Perspectively Equivariant Keypoint Learning for Omnidirectional Images

Yunjian Zhang, Yanwei Liu, Jinxia Liu, Antonios Argyriou, Liming Wang, Zhen Xu and Xiangyang Ji

***Abstract*—Robust keypoint detection on omnidirectional images against large perspective variations, is a key problem in many computer vision tasks. In this paper, we propose a perspectively equivariant keypoint learning framework named OmniKL for addressing this problem. Specifically, the framework is composed of a perspective module and a spherical module, each one including a keypoint detector specific to the type of the input image and a shared descriptor providing uniform description for omnidirectional and perspective images. In these detectors, we propose a differentiable candidate position sorting operation for localizing keypoints, which directly sorts the scores of the candidate positions in a differentiable manner and returns the globally top-K keypoints on the image. This approach does not break the differentiability of the two modules, thus they are end-to-end trainable. Moreover, we design a novel training strategy combining the self-supervised and co-supervised methods to train the framework without any labeled data. Extensive experiments on synthetic and real-world 360° image datasets demonstrate the effectiveness of OmniKL in detecting perspectively equivariant keypoints on omnidirectional images. Our source code are available online at https://github.com/vandeppce/sphkpt.**

***Index Terms*—Omnidirectional images, uniform descriptor, self-supervised learning, perspectively equivariant keypoint**

I. Introduction

WITH the rapid development of 360° vision, the matching between omnidirectional images (ODIs) and perspective images (PIs) has become a fundamental component in many computer vision tasks, including camera calibration [1], viewpoint estimation [2], and scene reconstruction [3]. At the core of the matching problem, the approaches for detecting and describing keypoints on planar PIs have been widely studied, while those on ODIs have not. The main barrier is the complex geometric structure of ODIs, leading to the inefficiency of perspective approaches on ODIs.

In the past few years, several approaches have been proposed to detect keypoints for ODIs. They either applied the conventional keypoint detection approaches on the planar panoramas mapped from the raw ODIs [4], [5], or specially designed algorithms on the spherical surface in allusion to the geometric characteristics of the ODIs [6], [7]. However, the former methods suffer from inaccuracy due to the deformation of the panoramas, while the latter ones have poor robustness to the perspective transformation. The seminal work in [8] focuses on detecting perspectively equivariant keypoints on ODIs, which means a keypoint can resist the perspective projection from ODIs to PIs and also the homography transformation among the PIs. A limitation of this approach is that it generates inconsistent descriptions for keypoints on ODIs and PIs, and thus it needs an additional projection to map the descriptions of ODIs and PIs to the same feature space, which increases the computational cost. In contrast, the studies in [9] and [10] represent the ODIs with geodesic grids, and generate the descriptions on the planar projected grids, which makes it possible to match between the ODIs and PIs. However, for matching PIs with ODIs in the spherical domain, these methods assume that the PI has a fixed focal length and a constant field of view (FoV) that do not conform to the actual situations and it limits the matching performance.

Inspired by the success of deep neural networks (DNNs) in keypoint learning on planar images [11]–[14], in this paper, we propose OmniKL, a framework towards learning perspectively equivariant keypoints for ODIs, addressing the aforementioned limitations of existing approaches. OmniKL consists of a spherical module and a perspective module, and each one uses an individual keypoint detector and a shared descriptor for processing a specific type of image. The perspective module is used to augment the perspective equivariance of the keypoints detected from the spherical module. To ensure the differentiability for the modules so that they can be trained by back-propagation, we propose a differentiable candidate position sorting (DCPS) operation for localizing keypoints, which also overcomes the dependence on the sliding window which is present in previous studies. Since it is unclear which locations in the input images should be labeled as keypoints, we do not rely on any hand-labeled datasets with keypoint annotations, and train OmniKL entirely in a self-supervised scheme. To this end, we devise a self-annotating layer that automatically creates ground truth data on-the-fly by taking advantage of the initial PI annotations. Moreover, we design a score loss that considers the score and number of keypoints, and we combine it with contrastive loss to train OmniKL to detect and describe keypoints accurately. In addition, we propose a co-supervision strategy that enforces the spherical and perspective modules to supervise each other allowing them to be jointly trained. As the descriptions generated by

This work was supported in part by Ningbo Natural Science Foundation under contract 2022J189, the Cooperation project between Chongqing Municipal undergraduate universities and institutes affiliated to CAS (HZ2021015) and NSFC under grant No. 61771469. (Corresponding author: Yanwei Liu.)

Y. Zhang, Y. Liu, L. Wang and Z. Xu are with Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China 100093, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: zhangyunjian@iie.ac.cn; liuyanwei@iie.ac.cn; wangliming@iie.ac.cn; xuzhen@iie.ac.cn).

J. Liu is with the Zhejiang Wanli University, Ningbo 315100, China (e-mail: liujinxia@zjwu.edu.cn).

A. Argyriou is with the Department of Electrical and Computer Engineering, University of Thessaly, Greece (e-mail: anargyr@uth.gr).

X. Ji is with the Department of Automation, Tsinghua University, Beijing 100049, China (email: xyji@tsinghua.edu.cn).

the shared descriptor are uniform and consistent across PIs and ODIs, the matching problems for ODIs vs. ODIs (ovo) and ODIs vs. PIs (ovp) can be solved by directly correlating descriptions from different images.

To summarize, the key contributions of this paper include the following points:

- To the best of our knowledge, we are the first to propose a learnable framework for detecting and describing perspectively equivariant keypoints on ODIs. Considering the deformation of ODIs, we construct the detector for omnidirectional keypoints based on a kernel-adaptive convolution that dynamically adjusts the receptive fields of the kernels according to the geometric and semantic information from the input ODI. We also propose a novel DCPS operation that localizes high-ranking and window-irrelevant keypoints accurately. Furthermore, we devise a spherical cropper that addresses the uneven sampling issue of the ODIs, and construct a shared descriptor across ODIs and PIs to encode the descriptions in the same feature space, providing uniform descriptions for ODIs and PIs.
- We propose a self-supervised strategy for training the complete framework without using any labeled keypoints. Specifically, we construct a novel self-annotating layer that automatically labels the detected keypoints, and the only prior is the camera pose corresponding to the input image, greatly reducing the workload on preprocessing training data. Moreover, in order to jointly train the modules in OmniKL to localize the corresponding keypoints between ODIs and PIs, we devise a co-supervision strategy achieving mutual supervision between them.

II. Related Work

A. Keypoint Detection on PIs

SIFT [15] is a classic algorithm implementing the entire pipeline for keypoints detection and description on PIs. Inspired by SIFT, various subsequent algorithms based on the hand-crafted features are proposed to optimize its efficiency, including Speeded-Up Robust Features (SURF) [16], DAISY [17], KAZE [18], FAST [19], and ORB [20]. These approaches heavily depend on hand-crafted features and are not robust to large angle shift. Due to the highly representative ability of DNNs, researchers attempt to detect keypoints with DNN-based approaches [13], [14], [21]–[23], which achieve better performance compared to traditional methods, and they can be easily extended to describe keypoints by combining with existing descriptors [24]. There are also many end-to-end learning networks implementing both the keypoint detection and description. LIFT [11] used the spatial softmax function to localize the positions of keypoints from the feature map calculated by the detector, and applied another network to generate descriptions from the patches cropped around the keypoints. LF-Net [25] proposed a novel training strategy to learn a non-differentiable pipeline, optimizing the network in a two-branch setup by confining it to one branch, while preserving differentiability in the other. MagicPoint [23] and SuperPoint [26] utilized salient detectors. RF-Net [27] optimized the receptive field of the kernels, and ASLFeat [28] utilized the deformable kernel [29] to extract features accurately. Different from PIs, ODIs contain latitude-varying deformation that ASLFeat can not capture well and thus ASLFeat has limited efficiency for keypoint detection on ODIs.

B. Keypoint Detection on ODIs

The recent advancements in omnidirectional vision applications [30] have urged the need for keypoint detection algorithms towards ODIs. The intuitive idea is to map the raw ODIs to the plane and then apply the conventional planar keypoint detection approaches on them [4], [5]. However, due to the deformation of the projective planar images, these approaches cannot obtain satisfactory performance. To solve this problem, researchers have proposed to detect keypoints on the non-deformed spherical domain. Specifically, Hanse *et al*. [6] developed a SIFT-like algorithm on the sphere to match points between wide-angle images. Arican *et al*. [7] built on the Riemannian geometry to define differential operators on non-Euclidian manifolds, addressing the inaccuracy induced by the sampling operation in [6]. Cruz-Mota *et al*. [8] proposed a framework that combines spherical SIFT and planar SIFT, in which the descriptions for keypoints on ODIs can be matched with those on PIs after a planar projection. Although the above algorithms are effective for their target tasks, they cannot detect the perspectively equivariant keypoints on ODIs with uniform feature descriptions on both the spherical domain and perspective domain. In [9] and [10], the authors proposed SPHORB and BRISKS that generate binary descriptions based on the geodesic subdivision of the ODIs. Although they can project the PIs to the spherical domain for matching them with the ODIs, they assume that the PIs have an unrealistic focal length and FoV, which introduces a deviation to the matching pipeline and limits the matching performance.

Since hand-crafted approaches depend on pre-defining patterns, they cannot fit the ODI features sufficiently. Instead, DNN-based approaches can mine more in-depth features from the given data, thus we work towards designing a data-driven DNN-based framework to capture the perspectively equivariant features of keypoints and generate consistent descriptions for them.

C. Deep Neural Networks on ODIs

Conventional DNNs targeting planar images are inefficient for ODIs because the underlying projection models of PIs and ODIs are different. Various studies have proposed to optimize the planar convolutions to make them adjust to the panoramas projected from the raw ODIs. For example, Su *et al*. [31] proposed a kernel transformed network that can directly map a pre-trained kernel-shared conventional CNN to a kernel-scaled network. [32]–[36] share the similar idea of rectifying the receptive field of the regular kernels to fit the distortion of omnidirectional images. Eder *et al*. [37] extended the icosahedron projection and rendered a spherical image to a set of image grids tangent to the subdivided icosahedron to mitigate the spherical distortion. Besides, a few of studies focus on processing the ODIs in the spherical domain. Cohen

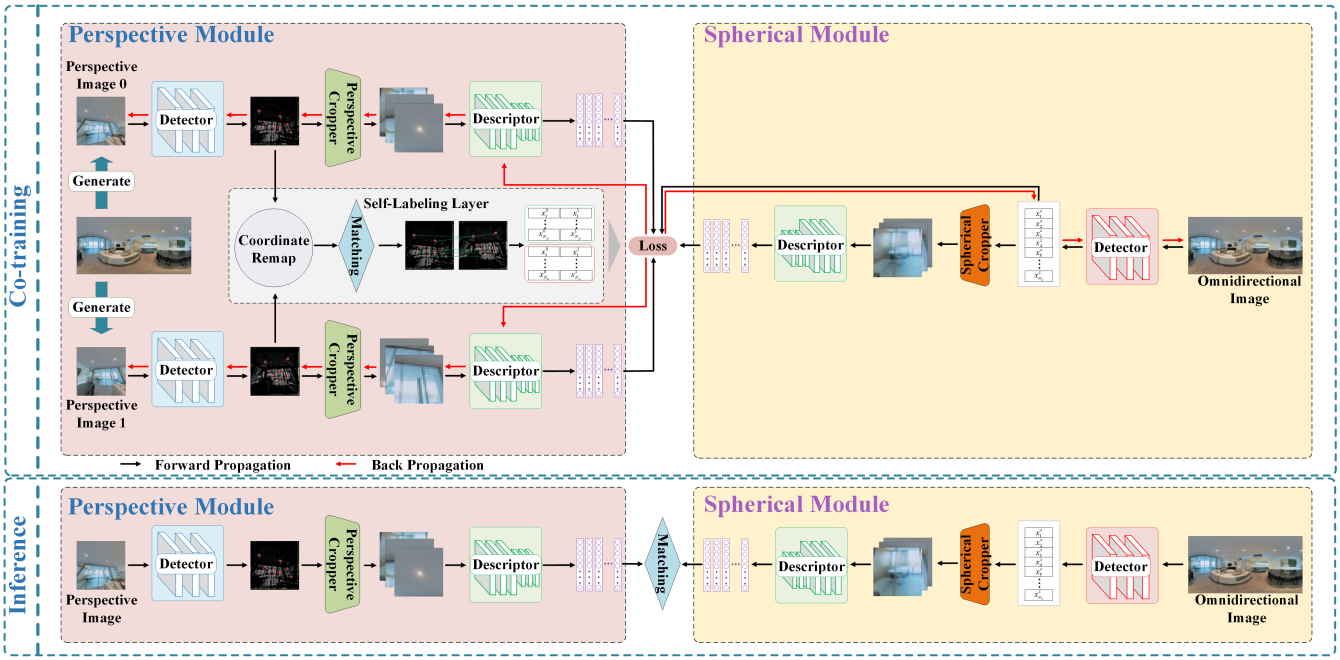


Fig. 1. The architecture of the proposed perspectively equivariant keypoint learning framework OmniKL in the training and inference stage.

et al. [38] proposed a spherical convolution called S2CNN that defines convolutional kernels on the spherical surface. Esteves *et al*. [39] used the spherical harmonic basis to transform the convolution operation to the frequency domain. Although the spherical frequency-domain approaches are more accurate than the schemes that operate in the spatial domain, calculations based on the spherical harmonic functions are very time-consuming, thus we follow the line of spatial domain approaches and design a deformation-aware kernel-adaptive convolution on ODIs.

III. Perspectively Equivariant Keypoint Learning Framework

A. Overview of OmniKL

This paper proposes a two-module framework OmniKL towards learning perspectively equivariant keypoints for ODIs. As shown in Fig. 1, OmniKL consists of a perspective module and a spherical module, detecting and describing the keypoints on PIs and ODIs, respectively. Indeed, the concept "perspectively equivariant" means that a keypoint can resist the perspective projection from ODIs to PIs and also the homography transformation on the PIs, which can be formulated as

$$\mathcal{F}(\phi_{\mathfrak{d}}^{(\mathbb{I})} I) = \phi_{\mathfrak{d}}^{(\mathbb{F})} \mathcal{F}(I), \tag{1}$$

where $\mathcal{F}(\cdot)$ is the detecting process, $\phi_{\mathfrak{d}}^{(\mathbb{I})}$ and $\phi_{\mathfrak{d}}^{(\mathbb{F})}$ are two geometric transformations from the same group (perspective projection, homography transformation, and also viewpoint changes) applied on the input image I. The equivariance property requires the coherence between the two modules. Thus, a co-supervision strategy is devised to train OmniKL. Specifically, in the training stage, OmniKL takes an ODI and two separate PIs rendered from the ODI as its input. Given the corresponding type of image, the two modules are independently trained and enforced to supervise each other in turn until reaching a trade-off in training losses. Based on that, the descriptions generated by the two modules are restricted in the same feature space, providing consistent descriptions for keypoints on ODIs and PIs. Furthermore, to train OmniKL without any labeled data, we construct a self-labeling layer, which maps the coordinates of the PI keypoints to the spherical domain, and labels them automatically. This aspect is described in detail in Section IV. During the inference stage, the ODIs and PIs go through their corresponding module in a single forward propagation, and finally the keypoints and descriptions are obtained.

In Fig. 1, we observe that the architectures of the two modules in OmniKL are similar and they both consist of three key building blocks: a detector for localizing keypoints, a patch cropper that crops a patch around each keypoint, and a feature descriptor encoding every patch to a vector. Indeed, due to the different geometric structure of ODIs and PIs, there exist a few differences between the two modules in their implementations, which will be discussed in the following subsections.

B. Keypoint Detector

The detector was designed to be robust to scale variations, and it consists of a feature extractor for generating the feature map of a given image and a DCPS component for localizing keypoints.

1) Feature Extractor: The architecture of the feature extractor is illustrated in Fig. 2. It first resizes the input image to three scales, and each one is fed into a six-layer neural network that is shared across different scales. The kernel size of the first five layers is 3×3, and that of the final layer is 1×1, while each one is followed by an instance normalization and a relu activation layer. Feature maps for all scales are upsampled and

concatenated together. Finally, a 1×1 convolution is applied to integrate different channels and obtain the feature map for localizing keypoints.

The architecture of the detector was engineered in light of the different properties of ODIs and PIs. The differences between the spherical detector and perspective detector are discussed next.

First, in the planar space, the input image is scaled directly by an interpolation filter. While for the panorama, due to the different sampling rates among different rows, this operation will introduce deformation on the scaled images. Therefore, we project the panorama to the spherical surface, then scale the number of samples on the sphere, and finally take the resulting image of equirectangular projection as the scaled ODI.

Second, the convolution in the perspective detector is the conventional one, while that in the spherical detector is a deformation-aware convolution that is adaptive to the spherical geometry. This is a common way that existing approaches follow so as to rectify the receptive field of the convolutional kernels according to the distortion level of different positions on the equirectangular projection ODI. However, they either interpolate the spherical features in the 2D plane [33] [35] or have to calculate the kernel shape for each central pixel on the sphere [34]. Other studies like [32] calculate the deformable kernels [29] [40] using the geometric knowledge of ODIs, which reduces the number of parameters when compared with other approaches. However, due to the intrinsic property of the ODIs, the receptive fields of the deformable kernels on them are non-convex. This reduces the overlaps of the receptive fields of the neighboring kernels, limiting the capability of the model to capture related information. Considering that the receptive fields in human perceptual systems are convex [41], [42], and current DNNs are often built to mimic the behavior of the human brain, we propose to calculate the convex-hull of the deformable kernel and then re-sample kernel to cover the region surrounded by the convex-hull. In addition, considering the varying contributions of different regions, we do not hard encode the shape of the kernels. Instead, we train the model so as to fine-tune the shape of the convex-hull, which leads to dynamic receptive fields that are adaptive to the region to be convolved.

Specifically, considering an ordinary planar kernel $\mathcal{P}$ of size $r_w \times r_h$, denoted as

$$\mathcal{P} = \{(p_x, p_y)^k\}^{k=1,\dots,k_n}, \tag{2}$$

where k_n is the number of the points in the kernel, given the location (v_θ, v_ϕ) on the ODI, the center of the kernel is aligned to it and then the positions of the elements in the kernel are projected to the sphere using the rectilinear projection [43]. Next, the spherical kernel is projected to the panorama with the equirectangular projection

$$(p_x, p_y) = ((\phi/2\pi + 0.5)W_p, (-\theta/\pi + 0.5)H_p), \tag{3}$$

where (p_x, p_y) is a position on the projected kernel, and W_p and H_p are the width and height of the panorama, respectively. Subsequently, we collect all the points on the projected kernel, and calculate the convex-hull for them. To ensure the kernel adjusts its receptive field adaptively, a scaling factor α_c^k and a biasing vector $(\beta_{x,c}^k, \beta_{y,c}^k)$ are defined to allow fine-tuning of the position of each vertex of the convex-hull. We define

$$\mathcal{C} = \{\alpha_c^k x_x^k + \beta_{x,c}^k, \alpha_c^k x_y^k + \beta_{y,c}^k\}^{k=1,\dots,k_c} \tag{4}$$

to be the adjusted convex-hull, with (x_x^k, x_y^k) being the k-th point on it, and k_c is the number of vertexes of the convex-hull. As the adjusting process does not break the differentiability of the model, α_c^k and $(\beta_{x,c}^k, \beta_{y,c}^k)$ can be optimized with gradient descent. Note that the deformed kernel obtained by projection is non-convex, thus only some of the points on it are considered for calculating the convex-hull, and as a result $k_n \neq k_c$. Next we define a re-sampling operation $\mathcal{S}_c$ with the bilinear sampling function, which is used to re-sample the original kernel to the deformed region covered by the convex-hull:

$$\mathcal{S}_c : \mathcal{P} \rightarrow \mathcal{C}. \tag{5}$$

The deformed kernel K_c is denoted as

$$K_c = \mathcal{S}_c(\mathcal{P}) = \{(x'^{(k)}_x, x'^{(k)}_y)\}^{k=1,\dots,k_n} \tag{6}$$

Finally, the convolution result on (v_θ, v_ϕ) is

$$\begin{aligned} F(v_\theta, v_\phi) &= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} f_m(i+\phi, j+\theta)\cdot K_c(i,j)\cdot\chi(i,j) \\ \chi(i,j) &= \begin{cases} 1, 0 \le i \le r_w, 0 \le j \le r_h \\ 0, \text{otherwise} \end{cases} \end{aligned}, \tag{7}$$

where f_m is the input feature map and F is the resulting feature map.

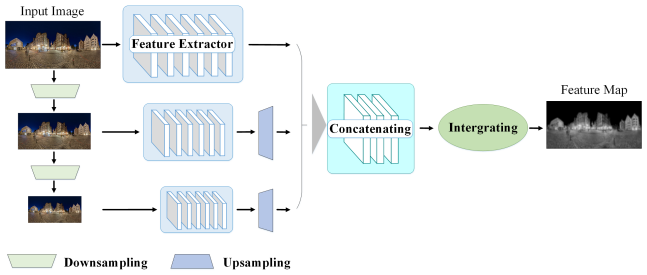


Fig. 2. The structure of feature extractor in the detector.

The convolution approach we propose above makes our model adaptive to any ODI in the equirectangular format regardless of the camera intrinsics or sensors. Even for catadioptric or fisheye images, in general, our approach can still process them if they are converted into equirectangular images.

2) Differentiable Candidate Position Sorting Operation: Previous studies geared towards extracting keypoints mainly relied on performing non-maximum suppression or spatial softmax with sliding windows. However, these methods limit the positions of the keypoints, thus it is tough for them to balance the number of false keypoints and collective keypoints: Increasing the window size can reduce the number of unnecessary keypoints, but it also reduces the total number of keypoints; decreasing window size leads to more keypoints, but it also results in more false predictions and introduces heavy training burden. The main reason is that the points

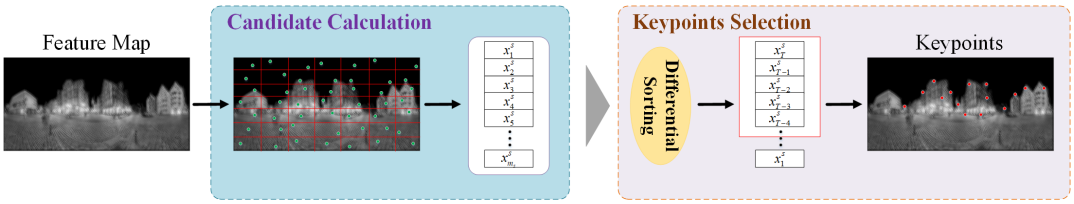


Fig. 3. The structure of the DCPS operation.

from all windows are treated equally during training, thus the detector does not gain the ability to determine the globally important keypoints.

To address the above problem, we propose a DCPS operation, which makes the network learn to rank the keypoints. As shown in Fig. 3, we first apply a sliding window with a small size on the feature map to calculate candidate positions for keypoints. Given a window w_i of size $N_w \times N_w$, the candidate keypoint position in it is calculated by

$$\Theta(w_i) = (x_i, y_i) = \sum_{u,v}^{N_w} (W \odot m_i, W^T \odot m_i) + c_w, \tag{8}$$

where W is a kernel of size $N_w \times N_w$ with index values $j = 1 : N_w$ along its columns, (u, v) denotes the coordinate of the pixel, $\odot$ is the point-wise product operation, c_w is the top-left corner coordinate of the window, and m_i is the normalized activation value in w_i, defined by

$$m_i(u, v) = \frac{e^{w_i(u,v)}}{\sum_{j,k}^{N_w} e^{w_i(j,k)}}. \tag{9}$$

The candidate keypoints are sorted in terms of their activation values (referred as scores) to reserve only the globally meaningful points. Consequently, only those keypoints with higher scores take part in optimizing the detector. The DCPS operation (denoted as $\Phi(\cdot)$) is captured as:

$$\Phi(F) = T_K(S([\Theta(w_1), \Theta(w_2), ..., \Theta(w_M)])), \tag{10}$$

where F is the feature map, T_K is a function only reserving the top-K elements, S is the sorting operator, and M denotes the number of the sliding window. In order to preserve the differentiability of the operation, we use differentiable sorting [44] to sort the candidates, and the sorting operator is projected as a linear program over the permutahedron:

$$P_Q(\mathbf{z}, \boldsymbol{\xi}) = \underset{\mu \in \rho(\boldsymbol{\xi})}{\arg\max} \langle \mathbf{z}, \mu \rangle - Q(\mu) = \underset{\mu \in \rho(\boldsymbol{\xi})}{\arg\min} \frac{1}{2} ||\mu - \mathbf{z}||^2, \tag{11}$$

where $\boldsymbol{\xi} = [\Theta(w_1), \Theta(w_2), ..., \Theta(w_M)]$, $\mathbf{z}$ is a permutation of $\boldsymbol{\xi}$, $\rho(\boldsymbol{\xi})$ is the convex hull of permutations of $\boldsymbol{\xi}$, and $Q(\mu) = \frac{1}{2}||\mu||^2$ is a convex regularization. Furthermore, the projection is reduced to the isotonic optimization for faster computation:

$$P_Q(\mathbf{z}, \boldsymbol{\xi}) = \mathbf{z} - v_Q(\mathbf{z}_{\delta(\mathbf{z})}, \boldsymbol{\xi})_{\delta^{-1}(\mathbf{z})}, \tag{12}$$

in the above v_Q is the isotonic regression function, δ is the argsort operation, and δ^{-1} is its inverse. Then the differential of the sorting operator can be calculated by its Jacobian matrix.

We note the candidates whose scores are larger than the K-th item of $S(\boldsymbol{\xi})$ (denoted as $S_K(\boldsymbol{\xi})$) in a vector $\boldsymbol{\kappa} \in \{0, 1\}^M$

$$\kappa_i = \begin{cases} 1, \Theta(w_i) \geq S_K(\boldsymbol{\xi}) \\ 0, others \end{cases} \tag{13}$$
$$\boldsymbol{\kappa} = \{\kappa_1, ..., \kappa_M\}$$

Finally we retain the candidates with $\boldsymbol{\kappa}_i$=1 as the keypoints

$$\Phi(F) = T_K(S(\boldsymbol{\xi})) = \boldsymbol{\kappa} \odot \boldsymbol{\xi}. \tag{14}$$

C. Patch Cropper

To extract the feature representations for describing keypoints, DNN-based approaches usually consider the encoding vector of the patch cropped around each keypoint as its corresponding description. An important requirement for the cropper in the DNN-based framework is its differentiability, which ensures the end-to-end property of the learning pipeline.

The perspective cropper is implemented with the bilinear sampling, which is a differentiable operation and applied directly on the input image to generate fixed-size patches around the keypoints. However, the sampling on the panorama is uneven, and it will introduce deformation when the patches are directly cropped from the panorama. Therefore, we design a novel spherical cropper that crops the patch in the perspective domain. As shown in Fig. 4, the panorama is first projected to the spherical domain, and then a patch is obtained by projecting a part of spherical surface around a keypoint into the perspective plane with a differentiable rectilinear projection. Due to the narrow view of the patches, they suffer from less deformation than the panorama.

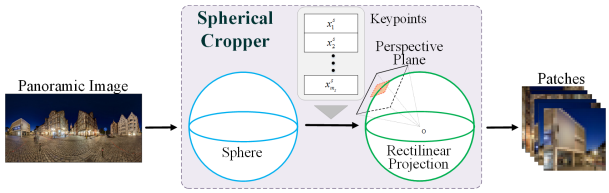


Fig. 4. The procedure to crop patches for the panoramic images.

D. Feature Descriptor

The descriptions of the keypoints are expected to be invariant to the image transformations, especially for those of the ODIs, whose patches are rotated with the rotation of the ODIs. This means that they remain unchanged during the transformations, and they can be formulated as

$$\mathcal{D}(\theta_{\mathfrak{I}}^{(\mathbb{I})} I) = \mathcal{D}(I), \tag{15}$$

where $\mathcal{D}(\cdot)$ is the function for generating descriptions from the input images, and $\theta_{\mathfrak{I}}^{(\mathbb{I})}$ is the geometric transform on them.

The architecture of the descriptor is presented in Fig. 5, comprising three convolutional layers and a pooling layer, followed by two fully connected layers. Considering the natural advantage of group convolution [45] in rotation-equivariance, we extend it to our descriptor to make the convolutions operate on the SO3 space, ensuring the descriptions to be robust to rotation. After the last convolutional layer, a MaxPolling3D layer is inserted so as to integrate the group channels. Finally, the 128-dimensional descriptions are obtained using the fully connected layer with 128 units. As the pooling layer and the fully connected layer are invariant to image transformations, the equivariant features obtained by the convolutional layers are encoded to invariant descriptions.

Since the patches for PIs and ODIs are all obtained in the perspective domain, there is no need to devise a descriptor that is specific for ODIs, instead, a shared descriptor is enough to encode the patches to descriptions. Therefore, the descriptions of the ODIs and PIs are in the same representation space, and they can be directly matched without any projection, something that simplifies significantly the matching process when compared to [8].

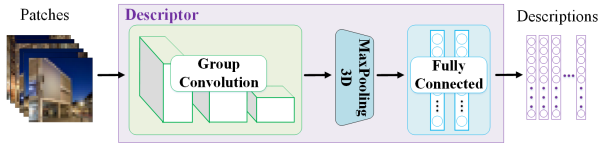


Fig. 5. The structure of the descriptor.

IV. Training Strategy

To capture the perspectively equivariant keypoint from ODIs, OmniKL is expected to achieve two matching objectives: ovo and ovp. Because both the perspective and spherical modules in OmniKL are end-to-end trainable, we propose to "preheat" OmniKL by pre-training them respectively. The training procedure includes three stages. In the first stage (Stage 1), we use PIs to pre-train the perspective module to make it learn robust keypoints on PIs without any dependency on the spherical module. Second (Stage 2), the pre-trained perspective module is used to help train the spherical module, in other words, the spherical detector is trained to select keypoints as close as possible to the output of the perspective detector. After pre-training the two modules we jointly optimize them with a rotational co-training strategy in the third stage (Stage 3), in which the two modules supervise each other to learn perspectively equivariant keypoints on ODIs.

A. Pre-training the Perspective Module

1) Self-labeling: To warm up the perspective module, we define a Siamese network that receives a pair of PIs $\{I^0, I^1\}$ and their corresponding camera pose information (named PI annotations) $\{(v^0, f^0), (v^1, f^1)\}$ as input, where $v^o (o = 0, 1)$ and $f^o (o = 0, 1)$ are the PI center position and FoV for rendering image I^o. As shown in the left part in Fig. 1, each branch in the network includes the complete processing pipeline, and independently performs the forward propagation. The two branches share the same weights and are related with a self-labeling layer for automatically annotating the keypoints. In the self-labeling layer, the coordinates of the perspective keypoints are projected to the spherical space with the PI annotations using the rectilinear projection [43]. Then the keypoints from different PIs are matched in terms of the distances between their spherical coordinates, that is,

$$\begin{aligned} F_n &= \{(x_i^0, d_i^0), (x_j^1, d_j^1)\} \\ &= \{X_n^{s0}, X_n^{s1}, D_n^0, D_n^1\}, n \in \{1, ..., N\} \\ j &= \underset{j \in \{1,...,N\}}{\arg\min} \, [Dis(x_i^0, x_1^1), ..., Dis(x_i^0, x_j^1), ..., Dis(x_i^0, x_N^1)], \end{aligned} \tag{16}$$

where $x_i^o (o = 0, 1)$ and d_i^o are the coordinate and description of the i-th keypoint from the image I^o, $Dis(\cdot)$ is the distance function, N is the number of keypoints per image, F_n is the n-th pair of matching keypoints, and (X_n^{s0}, X_n^{s1}), (D_n^0, D_n^1) are the coordinates of coupled keypoints mapped onto the sphere and their descriptions. Considering the geometry of the sphere, the distance function of $X_i^{s0} = (\theta_i^0, \phi_i^0)$ and $X_j^{s1} = (\theta_j^1, \phi_j^1)$ is defined as the haversine distance

$$\begin{aligned} Dis(X_i^{s0}, X_j^{s1}) = 2 \arcsin(\sin^2(\frac{\phi_i^0 - \phi_j^1}{2}) + \\ \cos\phi_i^0 \cos\phi_j^1 \sin^2(\frac{\theta_i^0 - \theta_j^1}{2}))^{\frac{1}{2}}. \end{aligned} \tag{17}$$

Finally, the pairs of keypoints and their corresponding patches are labeled as positive or negative with a distance threshold. If the distance is less than an empirical threshold, i.e. 1.0, it is labeled as positive, that is $l_n = l_i^0 = l_j^1 = 1$, otherwise it is denoted as negative with $l_n = l_i^0 = l_j^1 = 0$, where l_n is the label of the n-th pair, and $l_i^o (o = 0, 1)$ is the label for the i-th keypoint from the image I^o. Therefore, the self-labeling layer creates ground truth data by considering only the initial PI annotations of the input image pairs.

2) Pre-training Loss Functions: After labeling the description pairs, we calculate the loss function combining the contrastive loss and score loss to optimize the perspective module Specifically, we introduce a multi-task loss function with weighting parameters λ_c and λ_s:

$$L(\mathbf{x}^0, \mathbf{x}^1) = \lambda_c L_c(\mathbf{F}, \mathbf{l}) + \lambda_s L_s(\mathbf{x}^0, \mathbf{p}^0, \mathbf{l}^0, \mathbf{x}^1, \mathbf{p}^1, \mathbf{l}^1), \tag{18}$$

where $\mathbf{x}^o (o = 1, 2)$ is the collection of the keypoints on I^o, $\mathbf{p}^o$ and $\mathbf{l}^o$ are the corresponding collections of the scores and labels for the keypoints in $\mathbf{x}^o$, $\mathbf{F}$ and $\mathbf{l}$ are the collections of the matching keypoints and their labels, L_c is the contrastive loss function operating on the pairs of the keypoints and optimized over their descriptions, and L_s is the score loss function which uses the activations of the keypoints from positive pairs on the feature maps to optimize the detector.

The contrastive loss over the pairs of descriptions is:

$$\begin{aligned} L_c(\mathbf{F}, \mathbf{l}) = &\frac{\sum_{n=1}^{N} l_n ||D_n^0 - D_n^1||^2}{2N_p} + \\ &\frac{\sum_{n=1}^{N} l_n \cdot (\max\{0, t - ||D_n^0 - D_n^1||\})^2}{2N_n}, \end{aligned} \tag{19}$$

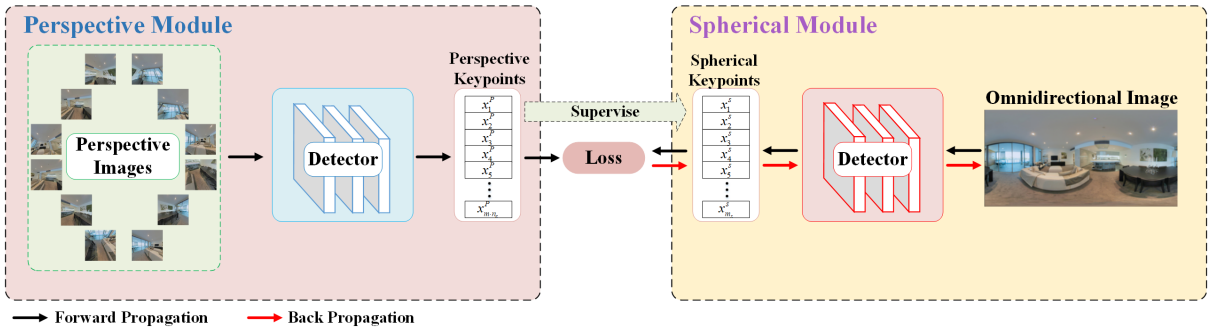


Fig. 6. Pre-training pipeline of the spherical module.

where t is the margin to control the distance between the descriptions of keypoints in a negative pair, N_p and N_n are the numbers of positive and negative pairs respectively, and $N_p + N_n = N$. The two terms in Eq. (19) make the model learn similar descriptions for positive pairs and discriminative descriptions for negative pairs. Because the perspective module is trained entirely with self-supervision, there is no available knowledge regarding the positive or negative pairs. Thus, we further design a score loss function to maximize the number of correspondences between two PIs. It is defined according to the scores of the keypoints from positive pairs as:

$$L_s(\mathbf{x}^0, \mathbf{p}^0, \mathbf{l}^0, \mathbf{x}^1, \mathbf{p}^1, \mathbf{l}^1) = \frac{1}{N_p + 1} - \frac{\epsilon \sum_{n=1}^{N} (l_n^0 p_n^0 + l_n^1 p_n^1)}{N_p + 1}, \tag{20}$$

where $p^o (o = (0, 1))$ is the feature map of the image I^o, and ϵ is a weighting parameter.

B. Pre-training the Spherical Module

Since the descriptor is shared across the two modules, we only train it with the perspective module, which improves the training efficiency. Therefore, only the spherical detector is pre-trained in the second stage. There are two available ways to warm up the spherical detector. One is to train it with a self-supervised strategy, and the other is to train it under the supervision of the perspective detector pre-trained in Stage 1. Considering the key challenge of our framework is to detect perspectively equivariant keypoints, we decided to alow the spherical detector to learn roughly the features of the two types of images from random initialization. Thus, we use the pre-trained perspective module to guide the training of the spherical module, and the training pipeline is shown in Fig. 6.

First, we feed n_c PIs $\{I^1, I^2, ...I^{n_c}\}$ that cover the whole spherical surface into the perspective detector, and obtain multiple groups of keypoints $\{\mathbf{x}^1, \mathbf{x}^2, ...\mathbf{x}^{n_c}\}$, each one corresponding to one PI. Next we collect them into the same group, denoted by $\mathbf{x}^P = \{x_1^1, ..., x_m^1, ..., x_1^{n_c}, ..., x_m^{n_c}\}$, where m is the number of keypoints in the image. To make the spherical detector be robust to the rotational variations of the ODIs, we randomly rotate the input panoramic image along the roll, pitch, or yaw axis with an angle ranged from $0°$ to $35°$ before feeding it into the spherical detector. Then, we use the spherical detector to localize its keypoints $\{x_1^s, x_2^s, ..., x_{m_s}^s\}$, where m_s is the number of spherical keypoints. For each keypoint x_i^s, we find its closest perspective keypoint from $\mathbf{x}^P$ on the sphere, and couple them to a pair $x_i^c = \{x_i^s, \widetilde{x_i^P}\}$.

$$\widetilde{x_i^P} = \arg\min_{x_j^P} [Dis(x_i^s, x_1^P), ..., Dis(x_i^s, x_{m \cdot n_c}^P)] \quad \text{with} \quad j \in \{1, ..., m \cdot n_c\}, \tag{21}$$

The training goal in this stage is to obtain the initial keypoints from ODIs and PIs and improve the repeatability of the spherical keypoints on planar space. Therefore, the loss function is to minimize the distance between the spherical coordinates of coupled pairs, which can be formulated as

$$L(\mathbf{x}^s, \mathbf{x}^P) = \frac{1}{m_s} \sum_{i=1}^{m_s} ||x_i^s - \widetilde{x_i^P}||^2, \tag{22}$$

where $\mathbf{x}^s$ is the collection of the keypoints on the ODI.

C. Co-supervised Training of the Perspective and Spherical Modules

1) Training strategy: After pre-training the two modules, we jointly train them further with a co-supervision strategy in which each module in OmniKL is trained alternatively with the other fixed, as shown in Fig. 1. Note that the feature descriptor is optimized with the perspective module, there are two reasons for this: First, the descriptor is shared among the two modules, thus it can be optimized only once in one training epoch with a proper loss function, which reduces the computational cost of training. Second, the pipeline of the spherical module is more complicated than that of the perspective module, thus if the descriptor is optimized with the spherical module, it will increase the difficulty of training.

While training the perspective module, we use a triplet input including a pair of PIs $\{I^0, I^1\}$ and an ODI I^s. We label three groups of description pairs for training, and the loss function jointly considers them. The first group comes from the two PIs, and the annotation method is the same as that in Stage 1. The second group is the collection of the description pairs consisting of a description from one PI and the other from the ODI, formed as $\{I^0, I^s\}$. The keypoints $\mathbf{x}^0$ on the PI are also projected onto the sphere, and then we find the closest spherical keypoint for each. Then the pair of keypoints and their corresponding descriptions $F_n^{s,0} = \{(x_i^0, d_i^0), (x_j^s, d_j^s)\} = \{X_n^s, X_n^0, D_n^s, D_n^0\}$ are labeled as positive with $l_n^{s,0} = l_i^0 = l_j^s = 1$ or negative with $l_n^{s,0} = l_i^0 = l_j^s = 0$ using the labeling

method in Stage 1. In the above d_j^s is the description of the j-th keypoint on the ODI, $l_n^{s,0}$ is the label of the n-th pair, $\mathbf{l}^0$ and $\mathbf{l}^s$ are the collections of the labels for the keypoints from the images, respectively. The generation of the third group and its labels denoted as $\{\mathbf{l}^1, \mathbf{l}^s\}$ are the same as those of the second group, except using the other PI.

When the spherical detector is optimized, the perspective detector and descriptor are all fixed, and we train the spherical detector to localize perspectively equivariant keypoints by training it to focus on the keypoints that are also detected on the PIs. Different from the scheme in Fig. 6 using a set of PIs covered the whole ODI, we only use the pair of PIs $\{I^0, I^1\}$. After obtaining the keypoints on the pair of PIs, we only consider the keypoints on the partial spherical surface overlapped by the two PIs. In other words, we enforce the locations of the keypoints on this partial sphere to approximate those on the PIs, and ignore the keypoints on other remaining areas. We randomly rotate the ODI before feeding it into the spherical module, aiming to improve the robustness of the model against the rotation of the ODIs.

2) Training Loss Functions:

a) Perspective loss function: The co-supervised training mainly aims at seeking the perspectively equivariant keypoints on the ODIs. Therefore, for the perspective module, its training objective consists of two parts respectively corresponding to ODIs and PIs, and thus its loss function is formulated as

$$L(I^0, I^1, I^s) = \lambda_p L_p(I^0, I^1) + \lambda_q L_q(I^0, I^1, I^s), \quad (23)$$

where L_p and L_q are the losses targeting the PI-to-PI and ODI-to-PI matchings respectively, and λ_p and λ_q are two regularization parameters.

Because we adopt the Siamese architecture to train the perspective module, which is similar to that in Stage 1, we can use the same loss function as Eq.(18), thus we have

$$L_p(I^0, I^1) = \lambda_c L_c(\mathbf{F}, \mathbf{l}) + \lambda_s L_s(\mathbf{x}^0, \mathbf{p}^0, \mathbf{l}^0, \mathbf{x}^1, \mathbf{p}^1, \mathbf{l}^1). \quad (24)$$

L_q is used to make the perspective module robust to the ODI-to-PI projection, thus it can be designed as the union of two contrastive loss functions, and each for one PI-to-ODI pair

$$L_q(I^0, I^1, I^s) = \sum_{o=0}^{1} \lambda_o L_q^o(I^o, I^s). \quad (25)$$

The contrastive loss L_q^o can be written as

$$\begin{aligned} L_q^o(I^o, I^s) &= L_c(F^{s,o}, l^{s,o}) \\ &= \frac{\sum_{n=1}^{N} l_n^{s,o} ||D_n^s - D_n^o||^2}{2N_p} + \\ &\quad \frac{\sum_{n=1}^{N} l_n^{s,o} \cdot (\max\{0, t - ||D_n^s - D_n^o||\})^2}{2N_n}. \end{aligned} \quad (26)$$

b) Spherical loss function: The training for the spherical module does not involve the descriptor, and it is used to fine-tune the spherical detector for improved ability to localize perspectively equivariant keypoints on the ODIs, thus the spherical detector is expected to focus on the keypoints that are also detected on the PIs. Our loss function for training the spherical detector is slightly modified from Eq.(22) by only considering two PIs and their overlapped spherical surface, denoted as

$$L(I^0, I^1, I^s) = \frac{1}{m_o} \sum_{i=1}^{m_o} ||x_i^s - \widetilde{x_i^{0,1}}||^2, \quad (27)$$

where $\mathbf{x}^s = [x_1^s, ..., x_{m_o}^s]$ is the collection of keypoints on the partial spherical surface covered by the two PIs, m_o is the number of spherical keypoints on the partial surface, and $\widetilde{x_i^{0,1}}$ is the closest perspective keypoint of the i-th spherical keypoint, which comes from the collection of the keypoints on the two PIs ($\mathbf{x}^{0,1}$). $\widetilde{x_i^{0,1}}$ is achieved by

$$\begin{aligned} \widetilde{x_i^{0,1}} = &\arg\min_{x_j^{0,1}} [Dis(x_i^s, x_1^{0,1}), ..., Dis(x_i^s, x_{2m}^{0,1})] \\ &\text{with } \ j \in \{1, ..., 2m\}. \end{aligned} \quad (28)$$

V. Evaluation and Analysis

A. Experimental Setup

We compare OmniKL with several classic hand-crafted and learning-based approaches for the keypoint matching task. For the hand-crafted approaches, we use five baselines including SIFT [15], ORB [20], Spherical SIFT [8], SPHORB [9], and BRISKS [10], all of which are entire frameworks including the keypoint detector and descriptor. For the learning-based approaches, we choose the state-of-the-art end-to-end trainable approaches SuperPoint [26], RF-Net [27], and ASLFeat [28] for comparison. Additionally, we also compare with T-ASLFeat, which performs ASLFeat on the tangent plane to detect ODI keypoints. Two matching scenarios including ovo and ovp are considered, and we use two metrics to evaluate the performance of all the approaches. The first is the keypoint repeatability rate, calculated as the ratio between the number of matched point-to-point correspondences of keypoints and the lower number of keypoints detected in the two images. The second metric is the mean keypoint matching accuracy, and it is computed as the ratio between the correct matchings (3-pixel threshold) and the total number of matchings [46]. Generally, the keypoint repeatability rate is used to measure the effectiveness of the detector, while the matching accuracy indicates the performance of the descriptor. For fair comparison, we enforce all the models to consider the same number of keypoints per image. Considering the training burden, the number is set to 64, which is also a popular setting for the batch size, making the model easy to process the cropped patches in parallel.

To prepare our data, we select twelve base PI-center-positions ($(\pm 30°, 0°)$, $(\pm 30°, \pm 60°)$, $(\pm 30°, \pm 120°)$ and $(\pm 30°, 180°)$) on the sphere, and then generate 36 PIs around each position with a random offset on latitudes and longitudes within $[-30°, 30°]$. The PIs for each base PI-center-position form a group. For each PI, the size is 128×128, and the FoV is $120°$. After obtaining a well-trained model, the FoV size of PIs will not affect its inference performance because it has been trained to learn general features on PIs, and thus its effectiveness is universal on PIs of different FoV. When pre-training the perspective module, we randomly choose three pairs of images from each group of PIs, totally 36 pairs for

TABLE I
PERFORMANCE OF IMAGE MATCHING UNDER THE ROTATIONS AROUND THREE AXES.

Task	Scene	Method	Repeatability Rate				Matching Accuracy			
			Roll	Pitch	Yaw	Average	Roll	Pitch	Yaw	Average
ovo	PanoContext	SIFT	0.669	0.688	0.832	0.730	0.607	0.625	0.792	0.675
		ORB	0.737	0.741	0.825	0.768	0.600	0.608	0.765	0.658
		Spherical SIFT	0.729	0.746	0.869	0.781	0.627	0.637	0.782	0.682
		SPHORB	0.742	0.751	0.869	0.787	0.641	0.643	0.808	0.697
		BRISKS	0.733	0.746	0.858	0.779	0.650	0.649	0.812	0.704
		SuperPoint	0.691	0.709	0.829	0.743	0.580	0.572	0.767	0.640
		RF-Net	0.710	0.702	0.840	0.751	0.597	0.594	0.762	0.651
		ASLFeat	0.724	0.706	0.843	0.758	0.605	0.603	0.793	0.667
		T-ASLFeat	0.739	0.738	0.875	0.784	0.653	0.635	0.829	0.709
		OmniKL	**0.759**	**0.767**	**0.891**	**0.806**	**0.679**	**0.683**	**0.885**	**0.749**
	Panoramic 3D Outdoor	SIFT	0.586	0.580	0.815	0.660	0.505	0.506	0.767	0.593
		ORB	0.661	0.678	0.781	0.707	0.481	0.485	0.729	0.565
		Spherical SIFT	0.710	0.715	0.826	0.750	0.565	0.526	0.790	0.627
		SPHORB	0.722	0.733	0.839	0.765	0.581	0.541	0.791	0.638
		BRISKS	0.706	0.707	0.822	0.745	0.563	0.521	0.776	0.620
		SuperPoint	0.651	0.663	0.769	0.694	0.484	0.477	0.742	0.568
		RF-Net	0.694	0.705	0.807	0.735	0.535	0.486	0.774	0.598
		ASLFeat	0.682	0.695	0.799	0.725	0.526	0.487	0.769	0.584
		T-ASLFeat	0.713	0.719	0.833	0.755	0.573	0.527	**0.806**	0.635
		OmniKL	**0.741**	**0.738**	**0.859**	**0.779**	**0.587**	**0.538**	0.802	**0.642**
ovp	PanoContext	SIFT	0.658	0.660	0.649	0.656	0.287	0.289	0.284	0.287
		ORB	0.549	0.543	0.540	0.544	0.259	0.257	0.264	0.260
		Spherical SIFT	0.794	0.808	0.827	0.810	0.329	0.313	0.321	0.321
		SPHORB	0.805	0.812	0.827	0.815	0.336	0.321	0.324	0.327
		BRISKS	0.780	0.792	0.801	0.791	0.317	0.304	0.299	0.307
		SuperPoint	0.763	0.764	0.776	0.768	0.202	0.199	0.218	0.206
		RF-Net	0.778	0.782	0.778	0.779	0.223	0.214	0.225	0.221
		ASLFeat	0.775	0.781	0.790	0.782	0.231	0.218	0.230	0.226
		T-ASLFeat	0.799	0.806	0.839	0.815	0.354	0.342	0.351	0.349
		OmniKL	**0.839**	**0.842**	**0.859**	**0.847**	**0.371**	**0.365**	**0.383**	**0.373**
	Panoramic 3D Outdoor	SIFT	0.629	0.618	0.603	0.617	0.262	0.251	0.252	0.255
		ORB	0.484	0.490	0.505	0.493	0.230	0.241	0.238	0.236
		Spherical SIFT	0.762	0.749	0.743	0.751	0.289	0.265	0.276	0.277
		SPHORB	**0.771**	0.759	0.758	0.763	0.297	0.262	0.283	0.281
		BRISKS	0.755	0.743	0.741	0.746	0.276	0.251	0.257	0.261
		SuperPoint	0.720	0.717	0.720	0.719	0.138	0.136	0.152	0.142
		RF-Net	0.731	0.727	0.727	0.728	0.163	0.173	0.173	0.170
		ASLFeat	0.728	0.722	0.716	0.722	0.169	0.171	0.179	0.173
		T-ASLFeat	0.755	0.739	0.763	0.752	0.307	0.256	0.273	0.279
		OmniKL	0.767	**0.769**	**0.772**	**0.769**	**0.310**	**0.285**	**0.289**	**0.295**

twelve base PI-center-positions for one ODI, while for pre-training the spherical module, we select one PI from each group of PIs for every ODI to generate ground-truth keypoints covering the whole spherical surface. In the co-training stage, an ODI and one of its corresponding perspective pairs are chosen to form the triplet input, totally 12 triplets for each ODI.

The models used for evaluation are trained on the PanoContext [47] and Multi-modal Panoramic 3D Outdoor dataset [48], respectively, towards keypoint detection in indoor and outdoor scenes. We randomly select 100 images from each scene contained in the dataset for training, and the remaining images are used for evaluation. The Adam optimizer [49] is used for tuning the weights, with a learning rate of 0.0001, and the total number of epochs is 100. For the scaling factors that balance the loss functions, we set $\lambda_c = 0.5$ and $\lambda_s = 0.1$ in Eq.(18) and Eq.(24), $\lambda_p = 0.1$ and $\lambda_q = 1.0$ in Eq.(23), and $\lambda_o = 0.5$ in Eq.(25).

B. Matching between Synthetic Images

1) ODIs vs. ODIs (ovo): OmniKL is evaluated for the keypoint matching between ODIs rotated from 5° to 35°. Since there are three axes for a 3D rotation, including roll, pitch and yaw, our experiments are performed individually on each axis.

Tab. I shows the average results on all the rotation angles. Among the three axes for this scenario, all the approaches perform the best on the yaw axis. It is because the rotation around the yaw axis for an ODI is equivalent to translating the projected panorama image along the horizontal axis, which reserves the most similar features compared to the rotations around other axes. We also notice that the spherical approaches (OmniKL, Spherical SIFT, SPHORB, BRISKS, and T-ASLFeat) perform better than the planar approaches. For example, on the indoor scene dataset, the average repeatability rates of SPHORB and ours are 0.787 and 0.806, respectively, while the maximum of those of the planar approaches is 0.768; the average matching accuracy of SPHORB and OmniKL are 0.697 and 0.749, while the maximum for the planar ones is 0.675. Besides, OmniKL is superior to SPHORB with

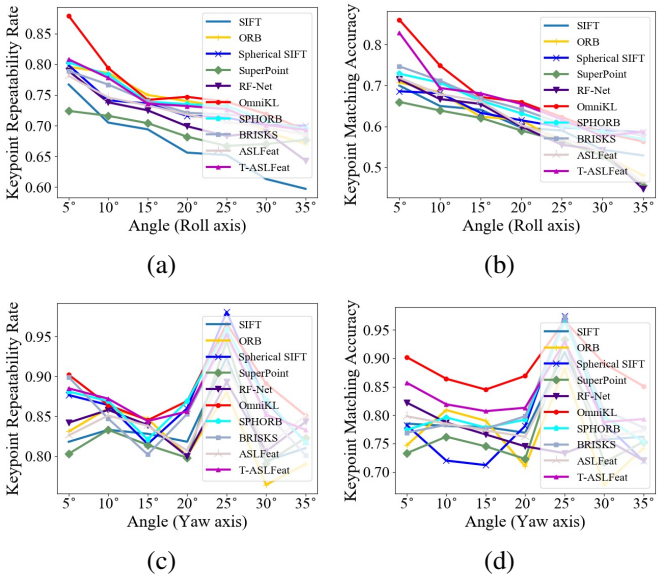


Fig. 7. The results of ovo matching on the indoor scene dataset, where the top row shows the results with roll rotation and the bottom row shows those for yaw rotation.

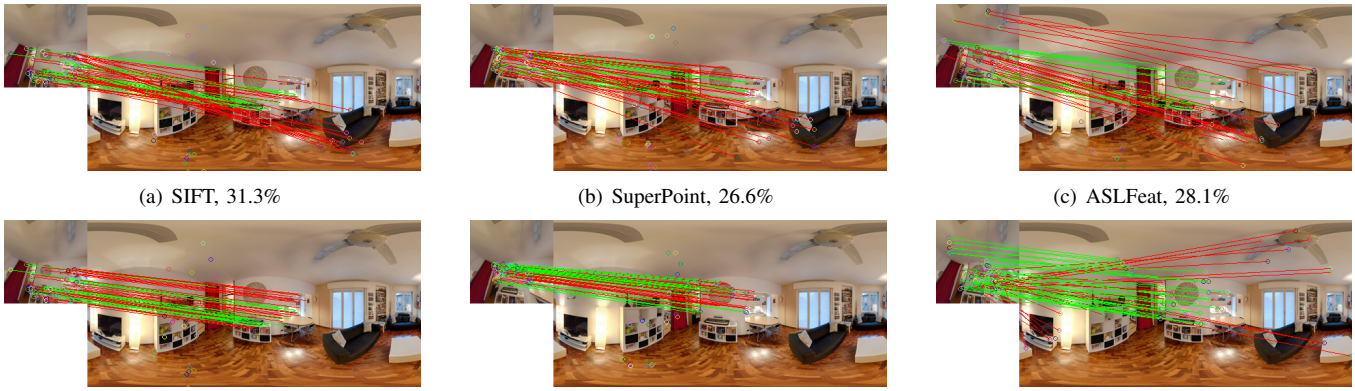
more than 1% and 5% in repeatability rate and matching accuracy, which indicates the effectiveness and robustness of OmniKL in learning ODI keypoints. Admittedly the matching under only 64 keypoints is a challenging work because fewer keypoints means also fewer candidates for calculating the correspondences. Therefore, the superiority of OmniKL also indicates its capability in maintaining the ranking of the keypoints against the geometric transformations. For our evaluation dataset SPHORB does not achieve results that are comparable with those reported in the original paper that included the code publicly on a website. A potential reason is that we used a different testing dataset when compared to [9] which means that a different data distribution may lead to results with minor differences. When compared with ORB, the improvements of SPHORB in our experiment and [9] are similar. It is also worth noting that T-ASLFeat that operates on the tangent plane also achieves a good performance which is due to the fact that suffers from lower distorted representations for the omnidirectional images. However, it relies on repeated operations on a collection of tangent images, and the recombining strategy for aligning or blending the keypoints from different tangent images is still an open problem.

Fig. 7 further presents the results by plotting the curves of the repeatability rate and the matching accuracy at different rotation angles on the indoor scene dataset. As the behaviors of the approaches on the roll and pitch axes are similar, we only show the results on the roll and yaw axes. It can be seen that the lines indicating the results of the four spherical approaches are always above the other lines, and the red line which corresponds to OmniKL in the beginning is quite ahead relative to the other lines. The behavior on the yaw axis is rather different from that on the roll axis. As we can see, the repeatability rates and matching accuracies do not always decrease with the increasing of the angles, instead, the change of the accuracies (or the rates) is periodic, and they reach their peaks when the rotational angle is approximately 25°. A possible reason for this is that the yaw rotation of the original ODI is equal to the periodic translation of the panorama along the longitude axis. As the pixels on the ODI occur cyclically during the translation, if the image is rotated to a position in which some key objects in it are split to the left and right sides of the image, the performance of the methods will decrease. Instead, when the image is rotated to reserve more integrated objects, the methods will perform better, and the rotation of a 25° angle may be one of the rotation angles that can retain more accurate features of the image for processing. Another reason may be that we only use the top-64 keypoints for evaluation, which makes it harder to explore the correspondences among the keypoints, especially on the ODIs, whose resolutions are often relatively high. We observe a similar phenomenon on the outdoor scene dataset, and due to space limitations, we only present the results on the indoor ODIs.

To evaluate the generalizability of OmniKL, we perform a cross validation for the models trained on different datasets. First, it is done for the model trained with the indoor ODIs on the test images from the outdoor dataset, and the repeatability rate and matching accuracy is 0.73 and 0.62, respectively. Then we exchange the training and testing data, and the repeatability rate and matching accuracy reduce to 0.7 and 0.58, respectively. The variation of the results is induced from the differences of the ODIs in the two datasets: The images in the indoor dataset are of high quality, then the model trained on them can capture more general features for ODIs, and thus it can generalize well to the outdoor dataset. On the other hand, the quality of the ODIs in the outdoor dataset are of relatively poor quality, which affects the capability of the trained model for extracting accurate features, and it further limits its generalizability to other datasets. Despite the differences on the two datasets, the results still indicate that OmniKL can generalize to unknown ODIs.

2) ODIs vs. PIs (ovp): We evaluate the performance of OmniKL for detecting perspectively equivariant keypoints by calculating the correspondences between PI and ODI keypoints.

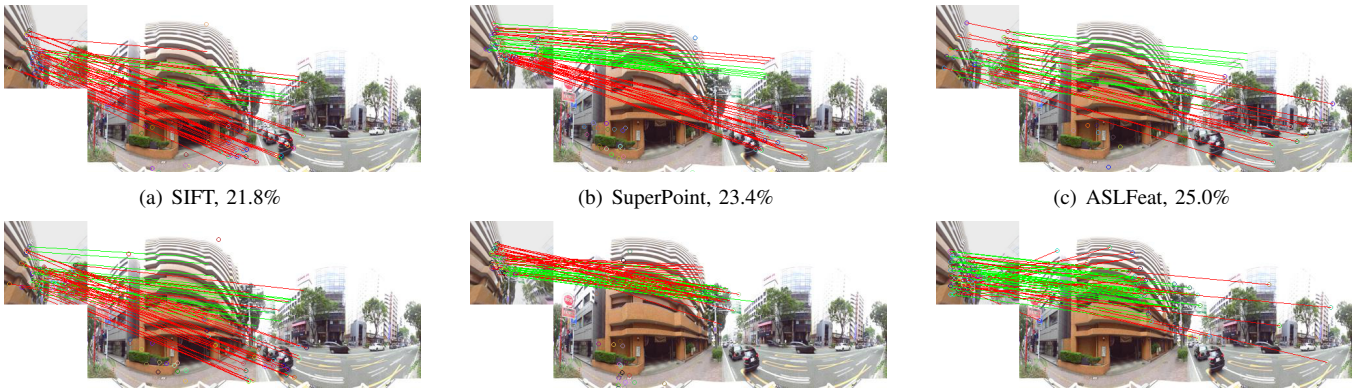
The results of this experiment are shown in Tab. I. Here we observe that the repeatability rate of OmniKL outperforms those of others by $1\% \sim 30\%$, which indicates the detector can localize the spherical keypoints with high correspondence to perspective keypoints. Compared to the repeatability rate, the matching accuracies of the approaches are obviously lower, due to the difficulty of generating consistent feature representations from two different domains. The average matching accuracy of OmniKL reaches 0.373, which outperforms those of other approaches by $2\% \sim 16\%$. The comparative performance in matching accuracies demonstrates that OmniKL can capture accurate features from the perspective and spherical domains synchronously. Additionally, the spherical baselines, especially SPHORB, achieve better performance in matching accuracy among the baselines, which indicates their effectiveness in capturing robust features against the projection between ODIs and PIs. However, Spherical SIFT needs an



(a) SIFT, 31.3% (b) SuperPoint, 26.6% (c) ASLFeat, 28.1%

(d) Spherical SIFT, 39.0% (e) SPHORB, 42.2% (f) OmniKL, 45.3%

Fig. 8. Matching between perspective and omnidirectional images for PanoContext dataset. The matching accuracies are listed for each approach.



(a) SIFT, 21.8% (b) SuperPoint, 23.4% (c) ASLFeat, 25.0%

(d) Spherical SIFT, 31.3% (e) SPHORB, 32.8% (f) OmniKL, 37.5%

Fig. 9. Matching between perspective and omnidirectional images for Panoramic Outdoor dataset. The matching accuracies are listed for each approach.

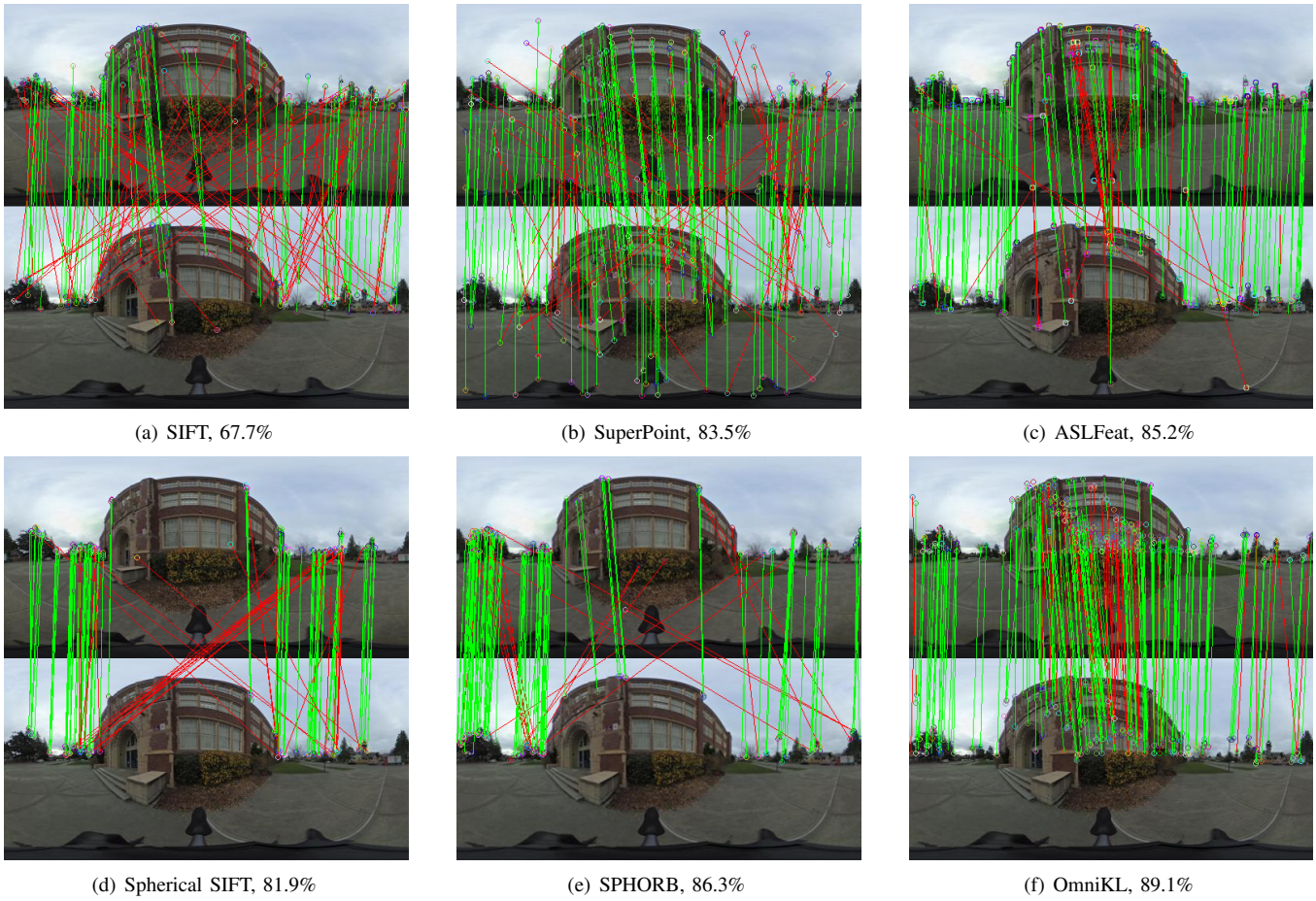
additional mapping step to align the spherical descriptions with the planar perspective ones that are generated by SIFT. The approach seems fussy and the mapping itself will introduce description alignment errors. Although SPHORB and BRISKS achieve a matching between ODIs and PIs by projecting the PIs to the sphere, since the specific pose parameters for PIs are not known in advance, they make a strong assumption on the PIs with a fixed focal length and a constant FoV. This does not conform with the actual case. Contrary to that, OmniKL does not require the assumption for the a-priori knowledge of the PIs, and directly makes a match for PI with a spherical one using consistent descriptions generated in a data-driven manner, which is more skillful in extracting in-depth and accurate features than the hand-crafted approaches.

Fig. 8 and Fig. 9 show some examples of ovp matching for SIFT, SuperPoint, ASLFeat, Spherical SIFT, SPHORB, and OmniKL. The correct matchings are plotted in green color and the false ones are in red. Admittedly, the great gap of the FoV between PIs and ODIs raises difficulties for ovp matching. Yet, OmniKL still keeps its competitiveness compared to the baselines in this scenario. In the PIs, the meaningful objects, including the bookshelf and wall hangings in Fig. 8 and buildings and trees in Fig. 9, still receive more attention than the background (floors, walls, skies). Not only that, the PI keypoints are also detected on the ODIs, and nearly 30%-40% of them are matched accurately through their descriptions. The visualization results indicate the effectiveness of OmniKL in detecting perspectively equivariant keypoints and generating consistent descriptions for them.

C. Matching between Realistic Images

The performance of OmniKL is also evaluated on the real captured images, and we show the matching results in Fig. 10 and Fig. 11. Fig. 10 shows the ovo matching results in the scene containing a building[1]. It can be observed that OmniKL outperforms other baselines almost 3%-20% under the condition of the moving camera. Besides, it is worth noting that the two images are captured from different viewpoints, which indicates that our proposed approach is robust to viewpoint changes. We further perform experiments on 200 images from the KASHIWA dataset [50] consisting of images captured with changing viewpoints, and the results are shown in Tab. II. It can be seen that even with the small-scale viewpoint changes, due to the preferable image quality all the approaches perform better than those in the conditions illustrated in Tab. I. Among the approaches, OmniKL exceeds the others on both repeatability rates and matching accuracies, which indicates its effectiveness in the case of viewpoint changes. Moreover,

[1]https://github.com/openMVG/Image-datasets



(a) SIFT, 67.7% (b) SuperPoint, 83.5% (c) ASLFeat, 85.2%

(d) Spherical SIFT, 81.9% (e) SPHORB, 86.3% (f) OmniKL, 89.1%

Fig. 10. Matching between ODIs captured around the building with different viewpoints. The matching accuracies are listed for each approach.

we do not re-train our model in this experiment, and it shows that OmniKL can generalize to unseen datasets with different scenes, viewpoints, and camera intrinsics.

TABLE II
THE REPEATABILITY RATES (REP.) AND MATCHING ACCURACIES (MAT.) OF THE OVO MATCHING ON ODIS WITH CHANGING VIEWPOINTS.

	SIFT	Spherical SIFT	SPHORB	SuperPoint	ASLFeat	OmniKL
Rep.	0.752	0.821	0.836	0.787	0.796	0.843
Mat.	0.602	0.763	0.776	0.729	0.751	0.788

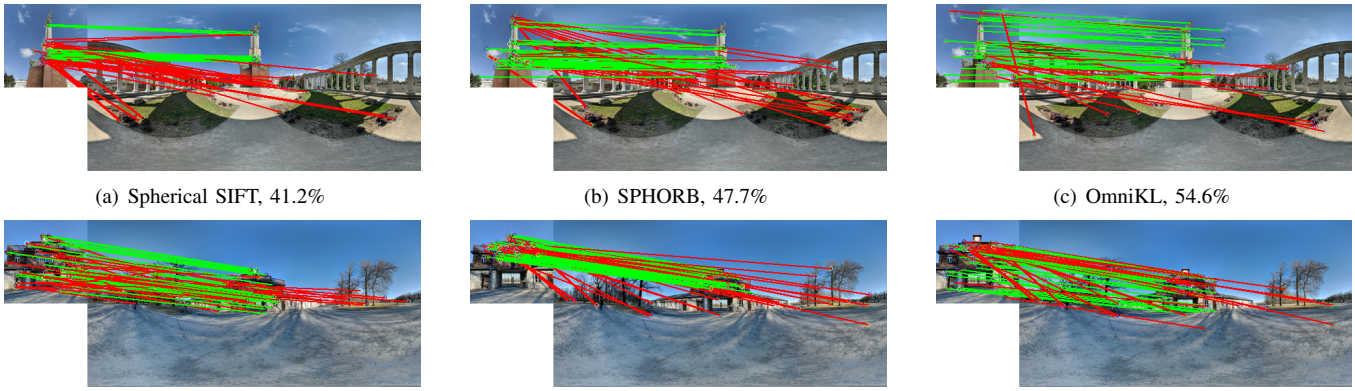
Fig. 11 shows the ovp matching results on two pairs of realistic images. Compared to Spherical SIFT and SPHORB, OmniKL achieves better matching performance in both images. Besides the matching accuracy, the distribution of the keypoints detected by OmniKL is more uniform than others. For example, in the top row, Spherical SIFT and SPHORB mainly focus on the statue, while OmniKL can detect the keypoints on other areas including the pedestal and the clouds, which helps it better estimate the camera pose. We also notice that all of the approaches perform better on the two real captured images than the synthetic datasets including indoor and outdoor ODIs. The reason is that the PIs corresponding to the realistic ODIs are captured around the equator of the ODIs, and they do not suffer from significant deformation when compared to PIs from other positions. Therefore, the features extracted from them are more accurate, and they lead to a good matching performance.

D. Ablation Study

We perform an ablation study to test the performance of the kernel-adaptive convolution, DCPS, and spherical cropper.

1) Kernel-adaptive Convolution: We replace the kernel-adaptive convolution in the spherical detector with EquiConv to evaluate its impact on the performance of OmniKL. The results presented in Tab. III indicate that the kernel-adaptive convolution improves both the repeatability rate and matching accuracy compared to the EquiConv, which only uses latitude-specific deformations to deform its kernels. For ovo matching on the PanoContext dataset, the improvement of the repeatability rate when applying kernel-adaptive convolution is 1.5%, and that of the matching accuracy is 0.8%. The ascendant performance of the kernel-adaptive convolution derives from its ability to extract accurate features from ODIs, which is the consequence of the convex-hull-based deformed kernels and the strategy to make the kernels learn to fine-tune their receptive fields to be adaptive to the semantic information.

2) DCPS: In this experiment, the DCPS operations in both the spherical and perspective modules are replaced by the traditional spatial softmax, with the other components fixed. The experimental results are shown in Tab. III.



(a) Spherical SIFT, 41.2% (b) SPHORB, 47.7% (c) OmniKL, 54.6%

(d) Spherical SIFT, 50.9% (e) SPHORB, 55.3% (f) OmniKL, 59.4%

Fig. 11. Matching between two pairs of real captured perspective and omnidirectional images. The matching accuracies are listed for each approach.

TABLE III
AVERAGE RESULTS ON THREE ROTATION AXES OF ABLATION EXPERIMENT.

Task	Scene	Approach	Repeatability Rate	Matching Accuracy
ovo	PanoContext	OmniKL (with EquiConv)	0.791	0.741
		OmniKL (with spatial softmax)	0.766	0.721
		OmniKL (with bilinear cropper)	0.775	0.679
		OmniKL	**0.806**	**0.749**
	Panoramic 3D Outdoor	OmniKL (with EquiConv)	0.772	0.637
		OmniKL (with spatial softmax)	0.736	0.631
		OmniKL (with bilinear cropper)	0.750	0.593
		OmniKL	**0.779**	**0.642**
ovp	PanoContext	OmniKL (with EquiConv)	0.836	0.365
		OmniKL (with spatial softmax)	0.813	0.334
		OmniKL (with bilinear cropper)	0.825	0.307
		OmniKL	**0.847**	**0.373**
	Panoramic 3D Outdoor	OmniKL (with EquiConv)	0.760	0.283
		OmniKL (with spatial softmax)	0.747	0.260
		OmniKL (with bilinear cropper)	0.738	0.249
		OmniKL	**0.769**	**0.295**

It can be observed that the DCPS operation dramatically improves the performance of OmniKL compared to that using the spatial softmax. Specifically, DCPS is superior to spatial softmax with repeatability rates and matching accuracies of 3%-5% on almost all the matching tasks. The remarkable capability of DCPS for localizing keypoints on the feature map benefits from the global ranking of candidate keypoints generated by DCPS in the training stage, and it makes the learning at the detector directly focus on the points with globally high scores on the whole feature map, while the spatial softmax can only provide local points from narrow windows.

3) Spherical Cropper: In our design, we project the panorama to the spherical surface and use rectilinear projection to generate patches for ODIs to avoid the deformation raised by directly performing bilinear crop on the panorama. To evaluate the necessity of this processing, we replace the spherical cropper in the spherical module with a bilinear cropper as in the perspective module, and the results are shown in Tab. III.

Generally, the spherical cropper outperforms the bilinear cropper, especially on the matching accuracy. Take the ovo matching as an example, the matching accuracy of the approach with bilinear cropper on the PanoContext dataset is only 0.679, while that with our proposed spherical cropper reaches 0.749, which verifies that the spherical cropper can generate patches with low deformation for ODIs, resulting in more robust descriptions for spherical keypoints. Additionally, although the two comparative frameworks in this experiment share the same architecture of detector, the approach with the spherical cropper still performs better than that with the bilinear cropper, and it indicates that the inaccurate descriptions negatively affect the training of the detector, and decrease its ability to localize keypoints.

Synthetically analyzing the results of four approaches in Tab. III, we conclude that kernel-adaptive convolution, DCPS, and spherical cropper are essential components in OmniKL for localizing keypoints and generating descriptions.

TABLE IV
AVERAGE PERFORMANCE OF DIFFERENT APPROACHES UNDER THE CONDITION OF 2K KEYPOINTS ON THREE ROTATIONAL AXES.

	Spherical SIFT	SPHORB	SuperPoint	ASLFeat	OmniKL
Rep.	0.934	0.949	0.907	0.916	0.963
Mat.	0.801	0.812	0.761	0.779	0.821

TABLE V
ROTATION ERROR RATES OF DIFFERENT APPROACHES FOR POSE ESTIMATION.

Threshold	SIFT	Spherical SIFT	ORB	SPHORB	SuperPoint	ASLFeat	OmniKL
0.2°	0.55	0.32	0.59	**0.25**	0.62	0.56	0.28
0.5°	0.29	0.18	0.21	0.14	0.26	0.17	**0.12**
1.0°	0.08	0.04	0.05	0.02	0.08	0.09	**0.01**

TABLE VI
TRANSLATION ERROR RATES OF DIFFERENT APPROACHES FOR POSE ESTIMATION.

Threshold	SIFT	Spherical SIFT	ORB	SPHORB	SuperPoint	ASLFeat	OmniKL
5°	0.60	0.57	0.57	0.54	0.55	0.58	**0.54**
10°	0.41	0.32	0.38	**0.30**	0.42	0.60	0.32
20°	0.34	0.30	0.34	0.29	0.67	0.31	**0.28**

TABLE VII
COMPUTATION TIMING FOR EACH APPROACH ON ODIS SIZED OF 1024×512.

Approach	SIFT	Spherical SIFT	ORB	SPHORB	SuperPoint	ASLFeat	OmniKL
Time (ms)	380	44179	122	857	667	6750	1433

E. Detecting More Keypoints

In this experiment, we increase the number of keypoints to 2k, and evaluate some well-performed approaches shown in Tab. I in detecting ODI keypoints on the indoor scene PanoContext dataset, and the average results over three rotational axes are shown in Tab. IV. We are surprised to notice that both the repeatability rates and matching accuracies improve compared to those in Tab. I when the number of detected keypoints is increased, the main reason of which is that the image transformations affect the ranking of the detected keypoints, and thus fewer number of keypoints means lower overlap of the keypoints on different images. Among all of the approaches, it can be observed that OmniKL is still superior to others under this setting. It is important to point out that we reuse the model that was trained with 64 keypoints, instead of re-training it with 2k keypoints in this experiment. Therefore, its generalization to the case of 2k keypoints indicates that our model learns not only the ranking of the keypoints, but also more general representations for the input ODIs. Additionally, re-training OmniKL with 2k keypoints will definitely improve further its performance, while the training procedures of other DNN-based approaches are number-irrelevant, and thus there is less space for improvement for them in this case.

F. Cross-View Camera Pose Estimation

Cross-view pose estimation with ODIs is a key step in several computer vision applications. The typical pipeline for solving this problem is to match the keypoints in two views and then use the point-level correspondences to estimate the camera pose. In this experiment, we reuse the models trained on the indoor scene dataset, rather than re-train them on the new dataset. We consider ODI pairs captured by a moving camera and estimate the rotation matrix or a translation vector from them. Note that rotations and translations are estimated separately, thus the transformations are also applied on the camera separately. We follow the methods in [51], and use Blender[2] to generate 100 realistic synthetic ODIs from public 3D models including Classroom, Urban, and Indoor. We change the camera pose to render ODI pairs, and the rotation angles range from 5° to 40° (the rotational axis is not fixed, which means the image is rotated around multiple axes simultaneously), while the translation distances range from 0.1m to 2m. Tab. V shows the average estimation error rates for ten runs under different rotational angles for the camera. It can be observed that the estimation error rates for our approach are lower than those of the baselines with up around 1% to 30%, which indicates the superior performance in localizing and describing keypoints of OmniKL. We also evaluate the localization ability of OmniKL for translating cameras. Since the estimated translation vectors are obtained up to a scale parameter, we compute the translation error as the angle from the ground truth. We evaluate the translation error rates on different angle thresholds, and the results are shown in Tab. VI. It can be observed that all the approaches achieved over 30% errors even though the threshold up to 20°. Despite the difficulties for localizing translating camera in this experiment, OmniKL still outperforms slightly the other schemes. As the model is not re-trained, it indicates the good generalization capability of OmniKL towards pose-changing matching.

G. Timing

We compare the timing of OmniKL with those of the baselines on a computer installed with an Intel Xeon E5-2680 CPU and an NVIDIA TITAN Xp GPU, and the results are presented in Tab. VII. For the same ODI sized of 1024×512, the two planar hand-crafted approaches, including SIFT and ORB, process faster than the other approaches. However, considering their limitation in accuracy against large viewpoint variations, the millisecond-level improvement on computational cost is insignificant. When it comes to the spherical approaches, SPHORB is the fastest, OmniKL is

[2]https://www.blender.org

slightly behind, and Spherical SIFT is much more time-consuming than them. A vital reason for the lower time cost for SPHORB is that it pre-computes the geodesic projection matrix for ODIs. However, the matrix is strongly correlated to the spatial size of the ODIs, and thus it should store massive matrices corresponding to different image sizes, which is not practical in real-world systems. In contrast, our approach can take images with any size as its input, and the processing speed is also competitive. Note that the most time-consuming step in OmniKL is generating candidate keypoints in DCPS, while the number of the retained keypoints does not affect this operation. Besides, the patch cropping step is essentially an interpolation operation, which can be realized in parallel. Therefore, OmniKL can maintain similar processing speeds for different number of detected keypoints.

VI. Conclusion

With the objective of detecting perspectively equivariant keypoints on the ODIs, in this paper, we propose a learnable framework, namely OmniKL for localizing and describing keypoints on the ODIs and PIs, generating universal and consistent descriptions for them. We introduce a kernel-adaptive convolution and a DCPS operation in the detector for improving its capability in localizing keypoints, and we devise a spherical cropper to generate non-deformed patches from ODIs for more accurate spherical descriptions. Furthermore, we propose a training strategy that combines self-supervision and co-supervision to train the whole framework from scratch without requiring any labeled data. Massive experiments performed on the synthetic and realistic 360° images for ovo and ovp matching show that OmniKL is superior to the state-of-the-art approaches.

References

[1] L. Puig, J. Bermúdez, P. Sturm, and J. J. Guerrero, "Calibration of omnidirectional cameras in practice: A comparison of methods," *Computer Vision and Image Understanding (CVIU)*, vol. 116, no. 1, pp. 120–137, 2012.

[2] J. Xiao, K. A. Ehinger, A. Oliva, and A. Torralba, "Recognizing scene viewpoint using panoramic place representation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 2695–2702.

[3] S.-T. Yang, F.-E. Wang, C.-H. Peng, P. Wonka, M. Sun, and H.-K. Chu, "Dula-net: A dual-projection network for estimating room layouts from a single rgb panorama," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3358–3367.

[4] T. Geodeme, T. Tuytelaars, G. Vanacker, M. Nuttin, and L. Van Gool, "Omnidirectional sparse visual path following with occlusion-robust feature tracking," in *OMNIVIS Workshop, IEEE Conference on Computer Vision (ICCV)*, 2005.

[5] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett, and A. Zell, "Localization of mobile robots with omnidirectional vision using particle filter and iterative sift," *Robotics and Autonomous Systems*, vol. 54, no. 9, pp. 758–765, 2006.

[6] P. Hansen, P. Corke, W. Boles, and K. Daniilidis, "Scale-invariant features on the sphere," in *IEEE International Conference on Computer Vision (ICCV)*, 2007, pp. 1–8.

[7] Z. Arican and P. Frossard, "Scale-invariant features and polar descriptors in omnidirectional imaging," *IEEE Transactions on Image Processing (TIP)*, vol. 21, no. 5, pp. 2412–2423, 2012.

[8] J. Cruz-Mota, I. Bogdanova, B. Paquier, M. Bierlaire, and J.-P. Thiran, "Scale invariant feature transform on the sphere: Theory and applications," *International Journal of Computer Vision (IJCV)*, vol. 98, no. 2, pp. 217–241, 2012.

[9] Q. Zhao, W. Feng, L. Wan, and J. Zhang, "Sphorb: A fast and robust binary feature on the sphere," *International Journal of Computer Vision (IJCV)*, vol. 113, no. 2, pp. 143–159, 2015.

[10] H. Guan and W. A. Smith, "Brisks: Binary features for spherical images on a geodesic grid," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4886–4894.

[11] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 467–483.

[12] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 821–11 830.

[13] N. Savinov, A. Seki, L. Ladicky, T. Sattler, and M. Pollefeys, "Quad-networks: Unsupervised learning to rank for interest point detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3929–3937.

[14] A. Barroso-Laguna, E. Riba, D. Ponsa, and K. Mikolajczyk, "Key.net: Keypoint detection by handcrafted and learned cnn filters," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 5835–5843.

[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, 2004.

[16] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European Conference on Computer Vision (ECCV)*, 2006, pp. 404–417.

[17] E. Tola, V. Lepetit, and P. Fua, "Daisy: An efficient dense descriptor applied to wide-baseline stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 5, pp. 815–830, 2009.

[18] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "Kaze features," in *European Conference on Computer Vision (ECCV)*, 2012, pp. 214–227.

[19] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision (ECCV)*, 2006, pp. 430–443.

[20] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2564–2571.

[21] K. Lenc and A. Vedaldi, "Learning covariant feature detectors," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 100–117.

[22] X. Zhang, F. X. Yu, S. Karaman, and S.-F. Chang, "Learning discriminative and transformation covariant local feature detectors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4923–4931.

[23] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Toward geometric deep slam," *arXiv preprint arXiv:1707.07410*, 2017.

[24] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 4826–4837.

[25] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "Lf-net: Learning local features from images," *arXiv preprint arXiv:1805.09662*, 2018.

[26] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, 2018, pp. 337–33 712.

[27] X. Shen, C. Wang, X. Li, Z. Yu, J. Li, C. Wen, M. Cheng, and Z. He, "Rf-net: An end-to-end image matching network based on receptive field," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8124–8132.

[28] Z. Luo, L. Zhou, X. Bai, H. Chen, J. Zhang, Y. Yao, S. Li, T. Fang, and L. Quan, "Aslfeat: Learning local features of accurate shape and localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6588–6597.

[29] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 764–773.

[30] F. Hawary, T. Maugey, and C. Guillemot, "Sphere mapping for feature extraction from 360 fish-eye captures," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2020, pp. 1–6.

[31] Y.-C. Su and K. Grauman, "Kernel transformer networks for compact spherical convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9434–9443.

[32] C. Fernandez-Labrador, J. M. Facil, A. Perez-Yus, C. Demonceaux, J. Civera, and J. J. Guerrero, "Corners for layout: End-to-end layout recovery from 360 images," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1255–1262, 2020.

[33] B. Coors, A. P. Condurache, and A. Geiger, "Spherenet: Learning spherical representations for detection and classification in omnidirectional images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 525–541.
[34] K. Tateno, N. Navab, and F. Tombari, "Distortion-aware convolutional filters for dense prediction in panoramic images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 732–750.
[35] Q. Zhao, C. Zhu, F. Dai, Y. Ma, G. Jin, and Y. Zhang, "Distortion-aware cnns for spherical images," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2018, pp. 1198–1204.
[36] Y.-C. Su and K. Grauman, "Learning spherical convolution for fast features from 360 imagery," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 529–539.
[37] M. Eder, M. Shvets, J. Lim, and J.-M. Frahm, "Tangent images for mitigating spherical distortion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 12 423–12 431.
[38] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical cnns," in *International Conference on Learning Representations (ICLR)*, 2018.
[39] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning so (3) equivariant representations with spherical cnns," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 54–70.
[40] G. Potje, R. Martins, F. Chamone, and E. Nascimento, "Extracting deformation-aware local features by learning to deform," in *Advances in Neural Information Processing Systems (NIPS)*, 2021, pp. 10 759–10 771.
[41] C. Curto, V. Itskov, A. Veliz-Cuba, and N. Youngs, "The neural ring: An algebraic tool for analyzing the intrinsic structure of neural codes," *Bulletin of Mathematical Biology*, vol. 75, 06 2013.
[42] C. Curto, E. Gross, J. Jeffries, K. Morrison, M. Omar, Z. Rosen, A. Shiu, and N. Youngs, "What makes a neural code convex?" *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 222–238, 2017.
[43] Y. Zhang, Y. Liu, J. Liu, J. Miao, A. Argyriou, L. Wang, and Z. Xu, "360-attack: Distortion-aware perturbations from perspective-views," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 15 015–15 024.
[44] M. Blondel, O. Teboul, Q. Berthet, and J. Djolonga, "Fast differentiable sorting and ranking," in *International Conference on Machine Learning (ICML)*, 2020, pp. 950–959.
[45] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International Conference on Machine Learning (ICML)*, 2016, pp. 2990–2999.
[46] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 27, no. 10, pp. 1615–1630, 2005.
[47] Y. Zhang, S. Song, P. Tan, and J. Xiao, "Panocontext: A whole-room 3d context model for panoramic scene understanding," in *European Conference on Computer Vision (ECCV)*, 2014, pp. 668–686.
[48] Ó. M. Mozos, K. Nakashima, H. Jung, Y. Iwashita, and R. Kurazume, "Fukuoka datasets for place categorization," *The International Journal of Robotics Research*, vol. 38, pp. 507 – 517, 2019.
[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representation (ICLR)*, 2015.
[50] S. Ji, Z. Qin, J. Shan, and M. Lu, "Panoramic slam from a multiple fisheye camera rig," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 159, pp. 169–183, 2020.
[51] P. K. Lai, S. Xie, J. Lang, and R. Laganière, "Real-time panoramic depth maps from omni-directional stereo images for 6 dof videos in virtual reality," in *IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 2019, pp. 405–412.

Yunjian Zhang received the B.S. degree in Northeastern University, China, in 2017, and now he is currently working toward the Ph.D. degree in Institute of Information Engineering, Chinese Academy of Sciences. His research interests include omnidirectional vision and multimodal fusion.

Yanwei Liu received the B.S. degree in applied geophysics from Jianghan Petroleum University, China, in 1998, the M.S. degree in computer science from China Petroleum University (Beijing) in 2004 and the Ph.D. degree in computer science from Institute of Computing Technology, Chinese Academy of Sciences in 2010. Currently, he is an Associate Professor with the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include omnidirectional vision, multiview/3D video/VR video processing, multimedia networking and digital twin. He serves in the TPC of several international conferences in the area of computer vision, multimedia, communications, and networking.

Jinxia Liu received the B.S. degree and M.S. degree in physics from Harbin Normal University, China, in 1994, and 2005, respectively. In 2005, she joined the Zhejiang Wanli University. Currently, she is a Professor with Zhejiang Wanli University. Her research interests include laser imaging, digital image/video processing, multiview and 3D video coding, and wireless communication.

Antonios Argyriou received the Diploma in electrical and computer engineering from Democritus University of Thrace, Greece,and the M.S. and Ph.D. degrees in electrical and computer engineering as a Fulbright scholar from the Georgia Institute of Technology, Atlanta, USA. Currently, he is an Associate Professor at the department of electrical and computer engineering, University of Thessaly, Greece. From 2007 until 2010 he was a Senior Research Scientist at Philips Research, Eindhoven, The Netherlands. Dr. Argyriou serves in the TPC of several international conferences and workshops. His research interests are in the areas of communication systems, and statistical signal processing.

Liming Wang received his Ph.D. degree from the Institute of Software, Chinese Academy of Sciences, Beijing, China in 2007. He is now a professor and working at the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. His current research interests include network security.

Zhen Xu received his Ph.D. degree from the Institute of Software, Chinese Academy of Sciences, Beijing, China in 2005. He is now a professor and a director of research center in the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. His research interests include network and system security.

Xiangyang Ji (Senior Member, IEEE) received the B.S. degree in materials science and the M.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. In 2008, he joined Tsinghua University, Beijing, where he is currently a Professor with the Department of Automation, School of Information Science and Technology. His research interests include signal processing, image/video processing, image/video compression and communication, and 3D representation and reconstruction.