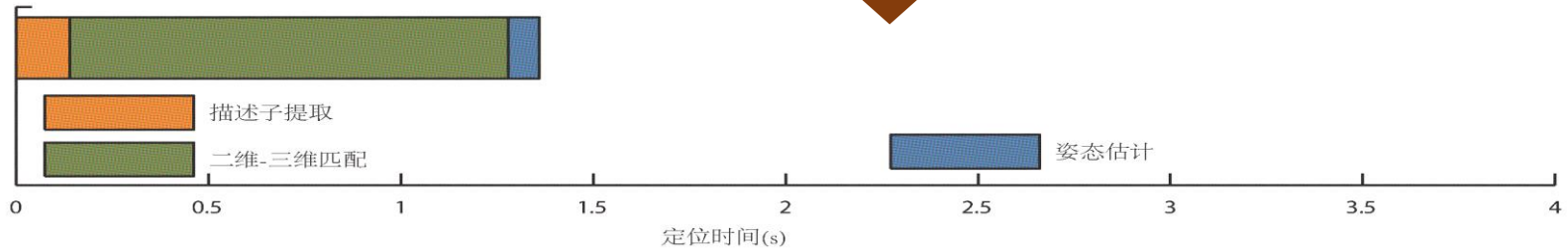
As can be seen, the descriptor extraction, which takes 3.61s, almost accounts for half of the localization time



- **One direct way**
  - Binary descriptors such as BRISK， ORB， FRIF to substitute SIFT.



SIFT+ approximate k-means



BRISK+LSH

现有的二进制特征索引方法是非监督的，忽略了定位数据库中的标签信息

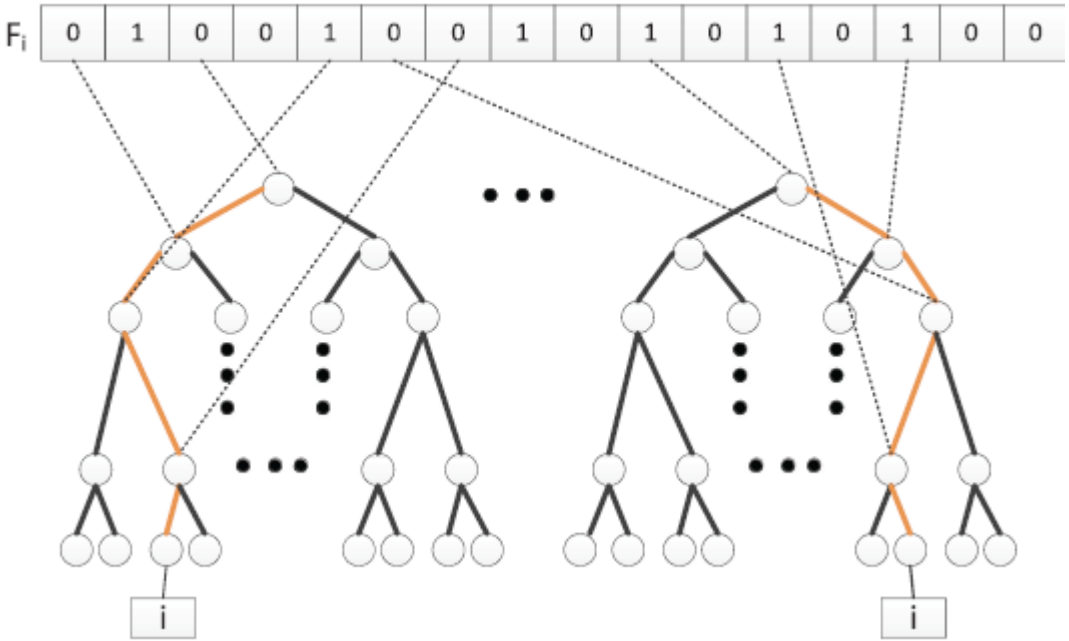
标签信息

在定位数据库中同一个三维点通常对应着多个特征，这些特征互为真实匹配，具有相同的标签

提出一种有监督的索引方法，有效的利用了这种标签信息

采用随机树对特征进行索引

每个非叶节点包含一个二值测试决定特征去往的子节点



（在离线阶段，为所有数据库特征建立索引）

二进制特征索引

- 影响搜索效率的因素
  - 叶节点大小的均匀性，即叶节点中数据库特征数量的分布
    - 通常，叶节点大小越均匀，搜索速度越快
  - 错误率，即相互匹配的特征落在不同的叶节点中的概率
    - 错误率越低，搜索精度越高
- 搜索效率由随机树的结构决定

- 目标
  - 获得尽量均匀的叶节点大小
  - 获得尽量低的错误率
- 通过有监督的训练过程和贪婪算法实现

在为每一个非叶节点选择测试，最小化如下损失函数

错误率　　均匀性

$$C = \lambda(1 - r_c) + \frac{1}{0.5 - |0.5 - \mu|}$$

其中r_c是匹配的特征对被划分在相同子节点的比例，$\mu = \frac{|S_l|}{|S_l| + |S_r|}$，$|S_l|$表示被划分在左子节点的数据库特征的数量，$\lambda$ 是经验参数。

优先搜索策略

- 提高搜索精度

每棵树查询一个叶节点 ➡ 每棵树查询多个叶节点

- 优先搜索策略

优先搜索包含真实匹配的概率较大的叶节点

We propose an indexing approach for approximate nearest neighbor search of binary features： to priorly search the leaf node with the largest success probability.

Algorithm 1 Leaf Nodes Retrieval

Input: a query feature $\mathbf{q}$ and a randomized tree $\boldsymbol{T}$.
Output: a set of leaf nodes $\boldsymbol{L}$ with N elements.
Initialization: $\boldsymbol{L} = \varnothing$; $n = 0$; $\omega_{root} = 1$; $nodeHeap \rightarrow$Insert($Root$, ω_{root}).
//$nodeHeap$ is a data structure that ranks the nodes in ascending order of the corresponding weight ω.
while $nodeHeap \rightarrow$size>0 and $n < N$ **do**
  $[node, \omega]$=$nodeHeap \rightarrow$popMax();
  //$popMax$ means extracting from $nodeHeap$ the node with the largest ω.
  while $node$ is not a leaf node **do**
    $P_l = P(\tau_{node}(\tilde{\mathbf{q}}) = 0 \,|\, \mathbf{q})$;
    $P_r = P(\tau_{node}(\tilde{\mathbf{q}}) = 1 \,|\, \mathbf{q})$;
    if $P_l > P_r$
      $node = childnode\_l$;
      $nodeHeap \rightarrow$Insert($childnode\_r$, $\frac{P_r}{P_l}\omega$);
    else
      $node = childnode\_r$;
      $nodeHeap \rightarrow$Insert($childnode\_l$, $\frac{P_l}{P_r}\omega$);
    end
  end
  $\boldsymbol{L} \rightarrow$Add($node$);
  $n = n + 1$;
end

Experiments

- On retrieval
- On localization

Setup: a laptop with 16GB RAM and an Intel Core i3-2310 2.1GHz CPU

On retrieval

Dataset: Dubrovnik, Rome, and Aachen

- 1.500000 3D points
- 5,270,000 database features
- 1,800,000 query features

Compared approaches

- RT_AP: the randomized trees with supervised training and with priority search
- RT： the randomized trees with supervised training and without priority search
- RT_b： the randomized trees without supervised training and without priority search
- LSH: locality sensitive hashing
- MLSH: multi-probe locality sensitive hashing
- HCT:hierachical clustering trees
- EWH: error weighted hashing
- MIH: multi-indexing hashing

On retrieval

RT_AP	(0.45, 0.02)	(0.58, 0.07)	(0.67, 0.23)
RT	(0.46, 0.03)	(0.56, 0.18)	-
RT_b	(0.44, 0.08)	-	-
HCT_16	(0.45, 0.10)	(0.55, 0.22)	-
HCT_32	(0.43, 0.11)	(0.58, 0.28)	(0.66, 0.78)
LSH	(0.46, 0.18)	(0.56, 1.06)	-
MLSH_1	-	(0.57, 0.48)	-
MLSH_2	-	(0.57, 0.62)	(0.67, 2.16)

The efficiency of the proposed RT_AP is remarkably higher than LSH, MLSH and HCT. Under similar retrieval accuracies,

- 3 to 4 times faster than HCT 32 and HCT 16,
- 9 to 14 times faster than LSH,
- 6 times faster than MLSH 1
- 8 times faster than MLSH 2.

On localization

- Dataset

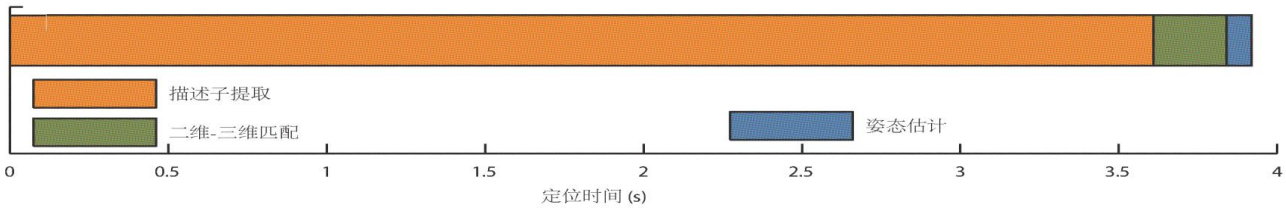


数据集	数据库中图像数目	数据库中三维点数目	数据库中特征数目	查询图像数目	单幅查询图像平均特征数目
Dubrovnik	6044	1,886,884	9,606,317	800	9768
Rome	16,179	4,312,820	21,515,110	1000	7279
Aachen	3047	1,540,786	7,281,501	369	8648

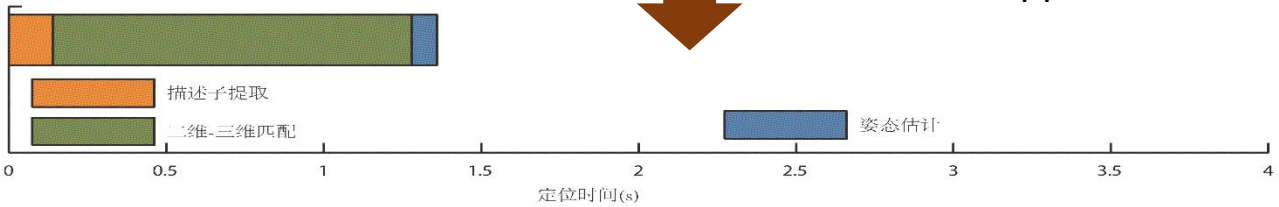
On localization

Dataset	Method	# registered images	descriptor extraction [s]	2D-3D correspondence search [s]	RANSAC [s]	total [s]
Dubrovnik	DM [7]	780.4	3.61	3.97+0.43	0.05	8.06
	P2F [6]	753	3.61	-	-	4.34
	RT_AP	784.9	0.13	0.18	0.10	0.42
	RT_AP(G)	781.6	0.13	0.18	0.11	0.43
Rome	DM [7]	972.7	2.69	2.98+0.61	0.02	6.30
	P2F [6]	921	2.69	-	-	3.60
	RT_AP	978.8	0.11	0.14	0.04	0.29
	RT_AP(G)	979.7	0.11	0.21	0.04	0.36
Aachen	DM [7]	295.4	3.20	3.54+0.47	0.06	7.27
	RT_AP	301.2	0.12	0.18	0.08	0.37
	RT_AP(G)	299.9	0.12	0.20	0.10	0.42

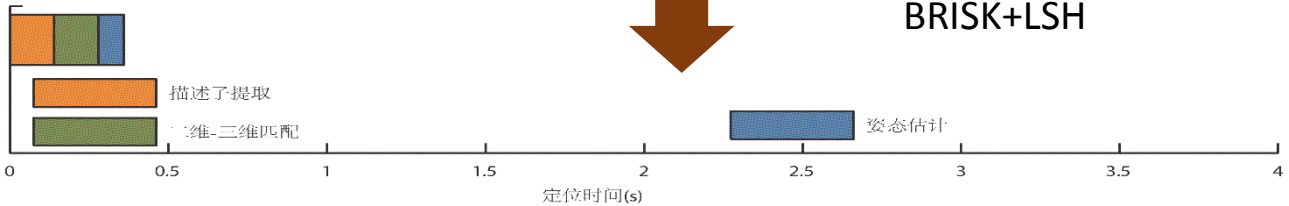
On localization



SIFT+ approximate k-means



BRISK+LSH



BRISK+RT_AP

On localization

- **Fast Camera Relocalization in SLAM**

Efficient global 2D-3D matching for camera localization in a large-scale 3D Map, ICCV 2017

- 在进行2D-3D匹配时，考虑了全局的上下文信息，不再是单个点对进行匹配
- 精度提高
- 速度下降
- 功耗增加

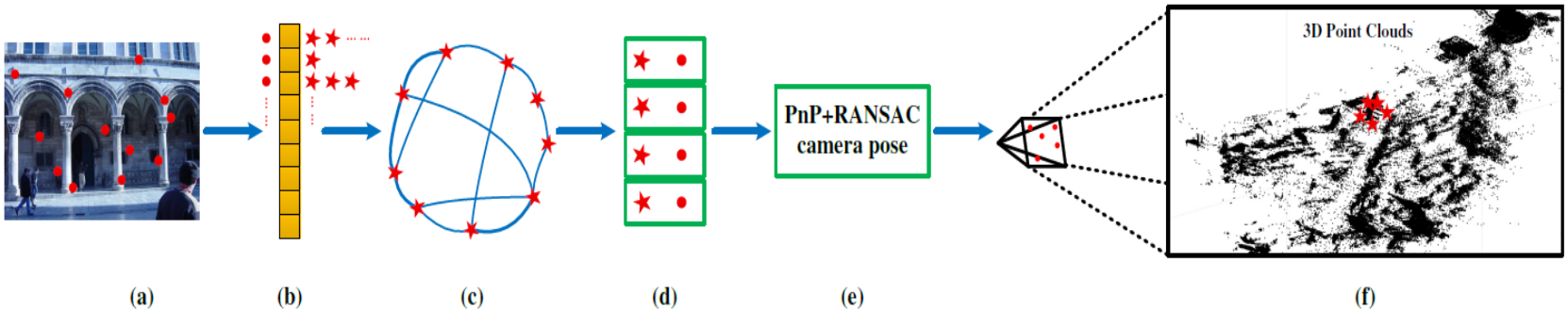
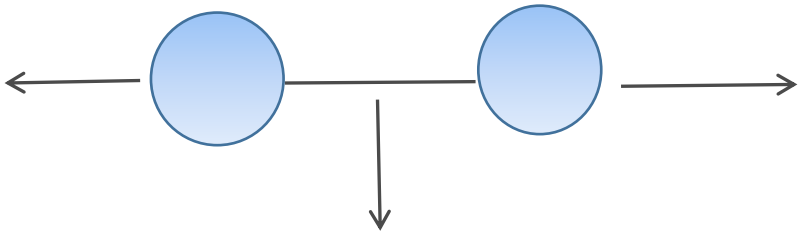


Figure 1. We solve a large-scale image-based localization problem by leveraging global contextual information manifested as co-visibility relationship between pairs of 3D map points. (a) Image features extracted from the query image; (b) Assign 2D features to visual words to obtain candidate 3D matches; (c) The matches are ranked based on global contextual information; (d) One-to-one 2D-3D matches are disambiguated; (e) PnP+RANSAC is used for 6-DoF camera pose recovery against the 3D map (f).

Step 1. Build a Map-Graph

- G(V, E): V graph nodes, E graph edges



Each edge is assign a weight c_{ij}

For the i-th 3D point, denote the set of database images that contain this point as Ai.

$$c_{ij} = \frac{|A_i \cap A_j|}{|A_j|}$$

$$\mathrm{C} = [c_{ij}]$$

状态迁移矩阵

Step 2. Compute query vector

- Given a 2D image, compute:

$$q_i = \sum_{f \in \mathcal{O}(i)} \frac{\sqrt{w_{fi}}}{N_i} \cdot \log\left(\frac{N}{N_f}\right)$$

$$w_{fi} = \exp(-H^2(f,i))/\sigma^2, \forall i \in [1..N]$$

a 2D query feature f and a 3D map point i

$$q_i \leftarrow \left(q_i / \sum_{i=1}^{N} q_i\right), \forall i \in [1..N]$$

Step 3. Random walk on map-graph

Given a map-graph G(V,E) along with a state transition matrix C：
a Markov Network or Markov Random Field

$$\mathbf{p}(\mathbf{t}+\mathbf{1}) = \alpha \mathrm{C}\mathbf{p}(\mathbf{t}) + (1-\alpha)\mathbf{q}.$$

Once the iteration converges, we sort this steady-state probability vector in descending order, which gives the final “matchability” of every 3D point to the set of 2D query features.

Experiments

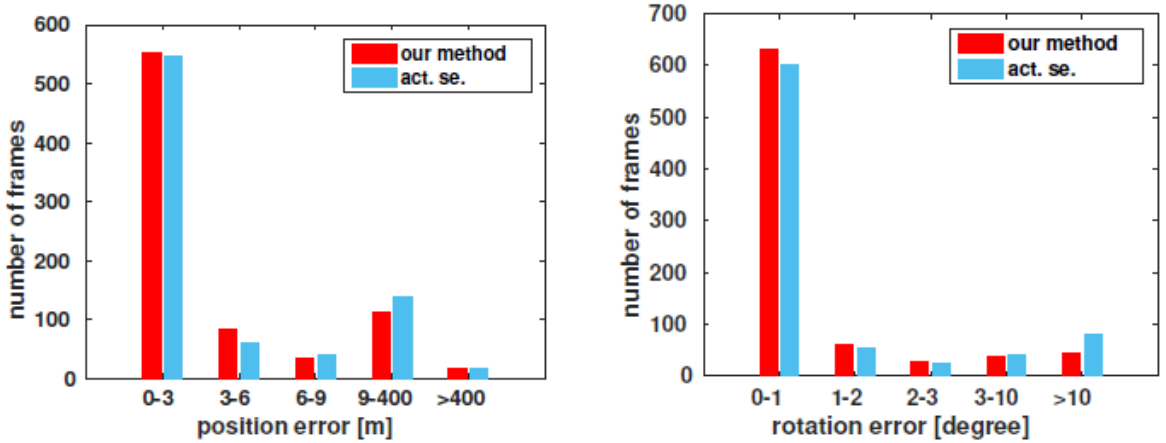


Figure 6. Localization Precision on the Dubrovnik dataset. Left: translation error histogram; Right: rotation error histogram. Results by our method in red, and by Active-Search in blue.

It localizes 743 (out of 800) query images under a localization error of < 18.3m. The average query time is 1.73s.

The average query time by our unoptimized code was 2.35s and 1.67s, which while slower than Active-Search.

Step 4. Camera pose computation

- **Ratio test: one to one**

  compare the descriptor distances between the 3D points and 2D feature points when they are at the same visual words in the vocabulary tree.

- **PnP RANSAC: P4P RANSAC**

Random Sample Consensus

关键概念

- 随机森林
- SIFT描述子
- 二值描述子
- Bag of Words
- Voronoi vector quatization
- Hamming distance
- Hamming embedding distance
- Ratio-test
- 马尔克夫随机场
- 随机游走

Eric Brachmann, Carsten Rother. Learning Less is More - 6D Camera Localization via 3D Surface Regression. CVPR 2018.

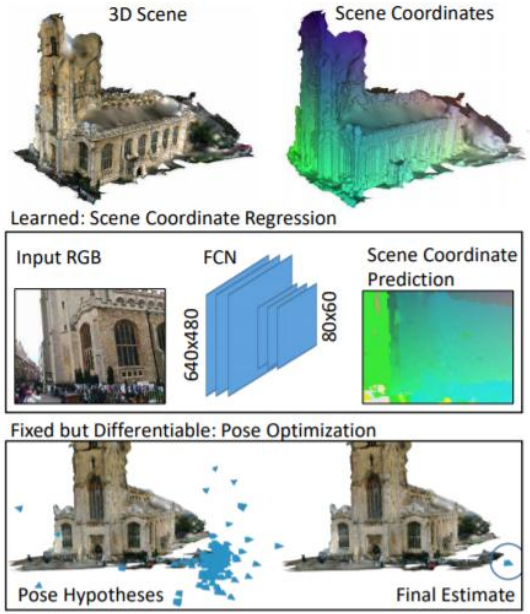


Figure 2. **System Overview.** Given an RGB image, we estimate the 6D camera pose in two stages. Firstly, a fully convolutional network (FCN) regresses 3D scene coordinates (XYZ mapped to RGB for visualization). This CNN is the only learnable component of our system. Secondly, the system optimizes the pose by sampling a pool of hypotheses, scoring them using a soft inlier count, selecting one according to the scores, and refining it as the final estimate. The second stage contains no learnable parameters but is fully differentiable.

Eric Brachmann and Carsten Rother
Visual Learning Lab
Heidelberg University (HCI/IWR)

https://github.com/vislearn/LessMore

a VGG-style architecture with ≈ 30M parameters

- Step 1： To regresses 3D scene coordinates by a fully convolutional network (FCN) .

Our FCN takes an RGB image of 640×480 px as input and produces 80×60 scene coordinate predictions, by using VGG

$$\tilde{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_i ||\mathbf{y}_i(\mathbf{w}) - \mathbf{y}_i^*||.$$

- Step2： To optimize the pose by sampling a pool of hypotheses, with no learnable parameters, but fully differentiable.

$$\tilde{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_i r_i(\mathbf{h}^*, \mathbf{w}), \qquad r_i(\mathbf{h}, \mathbf{w}) = ||C\mathbf{h}^{-1}\mathbf{y}_i(\mathbf{w}) - \mathbf{p}_i||,$$

2D-3D对应，可微分的RANSAC， 重投影误差最小化

$$j \sim P(j; \mathbf{w}, \alpha) = \frac{\exp(\alpha s(\mathbf{h}_j(\mathbf{w})))}{\sum_k \exp(\alpha s(\mathbf{h}_k(\mathbf{w})))},$$

通过softmax函数选择hyposis，使得RANSAC过程连续可微

with a sigmoid function:

$$s(\mathbf{h}) = \sum_i \text{sig}(\tau - \beta(r_i(\mathbf{h}, \mathbf{w}))),$$

通过sigmoid 函数进行打分

$$S(\alpha) = -\sum_j P(j; \mathbf{w}, \alpha) \log P(j; \mathbf{w}, \alpha).$$

自适应参数alpha求解

- Training:

  End to end:

$$\tilde{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{\mathcal{D}} \mathbb{E}_{j \sim P(j;\mathbf{w},\alpha)} \left[\ell(\mathbf{R}(\mathbf{h}_j(\mathbf{w})), \mathbf{h}^*) \right].$$

$$\mathbf{R}(\mathbf{h}) = \underset{\mathbf{h}'}{\operatorname{argmin}} ||\mathbf{r}(\mathbf{h}', \mathbf{w})||^2.$$

$$\ell(\mathbf{h}, \mathbf{h}^*) = \max(\angle(\boldsymbol{\theta}, \boldsymbol{\theta}^*), ||\mathbf{t} - \mathbf{t}^*||)$$

Table 1. **Median 6D Localization Errors.** We report results for the 7Scenes dataset [23] and the Cambridge Landmarks dataset [11]. We mark best results **bold** (if both, translational and rotational error, are lowest). Results of DSAC marked with an asterisk (*) are before end-to-end optimization which did not converge. A dash (-) indicates that a method failed completely.

Dataset	Training w/ 3D Model				w/o 3D Model		
7Scenes	PoseNet [10] (Geom. Loss)	Active Search [20]	DSAC [2] (RGB Training)	Ours	PoseNet [10] (Pose Loss)	Spatial LSTM [29]	Ours
Chess	0.13m, 4.5°	0.04m, 2.0°	0.02m, 1.2°	**0.02m, 0.5°**	0.14m, 4.5°	0.24m, 5.8°	**0.02m, 0.7°**
Fire	0.27m, 11.3°	0.03m, 1.5°	0.04m, 1.5°	**0.02m, 0.9°**	0.27m, 11.8°	0.34m, 11.9°	**0.03m, 1.1°**
Heads	0.17m, 13.0°	0.02m, 1.5°	0.03m, 2.7°	**0.01m, 0.8°**	0.18m, 12.1°	0.21m, 13.7°	**0.12m, 6.7°**
Office	0.19m, 5.6°	0.09m, 3.6°	0.04m, 1.6°	**0.03m, 0.7°**	0.20m, 5.8°	0.30m, 8.1°	**0.03m, 0.8°**
Pumpkin	0.26m, 4.8°	0.08m, 3.1°	0.05m, 2.0°	**0.04m, 1.1°**	0.25m, 4.8°	0.33m, 7.0°	**0.05m, 1.1°**
Kitchen	0.23m, 5.4°	0.07m, 3.4°	0.05m, 2.0°	**0.04m, 1.1°**	0.24m, 5.5°	0.37m, 8.8°	**0.05m, 1.3°**
Stairs	0.35m, 12.4°	**0.03m, 2.2°**	1.17m, 33.1°	0.09m, 2.6°	0.37m, 10.6°	0.40m, 13.7°	**0.29m, 5.1°**
Cambridge							
Great Court	7.00m, 3.7°	-	*2.80m, 1.5°	**0.40m, 0.2°**	6.83m, 3.5°	-	**0.66m, 0.4°**
K. College	0.99m, 1.1°	0.42m, 0.6°	*0.30m, 0.5°	**0.18m, 0.3°**	0.88m, 1.0°	0.99m, 1.0°	**0.23m, 0.4°**
Old Hospital	2.17m, 2.9°	0.44m, 1.0°	0.33m, 0.6°	**0.20m, 0.3°**	3.20m, 3.3°	1.51m, 4.3°	**0.24m, 0.5°**
Shop Facade	1.05m, 4.0°	0.12m, 0.4°	0.09m, 0.4°	**0.06m, 0.3°**	0.88m, 3.8°	1.18m, 7.4°	**0.09m, 0.4°**
St M. Church	1.49m, 3.4°	0.19m, 0.5°	*0.55m, 1.6°	**0.13m, 0.4°**	1.57m, 3.2°	1.52m, 6.7°	**0.20m, 0.7°**
Street	20.7m, 25.7°	**0.85m, 0.8°**	-	-	**20.3m, 25.5°**	-	-

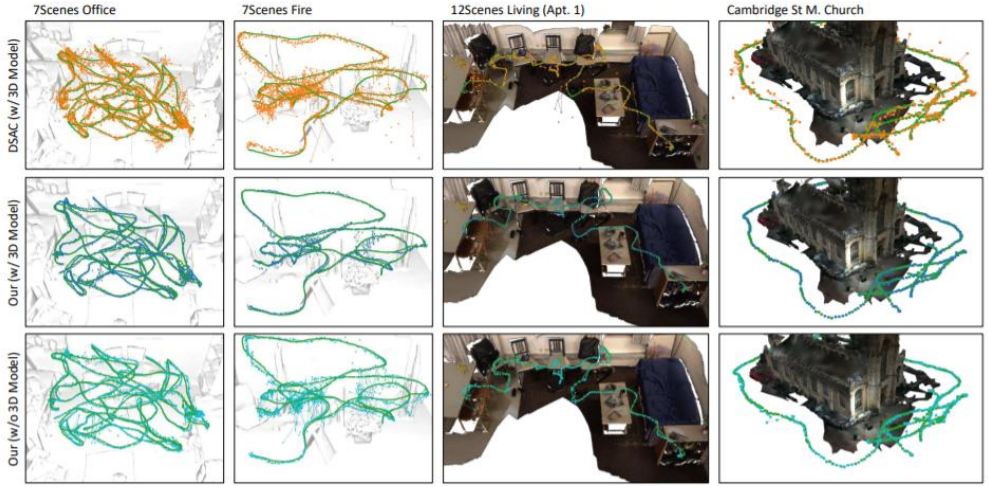


Figure 6. **Estimated Camera Trajectories**. We plot estimated camera locations as trajectories within the respective 3D scene model (untextured for 7Scenes). We show ground truth in green, DSAC [2] estimates in orange, and our results in blue and cyan when trained with and without a 3D model, respectively. Note that DSAC produces many wrong estimates despite being trained with a 3D model.

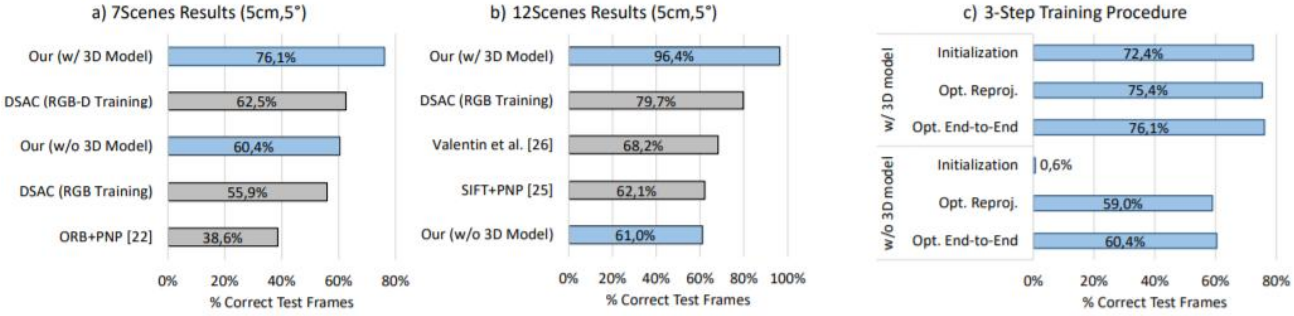


Figure 5. **Indoor Localization Accuracy.** We show the percentage of test frames of the 7Scenes **(a)** resp. the 12Scenes **(b)** dataset with a pose error below 5cm and 5°. We mark our method blue. Note that all competitors depend on a 3D model. For our method, we show results after each of our 3 training steps on the 7Scenes dataset **(c)**.

课后思考题

- 观看波士顿机器人SpotMini进行视觉定位导航的demo

- 问题：
  - 叙述场景特色
  - 叙述传感器
  - 叙述其中视觉定位导航中的各模块任务

谢 谢

yhwu@nlpr.ia.ac.cn
http://vision.ia.ac.cn/

模式识别国家重点实验室宣传小组
模识识别国家重点实验室综合办公室
2014年10月31日 制作