Information Sciences 581 (2021) 655–677

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Y-Net: Learning Domain Robust Feature Representation for ground camera image and large-scale image-based point cloud registration

Weiquan Liu [a], Cheng Wang [a,*], Shuting Chen [b], Xuesheng Bian [a], Baiqi Lai [a], Xuelun Shen [a], Ming Cheng [a], Shang-Hong Lai [c], Dongdong Weng [d], Jonathan Li [a,e]

[a] Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Informatics, Xiamen University, Xiamen 361005, China
[b] Chengyi University College, Jimei University, Xiamen 361021, China
[c] Department of Computer Science, National Tsing Hua University, Hsinchu 30013, Taiwan
[d] Beijing Engineering Research Center of Mixed Reality and Advanced Display, School of Optics and Photonics, Beijing Institute of Technology, Beijing 100081, China
[e] Departments of Geography and Environmental Management and Systems Design Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 3 December 2019
Received in revised form 26 September 2021
Accepted 3 October 2021
Available online 7 October 2021

Keywords:
Y-Net
Cross-domain image
Image patch matching
Domain Robust Feature Representation (DRFR)
Virtual-real registration
Outdoor Augmented Reality

ABSTRACT

Registering the 2D images (2D space) with the 3D model of the environment (3D space) provides a promising solution to outdoor Augmented Reality (AR) virtual-real registration. In this work, we use the position and orientation of the ground camera image to synthesize a corresponding rendered image from the outdoor large-scale 3D image-based point cloud. To achieve the virtual-real registration, we indirectly establish the spatial relationship between 2D and 3D space by matching the above two kinds (2D/3D space) of cross-domain images. However, matching cross-domain images goes beyond the capability of handcrafted descriptors and existing deep neural networks. To address this issue, we propose an end-to-end network, Y-Net, to learn Domain Robust Feature Representations (DRFRs) for the cross-domain images. Besides, we introduce a cross-domain-constrained loss function that balances the loss in image content and cross-domain consistency of the feature representations. Experimental results show that the DRFRs simultaneously preserve the representation of image content and suppress the influence of independent domains. Furthermore, Y-Net outperforms the existing algorithms on extracting feature representations and achieves state-of-the-art performance in cross-domain image retrieval. Finally, we validate the Y-Net-based registration approach on campus to demonstrate its possible applicability.

1. Introduction

Augmented Reality (AR), a supplement of the real world, overlays virtual objects (augmented components) into the real world. To date, AR has been widely employed in many different fields, including: education [32], medicine [4], remote sensing [25], etc. Virtual-real registration is an important research topic in AR applications. Precisely, virtual-real registration estimate 3D pose information from real-world images so that the virtual object can be accurately placed and integrated with

* Corresponding author.
E-mail address: cwang@xmu.edu.cn (C. Wang).

https://doi.org/10.1016/j.ins.2021.10.022
0020-0255/

the real-world environment [31]. The accuracy of virtual-real registration determines the performance of AR, making it considered to be one of the most fundamental problems of AR [10].

Currently, most AR applications are often restricted to indoor environments [2]; whereas, and few can be applied to outdoor environments. Because there are a number of uncontrolled factors in outdoor environments, such as the large range of outdoor scenes, high complexity of the large-scale data, and dramatic changes in illumination, which make it impractical to pre-place visual fiducial markers for assisting virtual-real registration. These situations make the sensors (e.g., Global Position System (GPS) and inertial and magnetic sensors) suffer from errors of precision, drift, and distortion [33]. Thus, it is challenging for virtual-real registration in outdoor AR applications.

1.1. The proposed virtual-real registration

In this paper, we develop a novel virtual-real registration approach for AR in large-scale outdoor environments. The pipeline framework is shown in Fig. 1. Our work involves three easily acquired image data types: aerial images, camera images, and synthetic images. Aerial images, captured by Unmanned Aerial Vehicles (UAVs) can be efficiently and inexpensively used for reconstructing a large-scale outdoor 3D image-based point cloud by using Structure-from-Motion (SfM) algorithms [37] (Fig. 1(a)). This mechanism provides powerful underlying data support for large-scale outdoor AR applications. Camera images, called "ground camera images", are captured from the ground by mobile devices. In detail, the GPS, Inertial Measurement Units (IMU), and magnetometer in the mobile devices (e.g., smartphone) provide an important clue for locating and estimating the orientation in a 3D environment (Fig. 1(b)). Synthetic images, rendered from the 3D image-based point cloud, are the 2D projections of the 3D environment model. In this paper, we refer to synthetic images as "rendered images". Intuitively, using the camera pose acquired from GPS and IMU as an initial estimation, a rendered image can be synthesized from the same viewpoint with the 3D image-based point cloud. The schematic of the synthetic process is shown in Fig. 1(b).

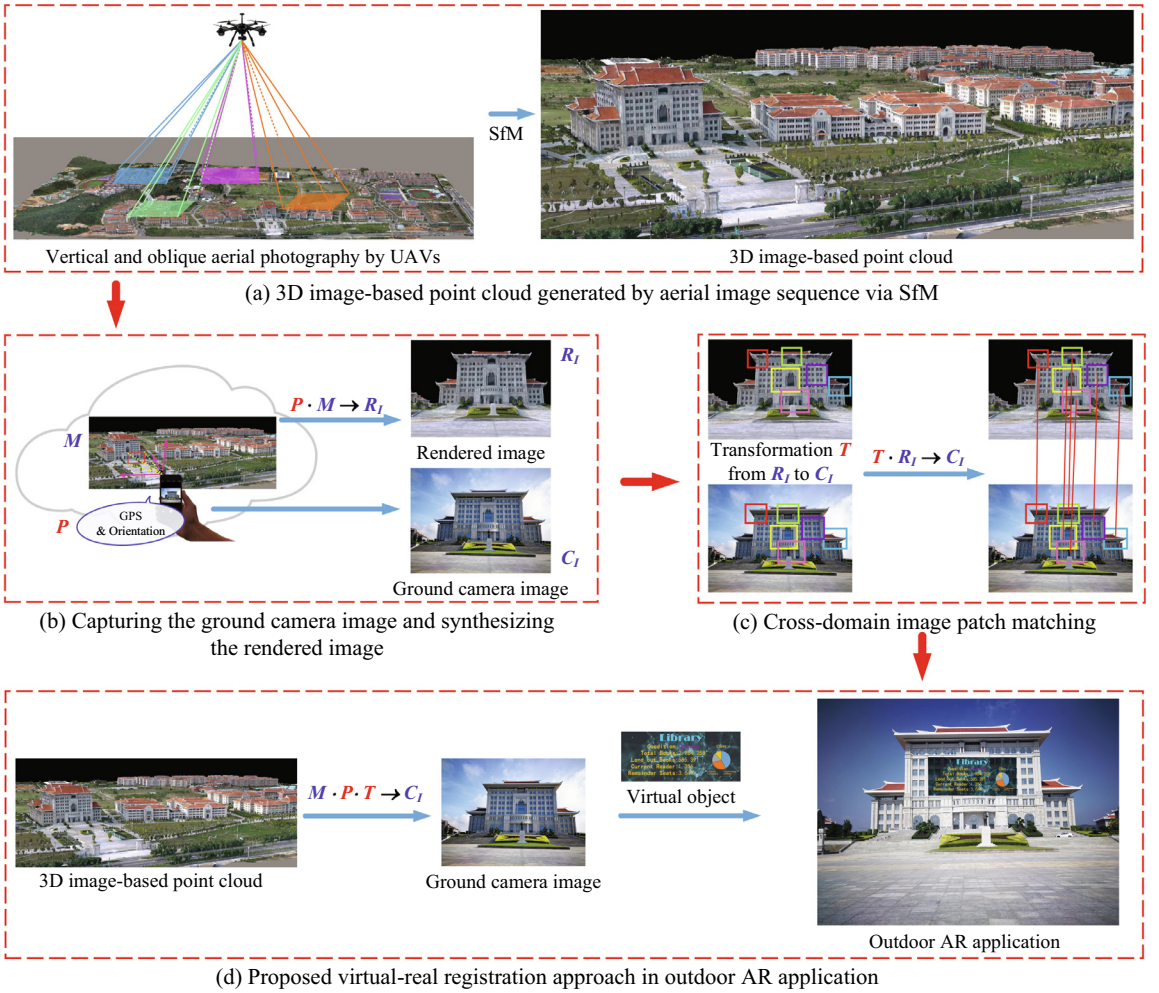


Fig. 1. The pipeline of the proposed virtual-real registration approach for outdoor AR application. The sequence of procedures is (a)→(b)→(c)→(d).

Because of significant differences in imaging mechanisms, local appearance, and the nature of ground camera images and rendered images, we consider these two types of images as "cross-domain images".

The motivation of the proposed virtual-real registration approach is to establish the spatial relationship between 3D and 2D space. Specifically, establishing the relationship between a 3D image-based point cloud and ground camera images. In Fig. 1, we denote the 3D image-based point cloud as M, the ground camera image as C_I, the rendered image as R_I, and the mapping matrix of projecting the 3D image-based point cloud to rendered image as P.Shown in Fig. 1(b)–(d), the detailed formulation of virtual-real registration is derived as follows:

First, P is obtained from the positioning information (GPS and orientation) of the ground camera image. The relationship between the 3D image-based point cloud and the rendered image is given by $P \cdot M \rightarrow R_I$. Second, the transformation matrix T (Fig. 1(c)) projecting the rendered image to the ground camera image is calculated as $T \cdot R_I \rightarrow C_I$. Finally, combining the above two transformation formulas, the spatial relationship between the 3D image-based point cloud (3D space) and a ground camera image (2D space) is $T \cdot (P \cdot M) \rightarrow C_I$ (Fig. 1(d)).

The derivation of the above formulas demonstrates that the spatial relationship between ground camera images and a 3D image-based point cloud is indirectly established through the registration relationship between the ground camera images and the rendered images. Essentially, the key problem of our proposed virtual-real registration approach is to estimate the transformation matrix T (Fig. 1(c)), i.e., the problem of matching the ground camera image and the rendered image. Thus, we consider the registration of ground camera images and rendered images as a retrieval problem. We collect a large number of rendered image patches and store their feature representations into a retrieval database. Then, we search best-matching image patches for a given ground camera image patches in feature representation space.

1.2. Challenges

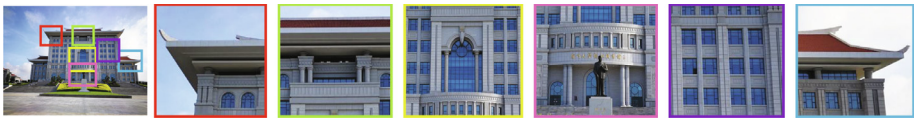
The core problem of the proposed virtual-real registration approach boils down to registrate the ground camera images to the rendered images. The first step is to retrieve the most similar feature representations for a ground camera image patch from the feature representations database consisting of all rendered image patches. It seems to be easy to register these two cross-domain images because they appear to be very similar. However, the following two challenges need to be addressed when registering ground camera images and rendered images:

(1) Gap between cross-domain images.

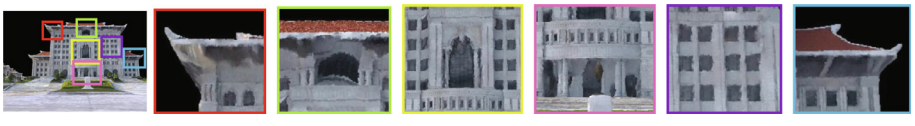
Because the ground camera images and rendered images are cross-domain images, there is a domain gap between them. The significant difference between the two domain images hinders the unity extraction of the feature representations from cross-domain images.

(2) Low quality in the rendered images.

As shown in Fig. 2, quite different from the ground camera images, the rendered images, generally of low quality, inevitably suffer from large distortion, blurred resolution, structural repetitiveness, and occlusions. The low-quality problem comes from the following three sources: i) Vertical and oblique aerial photography are two ways for UAVs to capture aerial images. However, they cannot cover all the terrain details in an outdoor environment, like structures under the eaves, bottoms of buildings, various occlusions, etc. ii) To balance the efficiency and cost of 3D reconstruction for large-scale outdoor environments, the number of aerial photographs cannot be infinitely dense, but limited. So, the resolution of a 3D image-based point cloud generated by the SFM algorithm is limited. iii) The aerial image distortion and pose estimation errors of the camera lead to distortions and errors in the 3D image-based point cloud reconstructed by the SfM algorithm.



(a) Detailed enlargement of ground camera image



(b) Detailed enlargement of rendered image

Fig. 2. The detailed enlargement of a paired corresponding ground camera image and rendered image.

(a) SIFT [24]

(b) DAISY [45]

(c) BRIEF [6]

(d) ORB [35]

Fig. 3. The failed matching results between ground camera images and rendered images by using SIFT [24], DAISY [45], BRIEF [6] and ORB [35]. Each pairs is the corresponding cross-domain images, the left is ground camera image, the right is rendered image. The yellow lines represent the right match and the red lines represent the wrong match.

As motivating examples, Fig. 3 demonstrates some failed matching results between ground camera images and rendered images by using handcrafted keypoints. These keypoints are generated by well-known handcrafted methods such as SIFT [24], DAISY [45], BRIEF [6] and ORB [35]. These results demonstrate that the low quality of the rendered images make these handcrafted keypoint methods inadequate.

1.3. Contributions

Inspired by the following two ideas, we consider registering ground camera images and rendered images by a learning scheme and using the image patch matching-based framework to replace the handcrafted keypoints. 1) SIFT descriptor [24] is calculated in a local area (actually a blob) [22], which is essentially an image patch-based framework. 2) Deep neural networks have been successfully applied to many computer vision tasks. Essentially, our goal is to learn local Domain Robust Feature Representations (DRFRs) for the cross-domain image matching. In detail, the learned DRFRs of cross-domain images should not only be invariant and robust in the two domains, but also should eliminate the cross-domain gap and can be measurable in the same metric space. Additionally, DRFRs should ensure fast retrieval in low-dimensional space, so that, by rapidly retrieving the best-matching rendered image patches for the ground camera image patches. Thus it can be used for AR in large-scale outdoor environments.

In this paper, we propose a novel end-to-end network, Y-Net, whose structural shape resembles the letter "Y", to learn DRFRs for ground camera images and rendered images. The Y-Net, contains two unshared encoders, one shared decoder, and a Spatial Transformer Network (STN) module [16], and it uses an autoencoder as the backbone framework. The two encoders extract feature representations for matching cross-domain image patches. The shared decoder is used to ensure that the generated paired image patches are similar. The STN transforms the input of a ground camera image patch, so the patch aligns with the structure of a rendered image patch. The training objective of Y-Net is to minimize the structure bias between ground camera images and rendered images. During training, the labeled raw matching cross-domain image patch pairs are fed into Y-Net; then, Y-Net is optimized by minimizing a cross-domain-constrained function consisting of content loss and feature consistency loss. The outputs of Y-Net are 128-dimensional compact feature vectors. They are DRFRs that narrow down and eliminate the domain gap for ground camera images and rendered images. In the end, we demonstrate the possibility of applying the proposed virtual-real registration approach in large-scale outdoor environments for AR applications.

The main contributions of this paper are summarized as follows:

(1) A novel virtual-real registration approach is developed for AR in large-scale outdoor environments. We indirectly establish the spatial relationship between 2D and 3D space by using the retrieval solution from the ground camera images for the whole 3D image-based point cloud.
(2) To learn DRFRs, we propose a novel end-to-end Y-Net framework, which achieves state-of-the-art retrieval performance with ground camera images and rendered images. Y-Net simultaneously learns to preserve the content representations and suppresses the domain variations between ground camera images and rendered images.
(3) The proposed cross-domain-constrained loss of Y-Net balances the loss of content and cross-domain consistency of the feature representations for ground camera images and rendered images.

2. Related work

In this section, we briefly review and discuss representative work for virtual-real registration and image patch matching.

2.1. Virtual-real registration

There are many effective virtual-real registration approaches based on visual fiducial markers for AR applications [2]. Such methods usually require a controlled environment (e.g. indoor environment) and must pre-place markers in advance. However, it is not practical to place markers in an uncontrolled outdoor environment. In addition, although these approaches achieve high registration accuracy, they are overly sensitive to the occlusion, unfocused camera, motion blur, uneven lighting, and have difficulty coping with multiple objects or detection at multiple scales or from multiple perspectives, frequently resulting in instability and even failure [33]. Thus, it is very difficult to apply the indoor virtual-real registration methods to outdoor environments.

The most commonly used virtual-real registration approaches in outdoor environments are vision-based methods combined with multiple sensors. They first use sensors (e.g., GPS, IMU) to obtain an initial coarse pose; then, vision-based methods are used to reduce drift and distortion in the sensors [9]. These methods are usually limited to static scenes and rely heavily on the accuracy of multiple sensor fusions.

Recently, deep learning-based methods for virtual-real registration in outdoor environments have been proposed. To realize mobile outdoor AR, [33] uses the deep learning approach to detect objects and infers the spatial relationship for geovisualization. However, it cannot handle poor lighting conditions and incomplete objects. For camera re-localization, other representative works use monocular images, such as PoseNet [19] and its variants. These deep learning methods achieve excellent performance in specific scenes with massive training images. However, the major limitation of these methods is that the training images must contain all the details of the environments, which means they do not work in untrained environments. For application in a new environment, these methods must train a new model, which has very poor generalization.

2.2. Image patch matching

Feature matching, which refers to establishing reliable correspondence between two sets of features, is a critical prerequisite in a wide spectrum of vision-based tasks. Much research has been conducted to find reliable correspondences between two feature sets and removing mismatches from given putative image feature correspondences. For example, LPM [27] maintains the local neighborhood structures of those potential true matches, and formulate the problem into a mathematical model, and derive a closed-form solution with linearithmic time and linear space complexities. As another example, LMR [26] casts the mismatch removal into a two-class classification problem, learning a general classifier to determine the correctness of an arbitrary putative match. RFM-SCAN [17] proposes a spatial clustering-based for robust feature matching. It can adaptively cluster a set of putative matches into several inlier groups in linearithmic time complexity with motion consistency.

In addition, from Section 1.2, it can be viewed that the low quality of rendered images extends beyond the reach of handcrafted descriptors (Fig. 3). Therefore, we only review the related neural networks of image patch matching and the learning feature representations. For a summary of handcrafted feature representations and a comparison of handcrafted and learning feature representations, please refer to [38] for details.

To date, the most common neural networks used for image patch matching tasks and feature representation learning are Siamese and triplet networks. Siamese networks consist of two Convolutional Neural Network (CNN) branches, which can be divided into two categories by a metric network usually composed of fully connected layers [44]. For learning the invariant feature representations, triplet networks are optimized by triplet loss [39]. The details of these networks are described as follows:

Siamese networks, with metric networks, usually treat the matching problem as binary classification by learning the similarity metrics. MatchNet [12] and Deepcompare [48] are typical Siamese networks that use two CNN branches to extract feature representations, which are fed into a metric network to measure the similarity of feature pairs. [48] explores different types of Siamese network-based variants, including the combination of branches (e.g, four branches) and the input form of data (e.g central-surround). Despite their excellent performance, the high computational cost prevents them from being

employed in many applications. Furthermore, the main limitation is that the learned feature representation is different from descriptors (e.g. invariant descriptors). Thus, they cannot be used for retrieval tasks.

Siamese networks, without metric networks, usually use loss function as a constraint for the feature representations extracted by the two CNN branches. DeepDesc [40] uses margin-based contrastive loss and outputs 128-dimensional feature vectors, which can be used as a drop-in replacement for any task involving SIFT. Previous work [46] proposed similar network frameworks and loss functions with DeepDesc. Furthermore, many variants of Siamese network, without metric network, have been proposed, such as DeepCD [47], L2-Net [44], etc. These various networks introduce a strategy for efficient data sampling and construct adaptive loss functions to learn feature representations. These networks learn high-performance feature representations, which can be used for Nearest Neighbor Search (NNS).

Triplet networks use triplet loss to minimize the distance between the anchor and positive images and maximize the gap distance between anchor and negative images [39]. [13] proposed a general-purpose learning method that optimizes the local feature representations. [18] proposed a new mixed-context loss and a scale-aware sampling strategy for the triplet network and achieved state-of-the-art results in patch verification, patch retrieval, and image matching. The performance of the feature representations extracted by the triplet network exceeds that of the feature representations extracted by previous Siamese networks without metric networks. However, the triplet networks are usually difficult to train because the following two reasons. First, the triplet loss makes the triplet networks converge slowly or sometimes even divergence; Second, hard triplet samples are difficult to find from the training data for training the triplet networks.

In summary, Siamese networks without metric networks, and with triplet networks achieve excellent performance for image matching on several datasets, such as Brown [5], Oxford [29], and Hpatches dataset [3]. However, they do not perform well on our task because they cannot learn satisfactory feature representations for ground camera images and rendered images (Results are given in Section 4).

In addition, H-Net and H-Net++ [23], the latest methods for cross-domain image patch matching, have excellent image patch matching performance for ground camera images and rendered images. H-Net incorporates autoencoder into the Siamese network, and feeds the penultimate feature map of the encoder into the metric network. The output of H-Net, binary judgment, has high accuracy; but the main drawback is that the extracted feature representations still cannot be used for retrieval. H-Net++, which replaces the metric network in H-Net with an Euclidean distance constraint, is an improvement over H-Net. The output of H-Net++ is 1024-dimensional feature representations, which can be handled by NNS. However, the feature representations learned by H-Net++ can be used for retrieval, but the retrieval accuracy is inadequate leading to more mismatches in the matching.

3. Y-Net framework construction

This section introduces how the manifold learning, autoencoder and Spatial Transformer Network (STN) inspire us, then describes the Y-Net structure and loss function in detail.

3.1. Manifold learning, autoencoder and Spatial Transformer Network

Manifold learning [43,7] is an approach for non-linear dimensionality reduction. Usually, the data is sampled from a low-dimensional manifold that is embedded in a high-dimensional space. Manifold learning attempts to find a low-dimensional representation of the high-dimensional data.

Ground camera images and rendered images are high-dimensional data, which are captured from the same scene. They have a potential relationship. Essentially, these two types of cross-domain images are different representations of the same object. Inspired by manifold learning, we consider the matched cross-domain images to have the same description in low-dimensional space. Thus, we aim to extract the robust low-dimensional representations for the matched cross-domain images.

Recently, autoencoders have emerged as an alternative to manifold learning [8] for non-linear feature dimensionality reduction. An autoencoder includes an encoder to extract feature representation and a decoder to reconstruct the feature representation. In details, as shown in Fig. 4, the input data, X, is mapped onto the encoding, f, via an encoder, represented as a mapping function $F : F(X) \rightarrow f$; then, the encoding, f, is in turn mapped to the reconstruction, Y, by the decoder, represented as the inverse mapping function, $G : G(f) \rightarrow Y$. The purpose of the autoencoder is to make the distribution of X and Y as similar as possible, that is, $G(F(X)) \approx X$. Essentially, the encoding, f, is the desired low-dimensional feature representation.

Spatial Transformer Network (STN) [16] allows for any transformation as long as it is differentiable. STN is a typically standard CNN followed by a set of fully-connected layers with the required number of outputs, i.e., the number of transformation parameters (e.g., two for translation, six for affine) [1]. Because rendered images are highly distorted, we consider using an STN for the input of ground camera images to align with the structure of the rendered images.

Fig. 4. General autoencoder structure.

Inspired by manifold learning, autoencoder, and STN, we use the autoencoder to simulate non-linear dimensional reduction of manifold learning. Based on the basic autoencoder framework and incorporating STN, we construct a network framework, Y-Net, to learn the DRFRs of the ground camera images and rendered images. Details are given as follows:

3.2. Y-Net structure

The Y-Net framework, based on the autoencoder, assembles two encoders, one shared decoder, and an embedding STN module (The architecture of Y-Net is depicted in Fig. 5). The training data for Y-Net are matching paired cross-domain image patches (See detailed explanation in Section 3.3).

The two encoders in Y-Net have the same structure, which consists of convolution layers with zero padding and max pooling layers without zero padding. Batch Normalization (BN) [15] is used after each convolution, and SeLU [20] is used as the non-linear activation function. Fig. 5(b) shows the structure of encoder, the detailed components of which are given as follows: C(32,5,2)-BN-SeLU-C(64,5,2)- BN-SeLU-P(3,2)-C(96,3,1)-BN-SeLU-C(256,3,1)-BN-SeLU-P(3,2)-C(384,3,1)-BN-SeLU-C(384,3,1)-BN-SeLU-C(256,3,1)-BN-SeLU-P(3,2)-C(128,7,1)-BN-SeLU. The shorthand notation, C(n, k, s), represents the convolution layer with n filters of kernel size $k \times k$ with stride s. P(k, s) denotes the max pooling layer of size $k \times k$ with strides. The inputs of encoder are local image patches which are resized to $256 \times 256 \times 3$. The outputs are feature representations which are 128-dimensional feature vectors.

For the shared decoder in Y-Net (Fig. 5(c)), transpose convolution is used to reconstruct the rendered images from the extracted feature representations. In detail, the 128-dimensional feature representation is first mapped to a 1024-dimensional vector by a fully connected layer before deconvolution. The detailed decoder structure is given as follows: FC(128,1024)-TC(128,4,2)-SeLU-TC(64,4,2)-SeLU-TC(32,4,2)-SeLU-TC(16,4,2)-SeLU-TC(8,4,2)-SeLU-TC(4,4,2)-SeLU-TC(3,4,2)-Sigmoid. The shorthand notation, FC(p, q), denotes the input p-dimensional feature vector map as a q-dimensional feature vector through the fully connected layer. TC(n, k, s) represents the transposed convolution with n output channels of size $k \times k$ and stride s.

In addition, the STN is applied only to one branch: Y-Net branch 2 as described in Fig. 6b) and Section 3.3. Ground camera image patches are the inputs for the STN. Before feeding the ground camera image patches into the encoder, we first use an STN to learn a transformation for those patches. The reason is that the straight line is easier to transform into a deformed line, and conversely, the deformed line is not easily corrected to a straight line. Thus, for an STN, a ground camera image patch and a transformed patch, both with size $256 \times 256 \times 3$, are the input and output, respectively.

In summary, the inputs of the Y-Net are matching paired cross-domain image patches, which are resized to $256 \times 256 \times 3$. The outputs of Y-Net are 128-dimensional feature representations extracted by encoders.

3.3. Loss function

Our goal is to extract the DRFRs for ground camera images and rendered images. First, we assume that the inputs of the Y-Net are pairs of matching cross-domain image patches, then we derive the formula for the loss function. In detail, Y-Net is

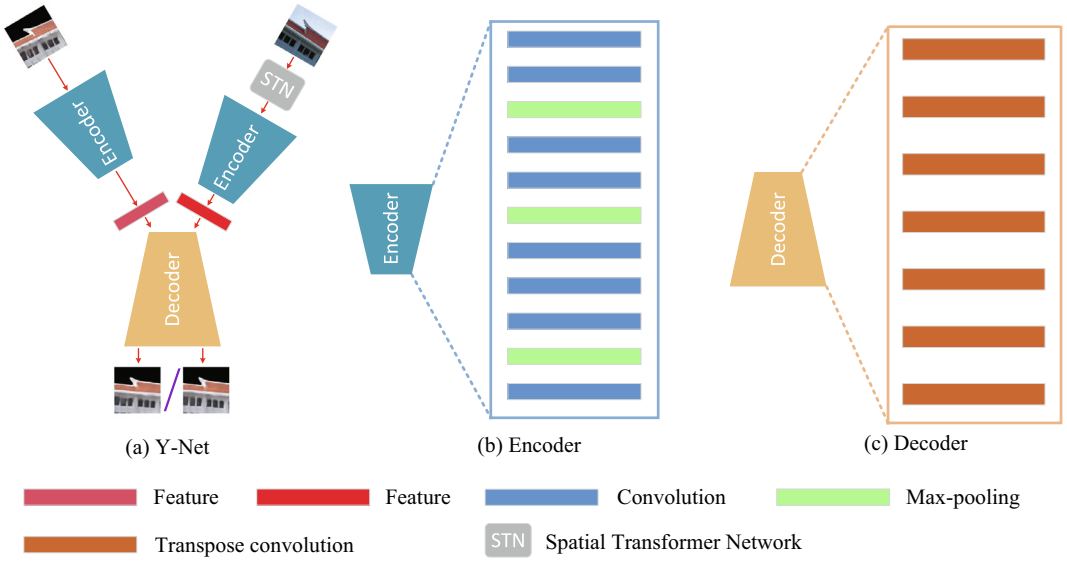
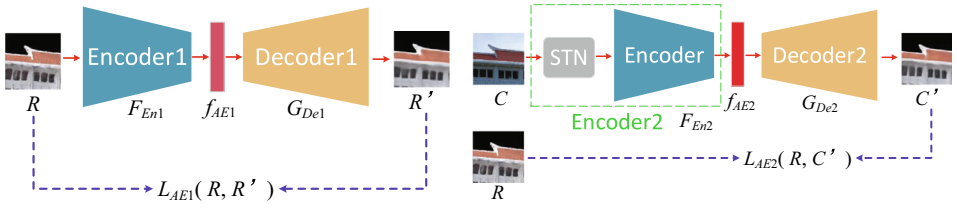


Fig. 5. Y-Net architecture.



(a) Y-Net branch 1 (b) Y-Net branch 2

Fig. 6. The decomposition of Y-Net.

decomposed into two separate parts: Y-Net branch 1 and Y-Net branch 2, which are composed of autoencoder with the same shared decoder, as shown in Fig. 6.

Rendered image patches are fed to Y-Net branch 1 (Fig. 6(a)). Denoting the rendered image patch as R, the corresponding generated image as R', the learned feature representation as f_{AE1}, the encoder as a mapping F_{En1} and the decoder as inverse mapping G_{De1}. Essentially, autoencoder aims to learn a feature representation of R, and expects the reconstructed R' as similar as R. The process is as follows:

$$F_{En1} : F_{En1}(R) \rightarrow f_{AE1} \tag{1}$$

$$G_{De1} : G_{De1}(f_{AE1}) \rightarrow R' \tag{2}$$

such that

$$G_{De1}(F_{En1}(R)) \approx R \tag{3}$$

which is equivalent to

$$R' \approx R \tag{4}$$

The inputs of Y-Net branch 2 (Fig. 6(b)) are ground camera image patches. We denote the ground camera image patch as C, the corresponding generated image as C', the learned feature representation as f_{AE2}, and the decoder as inverse mapping G_{De2}. The STN module is a typical CNN. Thus, we combine the STN with the encoder and denote them as a whole big encoder: named as a mapping F_{En2}. Importantly! Note that we do not make the generated C' similar to C, rather we let C' be similar to R. Thus, the process of Y-Net branch 2 is given as follows:

$$F_{En2} : F_{En2}(C) \rightarrow f_{AE2} \tag{5}$$

$$G_{De2} : G_{De2}(f_{AE2}) \rightarrow C' \tag{6}$$

such that

$$G_{De2}(F_{En2}(C)) \approx R \tag{7}$$

equals to

$$C' \approx R \tag{8}$$

From the above description of Y-Net branches 1 and 2, as expected, the generated images are all similar to R. Thus, by Eq. (4) and Eq. (8), the relationship between R' and C' is given by

$$R' \approx C' \tag{9}$$

In addition, the decoders in Y-Net branch 1 and branch 2 are shared, that is

$$G_{De1} = G_{De2} \tag{10}$$

thus, by Eq. (3), Eq. (7) and Eq. (10), we obtain the following:

$$F_{En1}(R) \approx F_{En2}(C) \tag{11}$$

Then, from Eq. (1), Eq. (5) and Eq. (11), the relationship between the feature representations, f_{AE1} and f_{AE2}, learned by the Y-Net for matching pair cross-domain image patches R and C is given as follows:

$$f_{AE1} \approx f_{AE2} \tag{12}$$

Thus, the above formulas can ensure that the feature representations of paired matching cross-domain image patches R and C extracted by Y-Net are invariant and robust.

However, the above-inferred formulae are all based on the assumption that R and C (the inputs of Y-Net) are matching cross-domain image patches. We argue the assumption is valid for the following reasons. In high-dimensional space, the distribution of the samples is discrete. In Y-Net, the learned feature representations of the matching paired image patches are constrained to be close. Originally, the distribution of the non-matching paired image patches is discrete; thus, Y-Net does not need to constrain them. If the training data contain non-matching paired cross-domain image patches, then Y-Net should constrain them with an empirically set margin, thus making Y-Net no longer end-to-end.

Then, to optimize Y-Net, we introduce and define a cross-domain-constrained loss function, which consists of content loss and feature consistency loss.

Content loss. To make the respective two images, described in Eq. (4), Eq. (8) and Eq. (9), to be similar, we use content loss, which is motivated by the perceptual similarity in pixel space. Thus, to minimize them, we use the pixel-wise mean squared error (MSE) loss as follows:

$$L_{AE1}(R,R') = \frac{1}{NWH}\sum_{n=1}^{N}\sum_{x=1}^{W}\sum_{y=1}^{H}(R_{x,y} - R'_{x,y})^2 \tag{13}$$

$$L_{AE2}(R,C') = \frac{1}{NWH}\sum_{n=1}^{N}\sum_{x=1}^{W}\sum_{y=1}^{H}(R_{x,y} - C'_{x,y})^2 \tag{14}$$

$$L_{GEN}(R',C') = \frac{1}{NWH}\sum_{n=1}^{N}\sum_{x=1}^{W}\sum_{y=1}^{H}(R'_{x,y} - C'_{x,y})^2 \tag{15}$$

where $W \times H$ is the size of the fed image patches, N is the channel of the images. Thus, the content loss is given by

$$L_{Content} = L_{AE1}(R,R') + L_{AE2}(R,C') + L_{GEN}(R',C') \tag{16}$$

Feature consistency loss. As shown in Eq. (12), because the desired cross-domain image patch feature representations are invariant and robust, we use Euclidean distance to minimize the paired feature representations defined as follows:

$$L_{Feature}(f_{AE1}, f_{AE2}) = \frac{1}{K}\sum_{k=1}^{K}(f_{AE1_k} - f_{AE2_k})^2 \tag{17}$$

where K represents the dimensions of f_{AE1} and f_{AE2}.

Finally, to sum up, the total cross-domain-constrained loss is defined as follow:

$$L_{Y-Net} = L_{Content} + \lambda * L_{Feature} \tag{18}$$

where λ is the weight.

3.4. Training strategy

Y-Net is implemented by the PyTorch framework and trained with the GPU of a NVIDIA Tesla P100. In detail, Y-Net is trained with the paired matching cross-domain image patches and optimized by the cross-domain-constrained loss L_{Y-Net} through the RMSprop Optimizer. The learning rate starts at 0.001 and decays with the factor 0.99 for every 4 epochs.

4. Experimental results

In this section, we first describe the cross-domain image dataset used in our experiments; then, we describe several experiments that were conducted to demonstrate the superiority of the proposed Y-Net. In addition, we also demonstrate that the DRFRs, learned by the Y-Net, are robust (invariant and continuous) through the t-SNE visualization and the *cosine* similarity feature representations.

4.1. Dataset

The data in this study include the ground camera images, 3D image-based point cloud, and rendered images, located in Xiangan campus of Xiamen University, China, where is about 3-square-kilometer and with more than 100 buildings.

As seen in the pipeline described in Fig. 1, we use UAVs to capture 30,000+ aerial images on the campus. Then, with the SfM algorithm, we reconstruct the campus as a 3D image-based point cloud, as shown in Fig. 1(a). In addition, we use a mobile phone HUAWEI Mate9 (with a Kirin 960 mobile CPU, 6 GB of memory, Leica Dual Camera with a 12MP RGB sensor and 20MP monochrome sensor, a 5.9-inch screen, a built-in GPS receiver, and IMU sensor) to capture 10,000 + ground camera images. Meanwhile, the corresponding similar viewpoints of the rendered images are obtained.

To create the ideal corresponding ground camera images and rendered images, we capture the ground camera images in open environment. In addition, to obtain more ideal training data for our proposed Y-Net, for the rendered image whose pose

differs greatly from the pose of the ground camera image, we manually adjusted the rendered image to be closer to the pose of the ground camera image. In addition, to make the collected patches more convenient, we did not capture the buildings very close to the camera. If the distance is too short, buildings in the image will be large, and the size of the image patches cannot be well controlled. Several matching cross-domain image pairs are shown in Fig. 7.

Then, to obtain massive cross-domain image patch pairs, we design a semi-manual method, given as follows: (1) We built a segmentation network using the framework of U-net [34], as shown in Fig. 8. Then, we use the LabelMe toolbox [36] to label the buildings in 200 ground camera images, and feed these 200 labeled images into the constructed segmentation network for training. Finally, we perform the trained segmentation network to segment the buildings in the captured ground camera image. The advantage of segmenting the building is to reduce the occlusion interference of trees, pedestrians, and vehicles on image data. (2) Because of the low quality of the rendered images (mentioned in Section 1), it is a challenge to extract meaningful handcrafted keypoints. Thus, we develop a software to manually click the corresponding keypoints for the corresponding cross-domain images. Next, we assume the transformation relationship between matching cross-domain images is perspective transformation. Thus, from the above manually selected corresponding keypoints, the perspective matrix of the matching cross-domain images is calculated. (3) We use the SIFT detector to extract the keypoints only from the segmented buildings of the ground camera images, and remove all SIFT keypoint within 30 pixels from a selected SIFT keypoint in each segmented building image. (4) We extract the ground camera image patch centering on the remaining SIFT keypoint, then, the matching patches in the rendered images are automatically captured through the perspective matrix.

Note that the size of the ground camera and rendered image patches, as the input of Y-Net, must be carefully designed. Ground camera images and rendered images are high-resolution images of approximately 4000*3000 pixels. Thus, if the size of an image patch is too small, information in the image patch is limited. On the contrary, if the size of image patch is too large, instead of a small image patch, the patch nearly becomes a whole image.

In detail, to illustrate how to choose the size of cross-domain image patches, we use a rendered image as an example. As shown in Fig. 9, two typical different locations in rendered image of various sizes (64*64, 128*128, 256*256 and 512*512) are selected. First, it can be observed that rendered image contains huge distortions in some regions. Besides, the image patches in blurred regions, e.g. the image patches of the blue bounding boxes, have a high probability of being selected (Fig. 9(a)). For small sized image patches, such as 64*64 or 128*128 (See Fig. 9(b), first row) these image patches are located in a completely blurred region. Thus, the loss of content information in these small image patches is too large. In fact, these blurred image patches are completely beyond the ability of humans to distinguish; therefore, small blurred image patches are worthless. Second, there are many repeating structures in the buildings, such as red bounding box image patches in Fig. 9(a). It can be viewed that, if the size of the image patches are too small (e.g the second row of 64*64 and 128*128 image patches in Fig. 9 (b)), the content information contained in the image patch is single, which results in many similar image patches, thus leading to mismatch for the matching. In summary, to collect more suitable cross-domain image patches, the sizes of the image patches are between 256*256 and 512*512 pixels.

Based on the above data sampling strategy, we choose 9,000+ corresponding ground camera images and rendered images, which are captured from 90 buildings in the campus, to collect 300,000+ matching image patches and 300,000+ non-matching image patches. Several matching examples are shown in Fig. 10.

4.2. Retrieval performance comparison

To demonstrate the superiority of Y-Net, we compared it with mainstream neural networks which are feature representations extraction. For Y-Net, the training data consists of the collected 300,000+ matching cross-domain image patches. For the compared neural networks, the training data are the collected 300,000+ matching and 300,000+ non-matching cross-domain image patches. We use TOP1 and TOP5 retrieval accuracy to measure the performance. For a fair comparison, we

Fig. 7. The corresponding image pairs of ground camera images (first row) and rendered images (second row).

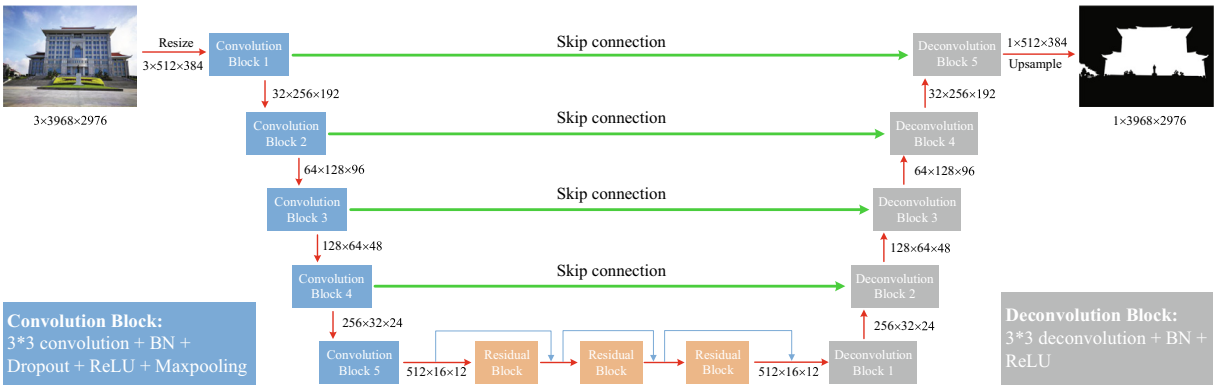


Fig. 8. Our segmentation network based on U-net [34] framework. The Residual Block is from ResNet [14].

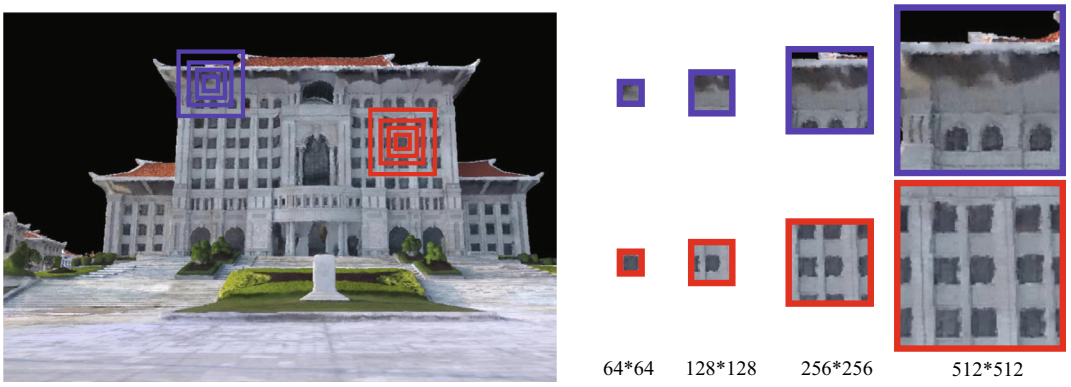


Fig. 9. Various size of patches in two typical locations in a rendered image.

Fig. 10. Matching cross-domain image patch pairs. Top: ground camera image patches. Bottom: rendered image patches.

additionally collected 8,000+ matching cross-domain image patch pairs from the remaining 10+ buildings of the campus as a retrieval benchmark dataset. Specifically, the retrieval benchmark dataset was not used in the training.

Thus far, we have considered the cross-domain image matching problem as a retrieval problem of the learned local image feature representations. Thus, the final application is framed as the following ranking or retrieval problem: given a query ground camera image patch and a repository of rendered image patches, which contains the match one. Then, we aim to rank the rendered image patches according to their relevance to the query, so that the true image patch is ranked as high as possible. Retrieval task is typically approached as follows: feature representation learning neural networks are applied to the query ground camera image patch and the repository of rendered image patches to obtain their feature representations; then, rendered image patches are ranked by sorting the distance from their feature representations to the feature represen-

tation of a query ground camera image patch. TOP1 retrieval is considered successful if the truly matched rendered image patch is ranked within a certain top percentile. If the truly matched rendered image patch is retrieved in the TOP5 ranked results, the TOP5 retrieval is considered successful.

The TOP1 and TOP5 retrieval performance of learned feature representations by Y-Net and compared networks are shown in Table 1. It can be viewed from Table 1, Y-Net achieves state-of-the-art retrieval performance on our cross-domain image dataset and shows significant improvement compared with the comparative networks. DeepDesc [40] and Siam_l2 [48] are Siamese networks with two simple CNN branches that use Euclidean distance as the loss function. DeepCD [47] uses two asymmetric CNN branches. L2-Net [44] establishes some constraints in data sampling strategy and uses intermediate feature maps to assist in optimizing the loss function. H-Net++ [23] incorporates autoencoder into the Siamese network. TNet [21] and DDSAT [18] use triplet loss with different data sampling strategy to learn the feature representations, and DOAP [13] extracts local feature representations optimized by average precision. Thus, whether for simple Siamese networks, Siamese networks with an improved dataset sampling strategy and loss function, or triplet networks, the retrieval performance is inferior for the cross-domain image feature representations learned by these compared networks. In addition, to replace the two encoders in Y-Net, we used VGG 16 [41], ResNet [14] and Inception V1 [42], named as VGG16 Y-Net, ResNet Y-Net and Inception V1 Y-Net, respectively. These are used to compare the performance of the encoder in Y-Net. The results demonstrate that, on retrieval accuracy, Y-Net also outperforms VGG16 Y-Net, ResNet Y-Net and Inception V1 Y-Net.

Furthermore, we show the visualization of some of the top ranked results of the 8,000+ matching cross-domain image patch pairs retrieved from the retrieval benchmark dataset. First, we selected 5 ground camera image patches as the query image patches, the top ranked results of Y-Net are shown in Fig. 11. Second, an image patch (the first line ground camera query image patch in Fig. 11) is used as an example to show the TOP5 retrieval comparison between Y-Net and compared networks, the top ranked results are shown in Fig. 12. For the top ranked results of the compared networks, it can be observed that some of the true matches are found in the TOP5 but not in the TOP1, as shown in Fig. 12. On the contrary, the feature representations extracted by Y-Net have excellent retrieval performance. The shape of the TOP5 ranked results from Y-Net are very similar (Fig. 11). These visualizations demonstrate the superiority of the DRFRs learned by Y-Net.

From the above experiments (Table 1, Figs. 11 and 12), we can conclude the superiority of the Y-Net backbone. Autoencoder is used as the backbone for Y-Net, because it not only extracts the feature representation of the input image, but also constrains the image generated by the decoder to be as similar to the input image as possible. Compared with CNN-based Siamese and triplet network, autoencoder-based network effectively capture the domain-specific information. Encoder works similarly to CNN, but decoder reconstructs the input image, which contains rich domain-specific information.

Finally, we also explore the ablation studies for the STN module in Y-Net and the weight (λ) of the cross-domain constrained loss (Eq. (18)), the results are shown in Table 2. As shown, the performance of Y-Net with a STN module is better than Y-Net without a STN module. When the weight (λ) of feature loss in the cross-domain constrained loss is 0.5, Y-Net achieves the best performance. In addition, compared with the comparable networks shown in Table 1, the retrieval performance of Y-Net, with or without a STN module, is superior to those of the comparable networks, thereby demonstrating the superiority of the Y-Net framework.

From the above ablation study on a STN module, we conclude that the STN must be embedded in the Y-Net branch 2 for the following three reasons: (1) Rendered images, whose detailed structures are quite different from ground camera images, are greatly distorted. (2) Compared with ground camera images, much detailed information is lost in rendered images. (3) There is pixel bias between the matching paired cross-domain image patches. Thus, by using a STN to learn a transformation for ground camera images, we can align the structure with the rendered images, so that the structure bias between ground camera images and rendered images would be minimized.

A detailed explanation of each term in the content loss (Eq. 16) is given as follows: Because the Y-Net branch 1 is a traditional autoencoder, the L_{AE1} (Eq. 13) is intuitive. However, for the L_{AE2} (Eq. 14), the generated image patch, C', is similarly limited to the input of the Y-Net branch 1 (R), not to the input of the Y-Net branch 2 (C). Just as C and R have pixel bias, the generated C' and R also have pixel bias in the same way. If the STN module is not embedded in the Y-Net branch 2, then the MSE in L_{AE2} (Eq. 14) will not be intuitive and will have pixel bias. The L_{GEN} (Eq. 15) is the same as the L_{AE2} (Eq. 14). The generated C' and R' also have pixel bias; thus, the STN module is indispensable in the Y-Net branch 2. In addition, we also attempted to embed STN in the Y-Net branch 1, but failed. Because of the information in rendered images is poor and limited; thus, it is difficult for rendered images to undergo transformation to align with the structures of ground camera images.

Table 1
The TOP1 and TOP5 retrieval accuracy of learned feature representations by Y-Net and comparative networks.

	Y-Net	H-Net++ [23]	DeepDesc [40]	Siam_l2 [48]	L2-Net [44]	DeepCD [47]
TOP1	**0.9562**	0.7124	0.5214	0.3597	0.4684	0.5674
TOP5	**0.9889**	0.8581	0.6563	0.4218	0.5063	0.6382
	TNet [21]	DOAP [13]	DDSAT [18]	VGG16 Y-Net [41]	ResNet Y-Net [14]	Inception V1 Y-Net [42]
TOP1	0.5987	0.6253	0.6059	0.6036	0.4925	0.5236
TOP5	0.6514	0.6849	0.6791	0.7558	0.5593	0.5781

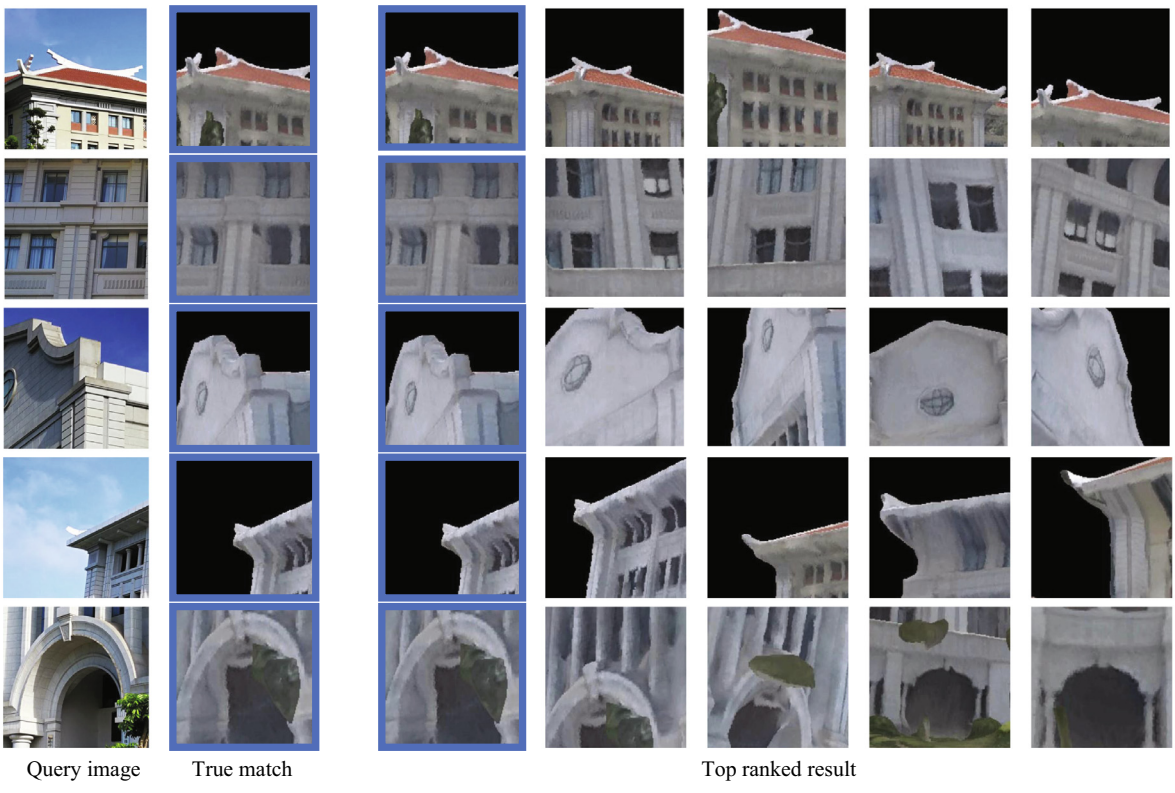


Fig. 11. Ranking result examples of the Y-Net. The corresponding true matches (ground truth) are labeled with blue bounding box.

On the contrary, it is easier for ground camera images to undergo transformation to align with the structures of rendered images.

In addition, it is crucial that both of the two generated images of the Y-Net be similar to the rendered images. Instead of rendered images as the constraint, we tried to use ground camera images, but failed. Because the details of ground camera images are rich, it is difficult for the images generated by the decoder to be as similar as ground camera images. On the contrary, the resolution of rendered images is blurry and lacks some details; therefore, it is easier to generate these images.

Furthermore, if the extracted paired feature representations, f_{AE1}, and f_{AE2}, are similar (Eq. (12)), the reconstructed images, R', and, C', by G (shared decoder) should be similar, too. Thus, Y-Net needs only one G (Eq. (10)). We tried using unshared decoders to train the Y-Net, but the results are worse than those for a Y-Net with shared decoders.

4.3. Invariant and continuous of DRFRs learned by Y-Net

To further demonstrate that the DRFRs learned by Y-Net are invariant and continuous, we use three kinds of visualization strategies to illustrate as follows: (1) Visualizing the image patches generated by Y-Net; (2) Using the T-distributed Stochastic Neighbor Embedding (t-SNE) [28] technique to visualize the learned cross-domain images feature representations; (3) Measuring the *cosine* similarity of the learned cross-domain image patch feature representations.

4.3.1. The visualization of generated image patches

The corresponding generated image patches of ground camera images and rendered images by the shared decoder of Y-Net are shown in Fig. 13. There are two reasons why the generated image patches are blurred. First, when extracting an image feature representation, the autoencoder experiences information loss. Second, and most importantly, Y-Net has a cross-domain-constrained loss constraint (Eq. (18)), when the two autoencoders make their generated image patches as similar as possible to the input for the rendered image patches (Eq. (4), Eq. (8), Eq. (13) and Eq. (14)), the autoencoders also consider the two feature representations to be as similar as possible (Eq. (12) and Eq. (17)). Essentially, this is a compromise. Not only the properties of the respective cross-domain image feature representations must be maintained, but the two cross-domain image feature representations are also constrained as similarly as possible. Thus, the resulting extracted feature representations are compromised, and the generated image patches are blurred.

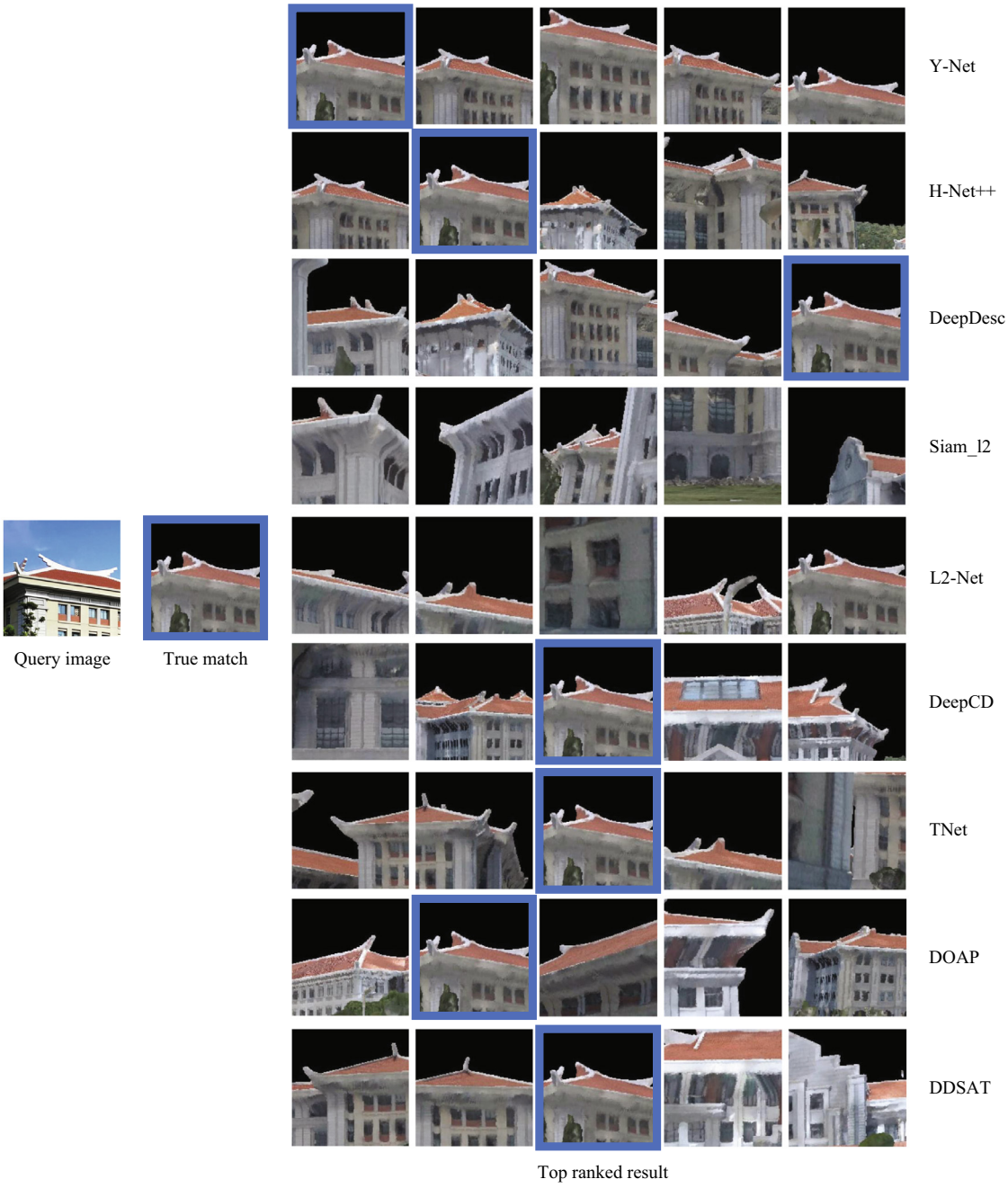


Fig. 12. Ranking result examples of the Y-Net and compared networks. The corresponding true match (ground truth) is labeled with blue bounding box.

In addition, it is seen that the style of generated image patches by ground camera images are transferred to the style of rendered images. Furthermore, image patches generated by ground camera image patches (Fig. 13(a)) are more blurred than image patches generated by rendered image patches (Fig. 13(b)). For example, the image patches generated by rendered image patches are clearer, and the details are retained better. This is because the image patches generated by ground camera image patches are transferred from one domain to another domain, but image patches generated by rendered image patches are still in the same domain.

Thus, the generated image patches are blurred, but image patch pairs generated by matching cross-domain image patch pairs are extremely similar. So combined with the shared decoder in Y-Net, the DRFRs learned by Y-Net are invariant.

Table 2
The TOP1 and TOP5 retrieval accuracy of ablation study for weight and STN module.

	λ	0.1	0.2	0.3	0.4	0.5
TOP 1	Y-Net	0.9423	0.9365	0.9152	0.9189	**0.9562**
	Y-Net w/o STN	0.9302	0.9131	0.9069	0.8985	0.9406
TOP 5	Y-Net	0.9756	0.9702	0.9417	0.9486	**0.9889**
	Y-Net w/o STN	0.9603	0.9581	0.9268	0.9274	0.9708
	λ	0.6	0.7	0.8	0.9	1
TOP 1	Y-Net	0.9416	0.9428	0.9257	0.9109	0.9501
	Y-Net w/o STN	0.9356	0.9277	0.9118	0.9024	0.94
TOP 5	Y-Net	0.9718	0.9706	0.9574	0.9527	0.9829
	Y-Net w/o STN	0.9597	0.9641	0.9382	0.9335	0.9632

(a) Ground camera image patches (b) Rendered image patches

Fig. 13. Image patches generated by Y-Net with 128-dimensional feature representations. Top: Input image patches; Bottom: Generated patches. The corresponding each column image patches in (a) and (b) are matching cross-domain image patches.

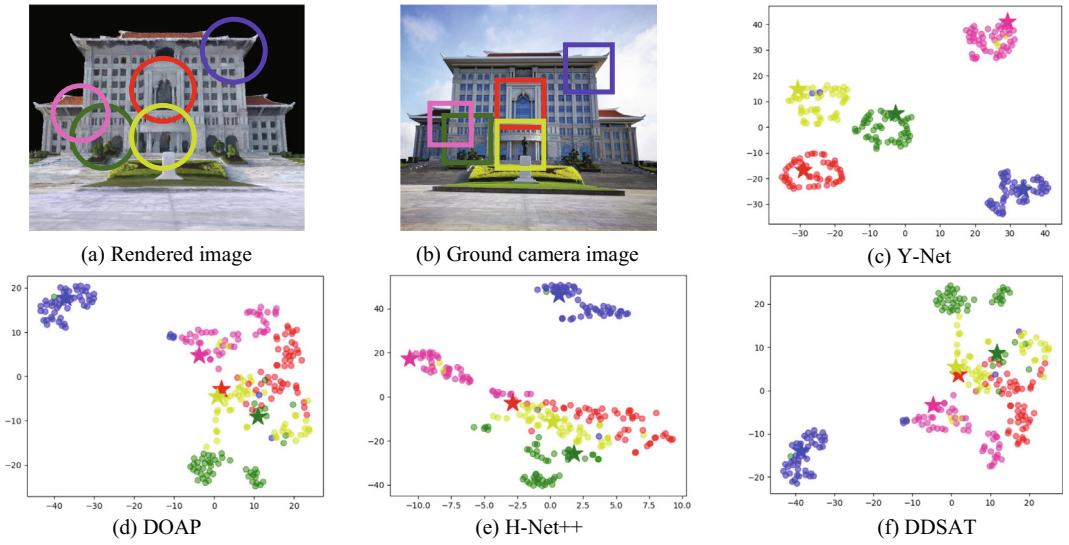


Fig. 14. Visualization of the learned feature representations by t-SNE. (a) is the circled area for selecting rendered patches. (b) is the selected ground camera image patches corresponding the circled area in (a).

4.3.2. The t-SNE visualization

The t-SNE visualizations of learned cross-domain image patch feature representations are shown in Fig. 14. The cross-domain image patches used for visualization are sampled as follows: First, 5 circle areas in the rendered image are selected (Fig. 14(a)); Second, 50 patches from each of the above five circles are randomly collected, respectively; Third, 5 ground camera image patches, corresponding to the above 5 circle areas in rendered image are selected (Fig. 14(b)).

Then, to extract the feature representations, the above 250 rendered image patches and 5 ground camera image patches are fed into the trained Y-Net and comparative networks. Here we only select three networks (H-Net++ [23], DOAP [13] and DDSAT [18]) with better retrieval results in the compared experiment. Next, the t-SNE algorithm is performed to the learned feature representations. As shown in Fig. 14(c), the 250 rendered image patches feature representations learned by the Y-Net

converge into 5 color-coded categories. Meanwhile, the corresponding 5 ground camera image patch feature representations learned by the Y-Net fall into the 5 categories of the aforementioned rendered image patches, respectively, which are labeled by the star symbol. Fig. 14(d)–(f) show the t-SNE visualization of the feature representations learned by H-Net++, DOAP and DDSAT. We can see that the feature representations learned by these three networks are not distinguished clearly.

In addition, some of the points in Fig. 14(c) are clustered into other categories (e.g. two blue points fall into the yellow class, and several yellow points fall into the pink class). The reason is that some regions of a rendered image are hugely distorted and blurred, resulting in difficulty (even beyond the ability of humans) for the Y-Net to learn robust feature representations. In addition, not only does the Y-Net distinguish the blue area, but also do the other networks. This is because the blue circular area in Fig. 14(a) contains the roof and part of the sky; thus, its structure is obviously different from the other circular areas. It is also quite different for the pink circular area in Fig. 14(a), because that area contains the main building, roof, and part of the sky. All the t-SNE visualizations in Fig. 14 are also separate in the pink area. The remaining three circular areas in Fig. 14(a) have overlapping or similar structures. Compared with the t-SNE visualizations, only the Y-Net clearly distinguishes the learned cross-domain image feature representations. Thus, the above t-SNE visualizations demonstrate that the DRFRs learned by Y-Net are robust.

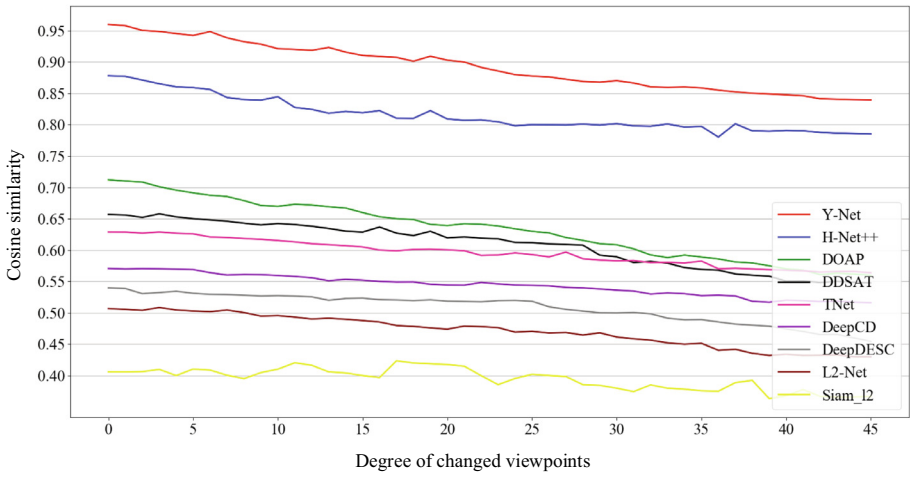
Finally, we conclude the DRFRs learned by the Y-Net are invariant, because, as shown in Fig. 14(c), the stars (ground camera image patch feature representations) fall into their corresponding categories (rendered image patch feature representations). In addition, the DRFRs learned by the Y-Net are also continuous. This is because (1) there is a slight offset between the rendered image patches collected from the same circular area (the patches overlap), and (2) the DRFRs learned by the Y-Net from these overlapping rendered image patches are well clustered.

4.3.3. The cosine similarity of cross-domain image feature representations

To further demonstrate the invariance and continuity of the DRFRs learned by Y-Net, the *cosine* similarity for the cross-domain image feature representations is investigated, as shown in Fig. 15. In detail, first, a fixed viewpoint ground camera image patch is taken (Fig. 15(a)). Then, corresponding rendered image patches with different viewpoints from 0 to 45 degrees are captured (Fig. 15(b)). Second, the *cosine* similarity of these paired cross-domain image feature representations learned by neural networks is calculated. In Fig. 15(c), the X-axis represents the degree of change in the viewpoints between the paired cross-domain image patches. Here, the Y-axis represents the *cosine* similarity between paired cross-domain image patch feature representations.

From the comparison with other comparable networks, e.g. the network represented by the red line in Fig. 15(c), the *cosine* similarity of paired cross-domain image feature representations learned by the Y-Net is the highest, which demonstrates DRFRs learned by the Y-Net are invariant. In addition, the Y-Net (See red line in Fig. 15(c)) shows a decreasing trend.

(a) Fixed (b) Gradually varied rendered image patches



(c) The cosine similarity of (a) and each patch in (b).

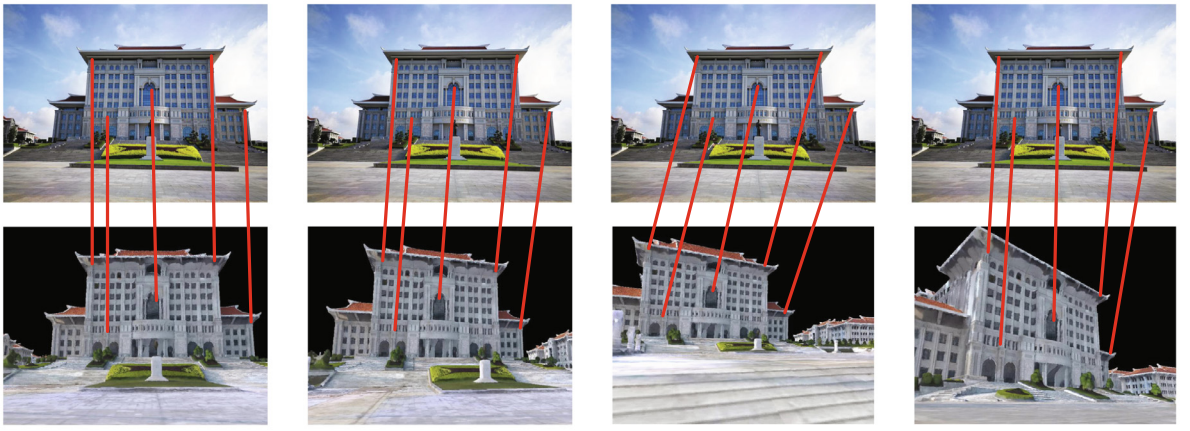
Fig. 15. The cosine similarity of fixed ground camera image patch and rendered image patches.

The reason is the training data (cross-domain image patch pairs) are collected from similar viewpoints, but cross-domain image patch of different viewpoints pairs are never fed into network before. Essentially, the decreasing trend of the Y-Net (red line) also demonstrates the continuity of the DRFRs, because as the viewpoint deviation of the cross-domain image patch pairs increases, the *cosine* similarity of the DRFRs decreases, but still maintains a high level of *cosine* similarity.

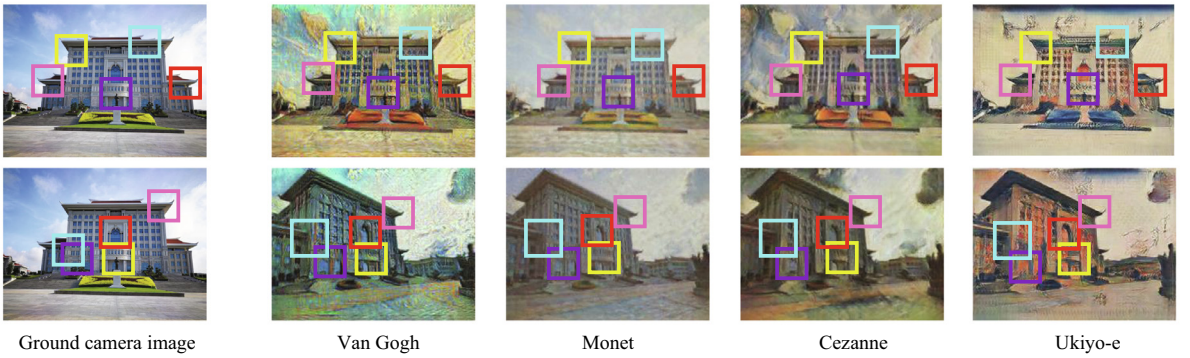
4.4. Matching results

With the DRFRs learned by Y-Net, we designed a matching strategy for ground camera images and rendered images, as follows: (1) We use the trained segmentation network (Fig. 8) to segment the buildings in a ground camera image. (2) We use the detector of SIFT to extract the keypoints only from the segmented buildings of the ground camera images, and remove all SIFT keypoint within 30 pixels from a selected SIFT keypoint in each segmented building image. Then, we extract the ground camera image patches centering at the remaining SIFT keypoints. (3) We randomly select 3,000 points in the corresponding rendered image, and extract the rendered image patches centering at these points. (4) Using the trained Y-Net,

(a) Matching results in varied viewpoint by proposed Y-Net



(b) Point matching results by the center point of matching patches in (a)



(c) Matching results of different style images by proposed Y-Net

Fig. 16. Matching result in different domain images.

the DRFRs of the above collected image patches were extracted. (5) NNS for the DRFRs are performed. Then, only the results with TOP1 retrieval and cosine similarity greater than 0.92 are retained. Finally, using RANSAC [11], significant mismatched results were filtered.

As shown in Fig. 16(a), the matching results of a fixed ground camera image and corresponding rendered images of different viewpoints are labeled with bounding boxes of the same color. Whether it is a cross-domain image with the same viewpoint or different viewpoints, the DRFRs learned by Y-Net achieve excellent matching performance, demonstrating the robustness and generalization of the Y-Net. In addition, as shown in Fig. 16(b), we connect the center points of the matching pair patches in Fig. 16(a). The matching results in Fig. 16(b) are far better than the matching results in Fig. 3. However, these point matching results show that the matching points have pixel offset errors for two reasons. One reason is because that the rendered image patches are obtained by random sampling strategy, the selected points may not be located at the same position with the SIFT keypoints detected from ground camera images. The other reason is that some randomly selected points are relatively close, resulting in similar patches.

Furthermore, to demonstrate the robustness and generalization performance of Y-Net, we tested four different domain images generated by CycleGAN [49], for which the styles are Van Gogh, Monet, Cezanne, and Ukiyo-e. The matching results of Y-Net in these four kinds of style images (See Fig. 16(c)) still achieve excellent performance.

4.5. The generalization of Y-Net

To further demonstrate the generalization of Y-Net, we test the Y-Net on the Brown dataset [5] and another scenario where located in Zhangzhou harbor of Fujian Province, China.

4.5.1. Y-Net tested on Brown dataset

Brown dataset [5] consists of three image patch subsets, Yosemite, Notre Dame, and Liberty, each of which contains more than 400,000 image patches (64×64 pixels) sampled around Difference of Gaussians feature points. For evaluations on this dataset, Y-Net and compared networks are trained on one subset and tested on the other two. We follow the standard evaluation protocol of [5] by using the 100K pairs provided by the authors and report the False Positive Rate at 95% recall (FPR95) to measure performance (the smaller the value of FPR95, the better the performance of the network in image patch matching), the results are shown in the following Table 3.

Specifically, it should be noted that there are minor changes of the Y-Net when using the Brown dataset to train Y-Net. First, Y-Net introduced non-matching image patch pairs when training with Brown dataset, and the feature consistency loss (Eq. 17) of the Y-Net is replaced by margin-based contrastive loss:

$$L_{Feature} = \frac{1}{2} l D_f^2 + \frac{1}{2}(1-l)\{max(0, m - D_f)\}^2 \tag{19}$$

where l is the label of the paired cross-domain image patches (if matched, $l = 1$; otherwise, $l = 0$); $D_f = \|f_{AE1} - f_{AE2}\|$ is the Euclidean distance between feature descriptors f_{AE1} and f_{AE2}, respectively. Such margin-based contrastive loss encourages the feature descriptors of matching pairs to be close and non-matching pairs to be separated by a distance of at least a margin m. In the experiment, the margin m is set as 0.2.

The reason for Y-Net introduced non-matching image patch pairs when training with Brown dataset is as follow: different with the cross-domain images of ground camera images and render images, the images of the Brown dataset are all captured from camera, which are the same domain images. In fact, the distribution of the non-matching paried cross-domain image patches is discrete, so, Y-Net needs not to constrain them. However, the feature representations of non-matching paired image patches of the Brown dataset may be close in the feature space, thus, a margin is required to punish the non-matching image patch pairs, so that the non-matching image patch pairs can be distinguished from the matching image patch pairs. In detail, we use the negative sample acquisition strategy in HardNet [30] to establish the non-matching patch pairs from the Brown dataset to train Y-Net.

Table 3
Patch correspondence verification performance on the Brown dataset [5]. The numbers are False Positive Rate at 95% recall (FPR95).

Train	Notredame	Yosemite	Liberty	Yosemite	Liberty	Notredame
Test	Liberty		Notredame		Yosemite	
DeepDesc [40]	10.9		4.40		5.69	
Siam_l2 [48]	13.24	17.25	6.01	8.38	19.91	12.64
MatchNet [12]	7.04	11.47	3.06	3.80	11.6	8.70
L2-Net [44]	3.64	5.29	1.15	1.62	4.43	3.30
DeepCD [47]	2.87	7.78	6.08	6.65	7.76	3.02
TNet [21]	9.91	13.45	3.91	5.43	10.65	9.47
DOAP [13]	1.54	2.62	0.43	0.87	2.00	1.21
DDSAT [18]	1.79	2.96	0.68	1.02	2.51	1.64
Y-Net (Ours)	**6.02**	**5.70**	**3.77**	**3.47**	**6.63**	**5.68**

From the Table 3, it can be viewed that although the image patch matching performance of our proposed Y-Net on the Brown dataset is not the best, Y-Net also can achieve a good image patch matching result, which also demonstrates the generalization of Y-Net on Brown dataset.

4.5.2. Y-Net tested on Zhangzhou habor scenario

Zhangzhou habor, about 40-square-kilometer, is reconstructed as 3D image-based point cloud by SfM algorithm with UAV aerial image sequences, as shown in Fig. 17. In detail, we choose two typical locations to collect training and testing data, as shown in Fig. 17. First, we created 10,000 paired matching cross-domain image patches from the above two areas to train the pre-trained Y-Net, which trained with the campus dataset in Section 4.1. Second, we selected two pairs of the corresponding ground camera image and rendered image, which are not in the created training data, to match by the DRFRs learned by Y-Net, the matching results are shown in Fig. 18.

5. Outdoor AR application

According to the cross-domain image matching in Section 4.4, we tested the proposed virtual-real registration with AR applications in a large-scale open outdoor environment (i.e. Xiangan campus of Xiamen University which mentioned in Section 4.1). To evaluate the virtual-real registration approach, several representative scenes from the campus were selected to demonstrate registration of virtual objects to real scenes. The distribution of the selected scenes is shown in Fig. 19.

After calculating the spatial relationship between ground camera images and 3D image-based point cloud (spatial relationship of 2D and 3D space is indirectly established through the matching results of ground camera images and rendered images), the virtual real-time library information (virtual object) is registered to the surface of the library building (real scene), as shown in Fig. 20(a)–(e). It can be viewed that the ground camera images of the library building are captured from different viewpoint. The experimental results demonstrate the proposed virtual-real registration approach can register virtual objects to the target scene from different viewpoints. In addition, in Fig. 20(f)–(j), we registered several virtual objects to the real scene, including Welcome label, cartoon animal, cartoon UFO, cartoon spider-man, and NO PARKING sign (For the cartoon animal, UFO, and spider-man, we register them to the real scene only at a specific viewpoint).

The above successful virtual-real registration results demonstrate the possible applicability of the proposed virtual-real registration approach for AR applications in large-scale, open, outdoor environments. However, there are some drawbacks in the results. First, the registered location is biased; for example, the location of the virtual information labels registered in Fig. 20(a)–(e) are different, they are slightly offset; the cartoon animal registered to the road in Fig. 20(g) is still separated by a gap to the ground; the cartoon spider-man in Fig. 20(i) is not tightly attached to the wall, there is still a small gap. Second, after registration, the virtual objects are slightly distorted.

6. Discussion

In this section, we discuss the benefits and limitations of the proposed Y-Net and virtual-real registration approach of AR applications in large-scale outdoor environments.

6.1. Benefits of Y-Net

Y-Net benefits from several components, including the network backbone, the STN module, the training data, the shared decoder, and the loss function. For each component, we briefly discuss its pros and cons based on the experiments and the construction of the Y-Net.

Advantage of the Y-Net backbone. The main backbone of Y-Net, especially the autoencoder backbone, can capture richer information, including image domain information embed in the feature representation.

Benefit of embedding STN module. The inputs of ground camera image patches are transformed by embedding the STN module. The STN module seek to minimize the structure bias between ground camera images and rendered images through a series of transformations.

Importance of matching training data. If the training data of Y-Net contains non-matching cross-domain image patch pairs, the loss function must be redesigned. The most common approach is to introduce an empirically set margin between non-

Fig. 17. The 3D image-based point cloud of Zhangzhou harbor and two enlargements of typical locations.

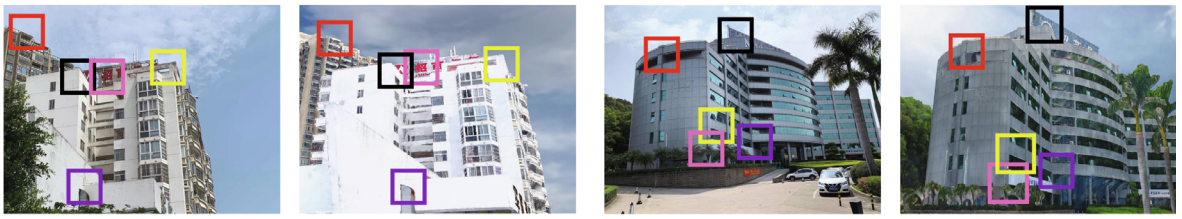


Fig. 18. The matching results of cross-domain images in Zhangzhou harbor.

Fig. 19. The distribution of selected scenes (red labels) on the Xiamen University Xiangan campus. The red dot represents the selected locations.

matching pairs. However, to this margin should be finetuned upon changed datasets, which makes Y-Net not flexible enough.

Crucial constraint in Y-Net. The constraint of the two generated images of the Y-Net to be similar to the rendered images is crucial. This constraint makes the Y-Net to balance the quality of the generated images and the learned feature representations of cross-domain image patches.

Advantage of shared decoder. The shared decoder of Y-Net infers invariant feature representations of cross-domain image patches based on the same constraints of the generated images. In addition, a Y-Net with a shared decoder has fewer parameters than those for unshared decoders, thereby improving efficiency in training.

Effectiveness of content loss. Content loss indirectly makes the cross-domain image feature representations be similar.

Compactness of feature consistency loss. Feature consistency loss is an intuitive loss term that makes the learned feature representations of cross-domain images united in Euclidean space. Thus, the learned cross-domain image features are compact.

6.2. Benefits of proposed virtual-real registration

The experimental results of the AR application demonstrate the possible applicability of the proposed virtual-real registration approach to large-scale, open, outdoor environments. The benefits are summarized as follows:

First, the proposed virtual-real registration approach for large-scale, open outdoor environments is portable and lightweight. Users can use a smart mobile phone to implement AR applications, with the pre-reconstructed 3D image-based point cloud, in large-scale, open outdoor environments. On one hand, with the huge range of uncontrolled outdoor environments, compared with the traditional methods, the proposed approach does not require multifarious sensors. On the other hand, compared with deep learning methods (e.g., PoseNet [19]), our approach does not require a huge amount of training data that contains all the details of the outdoor environments.

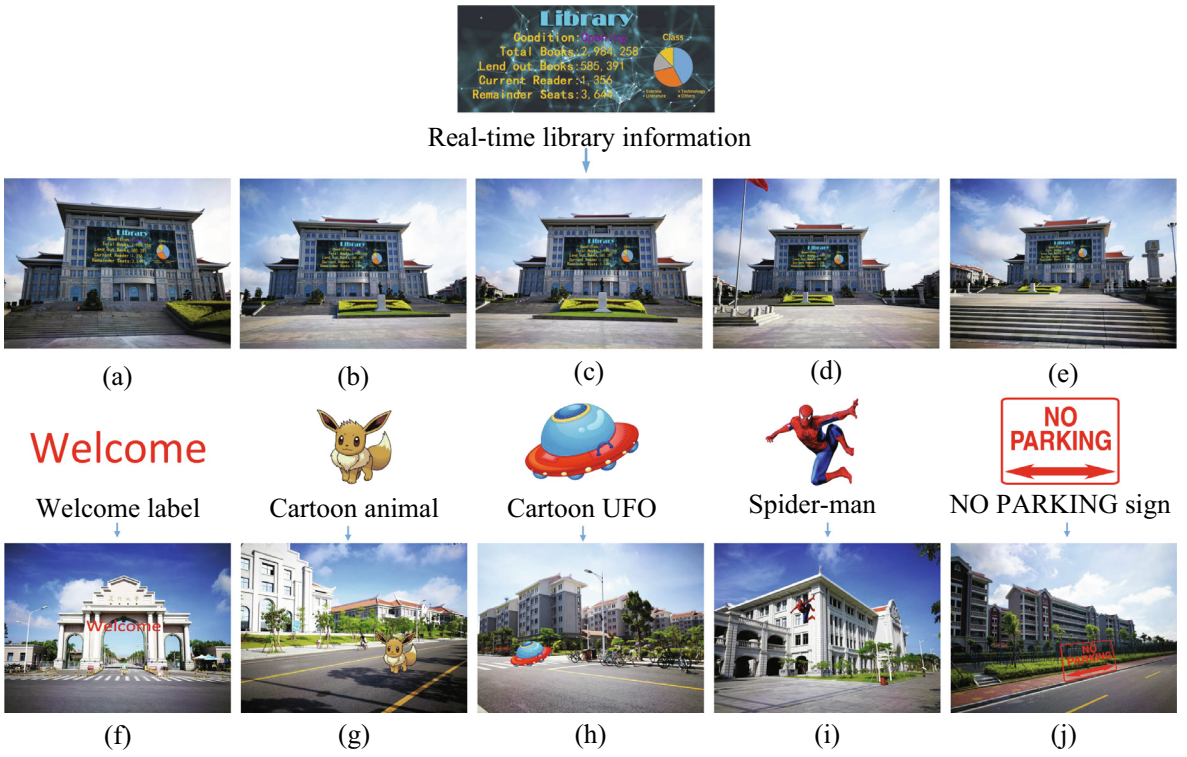


Fig. 20. AR applications by the proposed virtual-real registration approach.

Second, the indirectly established spatial relationship between a ground camera image and a 3D image-based point cloud is intuitive. from the mobile phone are very rough. Because the location and orientation information obtained from the mobile phone are not accurate rough, we match the ground camera images and rendered images. The key insight is that the spatial relationship between the ground camera images and the 3D image-based point cloud can be inferred from their transformation relationship. Such a solution addresses the slight location drift in the GPS and the slight deviation of orientation from the mobile phone.

Third, for the matching problem of ground camera images and rendered images, instead of traditional keypoint-based matching methods, an image patch matching-based strategy, which overcomes the challenges of huge distortion and blurred on rendered images, is used.

Fourth, in this paper, the cross-domain image patch matching problem is essentially feature representations extraction. The feature extraction of Y-Net is a vision-based deep learning approach that learns DRFRs from cross-domain images. These DRFRs are used for fast retrieval in large-scale, open, outdoor AR applications.

Fifth, the proposed virtual-real registration approach is easy to generalize. For a new outdoor environment, only some of the newly extracted cross-domain image pairs are introduced into our pre-trained Y-Net to transfer learning. Then, AR is applied to the new outdoor environment. Indeed, when there are large enough cross-domain image pairs with various styles, the proposed approach is easy to popularize.

6.3. Limitations of Y-Net

The Y-Net also has several limitations. First, if rendered images are largely distorted, or severely occluded in the ground camera images and rendered images, the feature representations learned by the Y-Net are problematic. Poor-quality cross-domain images, such as these, are sometimes beyond human capability. Second, cross-domain image patch matching using DRFRs learned by the Y-Net can only provide coarse matching. If the selected points are very close, the DRFRs will also be very similar; thus, the matching results will be slightly offset.

6.4. Limitations of proposed virtual-real registration

Although we demonstrate the possibility that it can be applied to outdoor environments, our proposed virtual-real registration method also has several limitations. First, AR applications rely heavily on the matching results of ground camera images and rendered images. Second, because of the difficulties in patch collecting, the proposed approach may fail when

the distance between the mobile phone and the building is excessively far or close. If the distance is too far, buildings in the images will be small, and image patches will be much smaller. If the distances are too close, buildings in the image will be large, and the patches will also be much larger. Third, because the rendered images rely heavily on positioning information from the mobile phone, the proposed approach is limited only to open, outdoor environments. If the deviation in positioning is large, we may not be able to obtain a reasonable approximation of the rendered image. In this case, the virtual-real registration approach would fail. Thus, in a dense building environment, sensors are needed to assist positioning.

7. Conclusion

In this paper, we proposed a novel end-to-end Y-Net framework to learn Domain Robust Feature Representations (DRFRs) for matching between ground camera images and rendered images. Using two autoencoders with a shared decoder as the network backbone and embedding an STN module, Y-Net simultaneously preserves the representation of content and suppresses the influence of the independent domain for cross-domain images. We also presented a cross-domain-constrained loss to optimize Y-Net, which balances the loss on content and cross-domain consistency of the feature representations for ground camera images and rendered images. Experimental comparisons clearly demonstrate that (1) DRFRs learned by the Y-Net outperform the other feature representation learning networks in image patch feature-based retrieval and (2) DRFRs are invariant and continuous. We used the learned DRFRs for cross-domain image patch matching and tested the proposed virtual-real registration approach. The AR application experiments demonstrate the possible application of the proposed virtual-real registration of AR in open and outdoor environments. In the future, we will focus on (1) improving the accuracy of image patch matching between ground camera images and rendered images and (2) applying Y-Net to other cross-domain data matching tasks.

CRediT authorship contribution statement

Weiquan Liu: Conceptualization, Methodology, Software, Writing – original draft. **Cheng Wang:** Conceptualization, Resources, Supervision, Funding acquisition. **Shuting Chen:** Validation, Data curation, Writing – original draft. **Xuesheng Bian:** Software, Formal analysis, Visualization. **Baiqi Lai:** Software, Formal analysis. **Xuelun Shen:** Software, Formal analysis. **Ming Cheng:** Data curation, Writing – review & editing. **Shang-Hong Lai:** Formal analysis, Writing – review & editing. **Dongdong Weng:** Formal analysis, Writing – review & editing. **Jonathan Li:** Resources, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by China Postdoctoral Science Foundation (No. 2021M690094); National Natural Science Foundation of China (Nos. U1605254, 41971424); the projects of Educational Project Foundation of Young and Middle-aged Teacher of Fujian Province of China (No. JT180884); the China Fundamental Research Funds for the Central Universities (No. 20720210074).

References

[1] H. Altwaijry, E. Trulls, J. Hays, P. Fua, S. Belongie, Learning to match aerial images with deep attentive architectures, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3539–3547.
[2] B. Bach, R. Sicat, J. Beyer, M. Cordeil, H. Pfister, The hologram in my hand: How effective is interactive exploration of 3d visualizations in immersive tangible augmented reality?, IEEE Transactions on Visualization Computer Graphics 24 (2018) 457–467
[3] V. Balntas, K. Lenc, A. Vedaldi, K. Mikolajczyk, Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5173–5182.
[4] S. Bernhardt, S.A. Nicolau, L. Soler, C. Doignon, The status of augmented reality in laparoscopic surgery as of 2016, Medical Image Analysis 37 (2017) 66–90.
[5] M. Brown, G. Hua, S. Winder, Discriminative learning of local image descriptors, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (2011) 43–57.
[6] M. Calonder, V. Lepetit, C. Strecha, P. Fua, Brief: Binary robust independent elementary features, in: European Conference on Computer Vision (ECCV), Springer, 2010, pp. 778–792..
[7] L. Cayton, Algorithms for manifold learning, Univ. of California at San Diego Tech. Rep 12, 1, 2005..
[8] D. Charte, F. Charte, S. García, M.J. del Jesus, F. Herrera, A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines, Information Fusion 44 (2018) 78–96.
[9] H.P. Chiu, V. Murali, R. Villamil, G.D. Kessler, S. Samarasekera, R. Kumar, Augmented reality driving using semantic geo-registration, in: 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (IEEE VR), 2018, pp. 423–430.
[10] Craig, Alan, B., 2013. Understanding augmented reality. ELSEVIER, 151–183..
[11] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (1981) 381–395.

[12] X. Han, T. Leung, Y. Jia, R. Sukthankar, A.C. Berg, Matchnet: Unifying feature and metric learning for patch-based matching, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3279–3286.
[13] K. He, Y. Lu, S. Sclaroff, Local descriptors optimized for average precision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 596–605.
[14] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
[15] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning (ICML), 2015, pp. 448–456.
[16] M. Jaderberg, K. Simonyan, A. Zisserman, et al, Spatial transformer networks, in: Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS), 2015, pp. 2017–2025.
[17] X. Jiang, J. Ma, J. Jiang, X. Guo, Robust feature matching using spatial clustering with heavy outliers, IEEE Transactions on Image Processing 29 (2019) 736–746.
[18] M. Keller, Z. Chen, F. Maffra, P. Schmuck, M. Chli, Learning deep descriptors with scale-aware triplet networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2762–2770.
[19] A. Kendall, M. Grimes, R. Cipolla, Posenet: A convolutional network for real-time 6-dof camera relocalization, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 2938–2946.
[20] G. Klambauer, T. Unterthiner, A. Mayr, S. Hochreiter, Self-normalizing neural networks, in: Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS), 2017, pp. 972–980.
[21] B. Kumar, G. Carneiro, I. Reid, et al, Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 5385–5394.
[22] K. Lenc, A. Vedaldi, Learning covariant feature detectors, in: European Conference on Computer Vision (ECCV), Springer, 2016, pp. 100–117..
[23] W. Liu, X. Shen, C. Wang, Z. Zhang, C. Wen, J. Li, H-net: Neural network for cross-domain image patch matching, in: International Joint Conference on Artificial Intelligence (IJCAI), 2018, pp. 856–863..
[24] D.G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2004) 91–110.
[25] G. Luchetti, A. Mancini, M. Sturari, E. Frontoni, P. Zingaretti, Whistland: An augmented reality crowd-mapping system for civil protection and emergency management, ISPRS International Journal of Geo-Information 6 (2017) 41.
[26] J. Ma, X. Jiang, J. Jiang, J. Zhao, X. Guo, Lmr: Learning a two-class classifier for mismatch removal, IEEE Transactions on Image Processing 28 (2019) 4045–4059.
[27] J. Ma, J. Zhao, J. Jiang, H. Zhou, X. Guo, Locality preserving matching, International Journal of Computer Vision 127 (2019) 512–531.
[28] L.v.d. Maaten, G. Hinton, Visualizing data using t-sne, Journal of Machine Learning Research 9 (2008) 2579–2605..
[29] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 1615–1630.
[30] A. Mishchuk, D. Mishkin, F. Radenovic, J. Matas, Working hard to know your neighbor's margins: Local descriptor learning loss, in: Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS), 2017, pp. 4826–4837.
[31] C. Panou, L. Ragia, D. Dimelli, K. Mania, An architecture for mobile outdoors augmented reality for cultural heritage, ISPRS International Journal of Geo-Information 7 (2018) 463.
[32] N. Pellas, P. Fotaris, I. Kazanidis, D. Wells, Augmenting the learning experience in primary and secondary school education: a systematic review of recent trends in augmented reality game-based learning, Virtual Reality 23 (2018) 329–346.
[33] J. Rao, Y. Qiao, F. Ren, J. Wang, Q. Du, A mobile outdoor augmented reality method combining deep learning object detection and spatial relationships for geovisualization, Sensors 17 (2017) 1951.
[34] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-assisted Intervention, Springer, 2015, pp. 234–241.
[35] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, Orb: An efficient alternative to sift or surf, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2012, pp. 2564–2571.
[36] B.C. Russell, A. Torralba, K.P. Murphy, W.T. Freeman, Labelme: a database and web-based tool for image annotation, International Journal of Computer Vision 77 (2008) 157–173.
[37] J.L. Schonberger, J.M. Frahm, Structure-from-motion revisited, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4104–4113.
[38] J.L. Schönberger, H. Hardmeier, T. Sattler, M. Pollefeys, Comparative evaluation of hand-crafted and learned local features, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6959–6968.
[39] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823.
[40] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, F. Moreno-Noguer, Discriminative learning of deep convolutional feature point descriptors, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 118–126.
[41] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014..
[42] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9.
[43] J.B. Tenenbaum, V. De Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.
[44] Y. Tian, B. Fan, F. Wu, et al, L2-net: Deep learning of discriminative patch descriptor in euclidean space, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 661–669.
[45] E. Tola, V. Lepetit, P. Fua, Daisy: An efficient dense descriptor applied to wide-baseline stereo, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2009) 815–830.
[46] N.N. Vo, J. Hays, Localizing and orienting street views using overhead imagery, in: European Conference on Computer Vision (ECCV), Springer, 2016, pp. 494–509..
[47] T.Y. Yang, J.H. Hsu, Y.Y. Lin, Y.Y. Chuang, Deepcd: Learning deep complementary descriptors for patch representations, in: Proceedings of the IEEE Conference on International Conference on Computer Vision (CVPR), 2017, pp. 3314–3322.
[48] S. Zagoruyko, N. Komodakis, Learning to compare image patches via convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 4353–4361.
[49] J.Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, 2017, pp. 2223–2232..