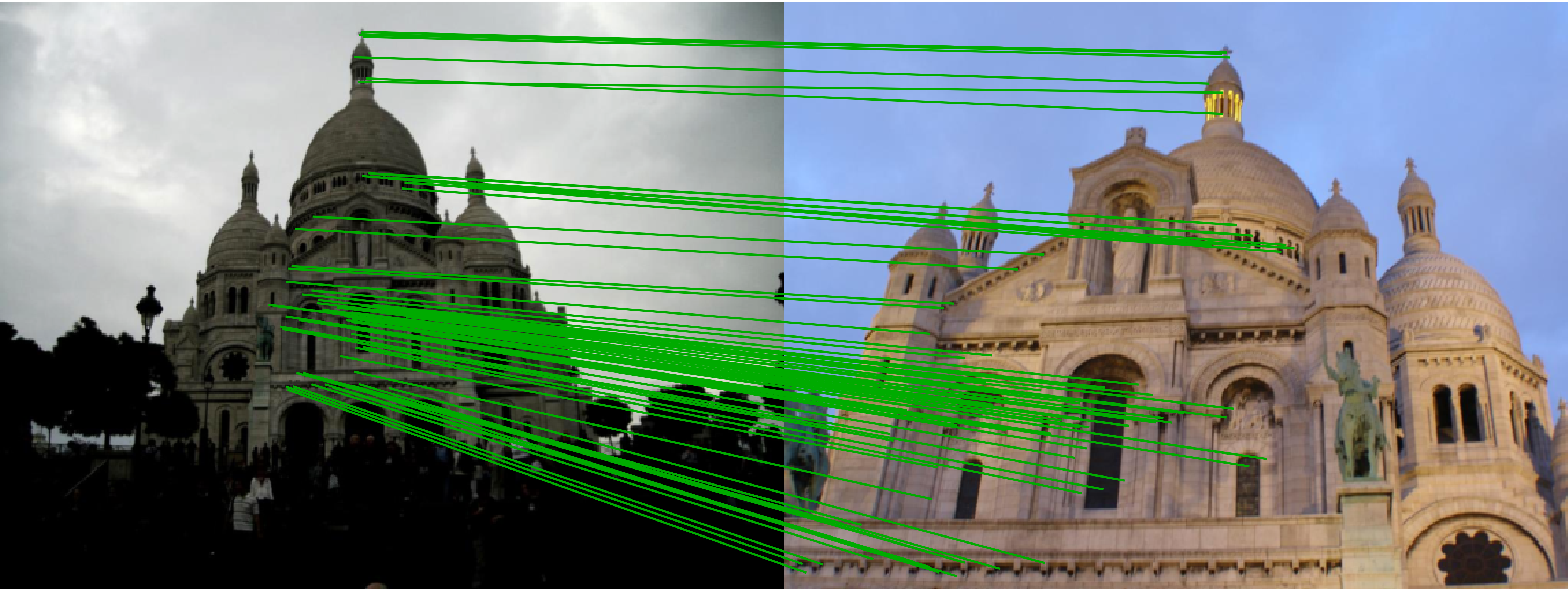
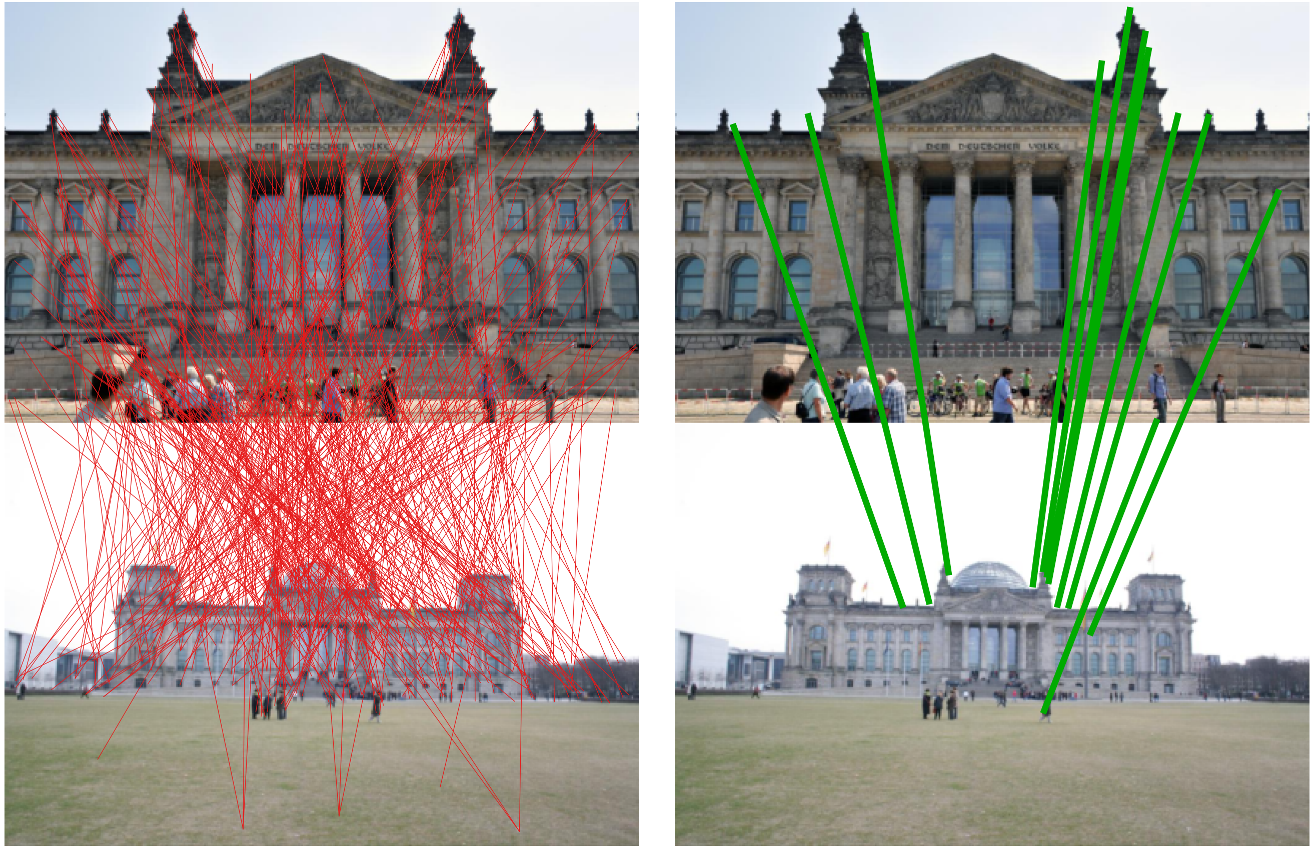


Learning to find good correspondences

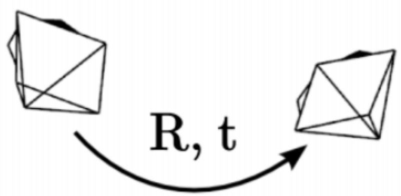
K.M. Yi, E. Trulls, Y. Ono, V. Lepetit, M. Salzmann, P. Fua

Matching with keypoints



(a) Find putative matches

(b) Find inliers (e.g. RANSAC)

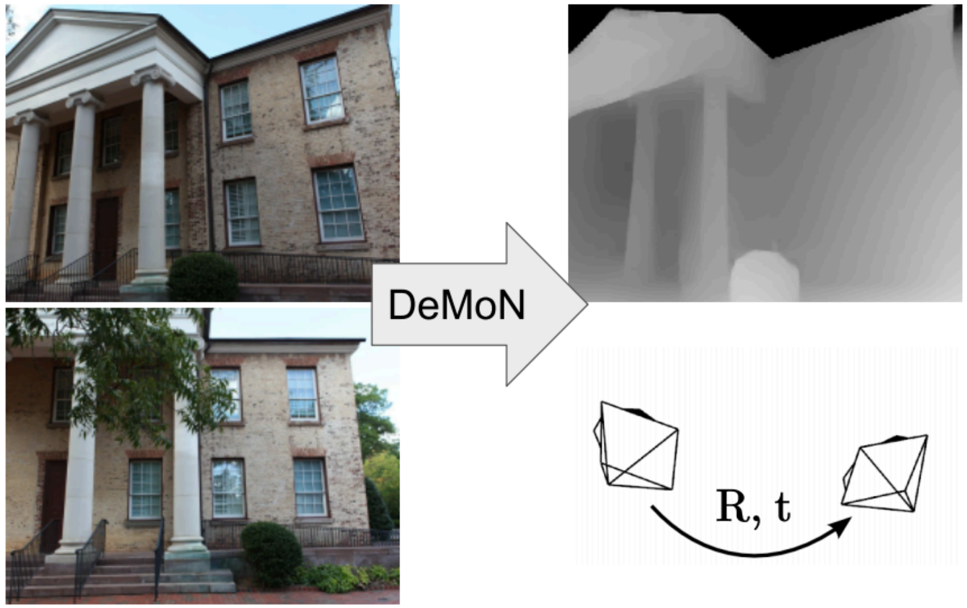


(c) Retrieve pose

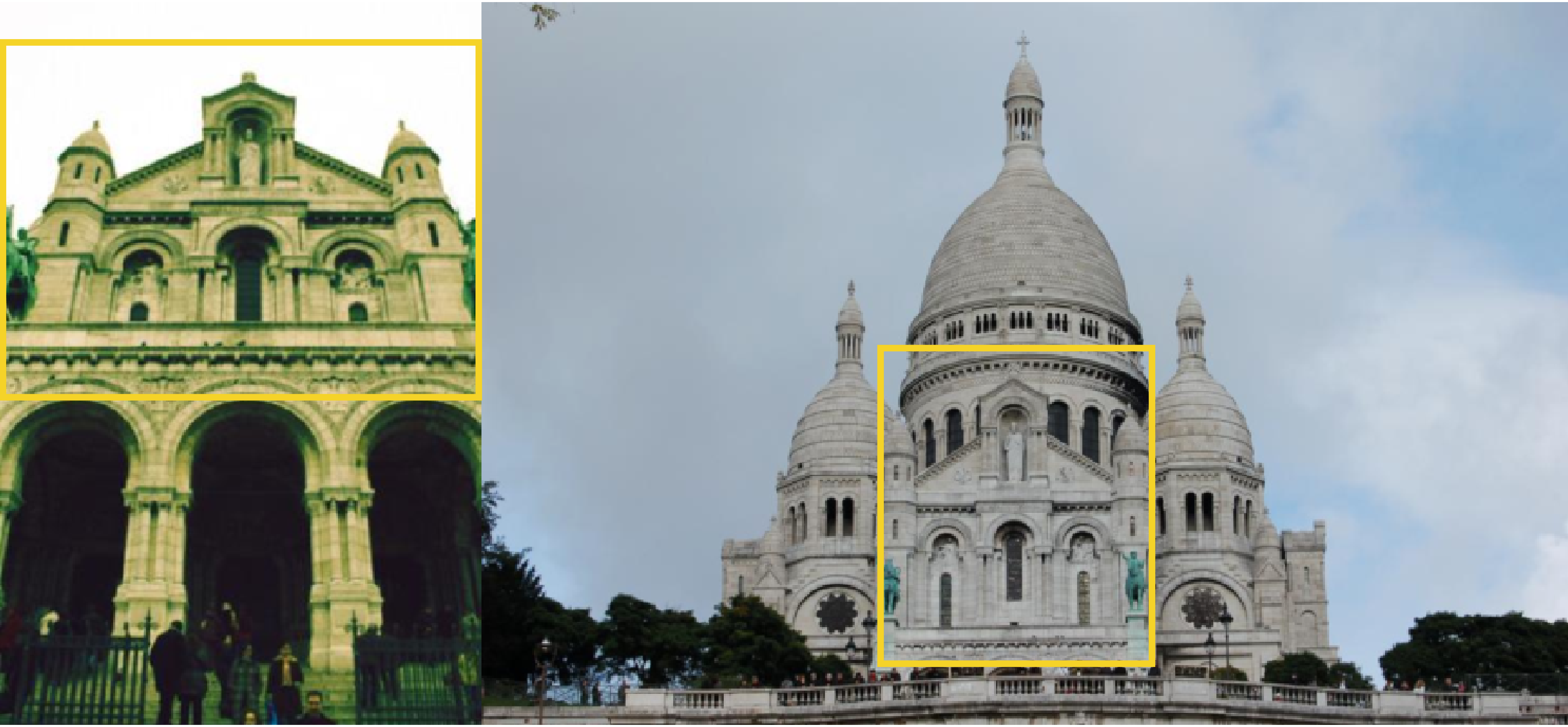
Fischler & Bolles, “Random Sample Consensus”. Comm. ACM, 1981

Dense matching with CNNs

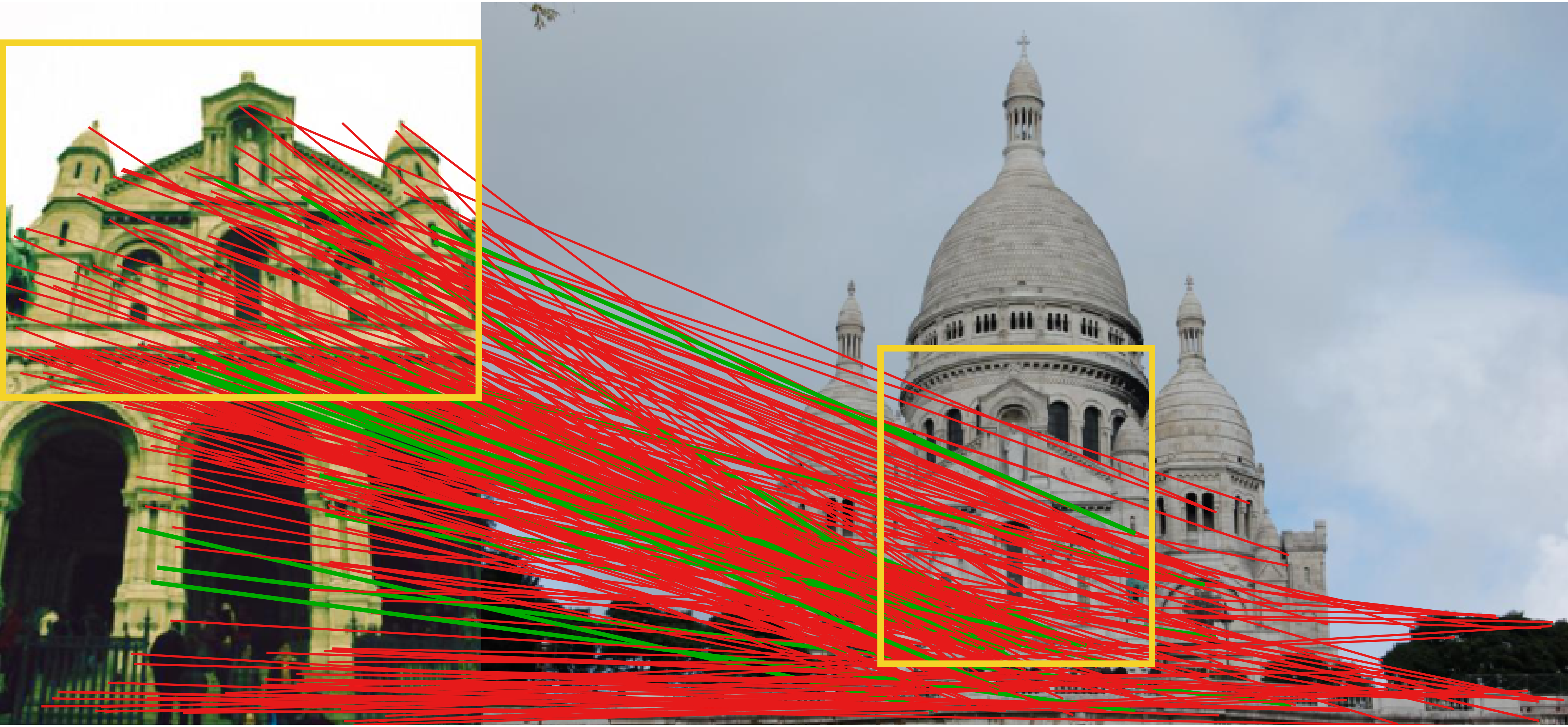
- Current focus of research:
  - ✤ Zamir et al, ECCV'16.
  - ✤ SfM-Net, arxiv'17.
  - ✤ DeMoN, CVPR'17.
  - ✤ Lowe et al, CVPR'17.
- Focus: video, small displacements.
- General case (wide baselines) remains unsolved.



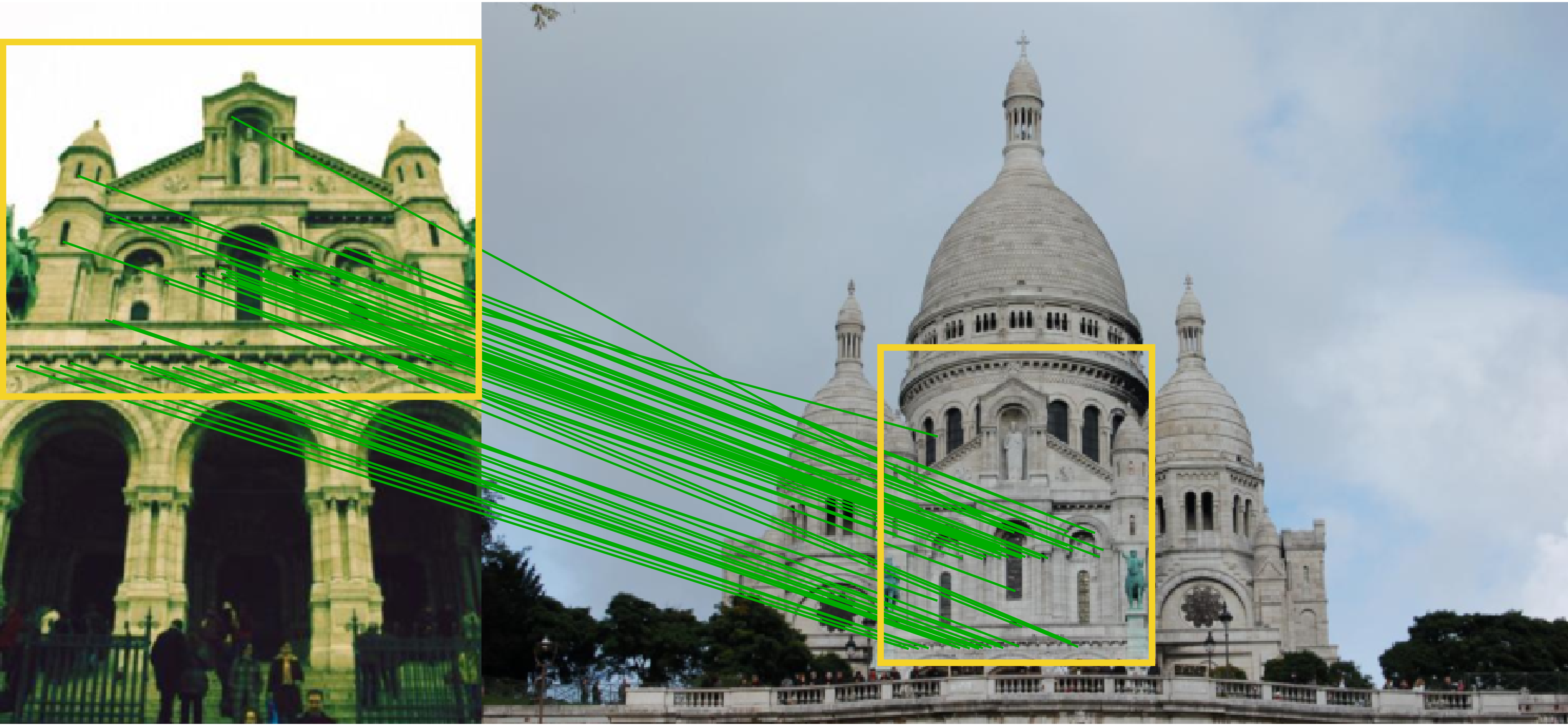
Where's the challenge?



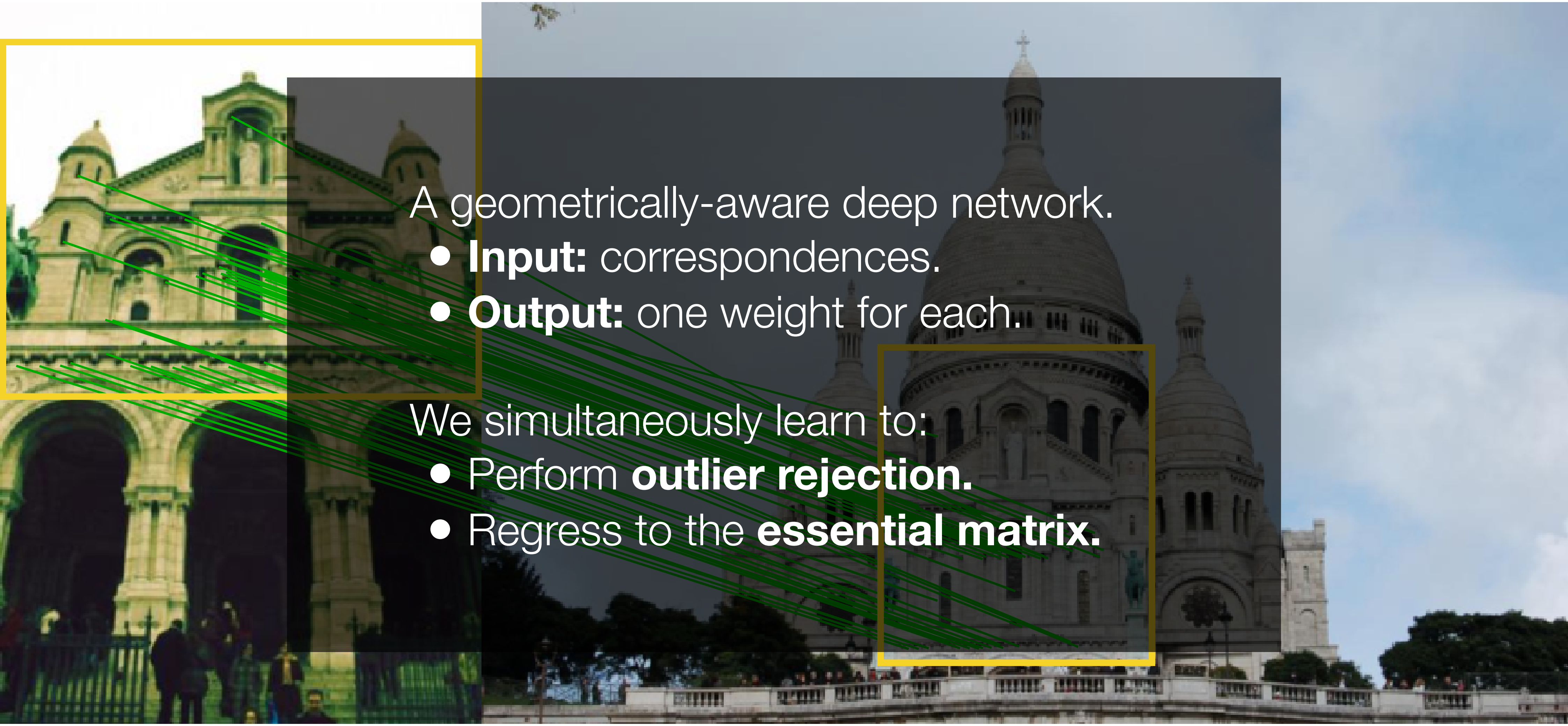
RANSAC: not always enough



Geometry to the rescue



Geometry to the rescue



A geometrically-aware deep network.

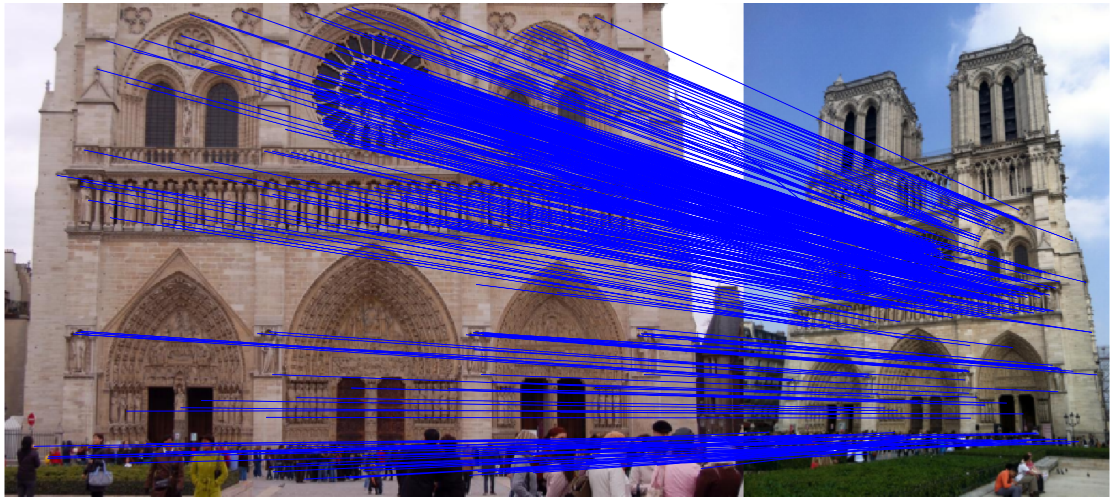
- **Input:** correspondences.
- **Output:** one weight for each.

We simultaneously learn to:

- Perform **outlier rejection.**
- Regress to the **essential matrix.**

Computing the Essential matrix

Closed form solution: **8-point algorithm**



N correspondences

Nx9 matrix

$\{uu', uv', u, \ldots\}$

9x9 matrix

$\mathbf{X}^\top \mathbf{X}$

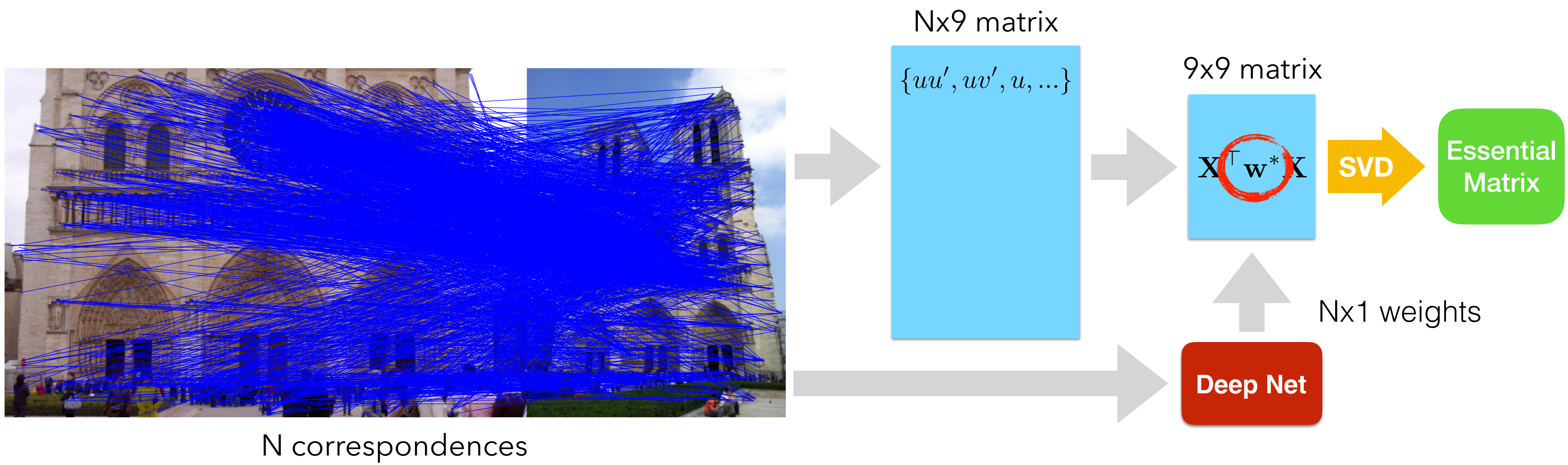
SVD

Essential Matrix

Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections". Nature, 1981.

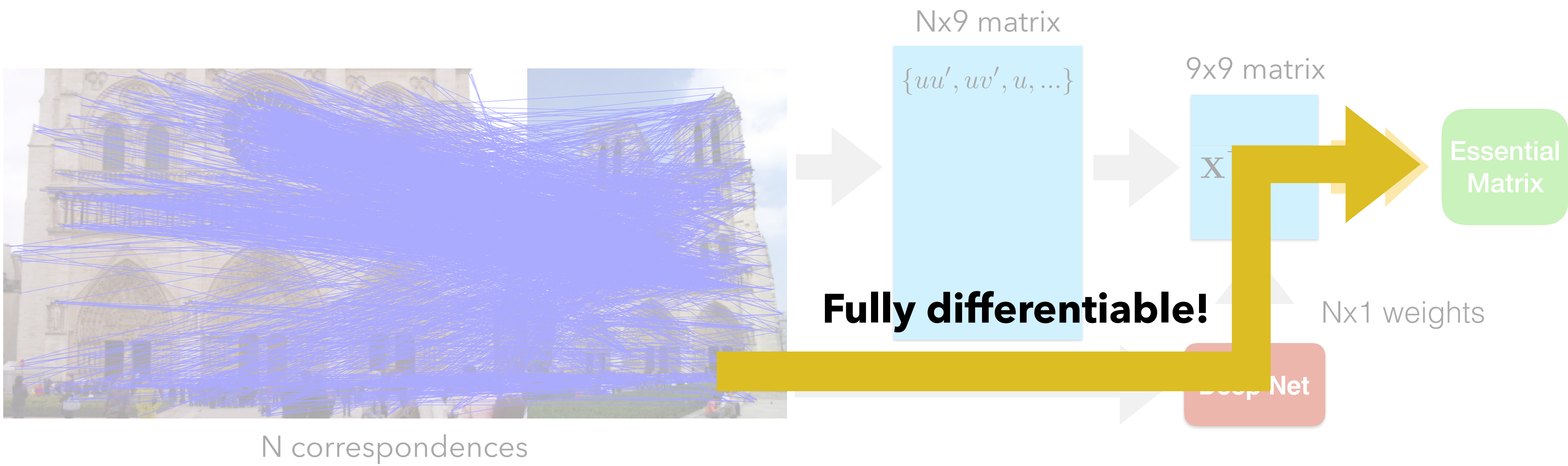
Learning to compute weights

We learn to compute **weights for the 8-point algorithm**



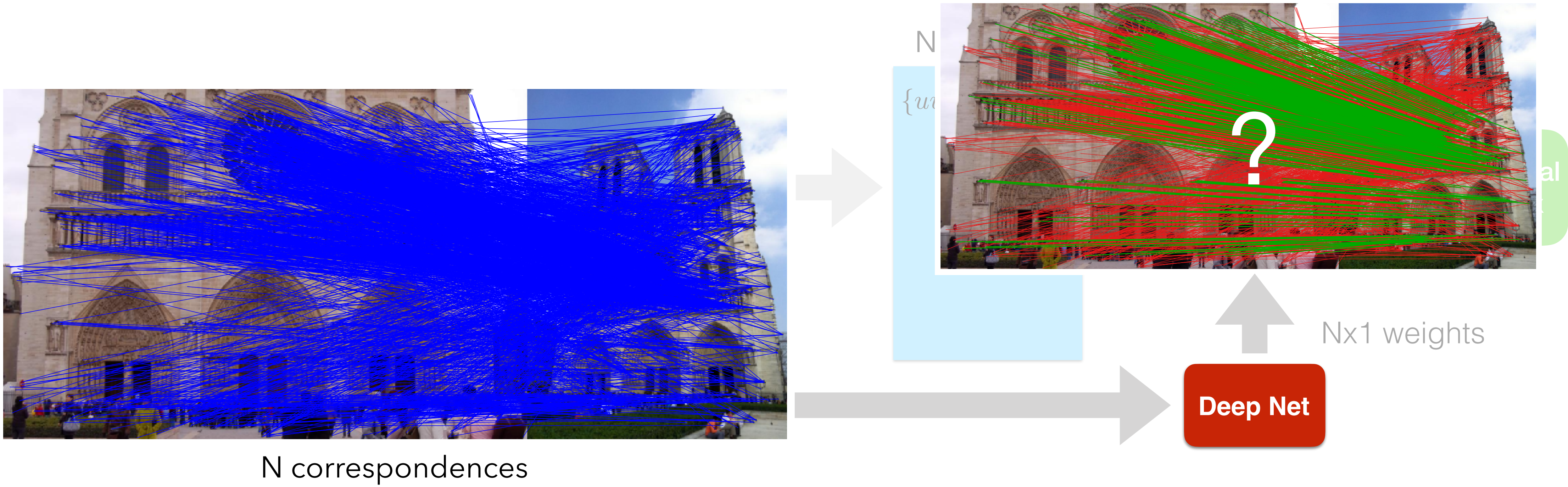
Learning to compute weights

We learn to compute **weights for the 8-point algorithm**

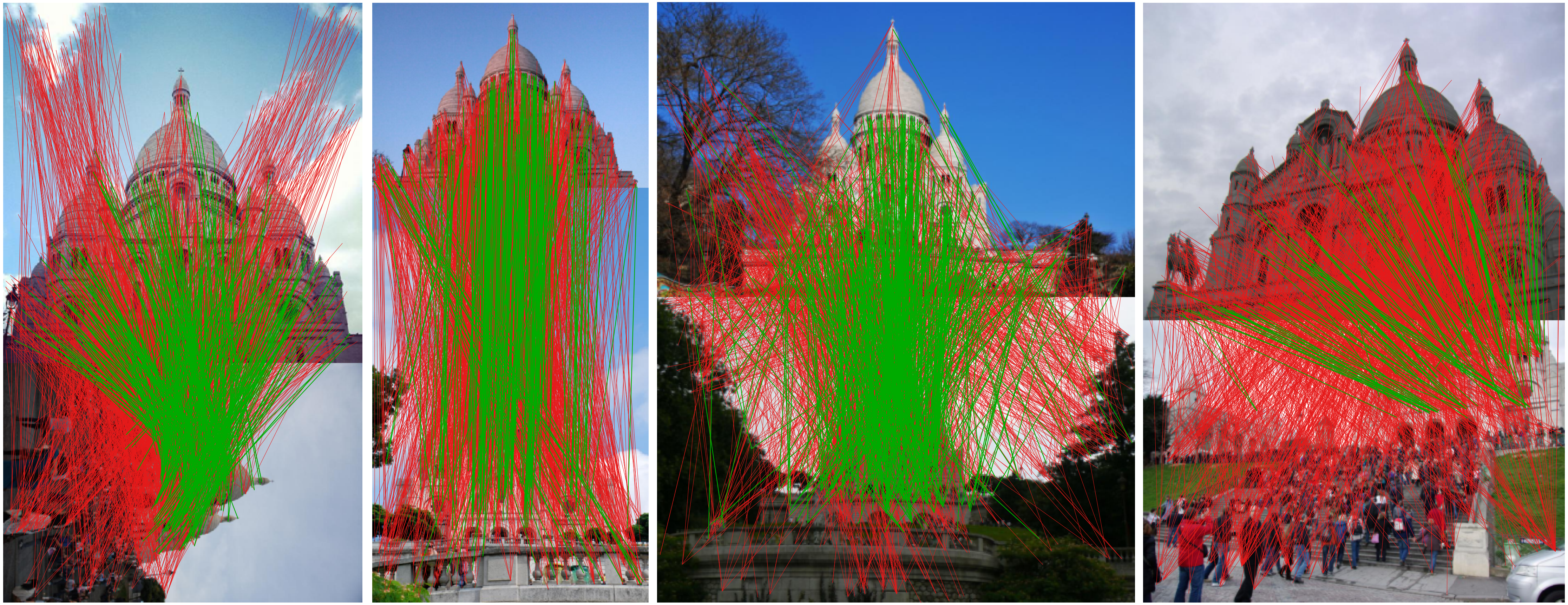


Learning to compute weights

We learn to compute **weights for the 8-point algorithm**



Adding a classification loss



We can build labels from epipolar geometry

Hartley & Zisserman, "Multiple view geometry in computer vision", 2000.

Adding a classification loss

We can build labels from epipolar geometry

Hartley & Zisserman, “Multiple view geometry in computer vision”, 2000.

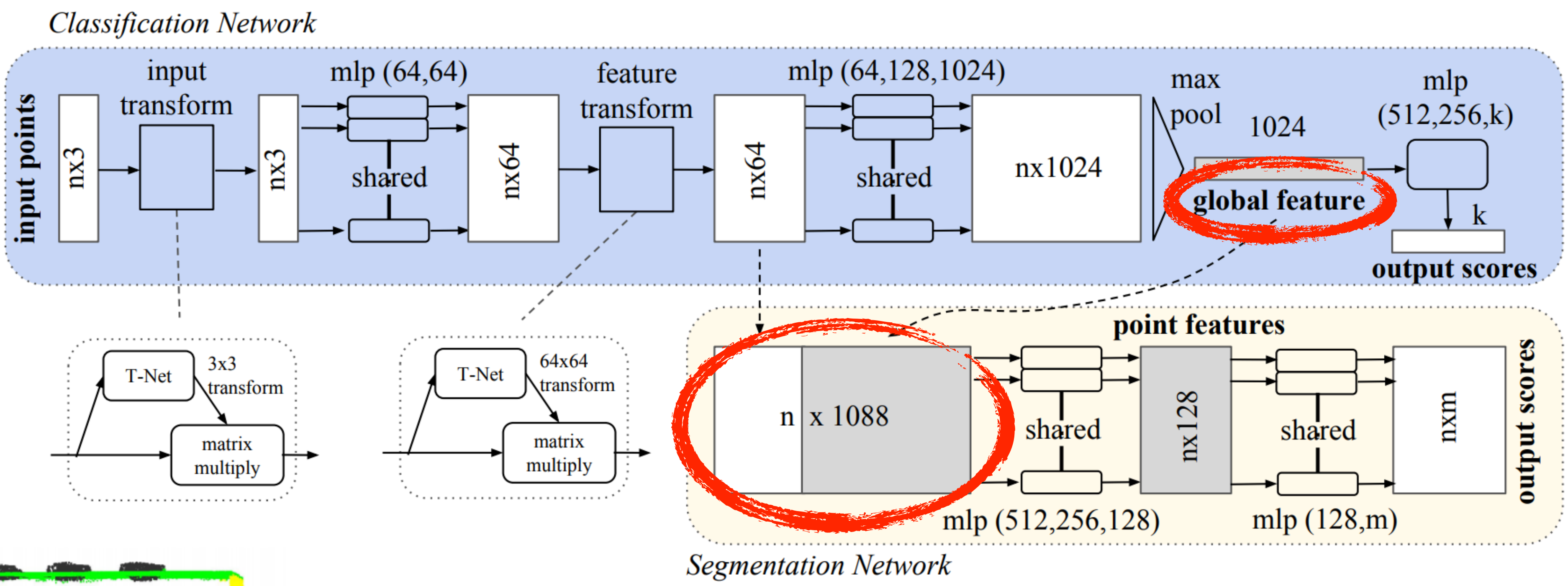
Complete formulation

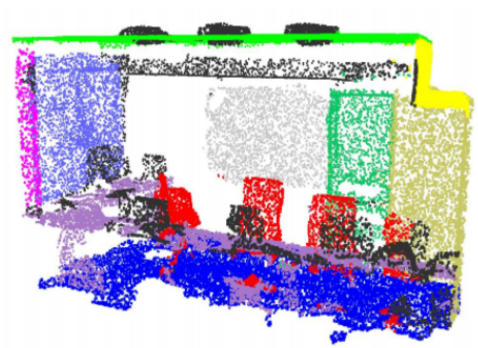
- We jointly train for outlier rejection and regression to the Essential matrix by minimizing the hybrid loss:

$$\mathcal{L}(\Phi) = \sum_{k=1}^{P} \Big(\underbrace{\alpha \mathcal{L}_x(\Phi, \mathbf{x}_k)}_{\substack{\textbf{Classification} \\ \text{(Inliers vs outliers)}}} + \underbrace{\beta \mathcal{L}_e(\Phi, \mathbf{x}_k)}_{\substack{\textbf{Regression} \\ \text{(which inliers help us} \\ \text{retrieve E?)}}} \Big)$$

- For optimal performance, we first minimize the classification loss alone, and then the weighted sum of the two losses.

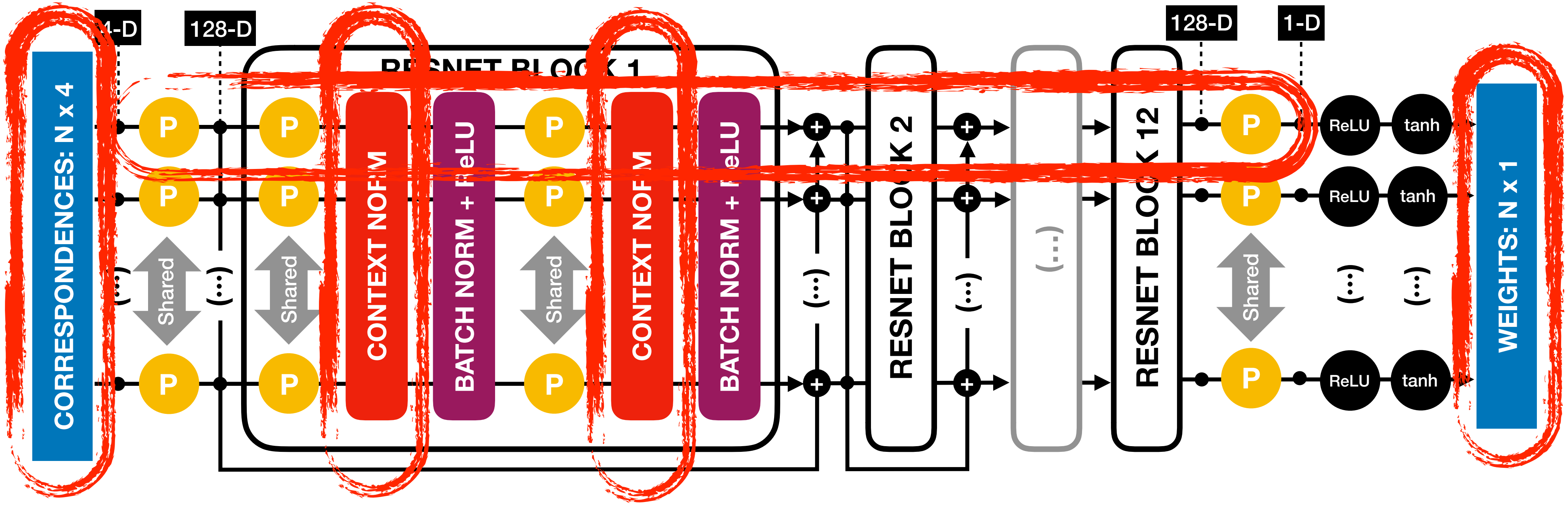
Unordered data





Qi et al. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. CVPR, 2017

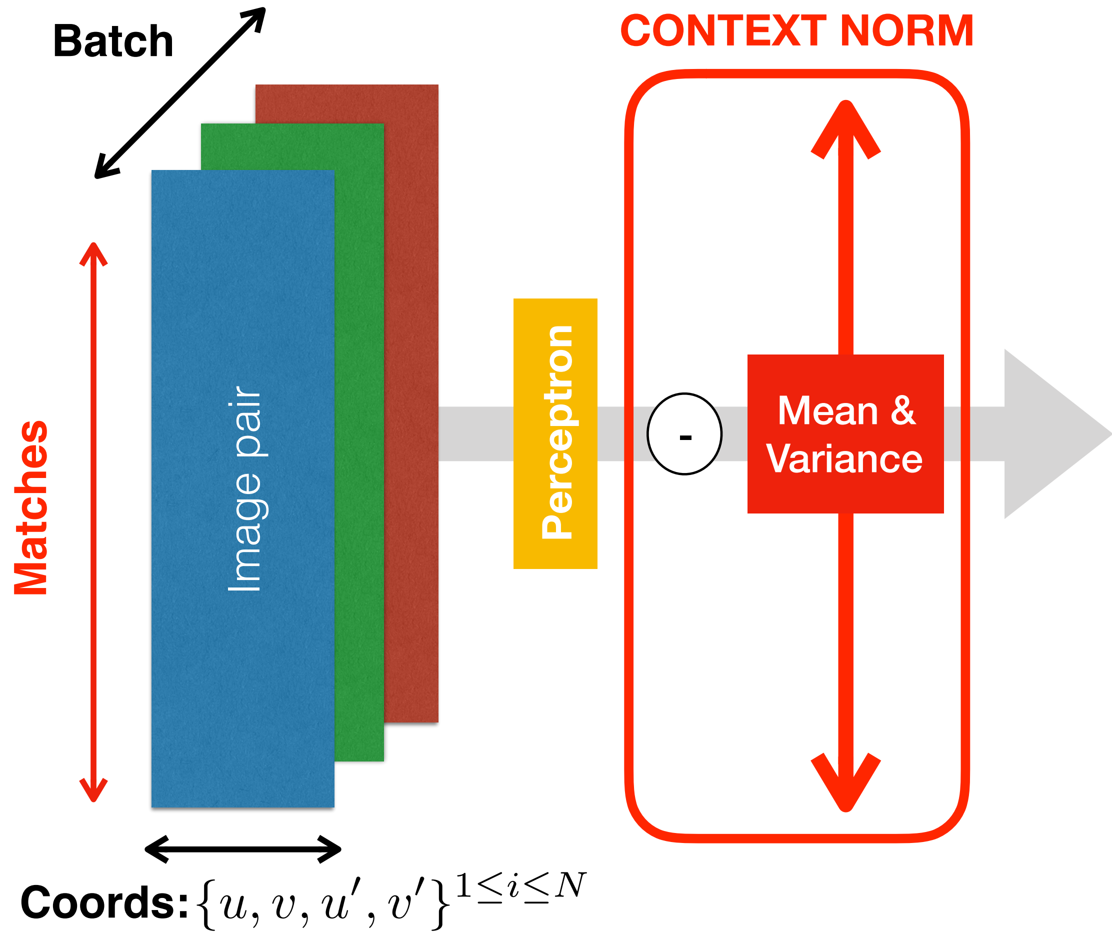
Our network



- **Input:** putative matches (SIFT+NN). Coordinates only: $\{u, v, u', v'\}^{1 \leq i \leq N}$
- **Output:** Weights, encoding inlier probability.
- **Network:** MLPs. Global context embedded via Context Normalization.

Embedding context

- Non-parametric normalization of the mean/std of feature maps.
- Applied over each image pair in the batch separately.
- Also known as Instance Norm, used in image stylization.



Training data

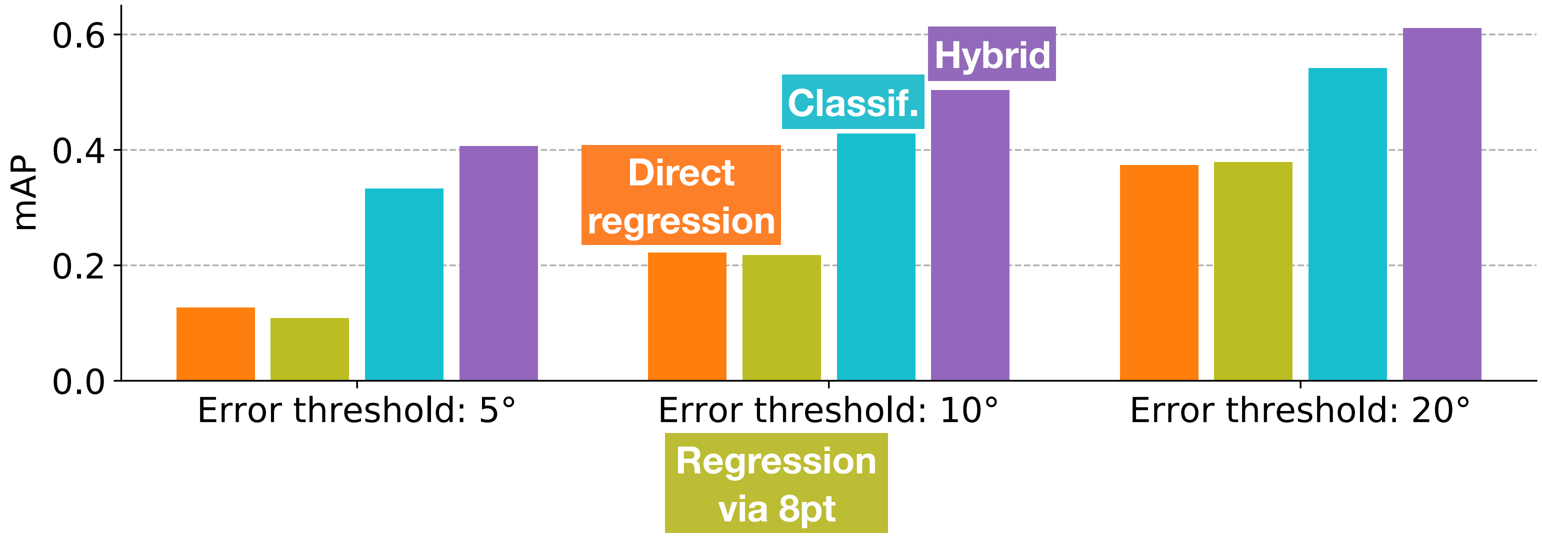
We need **only** the camera poses!

Indoors

Outdoors

Ablation test: hybrid loss

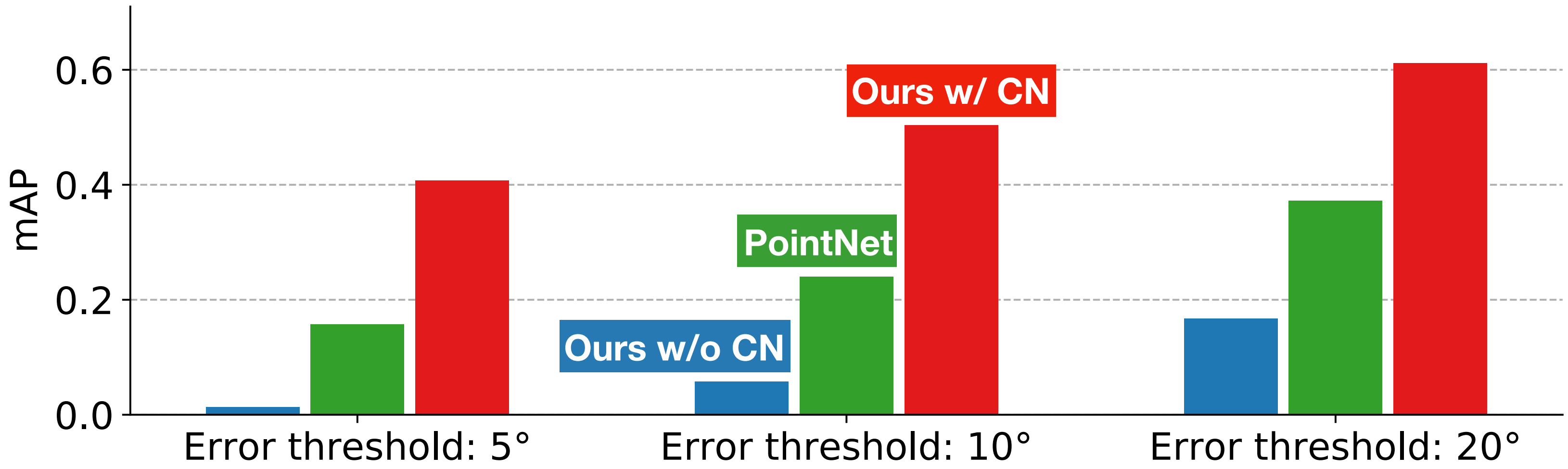
We build cumulative curves thresholding over the error in the estimated pose.
Metric: **mAP,** up to a certain angle (5°, 10°, 20°).



The **classification** loss works, but the **hybrid loss** does best.
Larger margin at smaller thresholds!

Ablation test: Context Norm

We build cumulative curves thresholding over the error in the estimated pose.
Metric: **mAP,** up to a certain angle (5°, 10°, 20°).



Context Normalization outperforms global features (PointNet).

Results

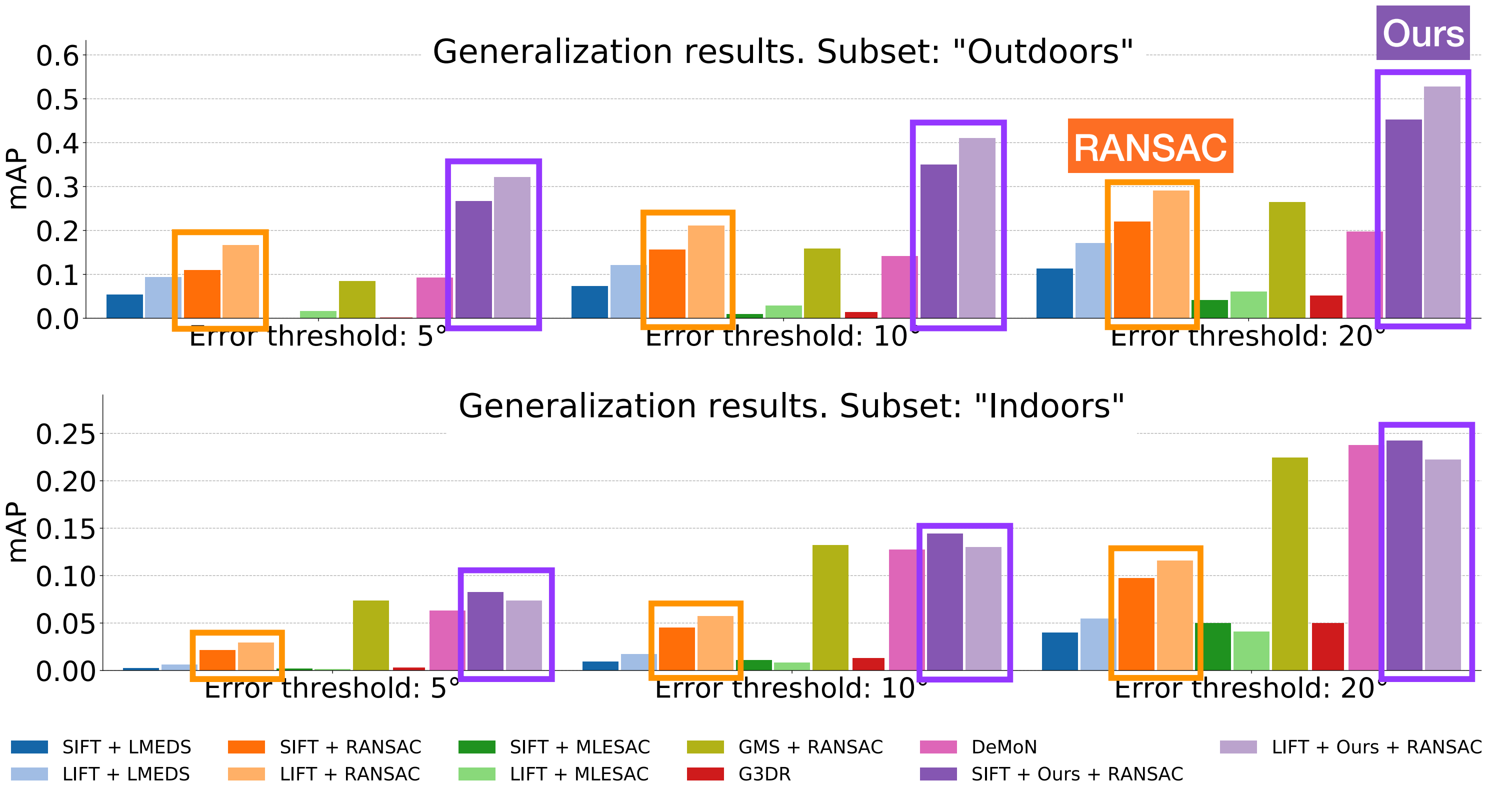
Train on only **two sequences:** one indoors & one outdoors (10k pairs from each):

(i) St. Peter's Square (2.5k images)

(ii) Brown (video, 8k images)

Test on **completely different** sequences (1k pairs from each):

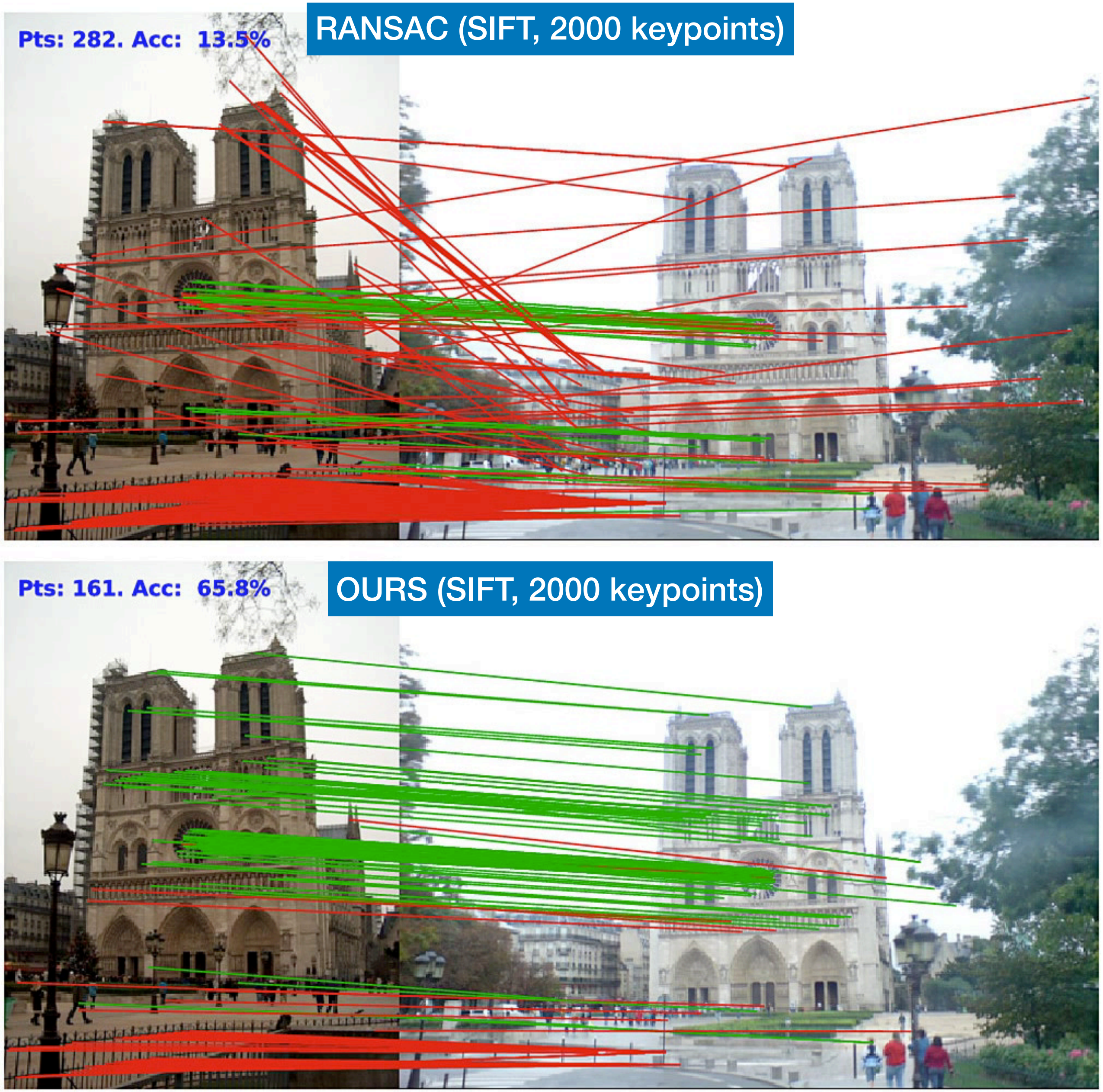
Results



Outdoors: great performance. **Indoors:** slightly better than dense methods.

RANSAC for inference

- At test time, we **do not require differentiability.** We can apply RANSAC!
- Our pipeline:
  1. Forward matches through the network.
  2. Threshold weights to filter them (~15% inliers).
  3. Run RANSAC (~67% inliers).
- **17x times faster** than standalone RANSAC! And **~2x better.**


Pts: 282. Acc: 13.5%
RANSAC (SIFT, 2000 keypoints)
Pts: 161. Acc: 65.8%
OURS (SIFT, 2000 keypoints)

Collaborators

Kwang Yi
(U. Victoria)

Eduard Trulls
(EPFL)

Yuki Ono
(Sony)

Mathieu Salzmann
(EPFL)

Vincent Lepetit
(U. Bordeaux)

Pascal Fua
(EPFL)

Code and models: github.com/vcg-uvic/learned-correspondence-release

Please visit the poster!