NinjaDesc: Content-Concealing Visual Descriptors via Adversarial Learning

Tony Ng[1,2]   Hyo Jin Kim[1†]   Vincent T. Lee[1]   Daniel DeTone[1]   Tsun-Yi Yang[1]

Tianwei Shen[1]   Eddy Ilg[1]   Vassileios Balntas[1]   Krystian Mikolajczyk[2]   Chris Sweeney[1]

[1]Reality Labs, Meta   [2]Imperial College London

Abstract

In the light of recent analyses on privacy-concerning scene revelation from visual descriptors, we develop descriptors that conceal the input image content. In particular, we propose an adversarial learning framework for training visual descriptors that prevent image reconstruction, while maintaining the matching accuracy. We let a feature encoding network and image reconstruction network compete with each other, such that the feature encoder tries to impede the image reconstruction with its generated descriptors, while the reconstructor tries to recover the input image from the descriptors. The experimental results demonstrate that the visual descriptors obtained with our method significantly deteriorate the image reconstruction quality with minimal impact on correspondence matching and camera localization performance.

1. Introduction

Local visual descriptors [7, 13, 56, 72, 74] are fundamental to a wide range of computer vision applications such as SLAM [15, 40, 42, 45], SfM [1, 64, 71], wide-baseline stereo [30, 43], calibration [49], tracking [24, 44, 51], image retrieval [3, 4, 32, 46, 47, 66, 77, 78], and camera pose estimation [5, 17, 54, 61, 62, 75, 76]. These descriptors represent local regions of images and are used to establish local correspondences between and across images and 3D models.

The descriptors take the form of vectors in high-dimensional space, and thus are not directly interpretable by humans. However, researchers have shown that it is possible to reveal the input images from local visual descriptors [10, 16, 80]. With the recent advances in deep learning, the quality of the reconstructed image content has been significantly improved [11, 53]. This poses potential privacy concerns for visual descriptors if they are used for sensitive data without proper encryption [11, 69, 80].

To prevent the reconstruction of the image content from visual descriptors, several methods have been proposed. These methods include obfuscating keypoint locations by

Figure 1. Our proposed content-concealing visual descriptor. **a)** We train ***NinjaNet***, the content-concealing network via adversarial learning to give ***NinjaDesc***. **b)** On the two examples shown, we compare inversions on SOSNet [74] descriptors *vs*. NinjaDesc (encoding SOSNet with NinjaNet). **c)** NinjaDesc is able to conceal facial features and landmark structures, while retaining correspondences. *Image credits: laylamoran4battersea & sgerner (Flickr)[1].*

lifting them to lines that pass through the original points [21, 65, 69, 70], or to affine subspaces with augmented adversarial feature samples [18] to increase the difficulty of recovering the original images. However, recent work [9] has demonstrated that the closest points between lines can yield a good approximation to the original points' locations.

In this work, we explore whether such local feature inversion could be mitigated at the descriptor level. Ideally, we want a descriptor that does not reveal the image content without a compromise in its performance. This may seem counter-intuitive due to the trade-off between utility and privacy discussed in the recent analysis on visual descriptors [11], where the utility is defined as matching accuracy, and the privacy is defined as non-invertibility of the descriptors. The analysis showed that the more useful the descriptors are for correspondence matching, the easier it is to invert them. To minimize this trade-off, we propose an

---

†Corresponding author.

adversarial approach to train visual descriptors.

Specifically, we optimize our descriptor encoding network with an adversarial loss for descriptor invertibility, in addition to the traditional metric learning loss for feature correspondence matching. For the adversarial loss, we jointly train an image reconstruction network to compete with the descriptor network in revealing the original image content from the descriptors. In this way, the descriptor network learns to hinder the reconstruction network by generating visual descriptors that conceal the image content, while being optimized for correspondence matching.

In particular, we introduce an auxiliary encoder network *NinjaNet* that can be trained with any existing visual descriptors and transform them to our content-concealing *NinjaDesc*, as illustrated in Fig. 1. In the experiments, we show that visual descriptors trained with our adversarial learning framework lead to only marginal drop in performance for feature matching and visual localization tasks, while significantly reducing the visual similarity of the reconstruction to the original input image.

One of the main benefits of our method is that we can control the trade-off between utility and privacy by changing a single parameter in the loss function. In addition, our method generalizes to different types of visual descriptors, and different image reconstruction network architectures.

In summary, our main innovations are as follows: **a)** We propose a novel adversarial learning framework for visual descriptors to prevent reconstructing original input image content from the descriptors. We experimentally validate that the obtained descriptors significantly deteriorate the image quality from descriptor inversion with only marginal drop in matching accuracy using standard benchmarks for matching (HPatches [6]) and visual localization (Aachen Day-Night [63,84]). **b)** We empirically demonstrate that we can effectively control the trade-off between utility (matching accuracy) and privacy (non-invertibility) by changing a single training parameter. **c)** We provide ablation studies by using different types of visual descriptors, image reconstruction network architectures and scene categories to demonstrate the generalizability of our method.

2. Related work

This section discusses prior work on visual descriptor inversion and the state-of-the-art descriptor designs that attempt to prevent such inversion.

Inversion of visual descriptors. Early results of reconstructing images from local descriptors was shown by Weinzaepfel *et al*. [80] by stitching the image patches from a known database with the closest distance to the input SIFT [37] descriptors in the feature space. d'Angelo *et al*. [10] used a deconvolution approach on local binary descriptors such as BRIEF [8] and FREAK [2]. Vondrick *et*

al. [79] used paired dictionary learning to invert HoG [85] features to reveal its limitations for object detection. For global descriptors, Kato and Harada [31] reconstructed images from Bag-of-Words descriptors [68]. However, the quality of reconstructions by these early works were not sufficient to raise concerns about privacy or security.

Subsequent work introduced methods that steadily improved the quality of the reconstructions. Mahendran and Vedaldi [39] used a back-propagation technique with a natural image prior to invert CNN features as well as SIFT [36] and HOG [85]. Dosovitskiy and Brox [16] trained up-convolutional networks that estimate the input image from features in a regression fashion, and demonstrated superior results on both classical [37, 48, 85] and CNN [34] features. In the recent work, descriptor inversion methods have started to leverage larger and more advanced CNN models as well as employ advanced optimization techniques. Pittaluga *et al*. [53] and Dangwal *et al*. [11] demonstrated sufficiently high reconstruction qualities, revealing not only semantic information but also details in the original images.

Preventing descriptor inversion for privacy. Descriptor inversion raises privacy concerns [11,53,69,80]. For example, in computer vision systems where the visual descriptors are transferred between the device and the server, an honest-but-curious server may exploit the descriptors sent by the client device. In particular, many large-scale localization systems adopt cloud computing and storage, due to limited compute on mobile devices. Homomorphic encryption [19,60,83] can protect descriptors, but are too computationally expensive for large-scale applications.

Proposed by Speciale *et al*. [69], the line-cloud representation obfuscate 2D/3D point locations in the map building process [20,21,65] without compromising the accuracy in localization. However, since the descriptors are unchanged, Chelani *et al*. [9] showed that line-clouds are vulnerable to inversion attacks if the underlying point-cloud is recovered.

Adversarial learning has been applied in image encoding [27,52,81] that optimizes privacy-utility trade-off, but not in the context of descriptor inversion, which involves reconstruction of images from a set of local features and has a much broader scope of downstream applications.

Recently, Dusmanu *et al*. [18] proposed a privacy-preserving visual descriptor via lifting descriptors to affine subspaces, which conceals the visual content from inversion attacks. However, this comes with a significant cost on the descriptor's utility in downstream tasks. Our work differs from [18] in that we propose a learned content-concealing descriptor and explicitly train it for utility retention to achieve a better trade-off between the two.

3. Method

We propose an adversarial learning framework for obtaining content-concealing visual descriptors, by introduc-
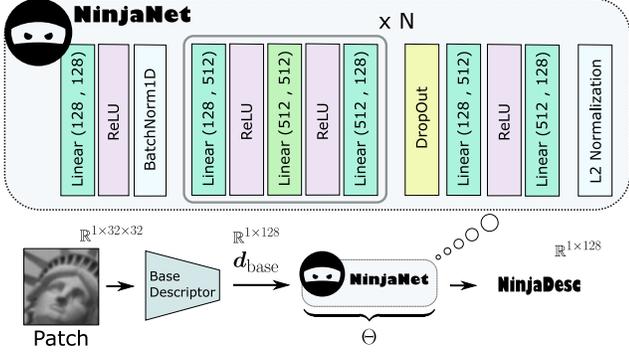
Figure 2. **Top:** Architecture of our content-concealing *NinjaNet* encoder Θ. **Bottom:** A base descriptor with dimensionality C is transformed to *NinjaDesc* of the same size, *e.g.* $C = 128$.

ing a descriptor inversion model as an adversary. In this section, we detail our content-concealing encoder NinjaNet (Sec. 3.1) and the descriptor inversion model (Sec. 3.2), as well as the joint adversarial training procedure (Sec. 3.3).

3.1. NinjaNet: the content-concealing encoder

In order to conceal the visual content of a local descriptor while maintaining its utility, we need a trainable encoder which transforms the original descriptor space to a different one, where visual information essential for reconstruction is reduced. Our NinjaNet encoder Θ is implemented by an MLP shown in Fig. 2. It takes a base descriptor d_{base}, and transforms it into a content-concealing NinjaDesc, d_{ninja}:

$$d_{\text{ninja}} = \Theta(d_{\text{base}}) \tag{1}$$

The design of NinjaNet is light-weight and plug-and-play, to make it flexible in accepting different types of existing local descriptors. The encoded NinjaDesc descriptor maintains the matching performance of the original descriptor, but prevents from high-quality reconstruction of images. In many of our experiments, we adopt SOSNet [74] as our base descriptor since it is one of the top-performing descriptors for correspondence matching and visual localization [30].

Utility initialization. To maintain the utility (*i.e.* accuracy for downstream tasks) of our encoded descriptor, we use a patch-based descriptor training approach [41, 73, 74]. The initialization step trains NinjaNet via a triplet-based ranking loss. We use the UBC dataset [22] which contains three subsets of patches labelled as positive and negative pairs, allowing for easy implementation of triplet-loss training.

Utility loss. We extract the base descriptors d_{base} from image patches x_{patch} and train NinjaNet (Θ) with the descriptor learning loss from [74] to optimize NinjaDesc (d_{ninja}).

$$\mathcal{L}_{util}(x_{\text{patch}}; \Theta) = \mathcal{L}_{triplet}(d_{\text{ninja}}) + \mathcal{L}_{reg.}(d_{\text{ninja}}), \tag{2}$$

where $\mathcal{L}_{reg.}(\cdot)$ is the second-order similarity regularization term [74]. We always freeze the weights of the base descriptor network, including the joint training process in Sec. 3.3.

3.2. Descriptor inversion model

For our proposed adversarial learning framework, we utilize a descriptor inversion network as the adversary to reconstruct the input images from our NinjaDesc. We adopt the UNet-based [58] inversion network from prior work [11, 53]. Following Dangwal *et al.* [11], the inversion model Φ takes as input the sparse feature map $\mathbf{F}_{\Theta} \in \mathbb{R}^{H \times W \times C}$ composed from the descriptors and their keypoints, and predicts the RGB image $\hat{\mathbf{I}} \in \mathbb{R}^{h \times w \times 3}$, *i.e.* $\hat{\mathbf{I}} = \Phi(\mathbf{F}_{\Theta})$. We denote $(H, W), (h, w)$ as the resolutions of the sparse feature image and the reconstructed RGB image, respectively. C is the dimensionality of the descriptor. The detailed architecture is provided in the supplementary.

Reconstruction loss. The descriptor inversion model Φ is optimized under a reconstruction loss which is composed of two parts. The first loss is the mean absolute error (MAE) between the predicted $\hat{\mathbf{I}}$ and input $\mathbf{I}$ images,

$$\mathcal{L}_{mae} = \sum_{i}^{h} \sum_{j}^{w} ||\hat{\mathbf{I}}_{i,j} - \mathbf{I}_{i,j}||_1. \tag{3}$$

The second loss is the perceptual loss, which is the L2 distance between intermediate features of a VGG16 [67] network pretrained on ImageNet [12],

$$\mathcal{L}_{perc} = \sum_{k=1}^{3} \sum_{i}^{h_k} \sum_{j}^{w_k} ||\psi_{k,i,j}^{VGG}(\hat{\mathbf{I}}) - \psi_{k,i,j}^{VGG}(\mathbf{I})||_2^2, \tag{4}$$

where $\psi_k^{VGG}(\mathbf{I})$ are the feature maps extracted at layers $k \in \{2, 9, 16\}$, and (h_k, w_k) is the corresponding resolution.

The reconstruction loss is the sum of the two terms

$$\mathcal{L}_{recon}(x_{\text{image}}; \Phi) = \mathcal{L}_{mae} + \mathcal{L}_{perc}. \tag{5}$$

where x_{image} denote the image data term that includes both the descriptor feature map $\mathbf{F}_{\Theta}$ and the RGB image $\mathbf{I}$.

Reconstruction initialization. For the joint adversarial training described in Sec. 3.3, we initialize the the inversion model using the initialized NinjaDesc in Sec. 3.1, This part is done using the MegaDepth [35] dataset, which contains images of landmarks across the world. For the keypoint detection we use the Harris corners [25] in our experiments.

3.3. Joint adversarial training

The central component of engineering our content-concealing NinjaDesc is the joint adversarial training step, which is illustrated in Fig. 3 and elaborated as pseudo-code in Algorithm 1. We aim to minimize trade-off between utility and privacy, which are the two competing objectives. Inspired by methods using adversarial learning [23, 59, 82], we formulate the optimization of utility and privacy trade-off as an adversarial learning process. The objective of the

A. Utility (Matchability)

Patches

Base Descriptor → NinjaNet → NinjaDesc
Θ

Dataset: UBC Patches

$\mathcal{L}_{util} = \mathcal{L}_{triplet} + \mathcal{L}_{reg.}$

B. Reconstruction

Input RGB Image → Detect Keypoints & Descriptors → Sparse Feature Map from NinjaNet → Descriptor Inversion Model Φ → Predicted RGB Image

Dataset: MegaDepth

$\mathcal{L}_{recon} = \mathcal{L}_{mae} + \mathcal{L}_{perc}$

Joint Adversarial Training

1. $\mathcal{L}_{util} - \lambda\mathcal{L}_{recon}$

weights update: NinjaNet Θ

2. $\mathcal{L}_{recon}$

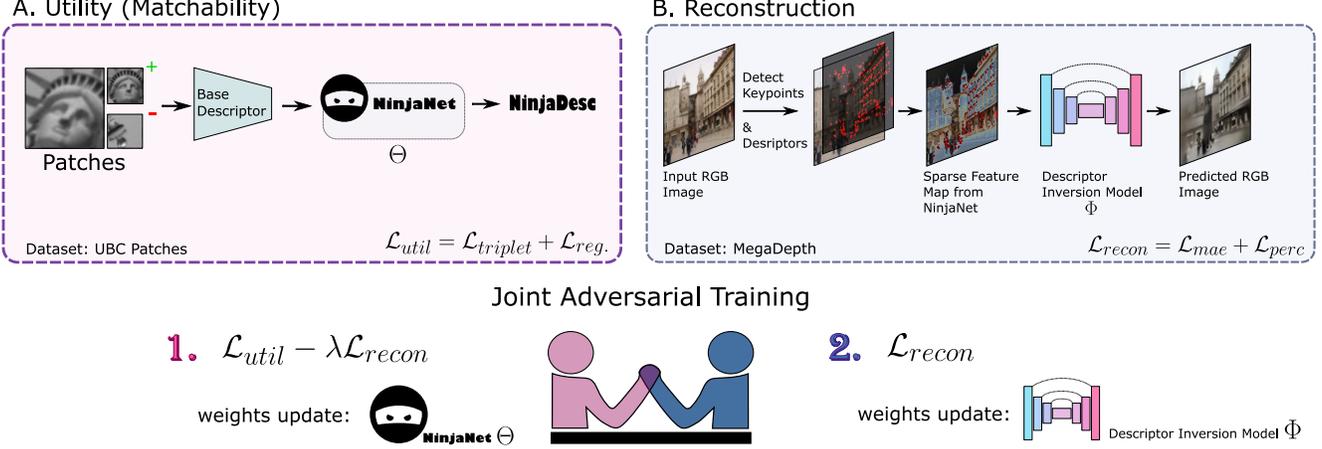weights update: Descriptor Inversion Model Φ

Figure 3. The pipeline for training our content-concealing NinjaDesc. **Top:** The two networks at play and their corresponding objectives are: **1.** NinjaNet Θ, which is for utility retention in A; and **2.** the descriptor inversion model, which reconstructs RGB images from input sparse features in B. **Bottom:** During joint adversarial training, we alternate between steps **1.** and **2.**, which is presented by Algorithm 1.

Algorithm 1 Pseudo-code for the joint adversarial training process of NinjaDesc

---

1: NinjaNet: $\Theta_0 \leftarrow$ initialize with Eqn. 2
2: Desc. inversion model: $\Phi_0 \leftarrow$ initialize with Eqn. 5
3: $\lambda \leftarrow$ set privacy parameter
4: **for** $i \leftarrow 1$, number of iterations **do**
5:     **if** $i = 0$ **then**
6:         $\Theta \leftarrow \Theta_0, \Phi \leftarrow \Phi_0$
7:     **end if**
8:     Compute $\mathcal{L}_{util}$ from $\boldsymbol{x}_{\text{patch}}$ and Θ.
9:     Extract sparse features on $\boldsymbol{x}_{\text{image}}$ with Θ,
       reconstruct image with Φ
       and compute $\mathcal{L}_{recon}(\boldsymbol{x}_{\text{image}}; \Theta, \Phi)$.
10:    Update weights of Θ:
         $\Theta' \leftarrow \nabla_\Theta (\mathcal{L}_{util} - \lambda\mathcal{L}_{recon})$.
11:    Extract sparse features on $\boldsymbol{x}_{\text{image}}$ with Θ',
       reconstruct image with Φ
       and compute $\mathcal{L}_{recon}(\boldsymbol{x}_{\text{image}}; \Theta', \Phi)$.
12:    Update weights of Φ:
         $\Phi' \leftarrow \nabla_\Phi \mathcal{L}_{util}$.
13:   $\Theta \leftarrow \Theta', \Phi \leftarrow \Phi'$
14: **end for**

---

descriptor inversion model Φ is to minimize the reconstruction error over image data $\boldsymbol{x}_{\text{image}}$. On the other hand, NinjaNet Θ aims to conceal the visual content by maximizing this error. Thus, the resulting objective function for content concealment $V(\Theta, \Phi)$ is a minimax game between the two:

$$\min_\Phi \max_\Theta V(\Theta, \Phi) = \mathcal{L}_{recon}(\boldsymbol{x}_{\text{image}}; \Theta, \Phi). \qquad (6)$$

At the same time, we wish to maintain the descriptor utility:

$$\min_\Theta \mathcal{L}_{util}(\boldsymbol{x}_{\text{patch}}; \Theta). \qquad (7)$$

This brings us to the two separate optimization objectives for Θ and Φ that we will describe in the following. For the inversion model Φ, the objective remains the same as in Eqn. 6:

$$\mathcal{L}_\Phi = \mathcal{L}_{recon}(\boldsymbol{x}_{\text{image}}; \Theta, \Phi). \qquad (8)$$

However, for maintaining utility, NinjaNet with weights Θ is also optimized with the utility loss $\mathcal{L}_{util}(\boldsymbol{x}_{\text{patch}}; \Theta)$ from Eqn. 2. In conjunction with the maximization by Θ from Eqn. 6, the loss for NinjaNet becomes

$$\mathcal{L}_\Theta = \mathcal{L}_{util}(\boldsymbol{x}_{\text{patch}}; \Theta) - \lambda\mathcal{L}_{recon}(\boldsymbol{x}_{\text{image}}; \Theta, \Phi), \qquad (9)$$

where λ controls the balance of how much Θ prioritizes content concealment over utility retention, *i.e.* the privacy parameter. In practice, we optimize Θ and Φ in an alternating manner, such that Θ is not optimized in Eqn. 8 and Φ is not optimized in Eqn. 9. The overall objective is then

$$\Theta^*, \Phi^* = \arg\min_{\Theta, \Phi}(\mathcal{L}_\Theta + \mathcal{L}_\Phi). \qquad (10)$$

3.4. Implementation details

The code is implemented using PyTorch [50]. We use Kornia [57]'s implementation of SIFT for GPU acceleration. For all training, we use the Adam [33] optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\lambda = 0$.

Utility initialization. We use the *liberty* set of the UBC patches [22] to train NinjaNet for 200 epochs and select the model with the lowest average FPR@95 in the other two sets (*notredame* and *yosemite*). The number of submodules in NinjaNet (N in Fig. 2) is $N = 1$, since we observed no improvement in FPR@95 by increasing N. Dropout rate is 0.1. We use a batch-size of 1024 and learning rate of 0.01.

Reconstruction initialization. We randomly split MegaDepth [35] into train / validation / test split of ratio

		NinjaDesc				
RGB	SOSNet	$\lambda=0.01$	$\lambda=0.1$	$\lambda=0.25$	$\lambda=1.0$	$\lambda=2.5$

Figure 4. Qualitative results on landmark images. First column: original images overlaid with the 1000 (red) Harris corners [25]. Second column: reconstructions by the inversion model from raw SOSNet [74] descriptors extracted on those points. The last five columns show reconstruction from NinjaDesc with increasing privacy parameter λ. The SSIM and PSNR w.r.t. the original images are shown on top of each reconstruction. Best viewed digitally. *Image credits: first 3 — Holidays dataset [29]; last — laylamoran4battersea (Flickr).*

0.6 / 0.1 / 0.3. The process of forming a feature map is the same as in [11] and we use up to 1000 Harris corners [25] for all experiments. We train the inversion model with a batch-size of 64, learning rate of 1e-4 for a maximum of 200 epochs and select the best model with the lowest structural similarity (SSIM) on the validation split. We also do not use the discriminator as in [11], since convergence of the discriminator takes substantially longer, and it improves the inversion model only very slightly.

Joint adversarial training. The dataset configurations for $\mathcal{L}_{util}$ and $\mathcal{L}_{recon}$ are the same as in the above two steps, except the batch size, that is 968 for UBC patches. We use equal learning rate for Θ and Φ. This is 5e-5 for SOS-Net [74] and HardNet [41], and 1e-5 for SIFT [37]. NinjaDesc with the best FPR@95 in 20 epochs on the validation set is selected for testing.

4. Experimental results

In this section, we evaluate NinjaDesc on the two criteria that guide its design — the ability to simultaneously achieve: (1) content concealment (privacy) and (2) utility

(matching accuracy and camera localization performance).

4.1. Content concealment (Privacy)

We assess the content-concealing ability of NinjaDesc by measuring the reconstruction quality of descriptor inversion attacks. Here we assume the inversion model has access to the NinjaDescs and the input RGB images for training, *i.e.* x_{image} in Sec. 3.2. We train the inversion model from scratch for NinjaDesc (Eqn. 5) on the train split of MegaDepth [35], and the best model with the highest SSIM on the validation split is used for the evaluation.

Recall in Eqn. 9, λ is the privacy parameter controlling how much NinjaDesc prioritizes privacy over utility. The intuition is that, the higher λ is, the more aggressive NinjaDesc tries to prevent reconstruction quality by the inversion model. We perform descriptor inversion on NinjaDesc that are trained with a range of λ values to demonstrate its effect on reconstruction quality.

Fig. 4 shows qualitative results of descriptor inversion attacks when changing λ. We observe that λ indeed fulfills the role of controlling how much NinjaDesc conceals the original image content. When λ is small, *e.g.* 0.01, 0.1,
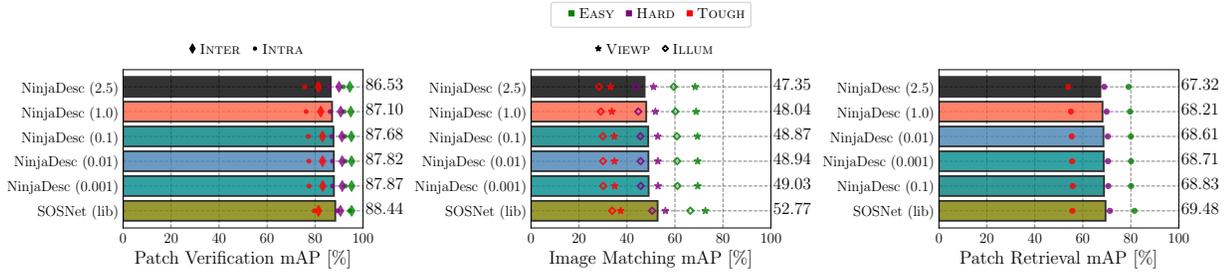
Figure 5 (three bar charts):

♦ Inter   • Intra

Patch Verification mAP [%]:
- NinjaDesc (2.5): 86.53
- NinjaDesc (1.0): 87.10
- NinjaDesc (0.1): 87.68
- NinjaDesc (0.01): 87.82
- NinjaDesc (0.001): 87.87
- SOSNet (lib): 88.44

★ Viewp   ◇ Illum

Image Matching mAP [%]:
- NinjaDesc (2.5): 47.35
- NinjaDesc (1.0): 48.04
- NinjaDesc (0.1): 48.87
- NinjaDesc (0.01): 48.94
- NinjaDesc (0.001): 49.03
- SOSNet (lib): 52.77

Patch Retrieval mAP [%]:
- NinjaDesc (2.5): 67.32
- NinjaDesc (1.0): 68.21
- NinjaDesc (0.01): 68.61
- NinjaDesc (0.001): 68.71
- NinjaDesc (0.1): 68.83
- SOSNet (lib): 69.48

Figure 5. HPatches evaluation results. We compare the baseline SOSNet [74] *vs.* NinjaDesc, with 5 different levels of privacy parameter λ (indicated by the number in parenthesis). All results are from models trained on the *liberty* subset of the UBC patches [22] dataset.

Metric	SOSNet (Raw)	NinjaDesc (λ)					
		0.001	0.01	0.1	0.25	1.0	2.5
MAE ($\uparrow$)	0.104	0.117	0.125	0.129	0.162	0.183	0.212
SSIM ($\downarrow$)	0.596	0.566	0.569	0.527	0.484	0.385	0.349
PSNR ($\downarrow$)	17.904	18.037	16.826	17.821	17.671	13.367	12.010

Table 1. Quantitative results of the descriptor inversion on SOSNet *vs.* NinjaDesc, evaluated on the MegaDepth [35] test split[2]. The arrows indicate higher / lower value is better for privacy.

the reconstruction is only slightly worse than that from the baseline SOSNet. As λ increases to 0.25, there is a visible deterioration in quality. Once equal / stronger weighting is given to privacy ($\lambda = 1, 2.5$), little texture / structure is revealed, achieving high privacy.

Such observation is also validated quantitatively by Table 1, where we see a drop in performance of the inversion model as λ increases across the three metrics: average mean absolute error (MAE), structural similarity (SSIM), and peak signal-to-noise ratio (PSNR) which are computed from the reconstructed image and the original input image.

4.2. Utility retention

We measure the utility of NinjaDesc via two tasks: image matching and visual localization.

Image matching. We evaluate NinjaDesc based on SOS-Net [74] with a set of different privacy parameter on the HPatches [6] benchmarks, which is shown in Fig. 5. NinjaDesc is comparable with SOSNet in mAP across all three tasks, especially for the *verification* and *retrieval* tasks. Also, higher privacy parameter λ generally corresponds to lower mAP, as $\mathcal{L}_{util}$ becomes less dominant in Eqn. 9.

Visual localization. We evaluate NinjaDesc with three base descriptors - SOSNet [74], HardNet [41] and SIFT [37] on the Aachen-Day-Night v1.1 [63, 84] dataset using the Kapture [28] pipeline. We use AP-Gem [55] for retrieval and localize with the shortlist size of 20 and 50. The keypoint detector used is DoG [37]. Table 2 shows localization results. Again, we observe little drop in accuracy for NinjaDesc overall compared to the original base descriptors, ranging from low ($\lambda = 0.1$) to high ($\lambda = 2.5$) privacies.

---

[2]Note that in [11], only SSIM is reported, and we do not share the same train / validation / test split. Also, [11] uses the discriminator loss for training which we omit, and it leads to slight difference in SSIM.

Query	NNs	Method	Accuracy @ Thresholds (%)		
			0.25m, 2°	0.5m, 5°	5.0m, 10°
		Base Desc	SOS / Hard / SIFT	SOS / Hard / SIFT	SOS / Hard / SIFT
Day (824)	20	Raw	85.1 / 85.4 / 84.3	92.7 / 93.1 / 92.7	97.3 / 98.2 / 97.6
		$\lambda = 0.1$	85.4 / 84.7 / 82.0	92.5 / 91.9 / 91.1	97.5 / 96.8 / 96.4
		$\lambda = 1.0$	84.7 / 84.3 / 82.9	92.4 / 91.9 / 91.0	97.2 / 96.7 / 96.1
		$\lambda = 2.5$	84.6 / 83.7 / 82.5	92.4 / 92.0 / 91.0	97.1 / 96.8 / 96.0
	50	Raw	85.9 / 86.8 / 86.0	92.5 / 93.7 / 94.1	97.3 / 98.1 / 98.2
		$\lambda = 0.1$	85.2 / 85.2 / 84.2	92.2 / 92.4 / 91.4	97.1 / 97.1 / 96.6
		$\lambda = 1.0$	84.7 / 85.7 / 83.4	92.2 / 92.6 / 91.6	97.2 / 96.7 / 96.7
		$\lambda = 2.5$	85.6 / 85.3 / 83.6	92.7 / 91.7 / 91.1	97.3 / 96.8 / 96.2
Night (191)	20	Raw	49.2 / 52.4 / 50.8	60.2 / 62.3 / 62.3	68.1 / 72.3 / 72.8
		$\lambda = 0.1$	47.6 / 43.5 / 44.0	57.1 / 54.5 / 51.3	63.4 / 61.8 / 61.3
		$\lambda = 1.0$	45.5 / 44.5 / 41.4	56.0 / 51.8 / 52.9	61.8 / 60.2 / 62.3
		$\lambda = 2.5$	45.0 / 44.5 / 43.5	55.0 / 54.5 / 49.7	61.8 / 61.3 / 61.3
	50	Raw	44.5 / 47.6 / 51.3	52.4 / 59.7 / 62.3	60.2 / 64.9 / 74.3
		$\lambda = 0.1$	39.8 / 39.8 / 41.9	47.6 / 48.7 / 50.3	57.6 / 56.0 / 59.7
		$\lambda = 1.0$	42.9 / 39.8 / 39.8	52.4 / 49.2 / 48.2	57.1 / 54.5 / 56.5
		$\lambda = 2.5$	41.9 / 38.2 / 40.3	49.2 / 47.1 / 49.2	56.6 / 55.0 / 57.1

Table 2. Visual localization results on Aachen-Day-Night v1.1 [84]. 'Raw' corresponds to the base descriptor in each column, followed by three λ vales (0.1, 1.0, 2.5) for NinjaDesc.

Comparing our results on HardNet and SIFT with Table 3 in Dusmanu *et al.* [18], NinjaDesc is noticeably better in retaining the visual localization accuracy of the base descriptors than the subspace descriptors in [18][3], *e.g.* drop in *night* is up to 30% for HardNet in [18] but $\approx 10\%$ for NinjaDesc.

Hence, the results on both image matching and visual localization tasks demonstrate that NinjaDesc is able to retain the majority of its utility w.r.t. to the base descriptors.

5. Ablation studies

Table 2 already hints that our proposed adversarial descriptor learning framework generalizes to several base descriptors in terms of retaining utility. In this section, we further investigate the generalizability of our method through additional experiments on different types of descriptors, inversion network architectures, and scene categories.

5.1. Generalization to different descriptors

We extend the same experiments from SOSNet [74] in Table 1 to include HardNet [41] and SIFT [37] as well. We

---

[3] [18] is evaluated on Aachen-Day-Night v1.0, resulting in higher accuracy in *Night* due to poor ground-truths, and the code of [18] is not released yet. We also report our results on v1.0 in the supplementary.
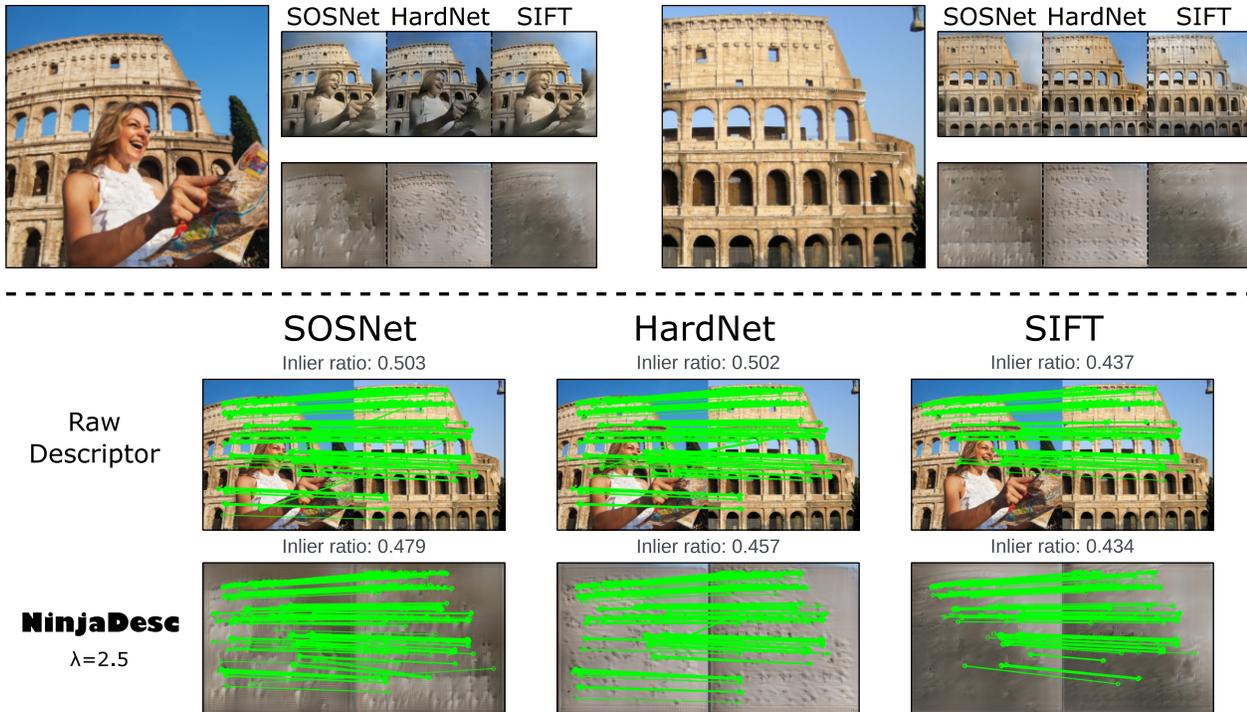
Figure 6. Illustration of our proposed adversarial descriptor learning framework's generalization across three different base descriptors. **Top.** We show two matching images. Two rows of small images to the right of each of them are the reconstructions. The top & bottom rows are, respectively, the reconstructions from the raw descriptor and from NinjaDesc ($\lambda = 2.5$) associated with the base descriptor above. **Bottom.** We visualize the matches between the two images on raw descriptors *vs*. NinjaDesc ($\lambda = 2.5$) for each of the three base descriptors. *Image credits: left — Tatyana Gladskih / stock.adobe.com; right — Urse Ovidiu (Wikimedia Commons, Public Domain).*

| Base Descriptor | SSIM (↓) | | | | | |
| | Raw (w/o NinjaDesc) | NinjaDesc (λ) | | | | |
		0.01	0.1	0.25	1.0	2.5
SOSNet	0.596	0.569	0.527	0.484	0.385	0.349
HardNet	0.582	0.545	0.516	0.399	0.349	0.312
SIFT	0.553	0.490	0.459	0.395	0.362	0.296

Table 3. Qualitative performance of the descriptor inversion model on the MegaDepth [35] test split with three base descriptors and the corresponding NinjaDescs, varying in privacy parameter.

| Arch. | UNet | | | UResNet | | |
	SOSNet	$\lambda = 1.0$	$\lambda = 2.5$	SOSNet	$\lambda = 1.0$	$\lambda = 2.5$
MAE (↑)	0.104	0.183	0.212	0.121	0.190	0.202
SSIM (↓)	0.596	0.385	0.349	0.595	0.427	0.380
PSNR (↓)	17.904	13.367	12.010	16.533	12.753	12.299

Table 4. Reconstruction results on MegaDepth [35]. We compare the UNet used in this work *vs*. a different architecture — UResNet.

report SSIM in Table 3. Similar to the observation for SOS-Net, increasing privacy parameter λ reduces reconstruction quality for both HardNet and SIFT as well. In Fig. 6, we qualitatively show the descriptor inversion and correspondence matching result across all three base descriptors. We observe that NinjaDesc derived from all three base descriptors are effective in concealing important contents such as person or landmark compared with the raw base descriptors. The visualization of keypoint correspondences between the images also demonstrates the utility retention of our proposed learning framework across different base descriptors.

5.2. Generalization to different architectures

So far, all experiments are evaluated with the same architecture for the inversion model - the UNet [58]-based network [11, 53]. To verify that NinjaDesc does not overfit to

this specific architecture, we conduct a descriptor inversion attack using an inversion model with drastically different architecture, called UResNet, which has a ResNet50 [26] as the encoder backbone and residual decoder blocks. (See the supplementary material.) The results are shown in Table 4, which depict only SSIM is slightly improved compared to UNet whereas MAE and PSNR remain relatively unaffected. This result illustrate that our proposed method is not limited by the architectures of the inversion model.

5.3. Content concealment on faces

We further show qualitative results on human faces using the Deepfake Detection Challenge (DFDC) [14] dataset. Fig. 7 presents the descriptor inversion result using the base descriptors (SOSNet [74]) as well as our NinjaDesc varying in privacy parameter λ. Similar to what we observed in Fig. 4, we see progressing concealment of facial features as we increase λ compared to the reconstruction on SOSNet.

Figure 7. Qualitative reconstruction results on faces. Images are cropped frames sampled from videos in the DFDC [14] dataset.

6. Utility and privacy trade-off

We now describe two experiments we perform to further investigate the utility and privacy trade-off of NinjaDesc.

First, in Fig. 8a we evaluate the mean matching accuracy (MMA) of NinjaDesc at the highest privacy parameter $\lambda = 2.5$, for both HardNet [41] and SIFT [37], on the HPatches sequences [6] and compare that with the sub-hybrid lifting method by Dusmanu *et al*. [18] with low privacy level (dim. 2). Even at a higher privacy level, NinjaDesc significantly outperforms sub-hybrid lifting for both types of descriptors. For NinjaDesc, the drop in MMA w.r.t. to HardNet is also minimal, and even increases w.r.t. SIFT.

Second, in Fig. 8b we perform a detailed utility *vs*. privacy trade-off analysis on NinjaDesc for all three base descriptors. The y-axis is the average difference in NinjaDesc's mAP across the three tasks in HPatches in Fig. 5, and the x-axis is the privacy measured by 1 - SSIM [11]. We plot the results varying the privacy parameter. For SOSNet and HardNet, the drop in utility ($< 4\%$) is a magnitude less than the gain in privacy (30%), indicating an optimal trade-off. Interestingly, for SIFT we see a net gain in utility for all λ (positive values in the y-axis). This is due to the SOSNet-like utility training, improving the *verification* and *retrieval* of NinjaDesc beyond the handcrafted SIFT. Full HPatches results for HardNet and SIFT are in the supplementary.

7. Limitations

NinjaDesc only affects the descriptors, and not the keypoint locations. Therefore, it does not prevent inferring scene structures from the patterns of keypoint locations themselves [38, 69]. Also, some level of structure can still be revealed where keypoints are very dense, *e.g*. the venetian blinds in the second example of Fig. 7.
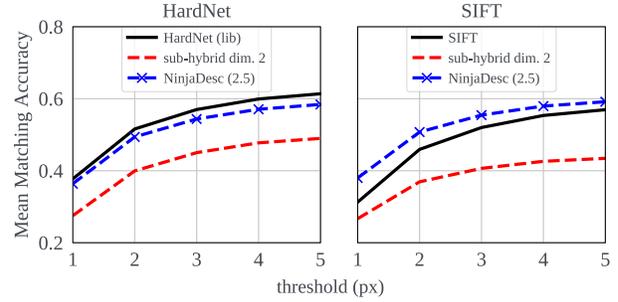
(a) Mean matching accuracy on HPatches [6] sequences. We compare NinjaDesc ($\lambda = 2.5$) to sub-hybrid lifting (dim. 2) in Dusmanu *et al*. [18].
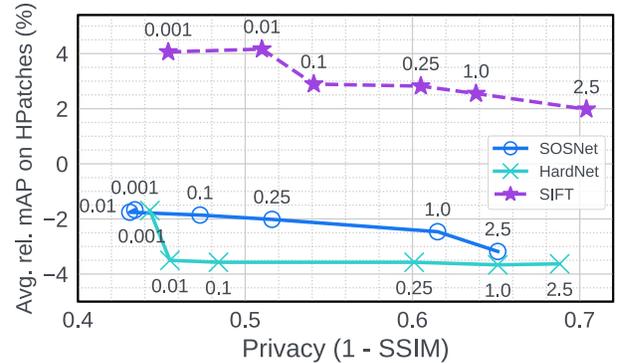
(b) For each descriptor we select NinjaDesc with varying privacy parameter values (annotated next to data points), and compare their utility *relative* to the raw descriptor *vs*. content concealment.

Figure 8. Utility *vs*. privacy trade-off analyses.

8. Conclusions

We introduced a novel adversarial learning framework for visual descriptors to prevent reconstructing original input image content from the descriptors. We experimentally validated that the obtained descriptors deteriorate the descriptor inversion quality with only marginal drop in utility. We also empirically demonstrated that we can control the trade-off between utility and non-invertibility using our framework, by changing a single parameter that weighs the adversarial loss. The ablation studies using different types of visual descriptors and image reconstruction network architectures demonstrate the generalizability of our method. Our proposed pipeline can enhance the security of computer vision systems that use visual descriptors, and has great potential to be extended for other applications beyond local descriptor encoding. Our observation suggests that the visual descriptors contain more information than what is needed for matching, which is removed by the adversarial learning process. It opens up a new opportunity in general representation learning for obtaining representations with only necessary information to preserve privacy.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 2011. 1

[2] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *CVPR*, 2012. 2

[3] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 1

[4] Relja Arandjelović and Andrew Zisserman. DisLocation: Scalable descriptor distinctiveness for location recognition. In *ACCV*, 2014. 1

[5] Sungyong Baik, Hyo Jin Kim, Tianwei Shen, Eddy Ilg, Kyoung Mu Lee, and Christopher Sweeney. Domain adaptation of learned featuresfor visual localization. In *BMVC*, 2020. 1

[6] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017. 2, 6, 8

[7] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key.Net: Keypoint detection by handcrafted and learned cnn filters. In *ICCV*, 2019. 1

[8] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. In *ECCV*, 2010. 2

[9] Kunal Chelani, Fredrik Kahl, and Torsten Sattler. How privacy-preserving are line clouds? Recovering scene details from 3d lines. In *CVPR*, 2021. 1, 2

[10] Emmanuel d'Angelo, Laurent Jacques, Alexandre Alahi, and Pierre Vandergheynst. From bits to images: Inversion of local binary descriptors. *TPAMI*, 36(5):874–887, 2013. 1, 2

[11] Deeksha Dangwal, Vincent T. Lee, Hyo Jin Kim, Tianwei Shen, Meghan Cowan, Rajvi Shah, Caroline Trippel, Brandon Reagen, Timothy Sherwood, Vasileios Balntas, Armin Alaghi, and Eddy Ilg. Analysis and mitigations of reverse engineering attacks on local feature descriptors. In *BMVC*, 2021. 1, 2, 3, 5, 6, 7, 8

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 3

[13] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *CVPR Workshops*, 2018. 1

[14] Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton-Ferrer. The deepfake detection challenge dataset. *CoRR*, abs/2006.07397, 2020. 7, 8

[15] Jing Dong, Erik Nelson, Vadim Indelman, Nathan Michael, and Frank Dellaert. Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach. In *ICRA*, 2015. 1

[16] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *CVPR*, 2016. 1, 2

[17] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable cnn for joint detection and description of local features. In *CVPR*, 2019. 1

[18] Mihai Dusmanu, Johannes L Schönberger, Sudipta N Sinha, and Marc Pollefeys. Privacy-preserving visual feature descriptors through adversarial affine subspace embedding. In *CVPR*, 2021. 1, 2, 6, 8

[19] Zekeriya Erkin, Martin Franz, Jorge Guajardo, Stefan Katzenbeisser, Inald Lagendijk, and Tomas Toft. Privacy-preserving face recognition. In *International symposium on privacy enhancing technologies symposium*, 2009. 2

[20] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L Schönberger, and Marc Pollefeys. Privacy preserving structure-from-motion. In *ECCV*, 2020. 2

[21] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L Schonberger, and Marc Pollefeys. Privacy preserving localization and mapping from uncalibrated cameras. In *CVPR*, 2021. 1, 2

[22] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *CVPR*, 2007. 3, 4, 6

[23] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *NIPS*, 2014. 3

[24] Sam Hare, Amir Saffari, and Philip HS Torr. Efficient online structured output learning for keypoint-based object tracking. In *CVPR*, 2012. 1

[25] Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, 1988. 3, 5

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[27] Carlos Hinojosa, Juan Carlos Niebles, and Henry Arguello. Learning privacy-preserving optics for human pose estimation. In *ICCV*, 2021. 2

[28] Martin Humenberger, Yohann Cabon, Nicolas Guerin, Julien Morat, Jérôme Revaud, Philippe Rerole, Noé Pion, Cesar de Souza, Vincent Leroy, and Gabriela Csurka. Robust image retrieval-based visual localization using kapture, 2020. 6

[29] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometry consistency for large scale image search. In *ECCV*, 2008. 5

[30] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *IJCV*, 2021. 1, 3

[31] Hiroharu Kato and Tatsuya Harada. Image reconstruction from bag-of-visual-words. In *CVPR*, 2014. 2

[32] Hyo Jin Kim, Enrique Dunn, and Jan-Michael Frahm. Learned contextual feature reweighting for image geo-localization. In *CVPR*, 2017. 1

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 4

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 2017. 2

[35] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 3, 4, 5, 6, 7

[36] Ce Liu, Jenny Yuen, and Antonio Torralba. SIFT flow: Dense correspondence across scenes and its applications. *TPAMI*, 2010. 2

[37] David G. Lowe. Distinctive image features from scale-invariant keypoints. In *IJCV*, 2004. 2, 5, 6, 8

[38] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. ContextDesc: Local descriptor augmentation with cross-modality context. In *CVPR*, 2019. 8

[39] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, 2015. 2

[40] Christopher Mei, Gabe Sibley, Mark Cummins, Paul Newman, and Ian Reid. Rslam: A system for large-scale mapping in constant-time using stereo. *IJCV*, 2011. 1

[41] Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenović, and Jiři Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *NIPS*, 2017. 3, 5, 6, 8

[42] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardos. ORB-SLAM: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, Oct 2015. 1

[43] Raul Mur-Artal and Juan D Tardós. ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 2017. 1

[44] Georg Nebehay and Roman Pflugfelder. Consensus-based matching and tracking of keypoints for object tracking. In *WACV*, 2014. 1

[45] Richard A. Newcombe, Steven J. Lovegrove, and Andrew J. Davison. DTAM: Dense tracking and mapping in real-time. In *ICCV*, 2011. 1

[46] Tony Ng, Vassileios Balntas, Yurun Tian, and Krystian Mikolajczyk. SOLAR: Second-order loss and attention for image retrieval. In *ECCV*, 2020. 1

[47] Hyeonwoo Noh, André Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Image retrieval with deep local features and attention-based keypoints. In *ICCV*, 2017. 1

[48] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *TPAMI*, 2002. 2

[49] Luc Oth, Paul Furgale, Laurent Kneip, and Roland Siegwart. Rolling shutter camera calibration. In *CVPR*, 2013. 1

[50] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 4

[51] Federico Pernici and Alberto Del Bimbo. Object tracking by oversampling local features. *TPAMI*, 2013. 1

[52] Francesco Pittaluga, Sanjeev Koppal, and Ayan Chakrabarti. Learning privacy preserving encodings through adversarial training. In *WACV*, 2019. 2

[53] Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *CVPR*, 2019. 1, 2, 3, 7

[54] Horia Porav, Will Maddern, and Paul Newman. Adversarial training for adverse conditions: Robust metric localisation using appearance transfer. In *ICRA*, 2018. 1

[55] Jerome Revaud, Jon Almazán, Rafael Sampaio de Rezende, and César Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. In *ICCV*, 2019. 6

[56] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 1

[57] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for PyTorch. In *WACV*, 2020. 4

[58] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer, 2015. 3, 7

[59] Proteek Chandan Roy and Vishnu Naresh Boddeti. Mitigating information leakage in image representations: A maximum entropy approach. In *CVPR*, 2019. 3

[60] Ahmad-Reza Sadeghi, Thomas Schneider, and Immo Wehrenberg. Efficient privacy-preserving face recognition. In *International Conference on Information Security and Cryptology*, 2009. 2

[61] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 1

[62] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *TPAMI*, 39(9):1744–1756, 2017. 1

[63] 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 *CVPR*, 2018. 2, 6

[64] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1

[65] Mikiya Shibuya, Shinya Sumikura, and Ken Sakurada. Privacy preserving visual SLAM. In *ECCV*, 2020. 1, 2

[66] Oriane Siméoni, Yannis Avrithis, and Ondrej Chum. Local features and visual words emerge in activations. In *CVPR*, 2019. 1

[67] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3

[68] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2

[69] Pablo Speciale, Johannes L Schonberger, Sing Bing Kang, Sudipta N Sinha, and Marc Pollefeys. Privacy preserving image-based localization. In *CVPR*, 2019. 1, 2, 8

[70] Pablo Speciale, Johannes L Schonberger, Sudipta N Sinha, and Marc Pollefeys. Privacy preserving image queries for camera localization. In *CVPR*, 2019. 1

[71] Chris Sweeney, Tobias Hollerer, and Matthew Turk. Theia: A fast and scalable structure-from-motion library. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, page 693–696, 2015. 1

[72] Yurun Tian, Axel Barroso-Laguna, Tony Ng, Vassileios Balntas, and Krystian Mikolajczyk. HyNet: Learning local descriptor with hybrid similarity measure and triplet loss. In *NeurIPS*, 2020. 1

[73] Yurun Tian, Bin Fan, and Fuchao Wu. L2-Net: Deep learning of discriminative patch descriptor in Euclidean space. In *CVPR*, 2017. 3

[74] Yurun Tian, Xin Yu, Bin Fan, Wu. Fuchao, Huub Heijnen, and Vassileios Balntas. SOSNet: Second order similarity regularization for local descriptor learning. In *CVPR*, 2019. 1, 3, 5, 6, 7

[75] Carl Toft, Will Maddern, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, Fredrik Kahl, and Torsten Sattler. Long-term visual localization revisited. *TPAMI*, 2020. 1

[76] Carl Toft, Daniyar Turmukhambetov, Torsten Sattler, Fredrik Kahl, and Gabriel J Brostow. Single-image depth prediction makes feature matching easier. In *ECCV*, 2020. 1

[77] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, 2013. 1

[78] Giorgos Tolias, Tomas Jenicek, and Ondřej Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *ECCV*, 2020. 1

[79] Carl Vondrick, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. Hoggles: Visualizing object detection features. In *ICCV*, 2013. 2

[80] Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. Reconstructing an image from its local descriptors. In *CVPR*, 2011. 1, 2

[81] Taihong Xiao, Yi-Hsuan Tsai, Kihyuk Sohn, Manmohan Chandraker, and Ming-Hsuan Yang. Adversarial learning of privacy-preserving and task-oriented representations. In *AAAI*, 2020. 2

[82] Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. In *NIPS*, 2017. 3

[83] Ryo Yonetani, Vishnu Naresh Boddeti, Kris M Kitani, and Yoichi Sato. Privacy-preserving visual learning using doubly permuted homomorphic encryption. In *ICCV*, 2017. 2

[84] Zichao Zhang, Torsten Sattler, and Davide Scaramuzza. Reference pose generation for long-term visual localization via learned features and view synthesis. *IJCV*, 2020. 2, 6

[85] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *CVPR*, 2006. 2