DescOver: Descriptor Overlap Leaks the Image

Simona Kocour[1,2] Torsten Sattler[1] Zuzana Kukelova[3] Marc Pollefeys[4,5] Pablo Speciale[4]

[1] Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague

[2] Faculty of Electrical Engineering, Czech Technical University in Prague

[3] Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague

[4] ETH Zurich [5] Microsoft Spatial AI Lab, Zurich

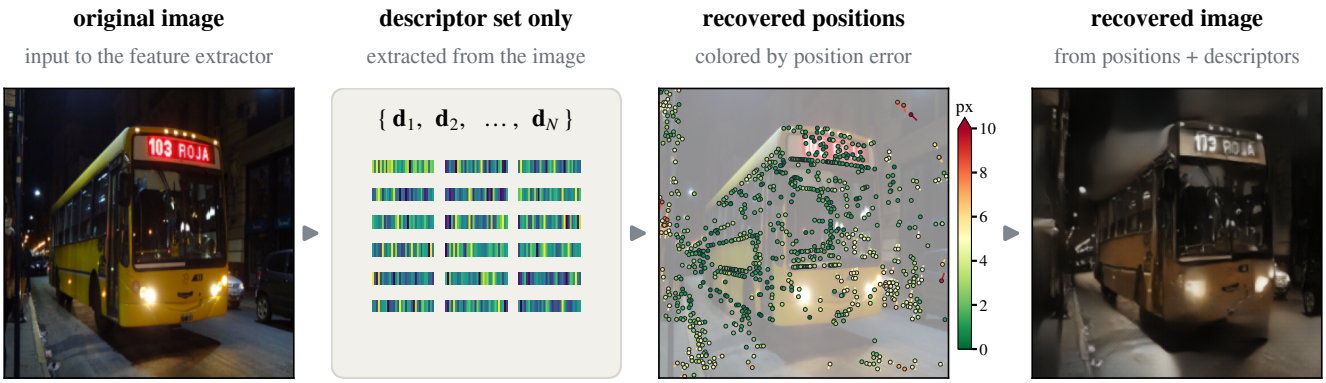


Figure 1. Can an image be reconstructed from its descriptors alone? Recognizable image content can be reconstructed from local features, *i.e.*, descriptors and their 2D keypoint positions. We show that 2D keypoints are not necessarily needed for the recovery process. From descriptors alone, our trained transformer predicts one 2D position per descriptor. An inversion model then reconstructs recognizable content from the descriptors at these predicted positions.

Abstract

Recognizable image content can be reconstructed from keypoint positions and descriptors of local features. This creates potential privacy risks when transmitting such data, e.g.*, in cloud-based visual localization. Obfuscation schemes aim to prevent these inversion attacks while retaining utility for tasks such as camera pose estimation. In this paper, we study the limiting case of geometric obfuscation, where all keypoint positions are discarded. Most modern learned descriptors are typically upright,* i.e.*, computed without rotating image regions to per-keypoint orientations, and for these we show that* ***keypoint positions can be recovered from the descriptors alone****, enabling image inversion. Our key observation is that neighboring descriptors describe overlapping image regions in a common image frame, so their shared content reveals where their keypoints are relative to each other, not only which keypoints are neighbors. A set transformer trained on public images exploits this shared content to jointly predict all keypoint positions. We recover keypoint positions for five learned feature extractors, and the trained solvers also work on image domains unseen during training. Recovery also persists in several tested settings after training on subsampled or quantized descriptors.* ***These results expose a limit of geometric obfuscation****: for the descriptors studied, recognizable image content can be reconstructed even with all keypoint positions discarded. We will make our code and trained models publicly available.*

1. Introduction

Many applications require users to send image data to external services, *e.g.*, cloud-based 3D reconstruction [1, 15] or client-server-based visual localization [30, 55]. Sending images leads to potential privacy risks. Thus, it is common to extract features from the images and send the features instead of the original image. For example, in the context of visual localization, a client might send local image features to a server, where they are used to estimate the camera pose w.r.t. a given 3D model of the scene [55]. Each local feature consists of a 2D keypoint position and a descriptor of the surrounding image content. However, inversion attacks can reconstruct recognizable image content from the positions and descriptors of sparse features [43, 45].

Inversion attacks can potentially be prevented by obfuscating the keypoint positions [15–17, 23, 24, 32, 39, 55] or descriptors [13, 36, 40, 41, 44]. Geometric obfuscation introduces ambiguities in the positions, *e.g.*, by replacing each

2D position with a random line through it [55]. Descriptor obfuscation introduces ambiguities into the descriptors such that they do not carry enough unique information to enable inversion [13, 41, 44]. Obfuscation strength needs to be balanced with utility [44], as the obfuscated entities still need to be informative enough for downstream tasks, *e.g.*, feature matching and camera pose estimation for localization.

Chelani *et al*. [5, 7] showed that, by estimating which keypoints are neighbors from the descriptors alone [7], positions close to the original keypoints can be recovered for many geometric obfuscation schemes [15–17, 24, 32, 39, 54, 55], while some schemes are still believed to be safe [23, 33].

In this paper, we consider the limiting case of geometric obfuscation, where only descriptors are available, without keypoint coordinates.[1] Our *solver* is a set transformer that predicts one 2D position per input descriptor. Using the recovered keypoint positions and the descriptors, the image content can then be recovered using existing image inversion methods. We train multiple solvers on public images, one for each type of local feature.

Our approach is based on a relatively simple key insight. Feature descriptors describe local patches around a keypoint. Thus, two descriptors with overlapping patches partially describe the same image content. This overlap in content can then be used to reason about the relative positioning of the two patches and thus the two keypoints.

The relation between the descriptors and the overlap of their corresponding patches can be very complex if each patch is rotated and rescaled to its own keypoint's orientation and scale, as in SIFT [29]. However, many modern learned extractors, including ALIKED [68], SuperPoint [10], XFeat [46], DISK [59] and LoMa [37], produce upright descriptors, *i.e.*, their image regions are aligned with the image axes. These extractors also omit explicit per-keypoint scale normalization. In this case, the relationship becomes easier to learn. The fact that our solvers are able to recover the keypoint positions quite accurately from the descriptors shows that they are able to exploit this shared information between descriptors (Sec. 4.1).

For the descriptors studied, our results show that geometric obfuscation can be circumvented by ignoring the obfuscated keypoints and recovering their positions with our solvers. Naturally, this raises questions about privacy. Since there is no clear definition of privacy [43], we use the following approach: We compare the images recovered using the keypoint positions predicted by our solvers to the images recovered using the ground truth keypoint positions. If the latter are privacy-sensitive, *i.e.*, they might contain content that a user might not want to share, then being able to accurately reconstruct them shows that there is a potential privacy leak. Furthermore, we show qualitatively that recognizable objects, faces and some document text can be recovered.

Contributions. **(1)** We show that upright local descriptors can reveal the relative 2D keypoint positions and recognizable image content even when all positions are discarded. **(2)** We explain how overlapping image regions in the same orientation and scale provide clues about relative keypoint positions, supported by controlled tests. We also recover positions and reconstruct image content from oriented SIFT [29] when each descriptor is transmitted with its keypoint's scale and orientation, computed by the extractor (Secs. 4.1 and 4.3). **(3)** We propose a set transformer-based solver that predicts one 2D position per descriptor in an unordered set, up to a common translation. We recover keypoint positions for five learned feature extractors, and the trained ALIKED solver also works on image domains unseen during training, including faces and documents (Sec. 4.1). **(4)** Solvers adapted to subsampled or quantized descriptors still recover keypoint positions in several tested settings; stronger modifications also reduce localization accuracy (Sec. 4.4). **(5)** We will release our source code and our trained models.

2. Related Work

Inverting local features. Prior work reconstructs images from local descriptors such as SIFT and HOG, and from spatial CNN feature maps [11, 31, 61, 65]. Images can also be reconstructed from sparse structure-from-motion points and their descriptors [43, 45], and from colored point clouds without descriptors [53]. These methods use known spatial information. Kato and Harada [22] discard feature positions but synthesize a texture rather than aiming to recover the original keypoint positions. We estimate 2D keypoint positions from descriptors alone, enabling image inversion.

Image and descriptor obfuscation. Descriptor obfuscation approaches introduce ambiguities into the descriptors, *e.g.*, by lifting descriptors to represent subspaces [13], through adversarial training designed to make image inversion quality worse [36], or through quantization and representing each feature through multiple descriptors [44]. Note that [13, 36] still allow image inversion [43, 44], while [44] does not make any guarantees that image content cannot be recovered from their obfuscated descriptors and keypoints. Recovering this geometry requires descriptors that are informative enough to reveal patch overlap. Recovery images from these obfuscated descriptor schemes is outside the scope of this work. In this paper, we focus on obfuscating keypoint positions, *i.e.*, the geometry of the features.

Obfuscating the whole image [40–42] or only privacy-relevant parts [8, 40, 62] are alternatives to descriptor obfuscation. *E.g.*, [40–42] introduce ambiguity by replacing color values through segmentation labels, since this many-to-one

[1] In the context of visual localization, this limit case is not practical as some information about the keypoints needs to be sent to the server to facilitate camera pose estimation. For other applications with privacy implications, just sending descriptors might be practical.

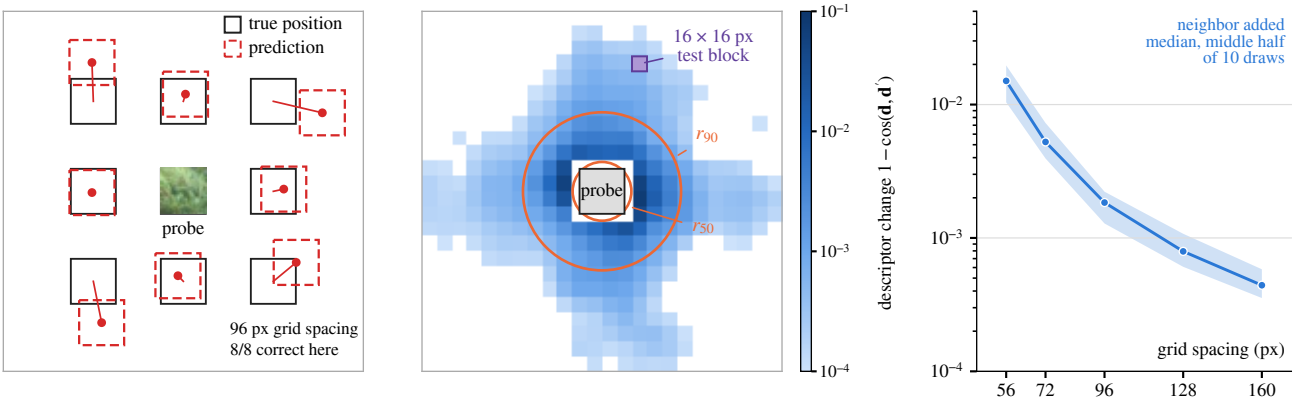


Figure 2. ALIKED descriptor influence and position recovery. A probe photograph stays fixed on a white background. **Left:** one neighbor photograph is added per trial; the solver uses only descriptors from both photographs. □ True and ⬚ predicted positions have the correct direction in all eight trials shown. **Middle:** ■ descriptor change when a 16×16 px block is added at each location, averaged over 15 probe keypoints. ○ r_{50} and r_{90} are median radii enclosing 50% and 90% of change in individual keypoint maps. **Right:** change in the probe's descriptors versus neighbor grid spacing.

mapping makes it harder to recover the original image content [43]. Panek *et al*. and Dangwal *et al*. [8, 40] identify potentially sensitive regions, such as text or faces, and either mask out the regions [40] or ensure that no keypoints are detected in these regions [8, 62]. This line of work is not affected by our approach.

Geometric obfuscation. Orthogonal to descriptor obfuscation, one can also obfuscate the keypoint positions. Geometric obfuscation schemes typically introduce ambiguities by lifting keypoints to 2D lines [15, 16, 23, 24, 39, 55]. This leaves one or more degrees of freedom in the position of each keypoint. As a result, camera pose estimation becomes more complicated, and potentially less accurate, compared to using 2D keypoint positions. We study the limit of geometric obfuscation for image queries by discarding every keypoint position. For the descriptors studied, we show that keypoint positions can be recovered from the descriptors alone, regardless of the geometric obfuscation scheme. Geometric obfuscation thus needs to be combined with techniques that make keypoint recovery harder, *e.g.*, sparsifying the keypoints or using less informative descriptors; adapted solvers still recover positions in several such settings (Sec. 4.4).

Related schemes obfuscate 3D maps [17, 24, 32, 39, 54]. Scene information can be recovered from obfuscated point clouds [5, 7], camera poses [6], or scene coordinate regression weights [34].

Recovering obfuscated geometry. Chelani *et al*. [5, 7] recover approximate keypoint positions from most geometric obfuscation schemes [17, 24, 39, 54, 55] using which keypoints were nearest neighbors, information they extract from the descriptors [7]. However, their approach needs to be adapted to each obfuscation scheme, and neighborhood information does not help in all cases [23, 33]. Our solver instead recovers keypoint positions from descriptors, enabling image reconstruction without the obfuscated geometry.

3. Method

Our goal is to estimate 2D keypoint positions from descriptors alone, then use those positions for image reconstruction (Fig. 1). The client extracts local features from a private query image I and transmits only the unordered descriptor set $\mathcal{D} = \{\boldsymbol{d}_i\}_{i=1}^{N}$, where $\boldsymbol{d}_i \in \mathbb{R}^D$. The corresponding positions $\boldsymbol{X} = \{\boldsymbol{x}_i \in \mathbb{R}^2\}_{i=1}^{N}$ are withheld. The recovered positions are defined up to a common translation.

We assume a known feature extractor, preprocessing and input image resolution, and train a separate solver for each extractor. Our main setting uses upright learned descriptors, such as ALIKED, SuperPoint, XFeat, DISK and LoMa [10, 37, 46, 59, 68], computed without per-keypoint rotation or scale normalization. We recover positions in pixels of the input image. SIFT descriptors accompanied by keypoint scale and orientation are evaluated as a separate input setting (Sec. 4.3).

The pipeline has two stages. A set transformer, the *solver*, predicts a position for each descriptor. An image inversion network, the *inverter*, reconstructs an image from the descriptors at those positions. During training, keypoint positions extracted from public images supervise the solver and provide the inverter's spatial input. For a private query, the solver receives only descriptors, and the inverter uses the predicted positions.

3.1. The global set transformer solver

Why a global solver? Nearby descriptors depend on overlapping image regions. In the common upright frame, shared content provides clues to the relative offset $\boldsymbol{o}_{ij} = \boldsymbol{x}_j - \boldsymbol{x}_i$ between keypoints. Fig. 2 (left) tests position recovery from the two photographs' descriptors. The extractor processes the whole image, so a blank gap between photographs does not isolate their descriptors.

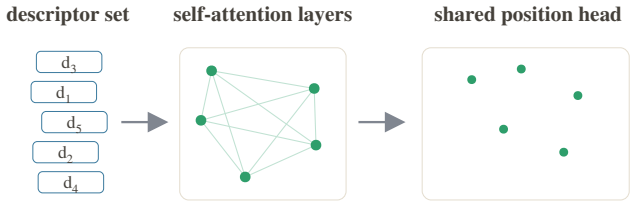


Figure 3. From unordered descriptors to keypoint positions. Each descriptor becomes a token, without positional encoding. Global self-attention combines the tokens; a shared head predicts one 2D position per descriptor in one forward pass.

The middle panel instead measures descriptor influence by adding a 16×16 px textured block at one grid location at a time. We measure change as $1-\cos(\boldsymbol{d},\boldsymbol{d}')$ at fixed probe keypoints, without ($\boldsymbol{d}$) and with ($\boldsymbol{d}'$) the added content. Repeating the neighbor-photograph experiment at different spacings (right) shows that its influence weakens with distance. In this example, the farther diagonal neighbors have weaker influence and larger placement errors. Joint prediction lets the solver combine weak responses from many descriptors.

Architecture. The solver $f_{\boldsymbol{\theta}}\colon \mathcal{D} \mapsto \{\hat{\boldsymbol{x}}_i \in \mathbb{R}^2\}_{i=1}^{N}$ predicts all positions jointly, with learned network parameters $\boldsymbol{\theta}$ (Fig. 3). It outputs one 2D position per descriptor and, optionally, a per-axis uncertainty estimate $\boldsymbol{\sigma}_i$ from a separate head. A shared linear embedding converts each descriptor into one token, without positional encoding. Self-attention lets every token use information from the whole set, and a shared output head predicts its position in one forward pass [25, 60, 66]. The solver is permutation-equivariant: reordering the descriptors reorders the predicted positions in the same way, preserving the descriptor-to-position correspondence. No per-query optimization is performed.

Training loss. We compare predicted and ground-truth (GT) positions after centering each set independently. The centroid $\boldsymbol{\mu}(\boldsymbol{X}) = \frac{1}{N}\sum_i \boldsymbol{x}_i$ is the mean keypoint position. The centered positions are $\boldsymbol{C}(\boldsymbol{X}) = \boldsymbol{X} - \boldsymbol{\mu}(\boldsymbol{X})$, obtained by subtracting this mean from every position. We define the position error as $\boldsymbol{\Delta} = \boldsymbol{C}(\hat{\boldsymbol{X}}) - \boldsymbol{C}(\boldsymbol{X})$ and train the position output with a smooth-L_1 (Huber) loss. An optional uncertainty head predicts per-axis standard deviations $\boldsymbol{\sigma}_i$ and is trained with a Gaussian negative log-likelihood on the detached position error. The combined loss is

$$\mathcal{L} = \mathrm{Huber}(\boldsymbol{\Delta}) + \lambda\,\mathrm{NLL}(\mathrm{sg}[\boldsymbol{\Delta}];\boldsymbol{\sigma}), \qquad (1)$$

where $\lambda = 1$ and sg stops gradients through $\boldsymbol{\Delta}$. Details of detachment and shared-backbone training are in Supp. S1.

3.2. From keypoint positions to an image

We place each descriptor on a feature grid as a small Gaussian centered at its predicted position. A U-Net decodes that grid into an image [47]. It is trained with GT positions from public images and uses predicted positions for query reconstruction. Pixel metrics and all reconstruction examples shift the predicted positions to the GT centroid (Sec. 4.2), except the teaser (Fig. 1), which places them at the image center, as an attacker without ground truth would. The recovery and inversion networks are trained separately. Other inverters accepting descriptors and positions can use the same predicted coordinates, but reconstruction quality depends on both the inverter and the position errors. Section 4.2 evaluates these effects with GT-position references and inversion controls.

4. Experiments

We first evaluate layout recovery across extractors and image domains and examine where the positional information comes from (Sec. 4.1). We then compare the image content reconstructed from recovered and from ground-truth (GT) positions (Sec. 4.2) and extend the attack to SIFT released with scale and orientation (Sec. 4.3). Finally, we test whether fewer or coarser descriptors prevent recovery while preserving localization accuracy (Sec. 4.4).

Setup. We refer to the assignment of a 2D position to each descriptor as the keypoint layout, defined up to a global translation. The solver is our set transformer (Sec. 3.1), trained separately for each extractor. We train on public COCO images [27] and evaluate on a held-out split.

Our main extractor is ALIKED [68], with its upright, fixed-scale `aliked-n16` descriptors. We use the authors' pretrained weights with detection threshold 0.2 and keep at most 2048 keypoints per image. Images are resized to 512 px on the longest side. ALIKED upsamples them internally to 1024 px during extraction and maps the keypoint positions back to the 512 px frame. We also evaluate four popular upright extractors, XFeat, SuperPoint, DISK and LoMa (Tab. 1; implementation details in Supp. S1).

Metrics. Each layout metric is computed per image; we report its median over test images. The *residual*, or position error, is the median distance in pixels between recovered and GT keypoint positions after aligning the layout centroids. The *spacing* is the median distance from a GT keypoint to its nearest neighbor. *Res./Sp.* divides the median residual by the median spacing; at 1, keypoints are typically misplaced by about one keypoint spacing. *dcorr* is the Pearson correlation between recovered and GT distances over all keypoint pairs. *kNN* is the fraction of each keypoint's five nearest neighbors in the GT layout that are also among its five nearest neighbors in the recovered layout, averaged over keypoints. Both (dcorr, kNN) are 1 for a perfect layout and $\approx$0 for random positions.

For image recovery, we use the reconstruction from GT keypoint positions as the *reference reconstruction*, since this represents what an inverter can recover when the original features are available. We compare reconstructions from recovered and random positions with this reference using PSNR and SSIM on color images, LPIPS [67] on grayscale

Table 1. Layout recovery across extractors and image domains. Top: 200 held-out COCO images [27]. Bottom: five other domains. Residual: median (25th–75th percentile) over images; Res./Sp.: residual divided by spacing (Sec. 4). Recovered positions assigned to keypoints at random: residual ≈190–230 px, kNN and dcorr 0.00 (COCO). In training and at test time, at most 2048 keypoints per image for every extractor.

	Spacing	Residual (px) ↓	Res./Sp.	kNN ↑	dcorr ↑
COCO, each extractor with its own solver					
ALIKED [68]	4.2	4.0 (3.3–5.1)	0.94	0.89	0.999
XFeat [46]	5.1	2.9 (2.6–3.4)	0.57	0.91	1.000
SuperPoint [10]	3.0	4.7 (4.1–5.5)	1.57	0.88	0.998
DISK [59]	5.7	3.2 (2.8–3.8)	0.57	0.87	0.999
LoMa [37]	3.5	7.0 (5.7–9.3)	1.99	0.83	0.997
ALIKED solver trained on COCO, out-of-domain, no retraining					
MegaDepth [26]	4.1	4.1 (3.4–4.9)	0.98	0.90	0.999
DIV2K [2]	4.3	4.3 (3.3–5.7)	1.00	0.89	0.999
ADE20K [69]	4.4	4.2 (3.4–5.3)	0.95	0.90	0.999
CelebA-HQ [21]	4.5	5.6 (4.4–6.8)	1.25	0.88	0.999
FUNSD forms [19]	3.4	5.9 (4.6–7.4)	1.73	0.88	0.998

images, and object recovery (*OR*). OR measures the fraction of objects identified by Qwen2-VL [63] in the reference that are also identified in the reconstruction. To measure the inverter's reconstruction error, we compare the original image with the image reconstructed using the GT keypoint positions.

4.1. Layout recovery

Recovery across extractors and domains. Using only unordered descriptors, the solver recovers the keypoint layouts of all five extractors with a median position error of 0.6–2.0 times the median nearest-neighbor distance between GT keypoints (Res./Sp., Tab. 1). For ALIKED, the residual is 4.0 px against a 4.2 px spacing, and on average 4.5 of each keypoint's five nearest neighbors are kept (kNN 0.89). Even LoMa, whose upright descriptors are partly robust to rotation, reveals its layout. The ALIKED solver also generalizes without retraining from COCO to five public image collections: 806 landmark photographs (all images of the two MegaDepth-1500 test scenes [26]), 300 high-resolution photographs (DIV2K training set [2]), 500 indoor and outdoor scenes (ADE20K validation set [69]), 200 faces (CelebA-HQ [21]) and the 50 FUNSD test forms [19]. Across these domains, the residual stays between 4.1 and 5.9 px (Tab. 1 Bottom, Fig. 5).

We compare our set transformer with four alternatives (Tab. 2). Classical MDS [57] and Isomap [56] embed descriptors according to their distances in descriptor space, but recover little of the layout. Deep Sets [66] processes each descriptor independently and combines it with a pooled set representation. It recovers the coarse arrangement but not the local neighborhoods. A graph network [64] that exchanges information between similar descriptors performs better, while our set transformer, which allows every descriptor to attend to all others, is the most accurate. These results show that accurate layout recovery benefits from combining information across the descriptor set.

Table 2. Comparing methods for recovering the keypoint layout. All methods use ALIKED descriptors and 200 held-out COCO images [27]; the learned ones are trained on the same 15k images for 80 epochs with about 14.2M parameters each. MDS and Isomap need no training and are aligned to the ground truth by rotation, reflection, scale and translation, the learned methods by translation only. Our set transformer is the most accurate; with the full training set it reaches 4.0 px residual error (Tab. 1).

Solver	Residual (px) ↓	dcorr ↑	kNN ↑
Classical MDS [57]	137.9	0.04	0.04
Isomap [56]	124.3	0.18	0.09
Deep Sets [66]	28.6	0.93	0.27
Descriptor-kNN GNN [64]	11.5	0.99	0.69
Set transformer (ours)	**6.1**	**1.00**	**0.84**

When does relative-position recovery work? We place two different or identical 64 px paintings [35] side by side or stacked on a plain wall, vary the gap between their edges and measure the distance between the centroids of their keypoints as recovered by the COCO-trained ALIKED solver (Fig. 4). On a 1024 px wall, at least 353 px from every edge, only the paintings themselves tell the solver where they lie relative to each other (Fig. 4c). This wall is larger than any training image we used, but small gaps are still recovered accurately. Different paintings keep their distance up to gaps of about 50 px and nearly coincide from about 100 px. While the gap is small, descriptors near it also capture part of the other painting, whose influence fades with distance (Fig. 2, right), and the solver uses this shared content to place the two paintings (Supp. S11). In real images, intervening keypoints can form chains between regions (Supp. S4). Identical copies, with nearly identical descriptors, lose their distance from 40–50 px. On a 512 px wall, as in our test images, the descriptors also change with where a painting lies in the image (Fig. 4b). This position cue keeps different paintings, and identical ones side by side, apart up to gaps of 160 px, but stacked identical copies collapse from about 64 px. The two copies differ only in where they lie in the image, and ALIKED's descriptors change less with vertical than with horizontal position. At a 128 px gap, stacked copies are therefore more alike than side-by-side ones (median cosine distance between their descriptors at the same keypoints 0.013 vs. 0.018), and the solver also needs a larger difference to separate copies vertically (Supp. S11).

4.2. Image inversion

Recovered positions preserve image content. We compare images reconstructed from recovered positions with those reconstructed from the GT positions, using the same descriptors and inverter (Tab. 3, Fig. 5). With our U-Net, the two reconstructions are close (LPIPS 0.10), and 63% of the objects identified in the reference reconstruction are also

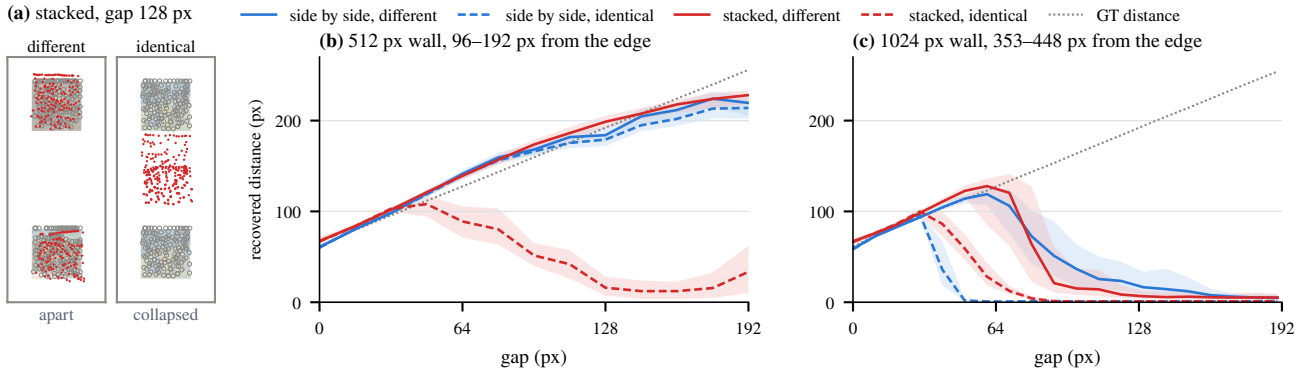


Figure 4. Far from the image border, two paintings stay apart only at small gaps. Two 64 px paintings [35] on a plain wall, 100 pairs per condition. (a) Stacked pairs at a gap of 128 px on the 512 px wall: different paintings stay apart, identical ones collapse; gray circles: GT keypoints, red dots: recovered keypoints. (b,c) Recovered distance against the gap on the 512 px wall, 96–192 px from the nearest image edge (b), and on a 1024 px wall, 353–448 px from it (c), for different (solid) and identical (dashed) paintings, side by side (blue) or stacked (red); dotted gray: GT distance. Medians over pairs with interquartile bands.

Table 3. Image reconstructions from recovered and GT positions are similar. Results on the 200 held-out COCO images [27]. Recovered and random positions are compared against reference reconstructions from GT positions, using the same descriptors and inverter. OR measures the fraction of reference objects recovered. Italic rows compare the reference reconstructions against the original images instead, providing a measure of inversion quality. Other domains: Supp. S7; every metric of the benchmark: Supp. S10.

Inverter	Positions	PSNR ↑	SSIM ↑	LPIPS ↓	OR ↑
U-Net (ours)	recovered	21.4	0.68	0.10	0.63
	random	12.1	0.23	0.60	0.07
	GT vs. original	*18.2*	*0.69*	*0.23*	*0.54*
Diffusion [43]	recovered	16.4	0.50	0.24	0.68
	random	11.6	0.18	0.66	0.07
	GT vs. original	*16.1*	*0.50*	*0.28*	*0.65*

found in the recovered one. Placing the same descriptors at random positions instead reduces object recovery to 7%. Thus, recovering the descriptor-to-position correspondence is essential for reconstructing the scene.

We also evaluate the diffusion inverter of [43]. It recovers a similar fraction of objects (68%), although its reconstructions differ more from the reference reconstruction (LPIPS 0.24). Part of this difference comes from stochastic sampling: two reconstructions from the same positions already differ by LPIPS 0.18. Figures use the deterministic U-Net, avoiding this variation and relying less on a generative prior.

Faces retain their identity. On CelebA-HQ [21], ArcFace [9] detects a face in 198 of 200 reconstructions from recovered positions and identifies 98% of them among the 200 originals, against 100% from GT positions (Fig. 5, Supp. S7). With random positions, no face is detected. The recovered layout thus preserves facial identity.

Text is harder to recover. The solver also recovers the layout of document images without retraining (Tab. 1), but recovering readable text requires greater positional accuracy. We train a separate U-Net on RVL-CDIP [18] documents and

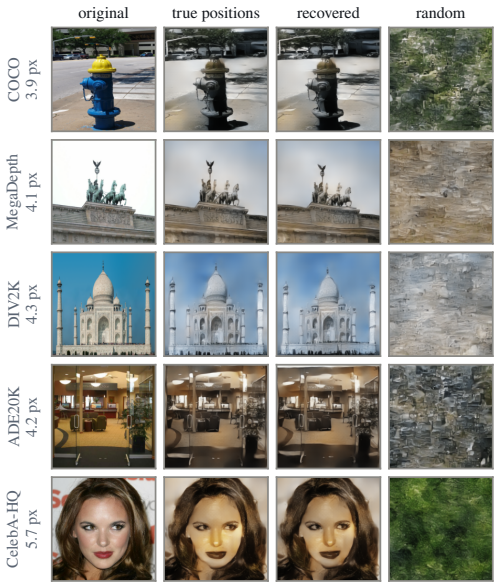


Figure 5. Recovered keypoint positions reveal recognizable image content across domains. Examples from COCO and four unseen domains (Tab. 1), selected near each domain's median residual. **True positions**: reference reconstruction, from the GT keypoint positions. **Recovered**: reconstruction using positions predicted from unordered descriptors. **Random**: reconstruction using the same descriptors at random positions. All three use the same U-Net inverter.

evaluate it on 50 unseen FUNSD [19] forms. We give the full reconstructed form to Qwen2-VL [63] and GPT-5.5 [38] and ask each model to transcribe all visible text, without cropping individual text regions (Tab. 4, Fig. 6). From the reference reconstructions they transcribe about half of the original words. From recovered positions, their recall drops to 15% and 2%, respectively, with low precision. The recovered images preserve the document structure and some larger text, but most words remain unreadable. Even moving keypoints independently by about 2 px already leaves

Table 4. Most text is lost when reconstructing from recovered positions. Qwen2-VL [63] and GPT-5.5 [38] transcribe 50 FUNSD [19] forms, either directly from the original image or from our document-trained U-Net's reconstructions. Recall is the fraction of GT words recovered; precision is the fraction of transcribed words that appear on the form. Medians over forms.

	Qwen2-VL		GPT-5.5	
Image	**recall**	**precision**	**recall**	**precision**
Original form	0.78	0.82	0.90	0.91
GT positions	0.50	0.58	0.51	0.68
Recovered positions	0.15	0.21	0.02	0.20
Random positions	0.02	0.10	0.00	0.00

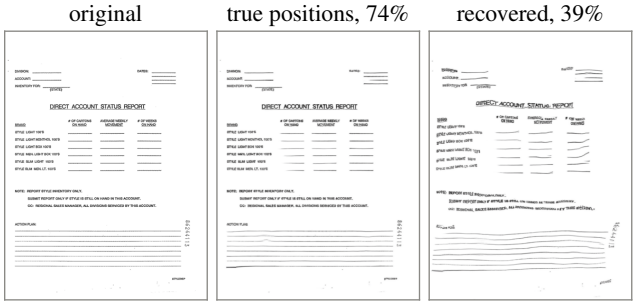


Figure 6. Reconstructing a document form. A FUNSD [19] form and its reconstructions from GT and recovered positions, by a document-trained U-Net using the same descriptors. Percentages: share of the form's words read by Qwen2-VL [63] in each image; medians over all 50 forms: 50% from GT and 15% from recovered positions (Tab. 4).

no word readable (Supp. S13), and the recovered FUNSD layouts have a median residual of 5.9 px.

4.3. SIFT with released scale and orientation

Oriented SIFT [29] computes descriptors in each keypoint's dominant orientation, unlike our upright setting (Sec. 1). Our solver does not reliably recover positions from these descriptors alone. We therefore consider SIFT records released with keypoint scale and orientation, as in OpenCV [3] and COLMAP [50, 51], while positions remain withheld. Further details in Supp. S8.

A small MLP, trained on public images, first estimates each upright descriptor from its oriented descriptor and orientation. The solver then predicts positions from these estimates, with scale as an additional input. An inverter trained on upright SIFT reconstructs images from the estimated descriptors at the recovered positions (Fig. 7). The median position error over test images is 22 px (dcorr 0.82), about five keypoint spacings. Recovery when orientation is withheld remains open.

4.4. Ablating Robustness

We test whether releasing fewer or coarser descriptors reduces layout leakage, and we also measure the associated localization cost. For each modification, we fine-tune an

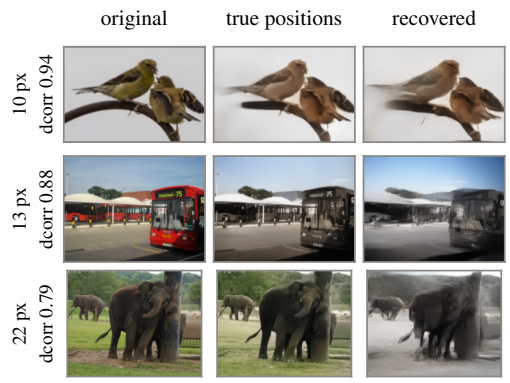


Figure 7. Oriented SIFT with scale and orientation. Two well-recovered scenes and a median one; position error and dcorr on the left. The same inverter uses upright descriptors computed from the image at GT positions, or MLP estimates at recovered positions.

adapted solver on descriptors modified in the same way, and compare it with an *unadapted* solver trained for the same number of epochs on unmodified descriptors (Fig. 8). These steps follow the adaptive-evaluation practice [4, 58]. On COCO, both start from the ALIKED solver of Sec. 4.1 and train for 10 more epochs; on Aachen, both start from the LoMa solver and train for 35 more epochs (Supp. S9).

Subsampling and quantization on COCO. We randomly retain 10–50% of each image's descriptors or quantize their 128 components to 3–8 bits (Fig. 8). Both modifications degrade the unadapted solver, but adaptation reduces the error, most strongly under quantization (Fig. 8). The illustrated reconstructions also become more recognizable after adaptation (Fig. 8). We also test farthest-first spatial selection against matched-count random subsets (Supp. S9).

Localization cost on Aachen. COCO lacks the map and ground-truth poses needed to evaluate localization, so we use Aachen Day–Night [49] (Tab. 5). We modify the 922 query descriptor sets by retaining the highest-scoring keypoints, selecting spatially separated keypoints, or compressing descriptors with optimized product quantization (OPQ) [12, 14, 20]. More details are in Supp. S10.

Reducing the number of keypoints gradually degrades localization (Tab. 5). Fewer keypoints make the reconstruction less recognizable already from GT positions, recovering 57% of the original objects with all keypoints, 27% with 256 and 14% with 64, the level of random positions. Hiding the positions barely changes this: from recovered positions, the reconstructions recover 56%, 25% and 10% and stay close to those from GT positions (LPIPS 0.16–0.20). Releasing fewer keypoints, not hiding their positions, thus reduces what the descriptors reveal; 64 keypoints still localize 71.6% of day queries. Spaced keypoints localize worse without protecting more: 119 keypoints at least 50 px apart localize fewer day queries than the 128 highest-scoring ones (61.0% vs. 78.2%), and their reconstructions recover about as many objects (17% vs. 18%). Quantization keeps the GT-position

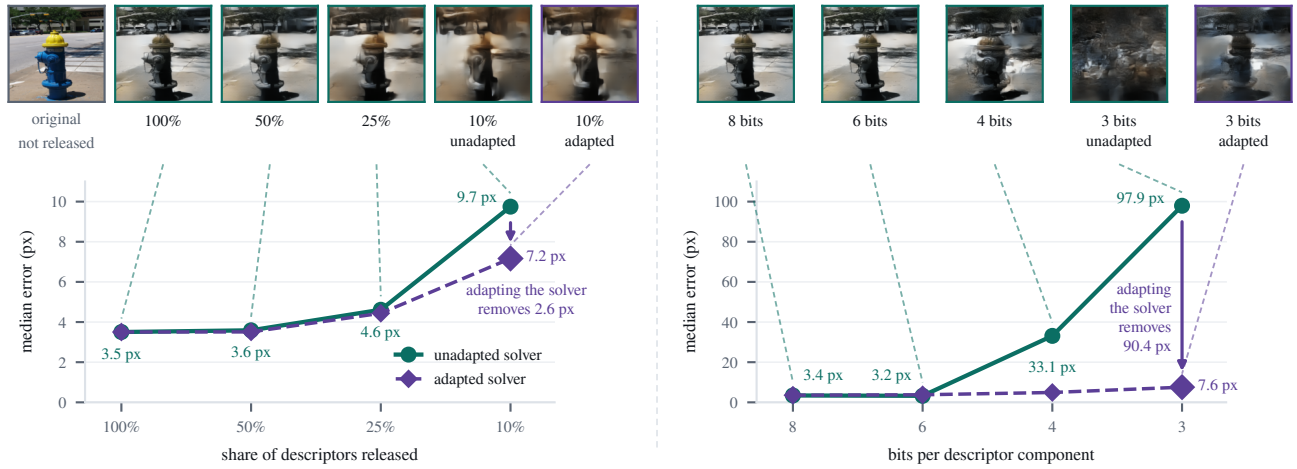


Figure 8. An adapted solver recovers layouts from sparse or quantized descriptors. Results on 200 held-out COCO images, showing median position error. **Left**: a random fraction of each image's descriptors is retained. **Right**: each descriptor component is quantized to 2^b equally spaced values in $[-1, 1]$ using b bits, then the descriptor is renormalized. Unadapted and adapted solvers receive 10 additional training epochs on unmodified and correspondingly modified descriptors, respectively. **Top**: U-Net reconstructions of one test image from recovered positions and the released descriptors. **Bottom**: layout errors for the two solvers.

Table 5. Localization and image recovery from modified query descriptors on Aachen Day–Night [49]. B: bytes per descriptor. N keypoints: the N highest-scoring keypoints. $\geq D$ px (n): keypoints taken in score order, each kept only if at least D px from those already kept; n is the median count. OPQ $m\times b$: m sub-vectors coded with b bits each, so B $= mb/8$. Loc.: queries localized within 25 cm and 2° against a 4-bit map. Error: median position residual (px) of the adapted solver; unadapted errors in Supp. S10. OR vs. orig.: object recovery of the U-Net reconstructions from GT and from recovered (Rec.) positions against the original image. LPIPS vs. GT: between the reconstruction from recovered positions and the one from GT positions; low with few keypoints, where both are mostly empty. Random positions: all 5,000 descriptors at uniformly random positions.

		Loc. (%) ↑		Error	OR vs. orig.		LPIPS
Query release	**B**	**Day**	**Night**	**(px) ↓**	**GT**	**Rec.**	**vs. GT**
5,000 keypoints	1024	88.2	90.8	8.9	0.57	0.56	0.16
1,024 keypoints	1024	87.3	82.7	9.0	0.50	0.47	0.17
512 keypoints	1024	86.0	79.6	10.6	0.38	0.37	0.20
256 keypoints	1024	83.5	76.5	13.1	0.27	0.25	0.20
128 keypoints	1024	78.2	73.5	16.2	0.21	0.18	0.18
64 keypoints	1024	71.6	56.1	21.3	0.14	0.10	0.16
≥50 px (119)	1024	61.0	43.9	13.7	0.16	0.17	0.18
≥100 px (35)	1024	24.1	9.2	25.5	0.00	0.01	0.17
≥200 px (11)	1024	2.2	0.0	50.6	0.00	0.01	0.10
OPQ 16×8	16	82.5	79.6	15.5	0.53	0.42	0.30
OPQ 8×8	8	61.2	37.8	27.7	0.49	0.34	0.43
OPQ 4×8	4	15.4	8.2	76.9	0.48	0.17	0.56
Random positions	1024	–	–	–	0.57	0.14	0.59

reconstructions recognizable (at least 48% of the objects) but makes the layout harder to recover: with 16-byte OPQ codes, localization stays above 79% and the reconstructions from recovered positions recover 42% of the objects; with 4 bytes, only 17%, but localization falls to 15.4%.

Distractor descriptors. We also test whether adding descriptors of unrelated images hides the layout. It does not: the solver flags descriptors drawn from many other images, and descriptors of one to three whole other images raise the error of the image's own keypoints only to 11–21 px (Supp. S9).

5. Conclusion

Withholding keypoint positions is meant to prevent image inversion; however, the upright descriptors we tested still contain enough information to recover them. Because neighboring descriptors describe overlapping image regions, a set transformer trained on public images can recover the keypoint layouts of five extractors from the descriptors alone, with median errors low enough to reconstruct recognizable images, even on image domains it was not trained on. Regarding oriented SIFT, recovery still works when scale and orientation are provided.

Sharing fewer or quantized descriptors does not make discarding the positions much better at preventing image inversion. On Aachen, reconstructions from recovered positions reveal nearly as much as those from GT positions. Quantizing the descriptors does widen this gap, but it also reduces localization accuracy.

Our results show that, for the descriptors studied here, discarding keypoint positions hides neither the keypoint position nor the recognizable image content. Evaluations of geometric obfuscation should compare what reconstructions from recovered and GT positions reveal, alongside localization accuracy.

Acknowledgments

This work was supported by the Czech Science Foundation (GACR) EXPRO (grant no. 23-07973X), the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254). The author(s) used generative AI to assist with editing the manuscript, cleaning up figures and coding. All was reviewed by the authors, who take full responsibility for its content.

References

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building Rome in a day. In *International Conference on Computer Vision (ICCV)*, pages 72–79, 2009.

[2] Eirikur Agustsson and Radu Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. In *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.

[3] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[4] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv:1902.06705*, 2019.

[5] Kunal Chelani, Fredrik Kahl, and Torsten Sattler. How privacy-preserving are line clouds? recovering scene details from 3D lines. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[6] Kunal Chelani, Torsten Sattler, Fredrik Kahl, and Zuzana Kukelova. Privacy-preserving representations are not enough: Recovering scene content from camera poses. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[7] Kunal Chelani, Assia Benbihi, Fredrik Kahl, Torsten Sattler, and Zuzana Kukelova. Obfuscation based privacy preserving representations are recoverable using neighborhood information. In *International Conference on 3D Vision (3DV)*, 2025.

[8] Deeksha Dangwal, Vincent T. Lee, Hyo Jin Kim, Tianwei Shen, Meghan Cowan, Rajvi Shah, Caroline Trippel, Brandon Reagen, Timothy Sherwood, Vassileios Balntas, Armin Alaghi, and Eddy Ilg. Mitigating reverse engineering attacks on local feature descriptors. In *British Machine Vision Conference (BMVC)*, 2021.

[9] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2018.

[11] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[12] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv:2401.08281*, 2024.

[13] Mihai Dusmanu, Johannes L. Schönberger, Sudipta N. Sinha, and Marc Pollefeys. Privacy-preserving image features via adversarial affine subspace embeddings. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[14] Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization for approximate nearest neighbor search. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[15] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L. Schönberger, and Marc Pollefeys. Privacy preserving structure-from-motion. In *European Conference on Computer Vision (ECCV)*, 2020.

[16] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L. Schönberger, and Marc Pollefeys. Privacy preserving localization and mapping from uncalibrated cameras. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[17] Marcel Geppert, Viktor Larsson, Johannes L. Schönberger, and Marc Pollefeys. Privacy preserving partial localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[18] Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2015.

[19] Guillaume Jaume, Hazım Kemal Ekenel, and Jean-Philippe Thiran. FUNSD: A dataset for form understanding in noisy scanned documents. In *International Conference on Document Analysis and Recognition Workshops (ICDARW)*, 2019.

[20] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(1): 117–128, 2011.

[21] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018.

[22] Hiroharu Kato and Tatsuya Harada. Image reconstruction from bag-of-visual-words. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[23] Jeonggon Kim, Heejoon Moon, and Je Hyeong Hong. Revisiting geometric obfuscation with dual convergent lines for privacy-preserving image queries in visual localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[24] Chunghwan Lee, Jaihoon Kim, Chanhyuk Yun, and Je Hyeong Hong. Paired-point lifting for enhanced privacy-preserving visual localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[25] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[26] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence

Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[28] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. LightGlue: Local feature matching at light speed. In *International Conference on Computer Vision (ICCV)*, 2023.

[29] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

[30] Simon Lynen, Bernhard Zeisl, Dror Aiger, Michael Bosse, Joel Hesch, Marc Pollefeys, Roland Siegwart, and Torsten Sattler. Large-scale, real-time visual–inertial localization revisited. *International Journal of Robotics Research (IJRR)*, 39(9):1061–1084, 2020.

[31] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[32] Heejoon Moon, Chunghwan Lee, and Je Hyeong Hong. Efficient privacy-preserving visual localization using 3D ray clouds. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[33] Heejoon Moon, Jeonggon Kim, Sudipta N. Sinha, and Je Hyeong Hong. 2D feature lattices for privacy-preserving image queries in visual localization. In *European Conference on Computer Vision (ECCV) Workshops*, 2026.

[34] Oleksii Nasypanyi, Jaemin Cho, Utku Ozbulak, Byungkon Kang, and Francois Rameau. Seeing through the weights: Privacy leakage in scene coordinate regression. In *European Conference on Computer Vision (ECCV)*, 2026.

[35] National Gallery of Art. Free images and open access. https://www.nga.gov/artworks/free-images-and-open-access. Accessed 2026-11-13.

[36] Tony Ng, Hyo Jin Kim, Vincent T. Lee, Daniel DeTone, Tsun-Yi Yang, Tianwei Shen, Eddy Ilg, Vassileios Balntas, Krystian Mikolajczyk, and Chris Sweeney. NinjaDesc: Content-concealing visual descriptors via adversarial learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[37] David Nordström, Johan Edstedt, Georg Bökman, Jonathan Astermark, Anders Heyden, Viktor Larsson, Mårten Wadenbäck, Michael Felsberg, and Fredrik Kahl. LoMa: Local feature matching revisited. In *European Conference on Computer Vision (ECCV)*, 2026.

[38] OpenAI. GPT-5.5. https://developers.openai.com/api/docs/models/gpt-5.5, 2026. Accessed: 2026-11-13.

[39] Linfei Pan, Johannes L. Schönberger, Viktor Larsson, and Marc Pollefeys. Privacy preserving localization via coordinate permutations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18174–18183, 2023.

[40] Vojtech Panek, Patrik Beliansky, Zuzana Kukelova, and Torsten Sattler. Privacy-preserving structureless visual localization via image obfuscation. In *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2026.

[41] Maxime Pietrantoni, Martin Humenberger, Torsten Sattler, and Gabriela Csurka. SegLoc: Learning segmentation-based representations for privacy-preserving visual localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[42] Maxime Pietrantoni, Gabriela Csurka, and Torsten Sattler. Gaussian splatting feature fields for (privacy-preserving) visual localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[43] Maxime Pietrantoni, Torsten Sattler, and Gabriela Csurka. Vulnerability of privacy-preserving visual localization methods against diffusion-based attacks. In *European Conference on Computer Vision (ECCV)*, pages 492–510, 2026.

[44] Francesco Pittaluga and Bingbing Zhuang. LDP-Feat: Image features with local differential privacy. In *International Conference on Computer Vision (ICCV)*, 2023.

[45] Francesco Pittaluga, Sanjeev J. Koppal, Sing Bing Kang, and Sudipta N. Sinha. Revealing scenes by inverting structure from motion reconstructions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[46] Guilherme Potje, Felipe Cadar, André Araujo, Renato Martins, and Erickson R. Nascimento. XFeat: Accelerated features for lightweight image matching. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[47] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[48] Torsten Sattler and Zuzana Kukelova. Simple but effective triplet-based compression strategies for compact visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29047–29059, 2026.

[49] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[50] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[51] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[52] Maximilian Seitzer, Arash Tavakoli, Dimitrije Antic, and Georg Martius. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. In *International Conference on Learning Representations (ICLR)*, 2022.

[53] Zhenbo Song, Wayne Chen, Dylan Campbell, and Hongdong Li. Deep novel view synthesis from colored 3D point clouds. In *European Conference on Computer Vision (ECCV)*, 2020.

[54] Pablo Speciale, Johannes L. Schönberger, Sing Bing Kang, Sudipta N. Sinha, and Marc Pollefeys. Privacy preserving image-based localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[55] Pablo Speciale, Johannes L. Schönberger, Sudipta N. Sinha, and Marc Pollefeys. Privacy preserving image queries for

camera localization. In *International Conference on Computer Vision (ICCV)*, 2019.

[56] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[57] Warren S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

[58] Florian Tramèr, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Conference on Neural Information Processing Systems (NIPS)*, 2020.

[59] Michał J. Tyszkiewicz, Pascal Fua, and Eduard Trulls. DISK: Learning local features with policy gradient. In *Conference on Neural Information Processing Systems (NIPS)*, 2020.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Conference on Neural Information Processing Systems (NIPS)*, 2017.

[61] Carl Vondrick, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. HOGgles: Visualizing object detection features. In *International Conference on Computer Vision (ICCV)*, 2013.

[62] Francesco Vultaggio, Predrag Djindjic, Markus Gerke, Sebastian Tschiatschek, and Phillipp Fanta-Jende. Misanthrope: A Privacy-Preserving Keypoint Detector. In *ECCV Workshops*, 2026.

[63] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv:2409.12191*, 2024.

[64] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph CNN for learning on point clouds. *Transactions on Graphics (TOG)*, 38(5), 2019.

[65] Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. Reconstructing an image from its local descriptors. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[66] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. In *Conference on Neural Information Processing Systems (NIPS)*, 2017.

[67] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[68] Xiaoming Zhao, Xingming Wu, Weihai Chen, Peter C. Y. Chen, Qingsong Xu, and Zhengguo Li. ALIKED: A lighter keypoint and descriptor extraction network via deformable transformation. *IEEE Transactions on Instrumentation and Measurement (TIM)*, 72:1–16, 2023.

[69] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

DescOver: Descriptor Overlap Leaks the Image

Supplementary Material

This supplement gives the details and further results behind the paper: the implementation (Sec. S1), the datasets (Sec. S2) and feature extractors (Sec. S3), the measurements behind the mechanism (Sec. S4), the reference reconstructions and metrics (Secs. S5 and S6), recovery and inversion across domains (Sec. S7), SIFT with released scale and orientation (Sec. S8), the defenses on COCO (Sec. S9) and on Aachen Day–Night (Sec. S10), the two paintings (Sec. S11), documents (Sec. S12) and layout precision (Sec. S13). Section, table and figure numbers without an S refer to the paper. We will make our code and trained models publicly available.

S1. Implementation details

Solver. The solver f_θ (Fig. 3) is a set transformer [25, 60, 66]. A linear layer turns each descriptor into a token, self-attention layers combine the tokens, and a shared head predicts one 2D position per token. The network has no positional encoding, so permuting its input permutes its output. Table S1 lists the training settings.

	Solver	**Inverter**
Network	set transformer, pre-LN	U-Net
Size	width 384, 8 layers, 8 heads	base width 64, depth 4
Input	K=768 descriptors	Gaussians (σ=2) on a 256×256 grid
Loss	Eq. 1, λ=1	L1 + 0.1 × VGG perceptual
Optimizer	Adam, lr 5×10^{-4}	Adam, lr 2×10^{-4}
Epochs, batch	80, 8	250, 16
Training data	COCO train2017 (117k)	COCO, GT positions

Table S1. Training settings. Both networks hold out 15% of their training images for validation. The DISK and LoMa solvers add a 2-epoch linear learning-rate warmup and gradient clipping at 1.0.

Loss. The loss of Eq. 1 combines a Huber term on the centered position error $\boldsymbol{\Delta}$ with a Gaussian negative log-likelihood for the uncertainty head, which predicts a log variance per axis. The likelihood sees $\boldsymbol{\Delta}$ only through the stop-gradient sg, which detaches it. This way the uncertainty head learns how large the error is without rescaling the position gradients, which can hinder training [52]. Both heads share the transformer backbone and are trained together with this one loss.

Extractors. ALIKED uses the `aliked-n16` model with the authors' weights and detection threshold 0.2 (Sec. 4), run through LightGlue [28] with its default preprocessing. ALIKED and LoMa (the LoMa-B model) see the image resized to 512 px on the longer side. XFeat, SuperPoint and DISK follow the extraction recipe of the inversion benchmark [43] and see a 512×512 square. Positions are measured in the frame each extractor sees. On COCO, every extractor keeps its 2048 highest-scoring keypoints.

Inverter. As in Sec. 3.2, each descriptor is placed on a feature grid as a small Gaussian centered at its position, and a U-Net [47] decodes that grid into an image. It is trained on ground-truth (GT) positions and applied to recovered ones, so its output reflects both its own quality and the error of the recovered layout.

Solvers of Table 2. All learned solvers are trained with our settings on the same images and with the same seed, without tuning per architecture. The GNN runs EdgeConv [64] on a graph that links each descriptor to its nearest neighbors in descriptor space, so it never sees a position. Classical MDS [57] and Isomap [56] embed the cosine distances between the descriptors, Isomap through geodesics on the same kind of graph. Their output has an arbitrary rotation, reflection and scale, so we align it to the ground truth with a similarity transform before measuring the residual. This alignment pulls any uninformative layout toward the centroid, so even a random layout gets a moderate residual (142 px). For these two methods we report dcorr and kNN evaluation values.

S2. Datasets

Training. The solvers and our U-Net are trained on public COCO train2017 images [27] (Table S1). The document U-Net is trained on RVL-CDIP [18] (Sec. S12). On Aachen, the solvers are adapted on the database images (Sec. S10).

Testing. Unless stated otherwise, results are on the 200 held-out COCO test images. Table S2 lists the other test sets, which we process the same way. Localization is evaluated on Aachen Day–Night v1.0 [49], with its 824 day and 98 night queries (Sec. S10).

Set	**Images**	**Which images**	**Keypoints**
COCO	200	held-out test images	1617
MegaDepth	806	both MegaDepth-1500 test scenes	1177
DIV2K	300	training images 0001–0300	1943
ADE20K	500	validation images 1–500	1162
CelebA-HQ	200	first 200 training faces	1528
FUNSD	50	all test forms	1300

Table S2. Test sets. Keypoints: median ALIKED keypoints per image. Subsets are the first images in each dataset's order, not selected by content.

S3. Feature extractors

We study five learned extractors with upright descriptors: ALIKED [68], XFeat [46], SuperPoint [10], DISK [59] and

LoMa [37]. Their image regions are aligned with the image axes, and they omit explicit per-keypoint scale normalization, so neighboring descriptors describe overlapping regions in a common frame. Their descriptors have 128 (ALIKED, DISK), 64 (XFeat) or 256 (SuperPoint, LoMa) dimensions. ALIKED is our main extractor, and Table 1 evaluates all five. SIFT [29] instead rotates and rescales each region to its keypoint's orientation and scale, and is treated on its own (Sec. 4.3 and Sec. S8). Sec. S1 gives the extraction settings, and Sec. S4 measures how far an ALIKED descriptor reaches.

S4. Mechanism

Measurements of Fig. 2. For the influence map in the middle, the probe photograph stays fixed on a white background, and we paste a small textured block at each point of a grid around it. For every block position we recompute the probe's descriptors at fixed keypoints and record how much they change ($1 - \cos$). The map averages three keypoints of the probe, at its left edge, center and right edge, over five probe photographs. The two circles mark the radii r_{50} and r_{90} that enclose half and 90% of a keypoint's descriptor change (31 and 84 px, medians over keypoints). For the curve on the right, each trial takes the median change over the probe's keypoints and the eight neighbor directions, and the line and band show the median and interquartile range over ten trials. The horizontal axis is the grid spacing, so diagonal neighbors are farther away.

Upright and oriented SIFT at the same keypoints. To isolate the orientation convention, we compute SIFT [29] with one large, fixed support at the ALIKED keypoints of each image and train the solver twice, once on upright and once on oriented descriptors, with nothing else changed. The solver places the upright descriptors to 12 px but fails on the oriented ones (141 px). The frame alone removes most of what the solver recovers. Native SIFT, whose support grows with the keypoint's scale, is studied in Sec. S8.

Connectivity. Two distant regions of an image can be linked through the keypoints between them (Sec. 4.1). Fig. S1 measures how soon the ALIKED keypoints of a test image form one connected graph when every pair closer than a threshold is linked. We use all detected keypoints, or a random sample of them as the solver sees them. A typical sampled set is connected at about 50 px, and the largest component holds most keypoints much earlier. The graph uses positions only. It shows that chains between distant regions exist in real images, and Fig. 2 shows that neighboring descriptors share information along them.

S5. Reference reconstructions

Image scores need a reference, and we use two. Tables 3 and S3 compare the reconstructions from recovered and from

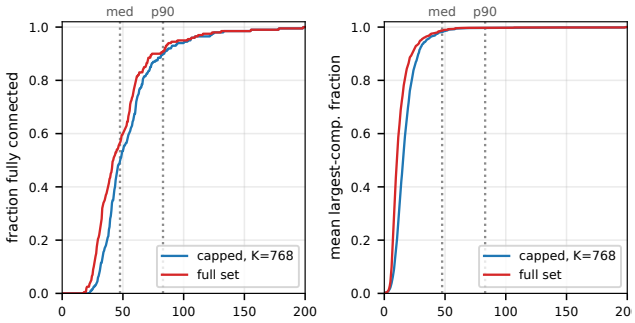


Figure S1. Keypoints of real images connect at small distances. ALIKED keypoints of the 200 test images, linked when closer than the threshold. Left: fraction of images whose keypoints form one component. Right: mean fraction of keypoints in the largest component. Blue: 768 sampled keypoints, as the solver sees them; red: all detected keypoints. Dotted: median and 90th percentile of the threshold at which a sampled set becomes connected (47.6 and 83.0 px).

random positions with the *reference reconstruction*, made from the ground-truth (GT) positions with the same descriptors and the same inverter (Sec. 4). The two reconstructions differ only in the positions, so these scores show what hiding the positions protects, separately from what the inverter can reconstruct at all. The italic rows of both tables give that second part: they compare the reference with the original image, so they depend only on the inverter.

All other image scores compare each reconstruction with the original image and give the reconstructions from GT and from recovered positions side by side: Table 5 and Tables S5, S8, S9 and S13. These experiments change how many descriptors are released or how precise their positions are, so a reconstruction from GT positions would not be a fixed reference. With few descriptors it loses most objects itself, and two poor reconstructions can agree on the same generic objects (Sec. S10). Faces are identified among the original images (Table S4).

S6. Metrics

Layouts. Each metric is computed per image, and we report its median over the test images (Sec. 4). The residual is the median distance between recovered and GT positions after aligning the layouts, by translation for upright descriptors and by rotation and translation for oriented ones. The spacing, the median distance from a GT keypoint to its nearest neighbor, puts the residual in context. dcorr is the Pearson correlation between recovered and GT distances over all keypoint pairs. kNN is the fraction of each keypoint's five nearest GT neighbors that are also among its five nearest recovered neighbors, averaged over keypoints. Both are 1 for a perfect layout and close to 0 for random positions. The MDS and Isomap baselines are aligned by a similarity transform instead (Sec. S1).

Images. PSNR and SSIM are computed on color images

and LPIPS [67] on grayscale images. Sec. S5 says which tables compare with the reconstruction from GT positions and which with the original image.

Object recovery. OR follows the benchmark of [43]. Qwen2-VL [63] lists up to ten objects in each image, leaving out walls, floors, sky, doors and trees. OR is the fraction of the objects listed for the reference image, the reconstruction from GT positions or the original image, that are also listed for the reconstruction, matched one to one by the similarity of their MiniLM sentence embeddings. Two more prompts of the benchmark give texture recall (TR) and scene similarity (SR), reported in Table S9. Decoding is greedy, so the scores are deterministic.

Faces and text. On faces, ArcFace [9] identifies each reconstruction among the original images (Table S4). On documents, word recall is the share of a form's words that appear in a model's transcription of the reconstruction (Sec. S12).

S7. Recovery and inversion across domains

Sec. 4.1 recovers keypoint positions on COCO and on five domains not seen in training (Table 1), and Sec. 4.2 reconstructs images from them (Table 3, Fig. 5). This section shows more layouts and reconstructions for the four image domains. Forms are in Sec. S12.

What the image scores compare. In Tables 3 and S3, each test image is reconstructed from the same descriptors with the same inverter, once at the GT positions and once at the recovered positions. PSNR, SSIM, LPIPS and object recovery (OR) compare the reconstruction from recovered positions, or from random positions, with the reconstruction from GT positions, not with the original image. An OR of 1 thus means that hiding the positions removed no object the inverter could otherwise show. The italic rows (GT vs. original) compare the reconstruction from GT positions with the original image and measure the inverter itself (Sec. S5).

Single images. Fig. S2 shows recovered layouts from the best to the worst of 120 test images. Even in the worst case, the arrangement of the scene holds.

Other domains. Fig. S3 overlays recovered and GT layouts for landmarks and faces, two of the domains of Table 1. Table S3 repeats Table 3 on the first 100 images of each domain, with the solver and the U-Net trained on COCO only. As on COCO, the reconstruction from recovered positions stays close to the one from GT positions, and random positions recover few objects. Fig. 5 shows, for each domain, an image within two ranks of the median residual, and Figs. S4 and S5 show more.

Diffusion inverter. For the diffusion rows of Table 3, we fine-tune the conditional diffusion inverter of [43], released for SIFT, SuperPoint and XFeat, on upright ALIKED for 50k steps.

Face identification. We run ArcFace [9] (SCRFD detector, w600k_r50 embedding) on every CelebA-HQ reconstruction and identify it among the 200 originals (Table S4). From recovered positions, almost every reconstruction contains a detected face, and almost all of these are identified correctly, close to the result from GT positions. From random positions, no face is detected. Some people appear more than once in CelebA-HQ, which can only lower these scores.

Domain	Positions	PSNR ↑	SSIM ↑	LPIPS ↓	OR ↑
MegaDepth	recovered	21.2	0.66	0.11	0.64
	random	12.7	0.19	0.60	0.20
	GT vs. original	*17.5*	*0.75*	*0.19*	*0.58*
DIV2K	recovered	20.5	0.60	0.11	0.62
	random	12.1	0.19	0.59	0.08
	GT vs. original	*17.4*	*0.67*	*0.24*	*0.51*
ADE20K	recovered	21.8	0.70	0.11	0.69
	random	12.0	0.24	0.60	0.09
	GT vs. original	*17.1*	*0.70*	*0.23*	*0.55*
CelebA-HQ	recovered	22.1	0.73	0.09	0.65
	random	10.8	0.24	0.62	0.08
	GT vs. original	*16.6*	*0.69*	*0.22*	*0.52*

Table S3. Table 3 on the other domains. Our U-Net and the ALIKED solver, both trained on COCO only, on the first 100 images of each domain. As in Table 3, recovered and random positions are scored against the reconstruction from GT positions, and the italic rows score that reference against the original image.

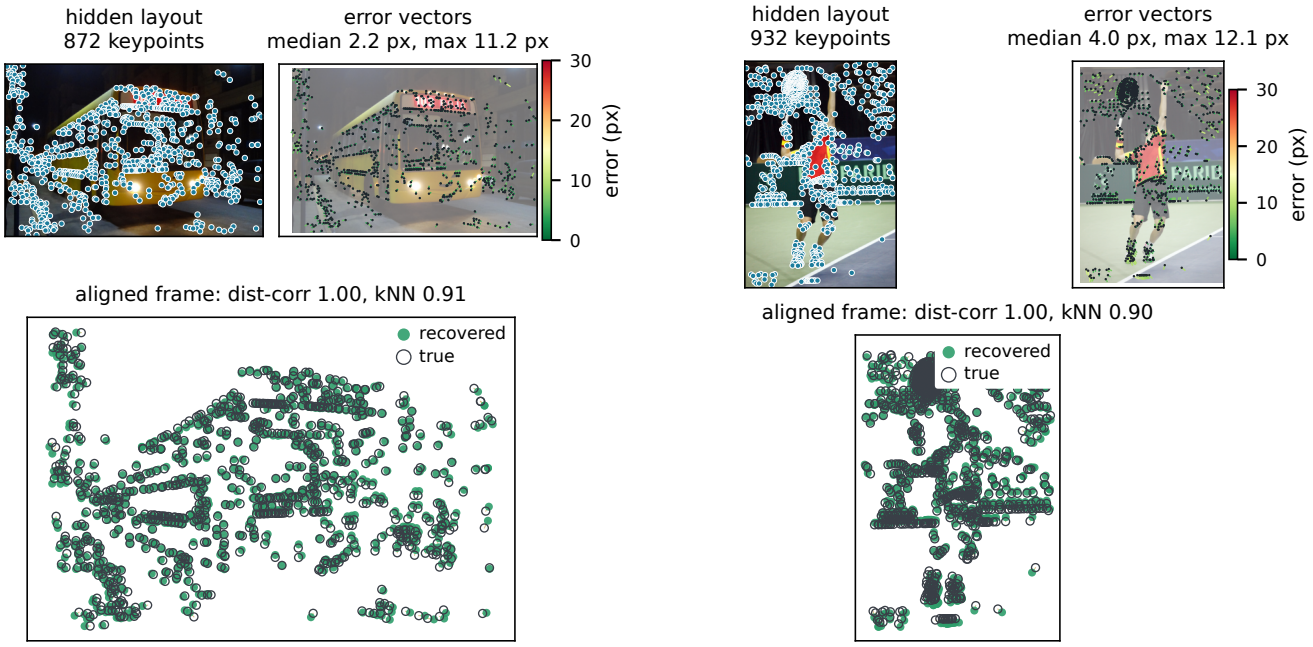
Positions	Det. ↑	Gen. cos ↑	AUC ↑	Rank-1 ↑	TPR@10^{-3} ↑
GT	1.00	0.55	1.000	1.00	0.995
Recovered	0.99	0.46	0.999	0.98	0.93
Random	0.00	–	–	–	–

Table S4. Face identification on CelebA-HQ. ArcFace [9] on 200 reconstructions, identified among the 200 originals. Det.: fraction with a detected face; gen. cos: mean cosine similarity to the original (to other people: 0.01); TPR at a false-accept rate of 10^{-3}.

S8. SIFT with released scale and orientation

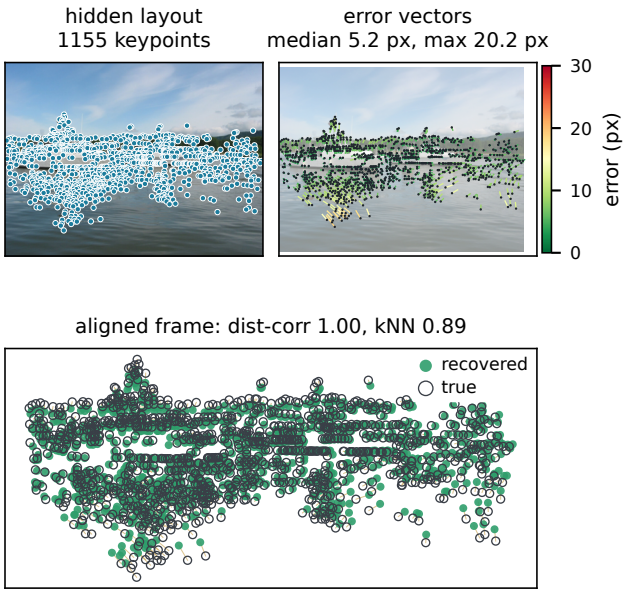
Release. Sec. 4.3 attacks SIFT records that keep each keypoint's scale σ and orientation θ, as OpenCV [3] keypoint files and COLMAP [50, 51] feature files store them. Each descriptor is computed in the frame rotated by θ, and the attacker gets everything but the positions. Keypoints are OpenCV's difference-of-Gaussians detections with a low contrast threshold, on images resized to 512 px on the longer side, capped at the number of ALIKED keypoints of the same image. The solver is the one of Sec. S1, trained on COCO images disjoint from the test images.

Undoing the orientation. The released angle tells the attacker in which frame each descriptor was computed. A small MLP maps the oriented descriptor and $(\cos\theta, \sin\theta)$ to the upright descriptor at the same keypoint. It is trained with a cosine loss on pairs computed from the attacker's own images. A fixed rearrangement of the descriptor would not work, because SIFT interpolates between its spatial and orientation bins. On the test keypoints, the MLP's output is close to the true upright descriptor (median cosine 0.99,

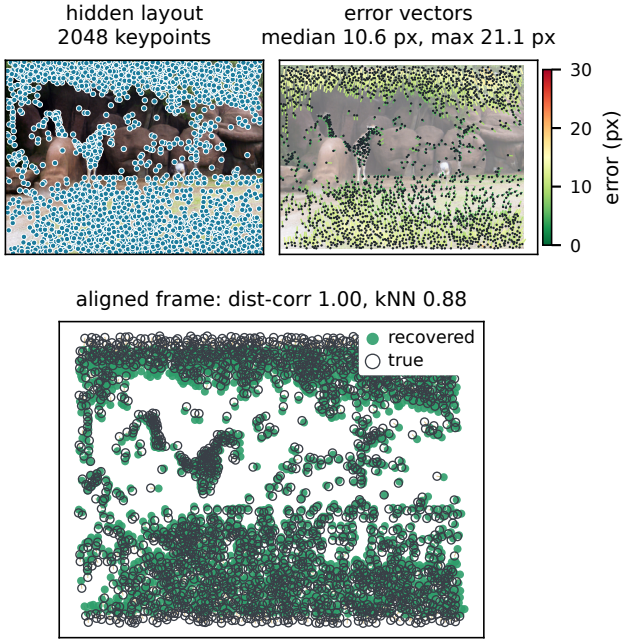


(a) best: 2.2 px

(b) median: 4.0 px



(c) 75th pct.: 5.2 px



(d) worst of 120 scanned: 10.6 px

Figure S2. Recovered layouts across the error distribution. GT keypoints in gray, recovered in green, after aligning the translation: the best, median, 75th-percentile and worst of 120 test images, with their median residuals.

against 0.62 for the released one).

Scale. We append $\log \sigma$ to each descriptor as one more input dimension, standardized and scaled to the spread of a RootSIFT entry.

Results. Fig. S6 shows that the attack improves with more training data. With 100k training images it reaches the 22 px of Sec. 4.3, and the curves have not yet flattened. The oracle, with descriptors computed upright from the image, which the attacker does not have, does somewhat better, so undoing the orientation costs a few pixels. Without the scale, the solver recovers nothing, from the released and from the upright descriptors alike. Fig. S7 shows reconstructions by a U-Net



Figure S3. Recovered layouts on landmarks and faces. Recovered (green) over GT (gray) keypoints, with the solver trained on COCO. Landmarks (MegaDepth, top) are recovered as precisely as COCO; faces (CelebA-HQ, bottom) less precisely, but their arrangement holds.



Figure S4. Faces from descriptors alone. CelebA-HQ, with the solver and the U-Net trained on COCO: original (top), reconstruction from GT positions (middle) and from recovered positions (bottom).

trained with our recipe on upright native-scale SIFT at GT positions.

S9. Defenses on COCO

Sec. 4.4 asks whether a client can protect the layout by changing the descriptors it sends, for example by sending fewer of them or quantizing them, while they stay useful for localization. This section gives the details of those experiments and tests two more modifications: releasing only keypoints far apart (Sec. S9.1) and hiding the descriptors among unrelated ones (Sec. S9.2).

Adapted and unadapted solvers. A modification protects the layout only if it holds against an attacker who knows it. Such an attacker can apply the same modification to its own public training images and fine-tune the solver on them. We call this solver *adapted*, following the practice of adaptive evaluation [4, 58]. The *unadapted* solver is fine-tuned for the same number of epochs on unmodified descriptors and then applied to modified ones. It shows what the modification does to an attacker who ignores it. On COCO, both solvers start from the ALIKED solver of Table 1 and train for 10 more epochs at a lower learning rate. The image panels of Fig. 8 show one test image near the median, inverted by the U-Net trained on unmodified descriptors. On Aachen (Sec. S10), the unadapted and the adapted query solvers start from the LoMa solver of Table 1 and train for 35 more epochs on the database images, released unmodified or modified like the queries. They are tested on the queries.

		Spaced		Random		OR, spaced ↑	
d (px)	Kept	px ↓	dcorr ↑	px ↓	dcorr ↑	GT	Rec.
–	1617	3.5	0.999	–	–	0.56	0.52
8	601	3.4	0.999	3.8	0.999	0.49	0.46
16	240	4.3	0.999	5.7	0.998	0.28	0.26
32	84	7.6	0.996	10.1	0.992	0.09	0.08
64	29	15.2	0.983	19.6	0.966	0.03	0.03
96	16	20.3	0.970	28.1	0.944	0.04	0.03
128	10	21.9	0.973	32.2	0.926	0.03	0.05

Table S5. Spaced releases of ALIKED descriptors on COCO. d: minimum distance between released keypoints (512 px frame); kept: median number of keypoints. Residual and dcorr of the solver adapted to each release, for the spaced release and for a random one of the same size. OR: U-Net at GT and at recovered positions (random positions: 0.05). The first row releases all keypoints. Images left with fewer than 8 keypoints are skipped (23 at d=128 px).

S9.1. Farthest-first spatial selection

If descriptors leak through the image regions they share, releasing only keypoints far from each other might help. How far apart is far depends on the extractor. We use ALIKED (Sec. S3), for which half of the change in a descriptor comes from content within $r_{50} = 31$ px of its keypoint and 90% from content within $r_{90} = 84$ px (Sec. S4). Even keypoints 128 px apart, the largest spacing we test, therefore still describe overlapping regions. We select keypoints farthest-first from a random start, each at least d from those already kept, and compare with a random subset of the same size (Table S5). Each release gets its own solver, adapted as above, and the U-Net inverts the first 100 test images.

Spreading the keypoints does not protect the layout. At every distance, the spaced release is recovered at least as well as the random one, and even the sparsest releases are placed near their GT positions (Fig. S8). Sparse releases lose content instead. With fewer than about 100 descriptors, the U-Net no longer draws recognizable objects, even at the GT positions.

S9.2. Distractor descriptors

A client could also hide its descriptors among those of other images. We add descriptors of other test images, either



Figure S5. Landmarks from descriptors alone. MegaDepth, with the solver and the U-Net trained on COCO: original (top), reconstruction from GT positions (middle) and from recovered positions (bottom).

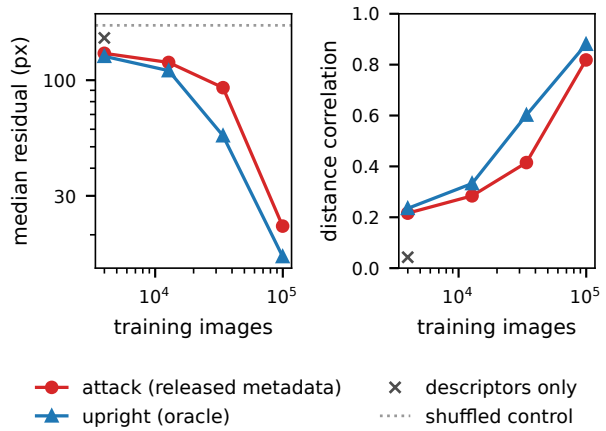


Figure S6. The SIFT attack improves with training data. Median residual (left) and dcorr (right) against the number of training images. Red: the attack, with the released descriptors de-rotated by the MLP and the released scale. Blue, the oracle: descriptors computed upright from the image, which the attacker does not have, with the released scale. It is the best case, a lower bound on the residual (left) and an upper bound on dcorr (right). Crosses: the released descriptors without scale; dotted: the solver given the descriptors in random order.

drawn from many images or taken from whole images, which form coherent scenes, up to ten times as many as the image's own (Table S6). The ALIKED solver runs on all descriptors together, without retraining, and we score the image's own keypoints. The solver also predicts a variance per keypoint, so an attacker who knows how many descriptors were added can drop that many of the most uncertain ones and run the solver again.

Descriptors from many images. These distractors disturb the solver, but its variance separates them from the image's own descriptors. After filtering, the error of the image's own keypoints is close to that of the clean release, as long as no more descriptors are added than the image has.

Whole scenes. Whole scenes cannot be separated this way, but they disturb the solver less. One to three added scenes raise the error only to 11 to 21 px, and even ten added scenes leave it around 30 px.

	From many images				From whole images			
Added	**px ↓**	**AUC ↑**	**Filt. ↓**	**Kept**	**px ↓**	**AUC ↑**	**Filt. ↓**	**Kept**
–	4.0	–	–	–	4.0	–	–	–
10%	5.1	0.97	3.9	0.98	5.2	0.94	4.4	0.96
25%	16.0	0.97	3.8	0.96	7.6	0.77	7.4	0.87
50%	34.7	0.96	4.5	0.93	10.7	0.63	10.1	0.73
100%	43.7	0.96	6.1	0.88	11.1	0.57	10.9	0.59
200%	47.8	0.95	8.1	0.84	15.8	0.54	13.8	0.41
300%	50.6	0.95	9.6	0.81	20.6	0.51	15.7	0.30
500%	50.9	0.94	13.4	0.77	24.6	0.52	16.6	0.21
1000%	50.6	0.92	26.3	0.65	29.8	0.53	16.9	0.12

Table S6. Distractor descriptors added to a release. Added: descriptors of other test images, as a share of the image's own count. px: median residual of the image's own keypoints for the ALIKED solver on all descriptors together. AUC: how well the solver's variance separates added from own descriptors. Filt. and kept: residual after dropping as many of the most uncertain descriptors as were added and running the solver again, and the share of own keypoints left. Images left with fewer than 8 keypoints count as failures (3, 6 and 23 of 200 at 300, 500 and 1000% from whole images).

Comparison with LDP-Feat. LDP-Feat [44] also hides each descriptor among unrelated ones, but unlike our distractors it changes the descriptor itself. It replaces the descriptor by its nearest word in a public dictionary, built by k-means clustering of descriptors from other images, and sends m words per keypoint. The keypoint's own word is among them only with a small probability, and otherwise all m words are random. We apply LDP-Feat to the COCO test images with its default privacy level, $\varepsilon = 10$, and a dictionary of 256k words built from the ALIKED descriptors of other COCO images. Only 8 to 26% of the keypoints then keep their own word ($m = 1$ to 4).

The two defenses behave differently. Distractors leave the image's own descriptors unchanged, so the ALIKED solver still recovers their positions (Table S6). Under LDP-Feat, the solver does not recover the positions, not even for the keypoints that kept their own word, and its variance does not single out the true words (Table S7). The solver also fails when every keypoint sends only its own word, so the random words are not what stops it. An attacker who knows the defense does not get much further. We fine-



Figure S7. Reconstructions from the SIFT attack. Test images at the 25th, 50th and 75th percentile of the residual (100k training images); row labels give residual and dcorr. Columns: original; U-Net at GT positions with upright descriptors; U-Net at the recovered positions with the de-rotated descriptors.

		Adapted to	
Release	**Unadapted**	**own words**	**LDP-Feat, m=2**
Own words	164	97	157
LDP-Feat, m=1	198	147	173
LDP-Feat, m=2	190	137	166
LDP-Feat, m=4	175	133	167

Table S7. LDP-Feat hides the layout, also from adapted solvers. Median distance (px) between the keypoints that sent their own word and their GT positions, after aligning the translation, on the 200 COCO test images ($\varepsilon = 10$, 256k words). The adapted solvers are fine-tuned as in Sec. S9, on 13,800 training images. Without LDP-Feat the solver reaches 3.9 px; placing every keypoint at the image center gives 182 px and random positions 231 px.

tune the solver as in Sec. S9, once on own words and once on LDP-Feat's word sets. On own words it then recovers only a coarse layout, and on the word sets the keypoints that kept their own word stay far from their positions with either adapted solver. Replacing the descriptors by dictionary words protects the layout, also against an adapted solver, while hiding unchanged descriptors among others does not. The random words, however, cost localization. On Aachen Day–Night with SIFT, LDP-Feat with $m = 2$ localizes 42% of the day queries, against 77% when each descriptor is only replaced by its own word [44].

S10. Defenses on Aachen Day–Night

Table 5 weighs what query releases cost in localization against what the attack still recovers, on Aachen Day–Night [49]. Here we give the setup and all results.

Setup. We use LoMa [37] features at 800 px on the longer side, with up to 5,000 keypoints per query and 512 per database image, on Aachen Day–Night v1.0. The map stores the database descriptors with 4 bits per value. For the localization results, the query descriptors are matched to the database by nearest neighbor search using the ratio test from [48], which does not use keypoint positions. Following [49], we report the percentage of query images localized within 25 cm and 2° of the ground truth pose. The server owns the database images, so the attacker adapts a solver on their features and tests it on the queries (Sec. S9). We invert 100 queries per release, with the diffusion inverter of [43] fine-tuned for LoMa and with our U-Net retrained for LoMa on COCO images.

Releases. A client can send fewer keypoints, spatially separated keypoints, or quantized descriptors. LoMa keeps its highest-scoring keypoints, so a release of N keypoints is a subset of the full one. A spaced release takes keypoints in score order and keeps each one only if it is at least D px from those already kept. A quantized release sends all keypoints through an OPQ codec trained on the database descriptors. Table S8 reports, for each release, the unadapted query solver

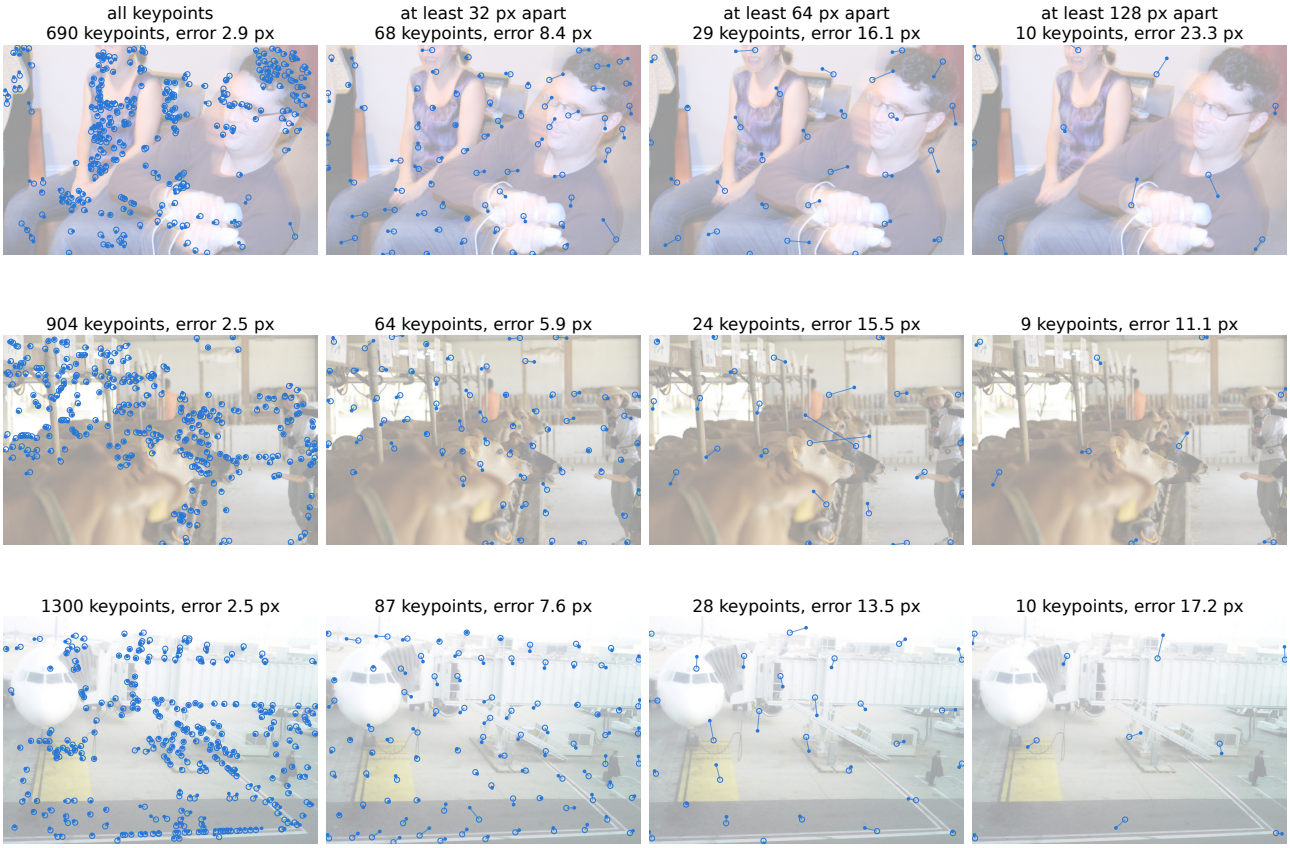


Figure S8. Spaced releases on three test images. Each column is attacked by the solver adapted to that release. Circles: GT keypoints; dots: recovered keypoints; lines: errors. Titles give the number of keypoints and the median residual; at most 300 keypoints are drawn.

		Loc. (%) ↑	Error (px) ↓ / dcorr ↑		OR, GT / attack ↑	
Query release	**B**	**day / night**	**unadapted**	**adapted**	**diffusion**	**U-Net**
5,000 keypoints	1024	88.2 / 90.8	8.9 / 0.998	8.9 / 0.998	0.63 / 0.64	0.57 / 0.56
1,024 keypoints	1024	87.3 / 82.7	10.6 / 0.997	9.0 / 0.998	0.62 / 0.58	0.50 / 0.47
512 keypoints	1024	86.0 / 79.6	14.0 / 0.995	10.6 / 0.997	0.51 / 0.51	0.38 / 0.37
256 keypoints	1024	83.5 / 76.5	21.9 / 0.989	13.1 / 0.994	0.41 / 0.41	0.27 / 0.25
128 keypoints	1024	78.2 / 73.5	41.7 / 0.966	16.2 / 0.990	0.31 / 0.29	0.21 / 0.18
64 keypoints	1024	71.6 / 56.1	71.1 / 0.909	21.3 / 0.983	0.22 / 0.19	0.14 / 0.10
≥50 px (119)	1024	61.0 / 43.9	18.5 / 0.991	13.7 / 0.996	0.12 / 0.14	0.16 / 0.17
≥100 px (35)	1024	24.1 / 9.2	72.9 / 0.831	25.5 / 0.981	0.11 / 0.08	0.00 / 0.01
≥200 px (11)	1024	2.2 / 0.0	140.0 / 0.550	50.6 / 0.917	0.13 / 0.12	0.00 / 0.01
OPQ 16×8	16	82.5 / 79.6	26.7 / 0.936	15.5 / 0.985	0.56 / 0.53	0.53 / 0.42
OPQ 8×8	8	61.2 / 37.8	86.6 / 0.642	27.7 / 0.893	0.54 / 0.48	0.49 / 0.34
OPQ 4×8	4	15.4 / 8.2	159.9 / 0.390	76.9 / 0.660	0.50 / 0.39	0.48 / 0.17

Table S8. Aachen query releases. All settings of Table 5, with the unadapted query solver next to the adapted one and with both inverters. B: bytes per descriptor (for 256 dimensional LoMa descriptors). Loc.: queries localized within 25 cm and 2°, matched without positions against a 4-bit map. Error: median residual over the 922 queries (800 px frame). OR: object recovery on 100 queries from GT positions and from the adapted solver's layout (random positions: 0.06 with the diffusion inverter, 0.14 with the U-Net). Top: the N highest-scoring keypoints. Middle: keypoints at least D px apart (n: median count). Bottom: all keypoints through an OPQ codec.

and the one adapted to the same release of the database. As the release shrinks or is quantized, the unadapted solver degrades quickly, while the adapted one is able to better recover the layout of the keypoints. At the same time, localization accuracy degrades as well, limiting the effectiveness of the different release strategies in terms of preventing the recovery of the keypoint position layout.

What the reconstructions show. Figs. S9 to S11 show four queries under each release. With all keypoints, the reconstructions from recovered and from GT positions recover

		Our U-Net						Diffusion inverter [43]					
Data / release	**Positions**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**OR ↑**	**TR ↑**	**SR ↑**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**	**OR ↑**	**TR ↑**	**SR ↑**
COCO	GT	0.69	0.22	179	0.54	0.21	0.78	0.49	0.28	120	0.68	0.59	0.80
	recovered	0.50	0.25	194	0.54	0.20	0.77	0.42	0.31	124	0.65	0.57	0.81
	random	0.16	0.62	342	0.06	0.00	0.42	0.15	0.66	364	0.05	0.01	0.38
5,000 keypoints	GT	0.60	0.38	119	0.57	0.20	0.77	0.49	0.29	94	0.63	0.52	0.79
	recovered	0.43	0.44	146	0.56	0.18	0.76	0.37	0.34	96	0.64	0.56	0.78
	random	0.22	0.70	358	0.14	0.01	0.50	0.09	0.68	369	0.06	0.01	0.46
1,024 keypoints	GT	0.52	0.46	170	0.50	0.06	0.73	0.42	0.34	93	0.62	0.51	0.77
	recovered	0.40	0.50	198	0.47	0.03	0.72	0.34	0.39	95	0.58	0.52	0.77
512 keypoints	GT	0.45	0.55	267	0.38	0.04	0.68	0.37	0.43	107	0.51	0.47	0.74
	recovered	0.38	0.59	287	0.37	0.02	0.66	0.32	0.47	111	0.51	0.45	0.74
256 keypoints	GT	0.39	0.67	319	0.27	0.03	0.60	0.32	0.57	132	0.41	0.32	0.69
	recovered	0.35	0.69	334	0.25	0.02	0.58	0.28	0.60	134	0.41	0.30	0.67
128 keypoints	GT	0.35	0.76	331	0.21	0.01	0.54	0.26	0.68	183	0.31	0.12	0.60
	recovered	0.33	0.78	337	0.18	0.00	0.52	0.25	0.70	197	0.29	0.09	0.57
64 keypoints	GT	0.33	0.81	329	0.14	0.00	0.50	0.23	0.72	247	0.22	0.05	0.52
	recovered	0.32	0.82	337	0.10	0.00	0.48	0.23	0.73	257	0.19	0.04	0.52
OPQ 16×8	GT	0.56	0.40	131	0.53	0.18	0.75	0.40	0.35	116	0.56	0.48	0.76
	recovered	0.36	0.52	195	0.42	0.03	0.69	0.27	0.44	120	0.53	0.47	0.75
OPQ 8×8	GT	0.54	0.42	135	0.49	0.19	0.73	0.36	0.39	118	0.54	0.49	0.76
	recovered	0.32	0.60	238	0.34	0.01	0.63	0.22	0.55	135	0.48	0.35	0.73
OPQ 4×8	GT	0.52	0.44	137	0.48	0.17	0.71	0.33	0.43	119	0.50	0.42	0.73
	recovered	0.31	0.68	296	0.17	0.00	0.52	0.23	0.67	197	0.39	0.11	0.65

Table S9. All metrics of the benchmark of [43], computed with its code: SSIM, grayscale LPIPS, FID, and the Qwen2-VL scores OR, TR and SR (Sec. S6), on 100 images per row. COCO: ALIKED with the unadapted solver of Sec. S9. Aachen: the LoMa query releases of Table S8. GT and recovered: the released descriptors at the GT positions and at those of the solver adapted to the release; random: all descriptors at random positions. Every reconstruction is compared with the original image, at its inverter's output size. FID on 100 images is noisy.

about the same share of objects with both inverters. Fewer keypoints reduce what is recovered already at the GT positions, and with the fewest neither inverter shows the original scene. The diffusion inverter then invents a plausible one, while the U-Net draws the released descriptors in place and leaves the rest empty. The spaced releases show the same contrast. With the two sparsest, the U-Net recovers no object, while the diffusion inverter, which completes a street scene from little evidence, still scores slightly above its random-position level. Quantization keeps most of the content at the GT positions, but the layout becomes harder to recover, and the attack loses content with it. Table S9 reports every metric of the benchmark of [43] for these releases and for COCO.

Object recovery against the GT reconstruction. Table 5 scores object recovery against the original image. Scored against the reconstruction from GT positions instead, as in Table 3, the attack keeps 69% of its objects with all keypoints and 23% with the strongest quantization, against 15% for random positions. For sparse releases this score is misleading, which is why Table 5 does not use it. Both reconstructions then show few real objects, and Qwen2-VL describes them with the same generic items ("milk with bubbles", "spheres"). The two sparsest spaced releases therefore score above 0.4, although neither reconstruction contains an object of the original image.

		Different		Identical	
Gap (px)	**GT**	**Side**	**Stacked**	**Side**	**Stacked**
0	62	59	67	58	66
20	83	83	86	84	87
40	104	104	111	36	86
60	123	119	128	1	28
80	144	72	64	1	4
100	164	37	15	1	1
120	184	23	8	1	1
160	224	8	5	1	1

Table S10. Two paintings far from the image border (Fig. 4c). Median GT and recovered distance between the paintings' keypoint centroids (px) against the gap between their edges, on a 1024×1024 wall, 100 pairs per condition.

S11. Two paintings on a plain wall

We give the measurements behind the painting experiment of Fig. 4 and test its explanation in Sec. 4.1.

Far from the border. Table S10 lists the numbers behind Fig. 4c. The paintings hang at the center of a 1024×1024 wall, far from every edge, and the features are extracted at twice the wall's size, as for the 512 px wall, so the paintings keep their size in pixels. At small gaps every pair keeps its distance. Different paintings keep it up to a gap of about

	Side by side		Stacked	
Gap (px)	Both	Alone	Both	Alone
0	2	53	5	57
32	3	55	7	88
64	14	31	13	110
96	12	20	15	92
128	16	20	12	44
160	21	29	14	24
192	43	44	32	34

Table S11. Each painting described without the other. Median error of the recovered offset between two different paintings on the 512 px wall (100 pairs), with descriptors computed in the scene with both paintings (both) or in an image without the other painting, at the same keypoints (alone).

50 px and nearly coincide from about 100 px, while identical copies merge earlier. A larger wall gives the same result. On the 512 px wall the image border helps. There, side-by-side copies stay apart up to 160 px, and stacked copies only near the top edge.

Why stacked copies merge. Two identical copies differ only in where they lie in the image, and ALIKED's descriptors change with that position, a little more horizontally than vertically. At the same gap, stacked copies are therefore more alike than side-by-side ones, as the paper reports. To see how much difference the solver needs, we keep the keypoints that have an exact counterpart on the other copy and scale the difference between the two copies' descriptors. Side-by-side copies differ by 0.018 and stay apart at that difference. Stacked copies differ by 0.013 and need 0.030, almost twice as much. Stacked copies thus both differ less and need a larger difference to stay apart. Most COCO images are wider than tall, which may explain the second effect, although we have not tested it.

Each painting described without the other. If nearby paintings are placed through what their descriptors see of each other, taking that view away should break the placement. We keep the keypoints of the scene with both paintings on the 512 px wall, but compute each painting's descriptors in an image without the other one (Table S11). At small gaps the error jumps, and touching paintings are recovered almost on top of each other. At large gaps the difference is small, and the image border keeps the paintings apart.

Plain walls are outside the training distribution, and the 1024 px wall is larger than any training image. Even there, close pairs are recovered at their true distance.

S12. Documents

Sec. 4.2 reads text from reconstructed FUNSD forms [19]. The ALIKED solver recovers their layout without retraining (Table 1), but Qwen2-VL reads almost no words in the reconstructions of our COCO U-Net and of the diffusion inverter, even from GT positions and even after fine-tuning the diffusion inverter on documents. We therefore train a document U-Net with the recipe of Sec. S1 on RVL-CDIP pages [18], after removing the FUNSD forms and their near-duplicates. It uses a finer grid than the COCO U-Net, since pages resampled to the coarser grid already lose about a fifth of their readable words.

Positions	Error (px) ↓	Words ↑	Chance
Original page	–	0.78	0.10
GT	0	0.50	0.08
Recovered	5.9	0.15	0.05
Random	207.1	0.02	0.02

Table S12. Words read by the document U-Net on the 50 FUNSD test forms. Error: median layout residual; words: median word recall; chance: recall against another form's transcription.

Moved independently			Moved together		
Error (px)	OR ↑	Words ↑	Error (px)	OR ↑	Words ↑
1.2	0.52	0.09	2.4	0.54	0.49
2.3	0.51	0.00	4.8	0.52	0.41
4.7	0.38	0.00	9.6	0.48	0.29
9.4	0.26	0.01	19.2	0.40	0.10
18.8	0.16	0.03	38.4	0.23	0.01
37.6	0.07	0.03	76.9	0.13	0.02

Table S13. Content against position error. Error: median distance of the moved keypoints from their GT positions (px, 512 px frame; the FUNSD errors differ from these by less than 1 px). Moved together: white noise smoothed with a Gaussian of 46 px. OR on 100 COCO test images; words: median word recall on the 50 FUNSD forms. GT positions give OR 0.56 and words 0.50, random positions 0.05 and 0.02. The attack on FUNSD reaches 5.9 px and words 0.15.

Measuring text. Qwen2-VL [63] transcribes each image, and word recall is the share of the form's distinct words of at least three letters or digits that appear in the transcription. Scoring a form's words against the transcription of another form gives the chance level, which is above zero because forms share words such as "date".

Result. In the document U-Net's reconstructions, Qwen2-VL reads half of the words at GT positions and 15% at recovered ones (Table S12). The page structure, the rules and large print survive (Fig. 6), but small print needs a more precise layout than the attack provides (Sec. S13).

S13. Layout precision

To find how much position error the content tolerates, we move the GT positions on purpose and invert the moved layout. Either each keypoint moves independently by a Gaussian offset, or nearby keypoints move together along a smooth random displacement field. The second setting is closer to the attack, whose errors are strongly correlated between nearby keypoints. The error of a moved layout is measured like the attack's. Objects are scored on 100 COCO test images with our U-Net, words on the 50 FUNSD forms with the document U-Net.

Result. Objects tolerate a few pixels of either kind of er-

ror (Table S13). Text does not tolerate independent errors. Moving each keypoint by about 2 px already leaves no word readable, while moving neighbors together by twice as much keeps many of the words. Some words survive the attack only because its neighboring keypoints move together.

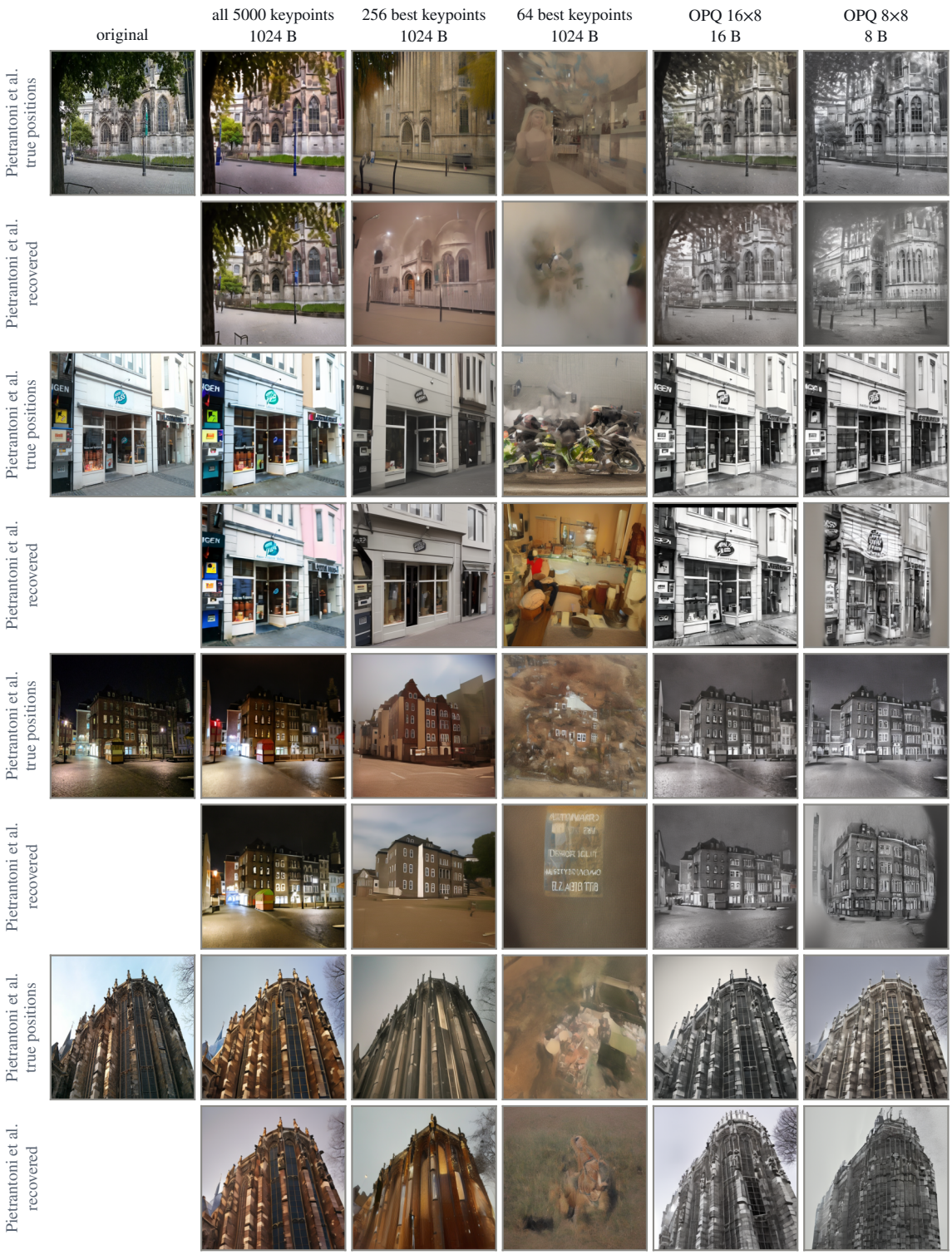


Figure S9. Aachen queries reconstructed by the diffusion inverter. Four queries drawn at random, under each release. For each query, top: GT positions with the released descriptors, the setting of [43]; bottom: the positions that the solver adapted to the release recovers. The diffusion inverter can add content that the descriptors do not carry; Fig. S10 shows the same queries with our U-Net.

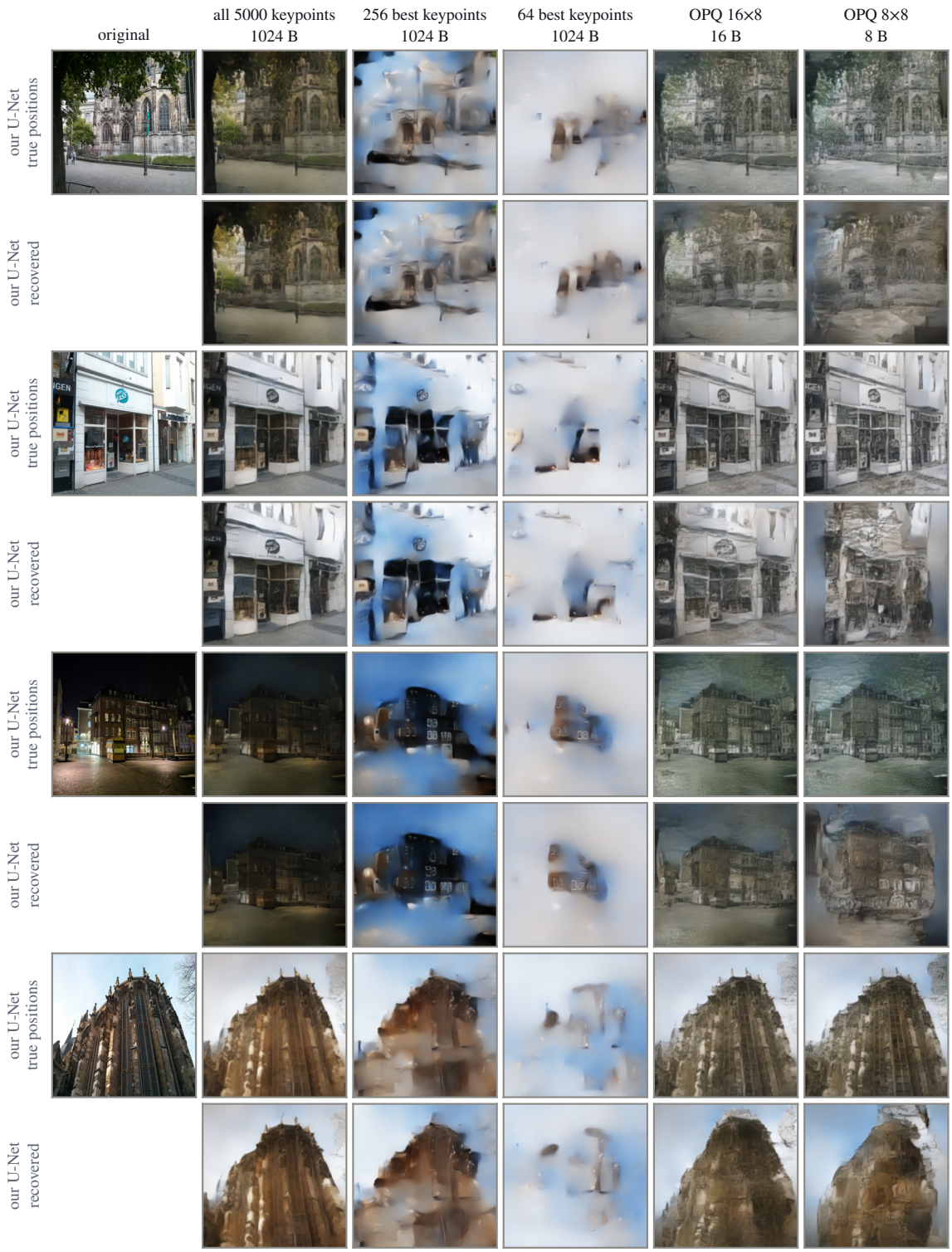


Figure S10. The queries of Fig. S9 reconstructed by our U-Net, retrained for LoMa. For each query, top: GT positions; bottom: recovered positions. The U-Net renders at 256×256.

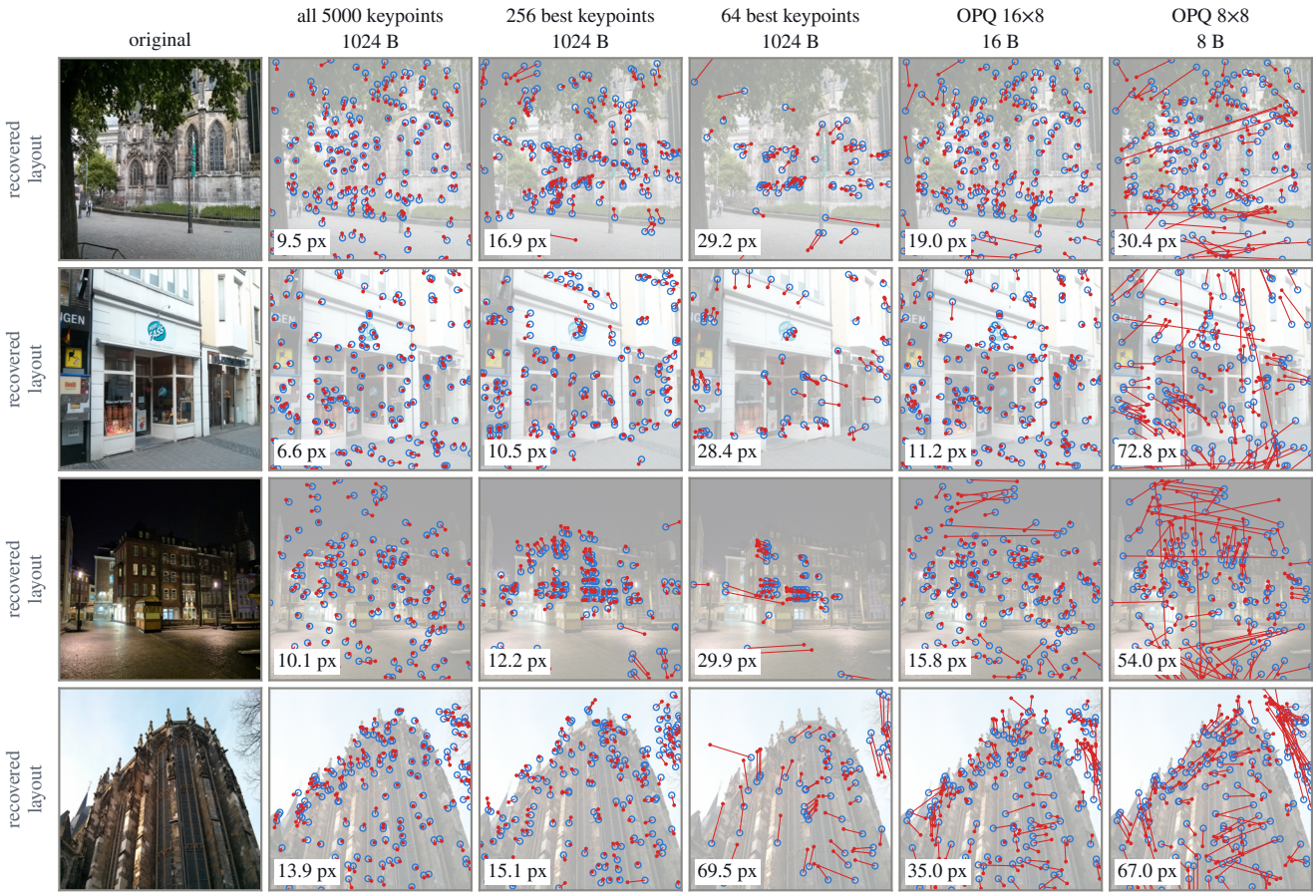


Figure S11. Recovered layouts of the queries of Fig. S9. GT (circles) and recovered (dots) positions of up to 150 keypoints, joined by their errors; the number is the median residual of the release (800 px frame).