Neural-Guided RANSAC: Learning Where to Sample Model Hypotheses

Eric Brachmann and Carsten Rother - Heidelberg University (HCI/IWR)

Visual Learning Lab Heidelberg

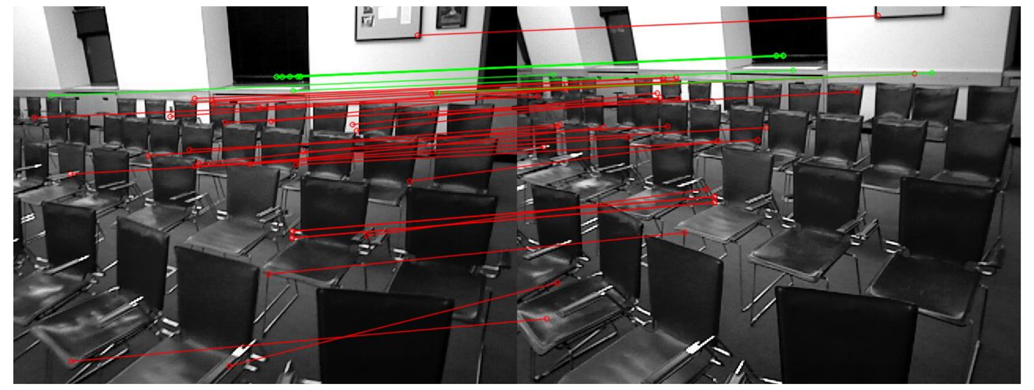
UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Code and trained models:

Problem Statement

Fit a parametric model to data points with **many outliers** using **RANSAC.**
For example, fit essential matrix to SIFT correspondences.

2000 SIFT correspondences, outlier ratio: 88%
RANSAC would need: > 185.000 iterations

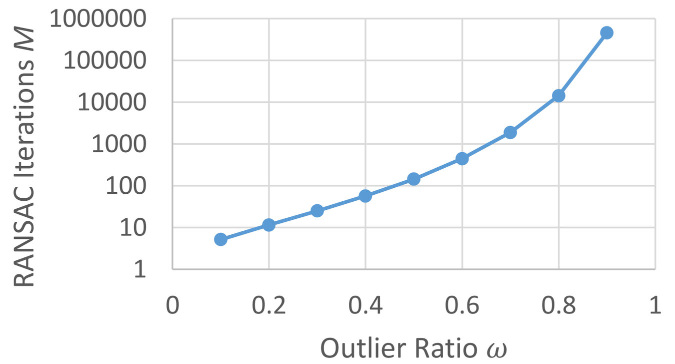


Result after 1000 iterations (OpenCV)

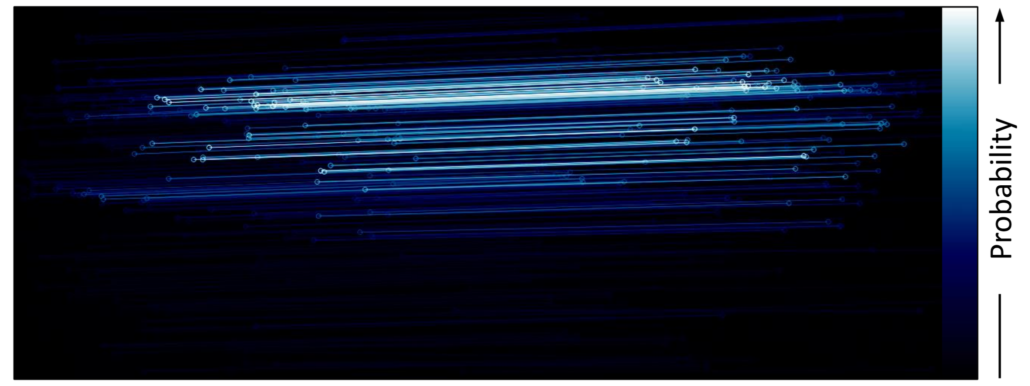
Number of RANSAC iterations grows **exponentially** with increasing outlier ratio.

$$M = \frac{\log(1-p)}{\log(1-(1-\omega)^n)}$$

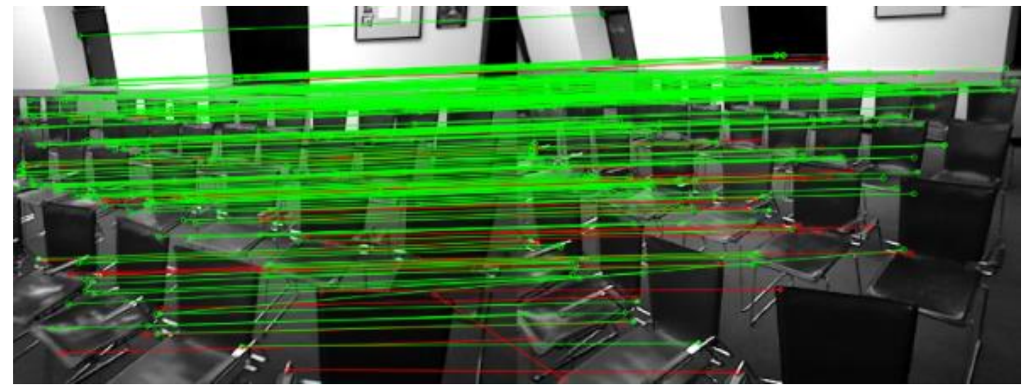
Target probability: $p = 0.99$
Minimal set size: $n = 5$



→ Removing **just enough** outliers makes the problem **exponentially easier**.
→ We let a **neural network** predict weights that **guide RANSAC** sampling.



239 correspondences get 90% probability mass, outlier ratio: 33%



NG-RANSAC result after 1000 iterations

Contributions

- NG-RANSAC: A **general, robust estimator** based on RANSAC with learned guidance of hypotheses sampling
- Principled learning formulation that directly **optimizes model quality**
- Training with **non-differentiable** minimal solver, refinement, loss etc., also training **self-supervised**
- Applied to estimation of **essential matrices**, **fundamental matrices**, **horizon lines**, and **camera poses**

European Research Council

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 647769)

European Commission

Horizon 2020 European Union funding for Research & Innovation

Brackground: RANSAC

1) Sampling Hypotheses:

$$p(\mathbf{h}) = \prod_{i=0}^{N} p(\mathbf{y}_i) \text{ with } p(\mathbf{y}) = \frac{1}{|\mathcal{Y}|}$$

uniform weights

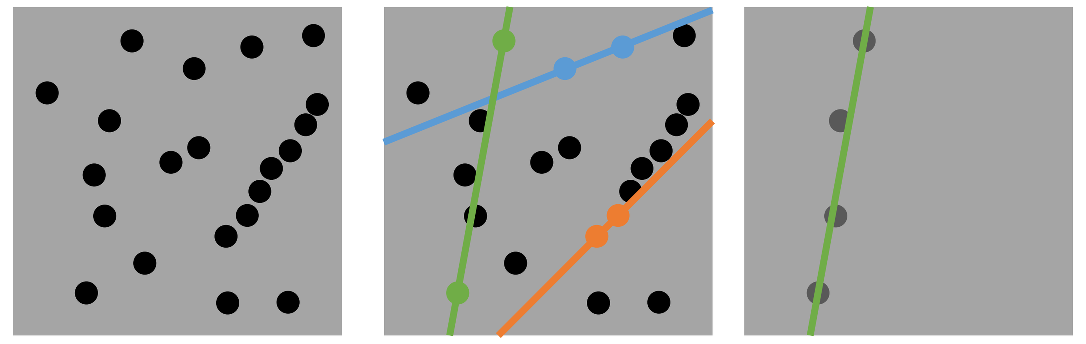
$$p(\mathcal{H}) = \prod_{j=0}^{M} p(\mathbf{h}_j)$$

2) Hypothesis Selection:

$$\hat{\mathbf{h}} = \operatorname*{argmax}_{\mathbf{h}\in\mathcal{H}} s(\mathbf{h}, \mathcal{Y})$$

inlier count:

$$s(\mathbf{h}, \mathcal{Y}) = \sum_{\mathbf{y}\in\mathcal{Y}} \mathbb{1}[d(\mathbf{y}, \mathbf{h}) < \tau]$$



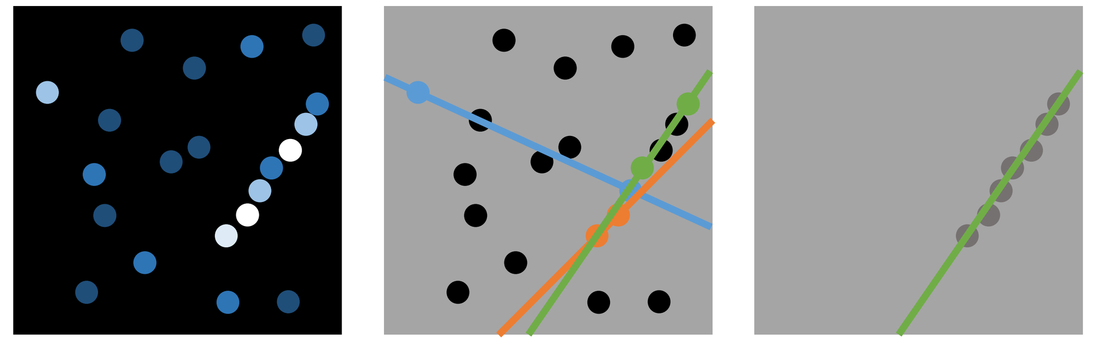
data points $\mathbf{y} \in \mathcal{Y}$ — hypotheses $\mathbf{h} \in \mathcal{H}$ — fitted model $\hat{\mathbf{h}}$

Neural-Guided RANSAC

Guided Sampling:

$$p(\mathbf{h}) = \prod_{i=0}^{n} p(\mathbf{y}_i) \text{ with } p(\mathbf{y}) = g(\mathcal{Y}; \mathbf{w})$$

learned weights



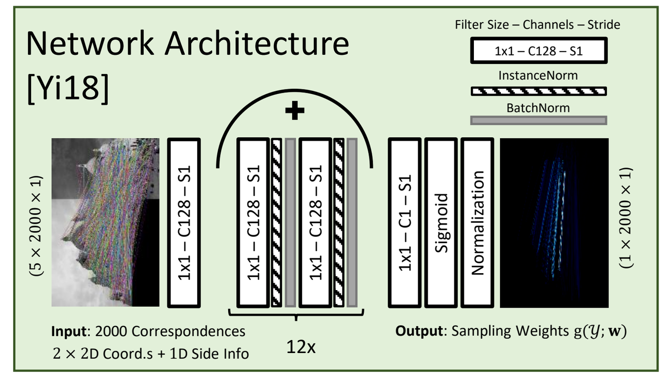
neural guidance $g(\mathcal{Y}; \mathbf{w})$ — hypotheses $\mathbf{h} \in \mathcal{H}$ — fitted model $\hat{\mathbf{h}}$

Training Objective:

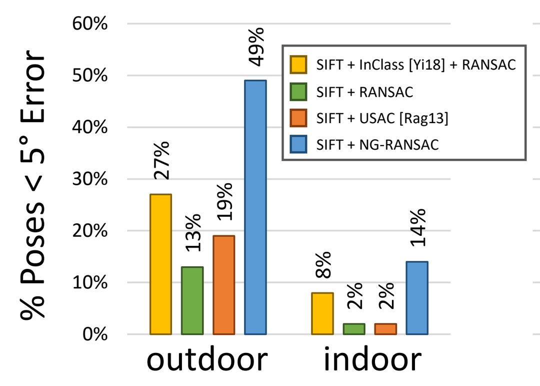
$$\underbrace{\frac{\partial}{\partial \mathbf{w}} \mathbb{E}_{\mathcal{H}\sim p(\mathcal{H};\mathbf{w})}\left[\ell(\hat{\mathbf{h}})\right]}_{\text{expected loss}} = \underbrace{\mathbb{E}_{\mathcal{H}\sim p(\mathcal{H};\mathbf{w})}\left[\ell(\hat{\mathbf{h}}) \frac{\partial}{\partial \mathbf{w}} \log p(\mathcal{H}; \mathbf{w})\right]}_{\text{gradients}} \approx \underbrace{\frac{1}{K}\sum_{k=1}^{K}\left[\ell(\hat{\mathbf{h}}) \frac{\partial}{\partial \mathbf{w}} \log p(\mathcal{H}_k; \mathbf{w})\right]}_{\text{approximate via sampling}}$$

Application: Epipolar Geometry

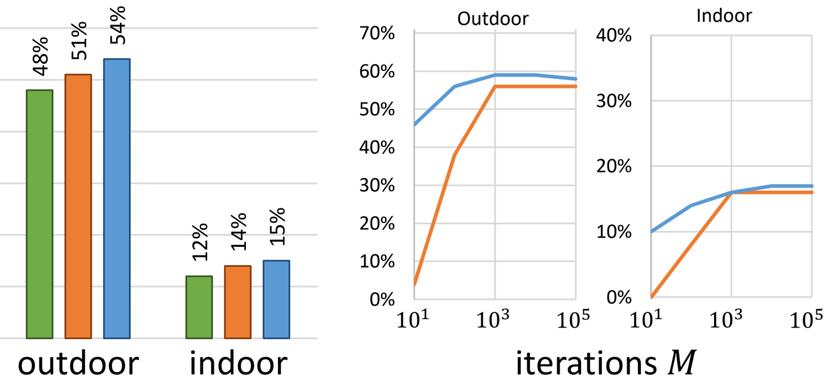
Essential matrix from SIFT correspondences:



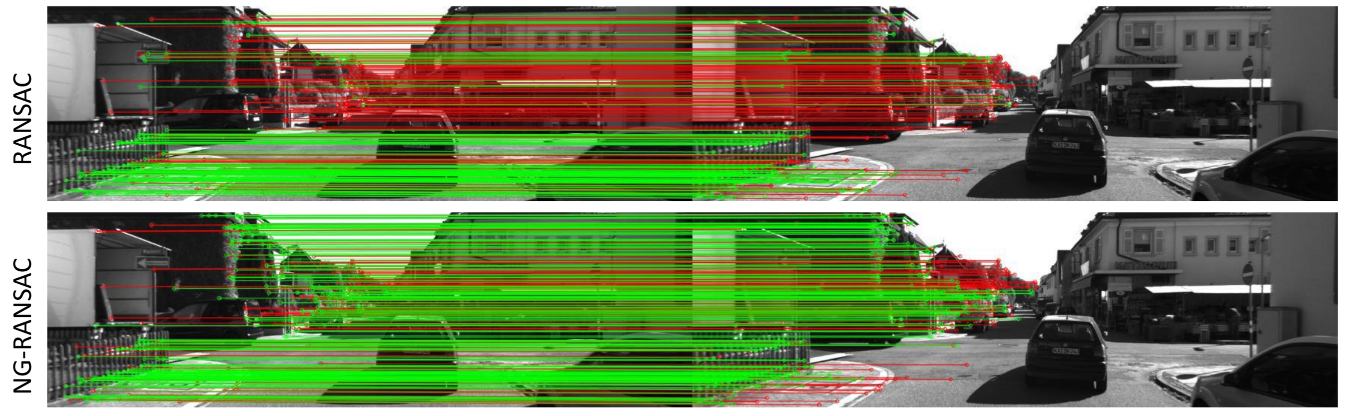
without side information



with side information and Lowe's ratio filter



Fundamental matrix from SIFT correspondences:



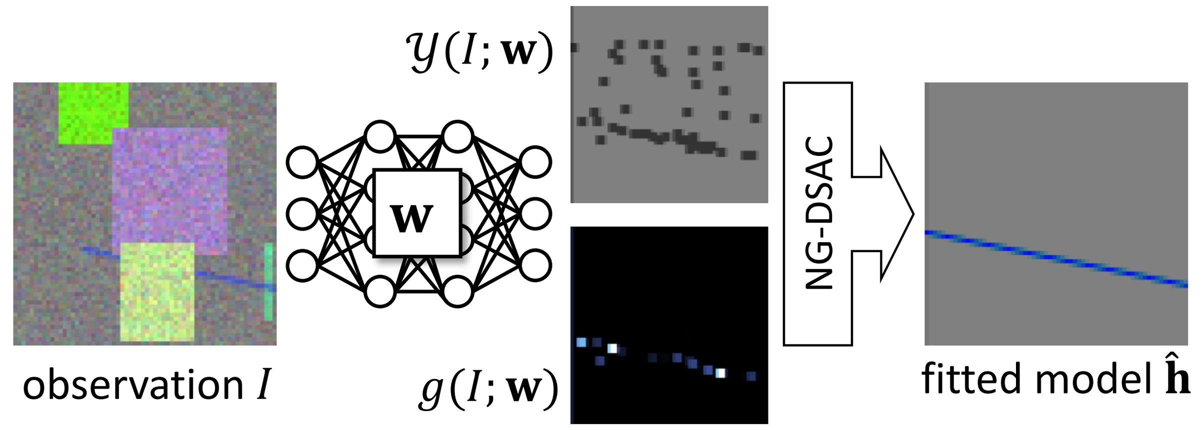
	Training Objective	% Inliers	F-score	Mean	Median
RANSAC	-	21.85	13.84	0.35	0.32
USAC [Rag13]	-	21.43	13.90	0.35	0.32
Deep F-Mat [Ran18]	Mean	24.61	14.65	**0.32**	**0.29**
NG-RANSAC	Mean	25.05	**14.76**	**0.32**	**0.29**
NG-RANSAC	F-score	24.13	14.72	0.33	0.31
NG-RANSAC	%Inliers	**25.12**	14.74	**0.32**	**0.29**

$$\ell(\hat{\mathbf{h}}) = -s(\hat{\mathbf{h}}, \mathcal{Y})$$

Self-Supervised!

Neural-Guided DSAC

Let a network **predict** not only **sampling weights** but **data points** themselves.



observation I — fitted model $\hat{\mathbf{h}}$

Background: DSAC [Bra17]

$$\hat{\mathbf{h}} = \mathbf{h}_j \text{ with } j \sim p(j) = \frac{\exp(s(\mathbf{h}_j, \mathcal{Y}))}{\sum_{j'} \exp(s(\mathbf{h}_{j'}, \mathcal{Y}))}$$

Training Objective

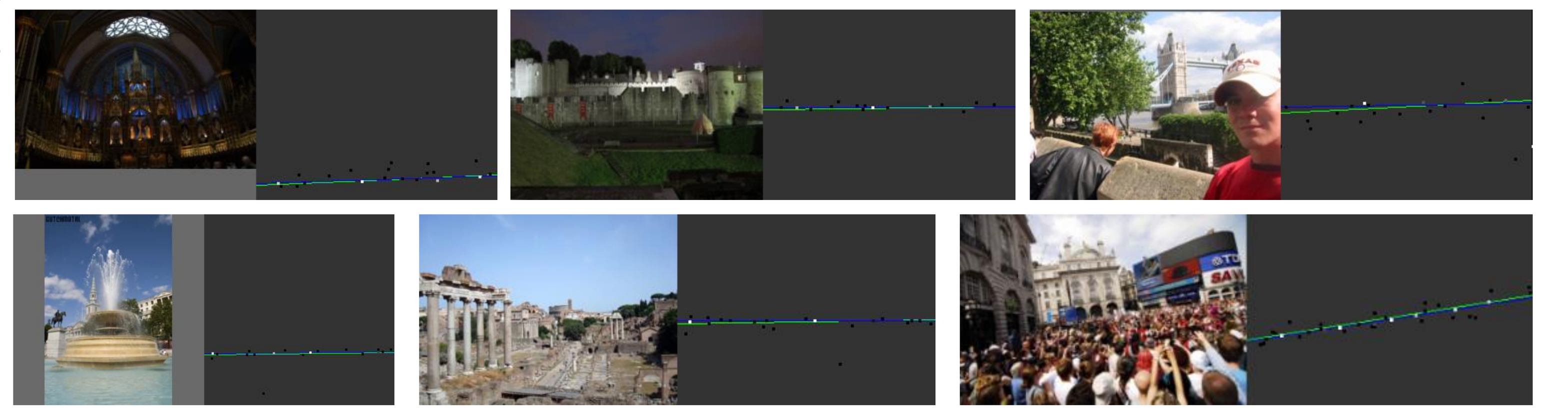
$$\mathcal{L}(\mathbf{w}) = \mathbb{E}_{j\sim p(j)}[\ell(\mathbf{h}_j)]$$

NG-DSAC Training Objective:

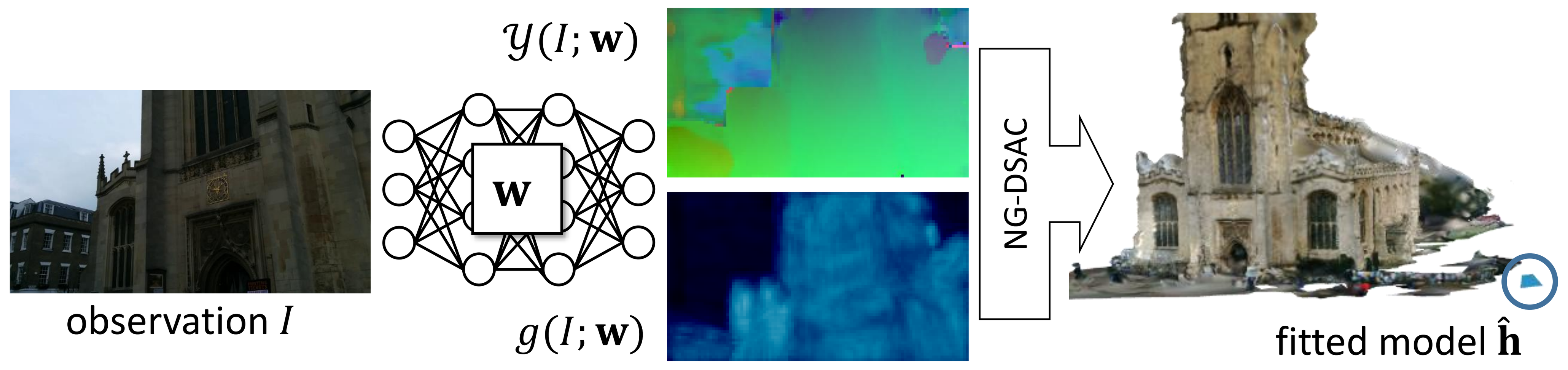
$$\underbrace{\frac{\partial}{\partial \mathbf{w}} \mathbb{E}_{\mathcal{H}\sim p(\mathcal{H};\mathbf{w})} \mathbb{E}_{j\sim p(j|\mathcal{H})}[\ell(\mathbf{h}_j)]}_{\text{expected loss}} \approx \underbrace{\frac{1}{K}\sum_{k=1}^{K}\left[\mathbb{E}_j[\ell] \frac{\partial}{\partial \mathbf{w}} \log p(\mathcal{H}_k; \mathbf{w}) + \mathbb{E}_j[\ell]\right]}_{\text{approximated gradients}}$$

Application: Horizon Line Estimation

on [Wor16]	AUC (%)
Simon et al.	54.4
Kluger et al.	57.3
Zhai et al.	58.2
Workman et al.	71.2
DSAC	**74.1**
NG-DSAC	**75.2**
SLNet	82.3



Application: Camera Re-Localization



observation I — fitted model $\hat{\mathbf{h}}$

on [Ken15]	PoseNet	Active Search	DSAC++ [Bra18]	NG-DSAC++
Great Court	700cm	-	40.3cm	**35.0cm**
Kings College	99cm	42cm	13.0cm	**12.6cm**
Old Hospital	217cm	44cm	22.4cm	**21.9cm**
Shop Facade	107cm	12cm	5.7cm	**5.6cm**
St M. Church	149cm	19cm	9.9cm	**9.8cm**

[Yi18] Learning to Find Good Correspondences, Yi et al., CVPR18
[Ran18] Deep Fundamental Matrix Estimation, Ranftl and Koltun, ECCV18
[Wor16] Horizon Lines in the Wild, Workman et al., BMVC16
[Rag13] USAC: A Universal Framework for Random Sample Consensus, Raguram et al., TPAMI13
[Bra17] DSAC - Differentiable RANSAC for camera localization, Brachmann et al., CVPR17
[Ken15] PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization, Kendall et al., ICCV15
[Bra18] Learning Less is More - 6D Camera Localization via 3D Surface Regression, Brachmann et Rother, CVPR18