Autoencoder-learned local image descriptor for image inpainting

Nina Žižakić[1]*, Izumi Ito[2], Laurens Meeus[1], and Aleksandra Pižurica[1]

[1] Department of Telecommunications and Information Processing, TELIN – GAIM, Ghent University – imec, Ghent, Belgium
{nina.zizakic, laurens.meeus, aleksandra.pizurica}@ugent.be
[2] Information and Communications Engineering, Tokyo Institute of Technology, Tokyo, Japan
ito@ict.e.titech.ac.jp

Abstract. In this paper, we propose an efficient method for learning local image descriptors suitable for the use in image inpainting algorithms. We learn the descriptors using a convolutional autoencoder network that we design such that the network produces a computationally efficient extraction of patch descriptors through an intermediate image representation. This approach saves computational memory and time in comparison to existing methods when used with algorithms that require patch search and matching within a single image. We show these benefits by integrating our descriptor into an inpainting algorithm and comparing it to the existing autoencoder-based descriptor. We also show results indicating that our descriptor improves the robustness to missing areas of the patches.

Keywords. Local image descriptors, patch descriptors, autoencoders, unsupervised deep learning, inpainting.

1 Introduction

Local image descriptors are a crucial component of many image processing tasks – image denoising, inpainting, stitching, object tracking, to name a few. In recent years, the approach to designing these patch descriptors has shifted from using hand-crafted features to the (deep) learning approach. Learned descriptors have been shown to outperform some hand-crafted ones on benchmarks [1, 7]. However, despite many advancements in the learning approach and their superior performance on benchmarks, hand-crafted descriptors still perform comparably or better than the learned descriptors in a practical context [16]. He et al. argue that descriptor learning should not be approached as a standalone problem, but rather as a component of a broader image processing task, which must also be taken into consideration [8].

*Corresponding author

In this paper, we present a descriptor whose learning process is specifically tailored for use in an image inpainting algorithm. The descriptor is based on our previous work [19] and has been extended so that it can be applied to image inpainting tasks. We show that our descriptor improves the inpainting result and makes the task computationally feasible in cases where it was previously not. The computational time of the inpainting is reduced by leveraging our specific autoencoder architecture that allows for the calculation of the intermediate representation of the image [19]. From the intermediate representation, we can obtain the descriptors using a simple operation. We hypothesise that the improved inpainting results are the result of (i) high robustness to missing parts of patches in comparison to the other descriptors (shown in section 4), and (ii) the fine-tuning of the descriptor on data similar to what will be used for the inpainting. This fine-tuning is possible since our descriptor is trained in an unsupervised fashion. To achieve such fine-tuning with other (supervised) descriptors, it would be necessary to have a labeled set for the type of images that need to be inpainted, which is not feasible in most cases. Furthermore, and in contrast to other local feature descriptors, our descriptor is intentionally designed to be translation and rotation sensitive. This property is not always desirable and many descriptors are designed to be translation invariant. However, translation invariant descriptors do not achieve good results in inpainting since they tend to retrieve patches that are rotated or scaled, and introduce irregularities in the inpainted edges.

In the next section we discuss the current state of local image descriptors and, in particular, descriptor learning. We also include a brief introduction to autoencoders. We describe our method in Section 3. Section 4 contains the experimental results on robustness to missing data and application to inpainting. We conclude our work in Section 5.

2 Related work

The classical approach uses hand-crafted features to design local image descriptors. The most well-known descriptors include SIFT [12], SURF [3], BRIEF [4], ORB [14], which continue to be relevant to modern approaches. In recent years, the development of deep learning techniques has resulted in numerous learned descriptors [21, 17, 2, 8]. These descriptors are mostly learned in a supervised fashion, with the labels on pairs of patches, indicating their similarity or dissimilarity.

While a few learned descriptors show high performance on benchmarks [1, 7], the established hand-crafted descriptors such as SIFT are consistently chosen over the learned ones in practical applications [16]. He et al. argue that this is due to the fact that the descriptors were trained too generally and that training a descriptor for an image processing task in mind can lead to the descriptor performing better than the general descriptors [8]. This effect is partially explained by the fact that different image processing tasks value different properties in descriptors. For example, in object tracking it is desirable that the descriptor is translation invariant. However, this property is not desirable in descriptors for inpainting.

Since supervised methods are dependent on labeled data, it is often unfeasible to create a supervised descriptor for a specific image processing task. Furthermore, since most labeled data for patch descriptors is designed for object tracking and hence is translation invariant, there are very few labelled datasets suitable for inpainting.

On the other hand, unsupervised learning methods such as autoencoders do not suffer from the problem of depending on labeled data. Chen et al. were the first authors to propose using autoencoders to the general problem of descriptor learning [5]. Their autoencoder-learned descriptor showed promising results, however, the computational time and memory usage make their method infeasible for high resolution image processing tasks. Their fully-connected network has more parameters to be trained and requires longer training times than convolutional autoencoder designs. Moreover, their descriptor does not allow different input sizes and therefore a separate autoencoder needs to be trained for every patch size, which renders the framework unusable in practical scenarios.

In our previous work [19], we proposed an autoencoder-based patch descriptor designed for applications with many patch comparisons within a single image. Our specific network architecture yielded a special image representation that we refer to as the intermediate representation (IR). The IR is a compact way of storing the descriptors of all the patches of an image because the descriptors of overlapping patches overlap themselves. Extracting a descriptor from the IR is done fast using a max-pooling operation. Moreover, the use of convolutional layers ensures a more efficient learning process and usability of the descriptor for all patch sizes.

In this paper, we build on our previous work to show that a descriptor designed specifically for an image processing task can outperform the general descriptors and can improve the performance of the task, both in terms of computational time and in terms of visual inspection. Furthermore, the unsupervised nature of our descriptor facilitates fine-tuning on the specific type of data to be used in the image processing task in order to further improve its performance.

2.1 Autoencoders

An autoencoder is a type of artificial neural network used to learn efficient data representations in an unsupervised fashion. An autoencoder consist of two parts: the encoder, which maps the input into an efficient representation, and the decoder, which maps the efficient representation back to the output (where, ideally, the output matches the input). Autoencoders are trained by minimising the reconstruction error between the input and output, while imposing some constraints on the representation layer. Formally, an autoencoder with encoder $\mathcal{E}$ and decoder $\mathcal{D}$ is trained to minimise the loss function $J(X, \mathcal{E}, \mathcal{D}) = \sum_{x \in X} \mathcal{L}(x, \mathcal{D}(\mathcal{E}(x))) + \Omega(\mathcal{E}(x))$, where $x \in X$ is a data sample, $\mathcal{L}$ is some reconstruction loss metric and $\Omega(\mathcal{E}(x))$ is an optional sparsity regularisation term

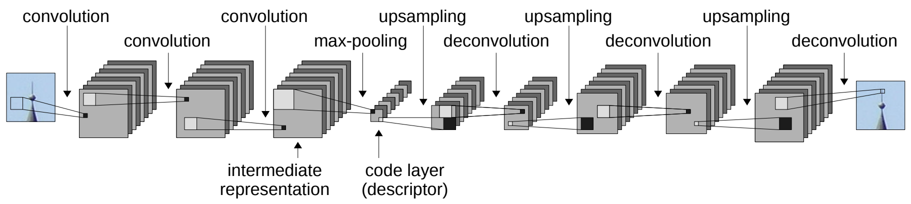

Fig. 1. The proposed autoencoder architecture. Max-pooling layers after the first two convolutional layers have been omitted in order to obtain an intermediate representation (IR) of the image that preserves the spatial information.

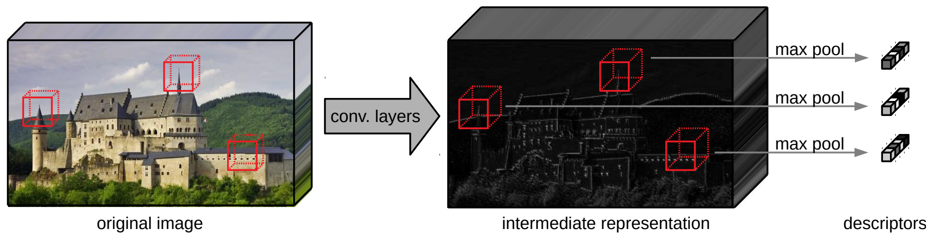

Fig. 2. Exploiting the proposed intermediate representation of an image in algorithms that require many patch comparisons. The intermediate representation is calculated once from the original image through the convolutional layers of the encoder. In algorithms that need to compare patches (e.g. inpainting), the descriptors are extracted from the IR using the fast max-pooling operation, and then compared.

imposed on the hidden (representation) layer. Autoencoders designed for working on image data are usually built by alternating convolutional and max-pooling layers. In the l-th convolutional layer, the output neuron at location (i, j) of the k-th channel is expressed as follows:

$$x_{ij}^{(l,k)} = \sum_{c \in C} \sum_{u=0}^{f^{(l)}-1} \sum_{v=0}^{f^{(l)}-1} w_{uv}^{(l,k,c)} x_{(i+u)(j+v)}^{(l-1,c)} + b^{(l,k)}, \tag{1}$$

where C is the set of input channel indices, $w^{(l,k,c)}$ is the convolutional kernel for the l-th layer and k-th channel applied on the c-th input channel, $b^{(l,k)}$ is the bias for the l-th layer of the k-th channel, and $f^{(l)}$ is the size of the convolutional kernel (filter) for the l-th layer.

3 Proposed method

The core contribution of this paper is a local image descriptor suitable for integration with image inpainting. This integration is possible due to (i) the specific autoencoder architecture that provides the intermediate representation (IR) of an image and (ii) the translation-sensitive design of our descriptor.

The introduction of the IR yields two main benefits: it is less memory intensive than storing the descriptors of all patches within an image and it allows a descriptor of a single patch to be extracted from the IR with minimal computation.

We accomplish this by proposing a novel convolutional neural network (CNN) architecture. While traditional CNNs consist of alternating convolutional and max-pooling layers, we introduce an architecture in which we remove all the max-pooling layers in the encoder except for the last one. This decision was encouraged by a recent study which showed that max-pooling layers are not necessary for a successful neural network [18]. Max-pooling is usually applied since it adds extra non-linearity and introduces dimensionality reduction. We omit max-pooling after the first two convolutions and instead employ non-linear activation functions. We leave only one max-pooling layer with a large spatial extent at the end of the encoder to reduce the dimensionality of the code layer.

Unlike the fully-connected architecture used in [5], the convolutional layers that we use in our network are able to exploit the self-similarity property of natural images and reduce the computational complexity and time, while achieving better results than fully connected-layers. Moreover, the use of convolutional layers is critical for the ability to extract patch descriptors from the IR, since convolutional layers preserve spatial information.

The reduction of computational time and memory usage follows from the IR in our network. The IR is obtained by propagating the complete image (containing patches of interest) through the convolutional layers in the encoder, but not the max-pooling layer. Figure 1 shows the architecture of our network and the IR, and Figure 2 shows how they can be used within an image processing task.

Mathematically, we define our intermediate representation as follows. Let $x := x^{(0,:)}$ be the input image. The intermediate representation is obtained as $\mathcal{IR}(x) = x^{(L,:)}$, with

$$x^{(L,k)} = \mathcal{A}(\mathcal{C}_L(\mathcal{A} \dots (\mathcal{C}_1(x^{(0,:)})))), \tag{2}$$

where L is the number of convolutional layers in the encoder $\mathcal{E}$, $x^{(l,k)}$ is the k-th channel of the output of the l-th layer, $x^{(l,:)}$ denotes all channels of the output of the l-th layer, $\mathcal{A}$ is some activation function, and $\mathcal{C}_l$ is the l-th convolutional layer. From the intermediate representation of an image $\mathcal{IR}(x)$, we obtain the descriptor for a patch $x_{(i,j)}$, whose upper left corner is positioned at (i, j), by performing the max-pooling on the patch of the IR, as follows

$$\mathcal{E}(x_{(i,j)}) = \mathcal{MP}(\mathcal{IR}(x)_{(i,j)}). \tag{3}$$

The activation function $\mathcal{A}$ from (2) is the rectified linear unit, $\mathcal{A}(x) = x^+ = \max(0, x)$. We trained the network with the Adadelta optimizer [22] and used binary cross entropy as the loss function, and scaling the pixel values to be in the range $[0, 1]$):

$$J(X, \mathcal{E}, \mathcal{D}) = \sum_{x \in X} \mathcal{L}(x, \hat{x}) = \sum_{x \in X} \left(x \log(\hat{x}) + (1 - x) \log(1 - \hat{x})\right), \tag{4}$$

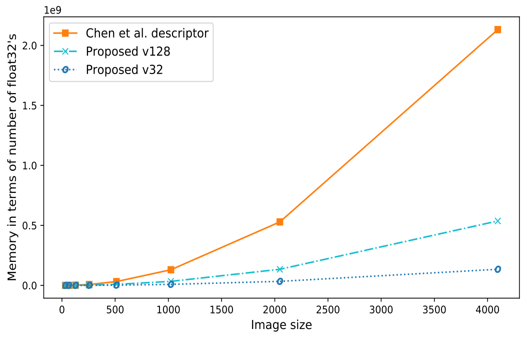

Fig. 3. A comparison of the memory requirements (expressed in the number of 32 bit floating points) as a function of image size in pixels, for the two versions of the proposed descriptor (v32 and v128) compared to Chen et al. [5].

where $\hat{x} := \mathcal{D}(\mathcal{E}(x))$ is the output of the autoencoder. We trained the autoencoder in two different ways, creating two versions of the descriptor, v32 and v128 (named after the dimensionality of the descriptor for 16×16 patches).

Image inpainting calls for sensitivity of the deployed patch descriptor to translation, rotation, and scaling. Although convolutional layers are known to be translation invariant, Kauderer-Abrams claimed that this invariance is due to the use of data augmentation, a large number of layers in the network, and the use of large filters in convolutional layers [9]. We have designed our autoencoder to avoid these sources of translation invariance.

Our autoencoder is implemented using the Keras library for neural networks. We initially trained our network on a total of 200k 16×16 patches cropped from images from the datasets ImageNet [6], KonIQ [20], and VisualGenome [11].

For the case of inpainting, descriptors play a crucial role in the performance of the algorithm, since retrieving similar patches is the most prevalent operation in the inpainting. The unsupervised nature of our descriptor makes it possible to fine-tune the descriptor to the type of data which will be used in the inpainting. For the fine-tuning, we use around 50k patches cropped from the images of interest. In our case, the images of interest were macro-photographs of the paintings from the Ghent Altarpiece painted by the Van Eyck brothers [10]. We describe the inpainting experiment in further detail in the following section.

In Figure 3 we compare the effective memory usage required by different descriptors with respect to the image size. The results indicate potential for a tremendous decrease in memory usage for applications on a single image. This decrease could make some algorithms, which use many patch comparisons, feasible in high resolution images.

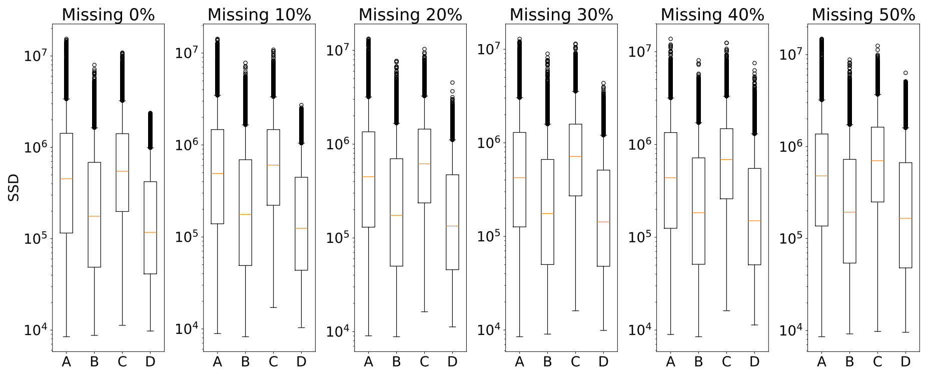

Fig. 4. Comparison of the descriptors' robustness to missing data. A – proposed descriptor v32, B – proposed descriptor v128, C – Chen et al. [5], D – exhaustive search on pixel intensity values. The plots are showing the sum of squared differences (SSD) of ground truth pixel values of patches found by the descriptors (in A-C) and exhaustive search (D), based on the percentage of missing area in a patch.

4 Experimental evaluation

The original motivation for implementing a local image descriptor was to improve the inpainting and to make it accessible for high-resolution images. In the first part of this section, we asses our descriptor's robustness to missing areas in the patch, a property important for inpainting application. In the second part, we describe the integration of our descriptor with the inpainting algorithm.

4.1 Robustness to missing areas in the patch

Robustness to missing regions is a desirable property of descriptors used for applications such as inpainting and scene reconstruction from multiview data. We compare our descriptor with the descriptor from [5] and the exhaustive search on pixel intensity values. We trained all the descriptor networks on the same set of colour patches.

The setup of the experiment is as follows. We select a set of query patches, which we edit in order to introduce missing areas. For each query patch, we retrieve the k most similar patches either by comparing their descriptors or by using exhaustive search over the pixel values. The quality of the patch retrieval (and thus the robustness to missing data) is evaluated based on the sum of squared differences (SSD) between the complete (undamaged) query patches and the retrieved ones.

The results are presented in Figure 4, comparing our two descriptor networks, descriptor from [5], and the exhaustive search over pixel intensity values. Visual examples of retrieved patches are shown in Figure 5.

When the missing area in a patch is small, the exhaustive search retrieves the patches that are the most similar to the (original) query patch. However, as

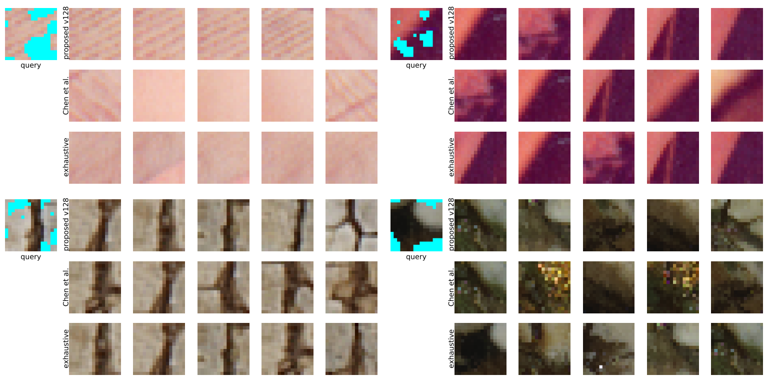

Fig. 5. Patch retrieval where the query has missing parts. For each query, the first row corresponds to the proposed v128 descriptors; the second row: the descriptors from [5], and the third row: exhaustive search. The missing parts of the query patches on the right are shown in cyan.

the missing area increases in size, our descriptor v128 begins to outperform the exhaustive search, showing more robustness to missing data.

Our descriptors also show superior performance compared to the existing descriptor learned with autoencoders. Our method v128 shows significantly better performance than the method implemented by Chen et al. [5], while having the same patch descriptor dimensionality. Moreover, our method v32 that shows similar results to [5] has an order of magnitude lower dimensionality of the descriptor when encoding a single patch. The dimensionality comparison changes even more in the favour of our method when encoding the whole image due to the usage of the IR (Figure 3).

4.2 Inpainting

We put our descriptor to a real-world test in the application of digital painting analysis, more specifically, digital inpainting of paint-loss areas. As a case study, we use images from the panels of Ghent Altarpiece [10]. The paint-loss areas to be inpainted are determined by the detection algorithm from [13].

We have modified the inpainting algorithm from [15] to use patch descriptors instead of their pixel values for comparisons. We fine-tuned the descriptor on the data from the patches extracted from the images of the paintings.

The inpainting results are presented in Figure 6, showing the inpainting of one panels, *the Prophet Zachary*, from the altarpiece. On this particular panel, the paint-loss areas are showing as light brown. Figure 7 shows the zoomed details of the panel. We have also used the inpainting algorithm without the descriptors, however, we were not able to obtain the inpainting results on the whole panel without using the descriptor due to the memory error on our computer.

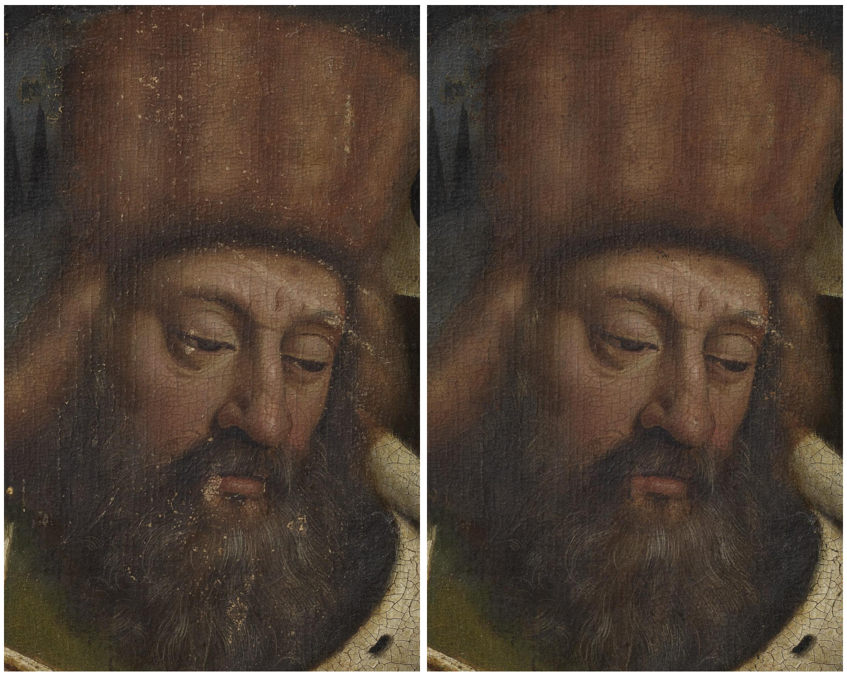

Fig. 6. Image inpainting results. Left: original; Right: Inpainted with a patch-based method using the proposed patch descriptors. Image copyright: Ghent, Kathedrale Kerkfabriek, Lukasweb; photo courtesy of KIK-IRPA, Brussels.

Moreover, we were not able to test the inpainting with the descriptor from [5] as it would require the retraining of their network to suit the needed patch size.

The inpainting results are very promising and show that our descriptor was not only able to visually improve the inpainted images, but also to improve on the computational aspect of the inpainting.

5 Conclusion

We propose a novel method for learning patch descriptors using autoencoders, for the use in image inpainting. Our approach saves computational memory and time in comparison to existing methods when used with algorithms such as those for inpainting that require patch search and matching within a single image. The proposed descriptor shows higher robustness to missing data when compared with an existing descriptor learned with autoencoders from [5] and exhaustive search over pixel intensity values. Furthermore, integrating our descriptor into an inpainting algorithm results in visual improvements over the inpainted images and the ability of the inpainting algorithm to handle higher resolution images.

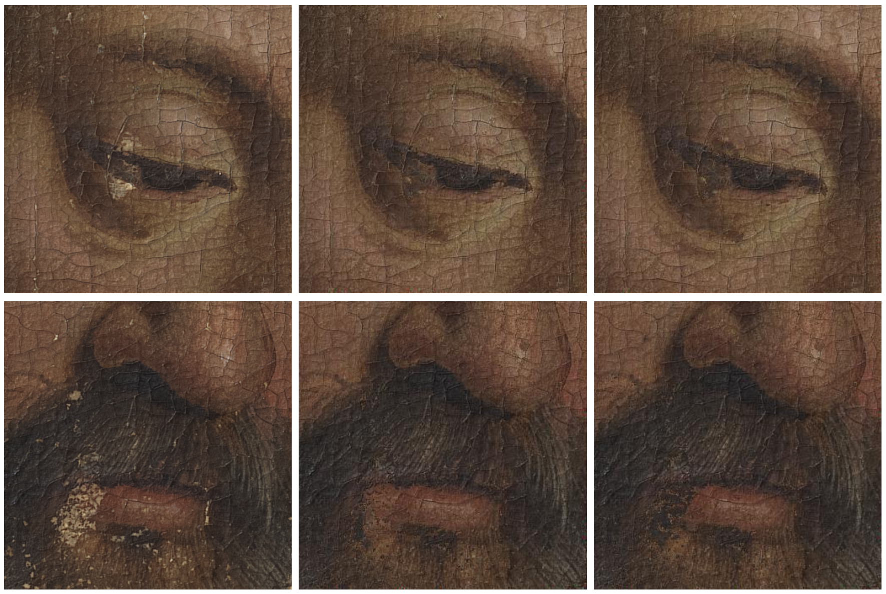

Fig. 7. Image inpainting results. Left column: original images containing paint-loss (showing as light brown); Middle column: Inpainted with a patch-based method using the proposed patch descriptor; Right column: Inpainted without using the descriptors. Image copyright: Ghent, Kathedrale Kerkfabriek, Lukasweb; photo courtesy of KIK-IRPA, Brussels.

References

1. Balntas, V., Lenc, K., Vedaldi, A., Mikolajczyk, K.: Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5173–5182 (2017)
2. Balntas, V., Riba, E., Ponsa, D., Mikolajczyk, K.: Learning local feature descriptors with triplets and shallow convolutional neural networks. In: Proceedings of the British Machine Vision Conference (BMVC). pp. 119.1–119.11 (2016), https://dx.doi.org/10.5244/C.30.119
3. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). Computer vision and image understanding **110**(3), 346–359 (2008)
4. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: Brief: Binary robust independent elementary features. In: European conference on computer vision. pp. 778–792. Springer (2010)
5. Chen, L., Rottensteiner, F., Heipke, C.: Feature descriptor by convolution and pooling autoencoders. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences-ISPRS Archives 40 (2015), Nr. 3W2 **40**(3W2), 31–38 (2015)

6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
7. Fischer, P., Dosovitskiy, A., Brox, T.: Descriptor matching with convolutional neural networks: a comparison to SIFT. arXiv preprint arXiv:1405.5769 (2014)
8. He, K., Lu, Y., Sclaroff, S.: Local descriptors optimized for average precision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
9. Kauderer-Abrams, E.: Quantifying translation-invariance in convolutional neural networks. arXiv preprint arXiv:1801.01450 (2017)
10. KIK/IRPA: Closer to van eyck: The ghent altarpiece. http://closertovaneyck.kikirpa.be/ghentaltarpiece/#home/ (2019)
11. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., et al.: Visual genome: Connecting language and vision using crowdsourced dense image annotations. International Journal of Computer Vision **123**(1), 32–73 (2017), https://doi.org/10.1007/s11263-016-0981-7
12. Lowe, D.G.: Object recognition from local scale-invariant features. In: ICCV. p. 1150. IEEE (1999)
13. Meeus, L., Huang, S., Devolder, B., Dubois, H., Pižurica, A.: Deep learning for paint loss detection with a multiscale, translation invariant network. In: Proceedings of the 11th International Symposium on Image and Signal Processing and Analysis (ISPA 2019). p. 5 (2019)
14. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.R.: ORB: An efficient alternative to SIFT or SURF. In: ICCV. pp. 2564–2571. IEEE (2011)
15. Ružić, T., Pižurica, A.: Context-aware patch-based image inpainting using Markov random field modeling. IEEE Transactions on Image Processing **24**(1), 444–456 (2015)
16. Schonberger, J.L., Hardmeier, H., Sattler, T., Pollefeys, M.: Comparative evaluation of hand-crafted and learned local features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1482–1491 (2017)
17. Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., Moreno-Noguer, F.: Discriminative learning of deep convolutional feature point descriptors. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 118–126 (2015)
18. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806 (2014)
19. Žižakić, N., Ito, I., Pižurica, A.: Learning local image descriptors with autoencoders. In: Proc. IEICE Inform. and Commun. Technol. Forum ICTF 2019 (2019)
20. Wiedemann, O., Hosu, V., Lin, H., Saupe, D.: Disregarding the big picture: Towards local image quality assessment. In: 10th International Conference on Quality of Multimedia Experience(QoMEX). IEEE (2018), http://database.mmsp-kn.de
21. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4353–4361 (2015)
22. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)