SphereCraft: A Dataset for Spherical Keypoint Detection, Matching and Camera Pose Estimation

Christiano Gava[1] Yunmin Cho[2] Federico Raue[1] Sebastian Palacio[1]
Alain Pagani[1] Andreas Dengel[1,3]
{christiano.gava,federico.raue,sebastian.palacio,alain.pagani,andreas.dengel}@dfki.de
min@aimmo.co.kr
[1] DFKI [2] Aimmo Germany GmbH [3] University of Kaiserslautern-Landau

(a) (b) (c) (d)

Figure 1. SphereCraft: we present a new dataset for spherical keypoint detection, matching and camera pose estimation. (a) Scene model and its 3D mesh. (b) Two rendered images from the scene in (a) (note the wide baseline and severe distortions on the second image), (c) their associated depth maps and (d) the ground truth keypoint correspondences colored according to confidence (red = high, blue = low).

Abstract

This paper introduces SphereCraft, a dataset specifically designed for spherical keypoint detection, matching, and camera pose estimation. The dataset addresses the limitations of existing datasets by providing extracted keypoints from various detectors, along with their ground truth correspondences. Synthetic scenes with photo-realistic rendering and accurate 3D meshes are included, as well as real-world scenes acquired from different spherical cameras. SphereCraft enables the development and evaluation of algorithms targeting multiple camera viewpoints, advancing the state-of-the-art in computer vision tasks involving spherical images. Our dataset is available at https://dfki.github.io/spherecraftweb/.

1. Introduction

Spherical images have gained significant attention in computer vision and deep learning due to their unique ability to provide information about the entire visible scene from a single vantage point. Previous studies have demonstrated the advantages of using panoramic images for scene understanding tasks, such as object detection [7, 35], semantic segmentation [5, 6] and room layout prediction [37], by leveraging the richer contextual information they offer.

However, current approaches rely on a single view to perform these tasks and neglect information provided by nearby views. We argue that reasoning about the underlying scene can be significantly improved when multiple images are available and the relative poses between the cameras are known. Therefore, *spherical* keypoint detection, matching and Structure from Motion (SfM) play a central role.

In this paper, we present SphereCraft, a novel dataset for keypoint detection, matching, and SfM on spherical images, which are critical for advancing the state-of-the-art in various computer vision tasks. Unlike existing datasets, SphereCraft includes extracted keypoints from a selection of popular handcrafted and learned detectors along with their ground truth correspondences, enabling researchers to develop and evaluate algorithms targeting multiple camera viewpoints. Following [17, 36], we leverage the computational power of graphics cards to generate photo-realistic scenes and overcome the limitations of scanning real-world scenarios. In addition, we provide highly accurate 3D meshes for all synthetic scenes (see Fig. 1). We also release a set of real scenes acquired with two different spherical cameras, thus providing both synthetic and real data for training and test-

ing. Finally, we propose a standard set of rendered scenes and a train-test split, facilitating evaluation and comparison of different approaches within the research community.

While this paper focuses primarily on keypoint detection and matching on spherical images, we emphasize the broader applicability of SphereCraft in diverse research areas, including single or multi-view depth prediction [18, 32, 33], geometric dense 3D reconstruction [25], indoor room layout prediction [35, 37], spherical novel view synthesis [14], and spherical light fields [13]. To encourage collaboration and reproducibility, we use free software [11] and openly accessible scene models, along with the code required to render images, depth maps, and camera poses.

2. Related Work

We start with an overview of relevant indoor and outdoor datasets. Then, we justify why none of them qualify as a dataset for spherical keypoint matching. Finally, we show that SphereCraft fulfills all necessary requirements.

Indoor Datasets. Using a Matterport camera [22], Armeni *et al*. [5] created the 2D-3D-S dataset. It comprises 6 large indoor areas aiming to support learning perception and the development of embodied agents. It was later included in the GibsonV2 dataset [16]. The Matterport3D dataset [6] captured 90 different rooms and along with [16] can be obtained through the Pano3D project [3]. Alternatively, one could use Replica [28]. Similar to [16], it supports the development of embodied agents and is compatible with AI Habitat [21, 29]. Nonetheless, Replica has many drawbacks and the provided SDK does not natively support the rendering of spherical images.

Instead of scanning real environments, [17, 31, 36] opted for completely synthetic scenes to bypass the effort of acquiring real-world data. Li *et al*. [17] created a photo-realistic renderer (ExaRenderer) to produce the InteriorNet dataset. The Structured3D dataset [36] aims to replace the human annotator by leveraging global or semi-global structures and symmetries. Won *et al*. [31] aim at omnidirectional stereo depth estimation. For that they used Blender [11] to create the OmniThings and OmniHouse datasets. However, the resolution of the images, among other limitations, is too low.

Addressing object detection, [7, 35] show the benefits of using 360° field of view images. Zhang *et al*. [35] use 500 images from SUN360 [34] to propose a whole-room 3D context model that outputs 3D bounding boxes and the detected objects along with their semantic classes. Chou *et al*. [7] created and released the 360-Indoor dataset with more than 3000 spherical images annotated with bounding boxes.

Outdoor Datasets. Targeting semantic segmentation of typical urban environments, [24] released a set of 600 outdoor spherical images annotated with semantic masks. The SUN360 dataset [34] offers a vast collection of indoor and outdoor spherical images obtained from the Internet [1] and grouped into place categories.

Discussion. All datasets discussed above are valuable contributions, but lack the required data for spherical keypoint matching and SfM in one way or another. For instance, some datasets [7, 24, 34, 36] are "single image per scene", *i.e*. it is impossible to establish meaningful feature correspondences between pairs of images, hence they are not suitable for keypoint matching and SfM. InteriorNet [17] offers several camera trajectories rendered as panoramic images, but lack the required ground truth depth maps and camera poses. In other cases [6, 16], apart from missing camera poses, the depth maps are not accurate enough to determine ground truth keypoint correspondences. Moreover, textures are often blurred [2] and impair keypoint detection and description.

The task of keypoint matching networks is to find correspondences between two sets of features extracted from a pair of images. To prevent bias and achieve robustness, the training data must contain a wide range of geometric transformations. In the context of spherical images, these transformations cause strong distortions in the distribution of the keypoint locations, which must be handled by the neural network. Even though [6, 16, 17] offer a large number of scenes, their panoramic images lack the required range of distortions. We offer an unprecedented wide range of geometric transformations, as described in Sec. 3.1. Additionally, considering the wide baseline between images in [6, 16], the amount of *meaningful* image pairs—*i.e*. those sharing a minimum number of correspondences—is significantly smaller than that provided by SphereCraft, which thanks to the use of anchor and satellite cameras, provides over 1.5M image pairs for training (see Table 2).

SphereCraft. The proposed dataset solves all aforementioned issues. We generate indoor and outdoor synthetic scenes with high-resolution RGB spherical images along with their depth maps and ground truth camera poses (Sec. 3.1). A selection of popular handcrafted and learned keypoints is then extracted from each image (Sec. 3.3) and accurate ground truth keypoint correspondences are established (Sec. 3.4). A highly accurate 3D mesh from each synthetic scene is also included. The resulting data (RGB images, depth maps, camera poses, 3D meshes, keypoints and their correspondences) is released along with a suggested train-test split to allow future approaches to be trained and evaluated on the same data. Additionally, we release all Blender [11] projects along with code so other researchers can easily render the same scenes at different resolutions or create their own version of the data according to their needs. Finally, we include a set of 9 real-world scenes acquired with two different spherical cameras for the evaluation on real data. Keypoints (handcrafted and learned) were extracted

[1] https://www.360cities.net/de

[2] https://aspis.cmpt.sfu.ca/scene-toolkit/scans/matterport3d/houses

Dataset	Keypoints	GT Correspondences	Depth Maps	GT Camera Poses	Indoors/Outdoors	Synthetic/Real	Semantic Masks	3D Meshes	Num. Scenes
Matterport3D [6]	-	-	1024x512	-	Indoor	Real	✓	✓	90
GibsonV2 [16]	-	-	1024x512	-	Indoor	Real	✓	✓	572
Replica [28]	-	-	-	✓	Indoor	Real	✓	✓	18
InteriorNet [17]	-	-	-*	-	Indoor	Synthetic	✓	-	704✠
Structured3D [36]	-	-	1024x512	-	Indoor	Synthetic	✓	-	3500⋆
SUN360 [34]	-	-	-	-	**Both**	Real	-	-	unknown⋆
360-Indoor [7]	-	-	-	-	Indoor	Real	-	-	3335⋆
CVRG-Pano [24]	-	-	-	-	Outdoor	Real	✓	-	600⋆
SphereCraft (**ours**)	✓	✓	**2048x1024**	✓	**Both**	**Both**†	-	✓	30

Table 1. Comparison of related datasets (GT = ground truth). SphereCraft is the only dataset with detected keypoints and their ground truth correspondences. In contrast to 16-bit depth maps from [36], like [6, 16] we offer 32-bit float depth maps, but at four times the resolution. * It is possible to render spherical depth maps with ExaRenderer [17], but they are not provided. ✠This is the number of rooms. The actual number of scenes may be significantly smaller as one scene may contain several rooms. ⋆ "Single image per scene" datasets. See text for details. †Although we provide synthetic and real scenes, the latter is reserved for testing as only images and keypoints are available for them.

and are also made available. Table 1 summarizes the differences between the discussed datasets.

3. Dataset Description

In this section, we provide details on how the scenes included in our dataset were rendered or captured. We also explain the strategy used to extract keypoints and map them onto the sphere, as well as how ground truth keypoint correspondences are established (in the case of synthetic scenes).

3.1. Synthetic Scenes

Our dataset comprises 21 synthetic scenes of different types, sizes, and complexity. A sample from each scene is shown in Fig. 5. In the following, we explain in detail how data from these scenes is created. For simplicity, we omit the index identifying the scene.

Anchor and satellite cameras. We start by manually placing a set of cameras throughout a scene to homogeneously cover it. We refer to this initial set as *anchor* cameras. For each anchor camera, we randomly generate a set of *satellite* cameras in its vicinity. Akin to data augmentation, the idea is to automatically produce several novel views of the scene from many different positions and orientations. With this, we generate the necessary data to train deep models robust to geometric transformations, especially regarding—but not limited to—spherical keypoint matching.

The procedure to create satellite cameras is as follows. Using the position of an anchor camera as the starting point, a random direction—expressed as unit 3D vector—and a random displacement in the range $(0, r_{max}]$ are selected to determine the location of the satellite camera. Next, using the orientation of the same anchor camera as reference, we combine rotations around x, y and z axes to define the orientation of the satellite camera. While rotations around the z axis (vertical) are randomly selected in the range $[-\pi, \pi]$, rotations around x and y axes are limited to the range $[\frac{-\pi}{4}, \frac{\pi}{4}]$. This yields images with severe distortions and is sufficient to represent the deformations resulting from geometric transformations, as shown in Fig 2. Together, the new position and

Figure 2. Anchor image (top left) and 3 (out of 9) of its satellites.

orientation define the pose of the satellite camera. This is repeated for the desired number of satellites before moving on to the next anchor camera. In summary, it creates a set of satellite cameras with random poses distributed within a sphere of radius r_{max} centered at the anchor camera, forming a *cluster*. For N_A anchors and N_S satellites per anchor, this procedure generates a total of $N = N_A(1 + N_S)$ cameras per scene. While N_A depends on the target scene, $N_S = 9$ is kept at all times. As a result, all scenes are covered by N_A possibly overlapping clusters of $(1 + N_S) = 10$ cameras.

Rendering. For each of the N cameras in a scene, we use Blender [11] to render a full panoramic ($360° \times 180°$) RGB image I^i and depth map D^i, with $i = \{0, .., N - 1\}$. To ensure accuracy, depth maps are stored as lossless 32-bit float `exr` images [15]. To strike a good balance between resolution and total size of the dataset, we chose to render all images and depth maps at 2048×1024 pixels. However, all Blender projects are included in the dataset. Along with each project, we also provide one or more configuration files. Together, Blender projects and configuration files enable researchers to render their own version of a specific scene or the entire dataset by simply manipulating the configuration files. See Supplemental Material for more details.

Camera poses. The pose of each camera C^i is given by $[R^i|t^i]$, where R^i and t^i are the rotation matrix and trans-

lation vector, respectively. In this paper, camera poses are used to establish ground truth keypoint correspondences and evaluate two-view pose estimation, but can also be used in multi-view pose estimation, novel-view synthesis, etc.

Local neighborhood. For each camera, we compute a set of neighbor cameras based on two factors: distance and number of ground truth keypoint correspondences. We first select a subset of cameras whose distance to the target camera is less than r_n. Next, we remove from this subset all cameras for which the number of ground truth keypoint correspondences with the target camera is below a threshold τ_n. While r_n is adjusted according to the scene, we keep $\tau_n = 100$ fixed. The remaining cameras are then regarded as neighbors of the target camera. This local camera neighborhood prevents training neural networks using several camera pairs that are unrelated, *i.e*. do not share enough visual information about the underlying scene, and would increase training time without necessarily improving the model.

This is especially useful in large and/or complex scenes, *e.g*. cameras in different rooms in a multiroom apartment or on opposite sides of a large outdoor scene. In the case of small scenes, r_n is chosen so that all or nearly all cameras fall within the desired range.

Train-test split. For each synthetic scene, we report its type (indoor/outdoor/mixed), the total number of cameras N, the scene area, camera density (ratio between number of cameras and area) and a score defined as the ratio between the average number of neighbor cameras—computed as described above—and the total number of cameras in the scene. We refer to this score as Normalized Direct Connectivity (NDC). While camera density conveys a sense of coverage, NDC encodes scene scale (size) and complexity and can be used to infer how challenging for multi-view SfM a scene is. The reason is as follows. For scenes with relatively wide open areas, typically outdoors or rooms with simple layout, the majority of cameras within r_n from a target camera will be regarded as its neighbors as they share significant visual information of their surroundings. In this case, smaller scenes tend to have higher NDC values whereas large scenes normally display lower NDC. On the other hand, for scenes with many rooms, narrow passages such as doors and corridors or cluttered environments, many cameras within r_n will be rejected due to a lower number of ground truth keypoint correspondences as they do not share enough visual information. This is a more challenging scenario for multi-view SfM and even small scenes may have low NDC values.

Based on the measurements outlined above, we suggest a train-test split. The idea is to prevent biases in training or testing data and to offer a standard and easy way for researchers to compare different learned models on our dataset. Tables 2 and 3 present the training and testing scenes, respectively.

Scene	Type	N	Num. Pairs	Area	Density	NDC
Bank	I	930	91,037	264.92	3.51	0.21
Barbershop	I	80	2652	35.39	2.26	0.83
Berlin	I	280	28,608	30.92	9.06	0.73
Classroom	I	370	57,073	51.10	7.24	0.83
Garage	M	1090	31,702	2823.37	0.39	0.05
Italian Flat	I	270	25,486	52.60	5.13	0.70
Kartu	I	640	65,687	70.87	9.03	0.32
Lone Monk	M	670	25,571	588.52	1.14	0.11
Medieval Port	O	2160	111,151	2093.38	1.03	0.05
Passion	I	600	56,545	95.74	6.27	0.31
Rainbow	I	930	47,755	2325.75*	0.40	0.11
Seoul	I	330	17,414	35.74	9.23	0.32
Simple	I	310	29,886	60	5.17	0.62
Urban Canyon	O	4090	870,718	133.84*	30.56	0.10
Vitoria	I	550	54,292	105.01	5.24	0.36
Warehouse	I	900	48,595	798.74	1.13	0.12

Table 2. Synthetic **training** scenes: we report type (I = Indoor, O = Outdoor, M = Mixed), total number of cameras N, number of training camera pairs, area (in m^2), density (cameras / m^2) and NDC. In total we provide over 1.5M training image pairs.
*Scales used for Rainbow and Urban Canyon in their original Blender projects are unfortunately over- and undersized, respectively. We included their areas for the sake of completeness. This is, however, not an issue as SfM is anyway up to scale.

Scene	Type	N	Num. Pairs	Area	Density	NDC
Tokyo	I	90	3976	15.22	5.91	0.98
Harmony	I	380	54,577	44.43	8.55	0.76
Shapespark	I	860	184,764	119.20	7.21	0.50
Showroom	I	1340	134,701	295.87	4.53	0.15
Middle East	M	4300	211,750	4634.98	0.93	0.02

Table 3. Synthetic **testing** scenes. See Table 2 for details.

3.2. Real Scenes

Along with synthetic scenes, we provide another 9 real scenes, captured with Civetta [2] and Ricoh Theta-S [8] cameras. They contain 4 indoor and 5 outdoor scenes of different sizes and complexity, with resolutions considerably higher than the synthetic images. Images captured with Civetta are 7070 $\times$ 3535 pixels, whereas those acquired with Theta-S are 5376 $\times$ 2688 pixels. Fig. 6 displays a representative image from each real scene. Unlike synthetic scenes, here ground truth depth maps, camera poses and keypoint correspondences are not available, but we provide keypoints extracted at the resolutions aforementioned.

3.3. Keypoint Detection

For all scenes in our dataset (synthetic and real), we extract both handcrafted and learned keypoints: Sift [20], Akaze [4], KP2D [30] and SuperPoint [9]. However, none of these keypoint detectors were designed for spherical images. Applying them directly to the equirectangular images would certainly yield suboptimal keypoints mostly due to the strong distortions near the poles. In the following, we describe a generic strategy that allows detecting virtually any keypoint

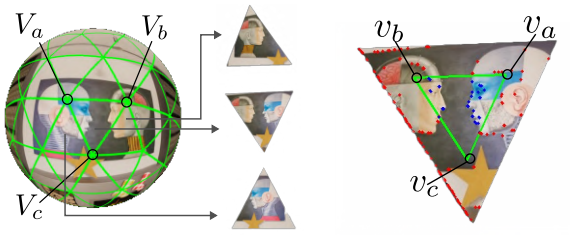


Figure 3. Shard images (left) and local keypoint detection (right). Blue and red dots indicate inlier and outlier keypoints, respectively.

on the sphere, especially suited for high-resolution images.

Shard images. To ensure that a variety of keypoints can be reliably detected on the sphere, we adopt a divide and conquer strategy. We follow [10] and produce a tessellation of the unit sphere to approximate the spherical image with multiple perspective images computed as local planar projections. We refer to these local projections as *shard* images. They are then fed into each keypoint detector to extract local keypoints that are later mapped back onto the sphere.

The advantages of this strategy are threefold. First, it practically removes the influence of distortion on keypoint detection, making it possible to include in this dataset both traditional (handcrafted) and recent learn-based detectors that have been originally devised for perspective images, such as those listed above. Second, it leads to an uniform and consistent way to extract keypoints, allowing the extension of this dataset by adding other keypoint detectors as new approaches are proposed. Finally, it is scalable. While handcrafted keypoint detectors scale well with image resolution, current learn-based models face technical issues when the resolution of the images increases. They tend to allocate significant amounts of hardware resources—especially GPU RAM—and become impractical if applied directly on the full image. Our tessellation algorithm automatically adapts to the resolution of the spherical images, increasing the number of subdivisions as necessary to keep the shard images free of distortion and limited in size.

The tessellation algorithm produces vertices on the unit sphere, which in turn define triangular facets. Each facet is used to compute a corresponding shard image. As illustrated in Fig. 3, consider a facet formed by vertices V_a, V_b and V_c on the surface of the unit sphere. The centroid of these vertices defines the tangent point for the plane that will contain the associated shard image. However, before projecting the facet onto this tangent plane, it is necessary to ensure that keypoints lying near the border of the facet have enough visual context (pixel neighborhood) so that their descriptors can be properly computed. Then, we define a "descriptor border" as a sufficiently wide region surrounding the facet. The projection of the facet and the descriptor border onto the tangent plane creates the shard image.

Note that as long as the resolution of the spherical images remains constant, the tessellation only needs to be computed once. The resulting vertices are then reused to generate shard images for all available spherical images—for instance, in a given scene rendered or captured with the same camera.

Local keypoint detection. Each shard image is processed independently by the keypoint detectors. We used OpenCV to extract Sift [20] and Akaze [4] keypoints. For KP2D [30] we used the code released by the authors. SuperPoint [9] keypoints are extracted using two implementations: the Pytorch [26] demo provided by the authors and an alternative TensorFlow [1] implementation [3].

Filtering. Naturally, when extracting keypoints on shard images, some will be located in the descriptor border, *i.e.* they fall outside the facet. Fortunately, it is straightforward to filter them out. If a keypoint is inside the triangle formed by v_a, v_b and v_c—the projections of the facet vertices V_a, V_b and V_c—a keypoint is regarded as inlier (shown in blue in Fig. 3); else it is an outlier (red). This simple but effective procedure guarantees that only keypoints lying inside the original triangular facet are mapped onto the sphere and prevents duplicated keypoints.

Mapping keypoints onto the sphere. Once keypoints are detected and filtered on all shard images, they are mapped onto the sphere. Essentially, keypoint locations are back-projected from shard images to the sphere. This is achieved by associating each shard image with the vertices (V_a, V_b and V_c in Fig. 3) used to create it so that the locations of the keypoints can be described in the spherical camera coordinate system. Finally, the resulting keypoint locations are converted to spherical coordinates and become independent of the image resolution. Hence, keypoints extracted from images with different resolutions can be seamlessly integrated into training of learn-based models.

Non-maxima Suppression. The last step is to account for keypoints that are too close to each other and are thus redundant and ambiguous. This can happen for two reasons: 1. the keypoint detector itself does not perform non-maxima suppression; and/or 2. keypoints detected on neighbor shard images land close to each other after the mapping performed in the previous step. We then apply non-maxima suppression directly on the unit sphere using the geodesic distance between keypoints and their associated scores (strength or keypoint response). For the geodesic distance we use a threshold of 15.34×10^{-3} radians, equivalent to 5 pixels at 2048×1024 resolution.

3.4. Ground Truth Keypoint Correspondences

Computation of ground truth keypoint correspondences requires depth maps and pose of both cameras. Hence, this computation is performed exclusively for the synthetic scenes. Assuming the distance between a pair of cameras C^i

[3] https://github.com/rpautrat/SuperPoint

Architecture	Tokyo			Harmony			Shapespark			Showroom			Middle East		
	MS	P	R	MS	P	R	MS	P	R	MS	P	R	MS	P	R
SuperGlue [27]	21.49	43.44	22.94	26.03	42.33	27.66	21.04	27.36	23.52	23.12	40.65	24.21	34.89	46.58	36.33
SphereGlue [12]	**44.87**	**59.10**	**51.00**	**55.85**	**65.40**	**62.05**	**53.65**	**65.89**	**58.96**	**51.16**	**66.24**	**56.20**	**64.62**	**71.88**	**68.74**

Table 4. Matching Score (MS), Precision (P) and Recall (R), in percent, measured on the proposed synthetic testing scenes (see Table 3) using the SuperPoint [9] detector. SphereGlue consistently outperforms SuperGlue as it properly models keypoints on the sphere.

Architecture	Tokyo			Harmony			Shapespark			Showroom			Middle East		
	5°	10°	20°	5°	10°	20°	5°	10°	20°	5°	10°	20°	5°	10°	20°
SuperGlue [27]	83.28	88.42	92.36	74.66	80.60	85.93	36.31	42.10	51.07	63.47	70.05	76.39	81.83	84.00	86.33
SphereGlue [12]	**89.35**	**93.64**	**96.62**	**94.20**	**96.64**	**98.64**	**87.59**	**92.84**	**97.27**	**77.34**	**83.53**	**88.92**	**93.83**	**94.98**	**95.91**

Table 5. AUC of the two-view pose error, in percent, measured on the proposed synthetic testing scenes (see Table 3) using SuperPoint [9]. SphereGlue's performance is relatively independent of the scene whereas SuperGlue performs comparatively poorly in Shapespark.

and C^j is below r_n (see Section 3.1), ground truth keypoint correspondences are established as follows.

To simplify this explanation, consider a single keypoint κ^i detected on image I^i as described in Section 3.3. Denoting keypoint coordinates as unit Cartesian vectors $\mathbf{x} = (x, y, z)$, the first step is to sample the depth map D^i and recover κ^i's depth value d^i. The corresponding 3D point is obtained as $\mathbf{P}^i = d^i\mathbf{x}^i$. The next step is to project $\mathbf{P}^i$ onto C^j, which is achieved using the relative pose between C^i and C^j given as $[\mathbf{R}^{ji}|\mathbf{t}^{ji}]$, with $\mathbf{R}^{ji} = \mathbf{R}^j\mathbf{R}^{iT}$ and $\mathbf{t}^{ji} = \mathbf{t}^j - \mathbf{R}^{ji}\mathbf{t}^i$. The projection yields a point $\boldsymbol{\rho}^{ji}$ on the surface of the unit sphere computed as

$$\boldsymbol{\rho}^{ji} = \frac{\mathbf{P}^{ji}}{\|\mathbf{P}^{ji}\|} = \frac{\mathbf{R}^{ji}d^i\mathbf{x}^i + \mathbf{t}^{ji}}{\|\mathbf{R}^{ji}d^i\mathbf{x}^i + \mathbf{t}^{ji}\|}. \quad (1)$$

Akin to a radius search, an angular deviation ω (shown in yellow in Fig. 4) is used to identify possible keypoint correspondences among those extracted from I^j. The coordinates of all keypoints detected on I^j are embedded in a 3D KDTree [23] for fast nearest neighbor search. Then, if no keypoint is found within the region defined by ω and centered at $\boldsymbol{\rho}^{ji}$, κ^i has no correspondence in image I^j. This happens due to occlusion or failure of the keypoint detector. If, however, a candidate κ^j is found, before assigning it as the ground truth correspondence of κ^i, it is necessary to apply the occlusion filter, which is described as follows. The depth map D^j is sampled to recover κ^j's depth value d^j and the 3D point $\mathbf{P}^j = d^j\mathbf{x}^j$ is determined. If $\|\mathbf{P}^{ji} - \mathbf{P}^j\|$ is smaller than a threshold δ, κ^i and κ^j are regarded as correspondences (Fig. 4-(a)); else $\mathbf{P}^i$ is considered occluded in I^j (Fig. 4-(b)). This process is repeated for all keypoints detected in image I^i. It may happen that the same keypoint κ^j is assigned as ground truth correspondence to more than one keypoint in image I^i. These duplicates are then removed.

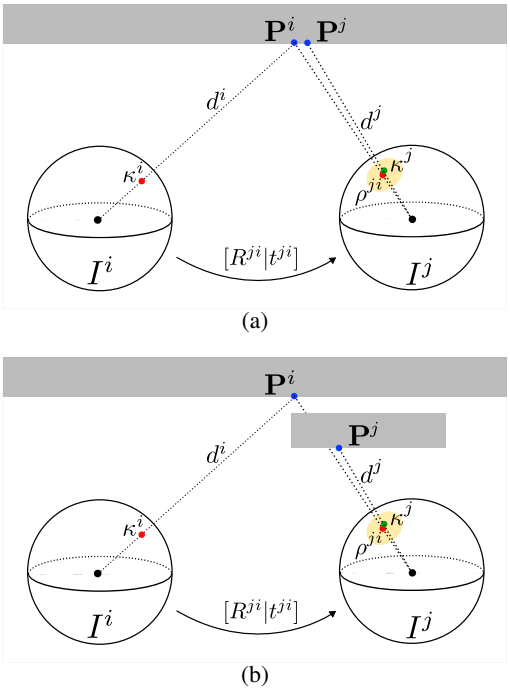


Figure 4. Computation of ground truth keypoint correspondences with occlusion filtering. (a) A match is found. (b) No match is found as the associated 3D point is occluded on the target image.

4. Evaluation

The goal of this section is twofold. First, we aim to demonstrate the importance of a keypoint matcher dedicated to spherical images (and in a broader sense, of algorithms that properly handle the unique characteristics of spherical images, such as, but not limited to, neighborhood across image borders and severe distortion under geometric transformations). Second, to provide baseline evaluations so that future approaches can be compared to existing methods.

We then evaluate two keypoint matching approaches on the synthetic test scenes (see Table 3), namely SuperGlue [27] and SphereGlue [12] using pre-trained weights provided by the respective authors. The former is the state-of-the-art attention-based approach designed primarily for perspective images. We use, however, the *outdoor* weights, *i.e.* those obtained by training SuperGlue on the MegaDepth


Bank (930)


Barbershop (80)


Berlin (280)


Classroom (370)


Garage (1090)


Harmony (380)


Italian Flat (270)


Kartu (640)


Lone Monk (670)


Medieval Port (2160)


Middle East (4300)

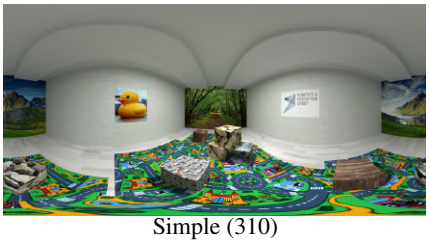

Passion (600)

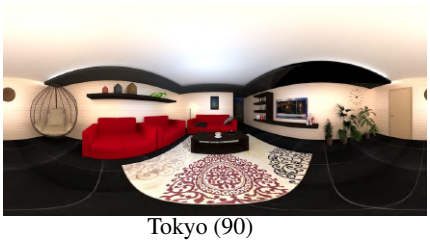

Rainbow (930)


Seoul (330)


Shapespark (860)


Showroom (1340)


Simple (310)


Tokyo (90)


Urban Canyon (4090)


Vitoria (550)


Warehouse (900)

Figure 5. Sample images from each synthetic scene. Number of images indicated in parenthesis.

Figure 6. Sample images from each real scene. Number of images indicated in parenthesis. See Supp. Material for more details.

dataset [19] as they yield better results for panoramic images than the indoor weights. SphereGlue is a recent approach inspired by SuperGlue but dedicated exclusively to spherical images. Here we use only SuperPoint keypoints [9] as SuperGlue pre-trained weights exist only for this detector.

We report Matching Score (MS), Precision (P) and Recall (R) in Table 4. Table 5 summarizes the results obtained by measuring the area under the curve (AUC) of the two-view pose error at $5°$, $10°$ and $20°$. The pose error is measured as the maximum angular deviation between rotation and translation. All values in Tables 4 and 5 were calculated using the ground truth keypoint correspondences (see Section 3.4).

Even though SphereGlue outperforms SuperGlue — as expected when evaluated on spherical images — its performance is lower than that reported by the authors. A possible explanation is the wider baseline and stronger distortions present in our dataset. Also, SphereGlue was trained on a relatively small set of 15k image pairs whereas our training set is two orders of magnitude bigger (see Table 2).

5. Societal Impact

This work focuses on a dataset of everyday scenes, in an effort to advance research about fundamental computer vision problems. As such, there is no immediate threat posed by these scenes for individuals or organizations. Images were properly anonymized and do not depict any human or personal data that can be associated to any particular person.

6. Conclusion

We believe that tasks related to scene understanding can be further improved when aggregating information from nearby views. This can be achieved with spherical SfM, which in turn depends on keypoint detection and matching.

Therefore, we introduced SphereCraft, a new dataset to support the development of data-driven models for spherical keypoint detection, matching and SfM. It consists of 21 synthetic and 9 real scenes. For each real scene, we provide high-resolution RGB images along with several detected keypoints (handcrafted and learned). For each synthetic scene, in addition to the RGB images and keypoints, we also provide ground truth camera poses, high-resolution and highly accurate depth maps, keypoint correspondences and 3D meshes. We propose a train-test split seeking to strike a good balance between what the networks see during training and testing time. We then used the testing set to provide a baseline evaluation of two learn-based keypoint matchers.

Future work includes evaluation of existing models for keypoint detection, depth prediction and multi-view camera pose estimation. We also consider adding other data modalities, such as spherical optical flow.

Acknowledgements

This work was supported by the projects SustainML (101070408) and XAINES (01IW20005).

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. Software available from tensorflow.org. 5

[2] Weiss AG. Civetta. https://weiss-ag.com/civetta360camera/. Retrieved June, 2023. 4

[3] Georgios Albanis, Nikolaos Zioulis, Petros Drakoulis, Vasileios Gkitsas, Vladimiros Sterzentsenko, Federico Alvarez, Dimitrios Zarpalas, and Petros Daras. Pano3D: A Holistic Benchmark and a Solid Baseline for 360° Depth Estimation. *CoRR*, 2021. 2

[4] Pablo Fernández Alcantarilla, Jesús Nuevo, and Adrien Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In *BMVC*, 2013. 4, 5

[5] I. Armeni, A. Sax, A. R. Zamir, and S. Savarese. Joint 2D-3D-Semantic Data for Indoor Scene Understanding. *ArXiv e-prints*, Feb. 2017. 1, 2

[6] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D Data in Indoor Environments. *International Conference on 3D Vision (3DV)*, 2017. 1, 2, 3

[7] Shih-Han Chou, Cheng Sun, Chang Wen-Yen, Wan-TIng Hsu, Min Sun, and Jianlong Fu. 360-Indoor: Towards Learning Real-World Objects in 360° Indoor Equirectangular Images. In *WACV*, 2020. 1, 2, 3

[8] Ricoh Company. Ricoh Theta S. https://theta360.com/en/about/theta/s.html. Retrieved June, 2023. 4

[9] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-Supervised Interest Point Detection and Description. In *IEEE/CVPR*, pages 224–236. Computer Vision Foundation / IEEE Computer Society, 2018. 4, 5, 6, 8

[10] Marc Eder, Mykhailo Shvets, John Lim, and Jan-Michael Frahm. Tangent Images for Mitigating Spherical Distortion. In *IEEE/CVPR*, June 2020. 5

[11] Blender Foundation. Blender. http://www.blender.org/. Retrieved March, 2023. 2, 3

[12] Christiano Gava, Vishal Mukunda, Tewodros Habtegebrial, Federico Raue, Sebastian Palacio, and Andreas Dengel. SphereGlue: Learning Keypoint Matching on High Resolution Spherical Images. In *CVPR Workshops*, pages 6133–6143, June 2023. 6

[13] Christiano Gava, Didier Stricker, and Soichiro Yokota. Dense Scene Reconstruction from Spherical Light Fileds. In *ICIP*, October 2018. 2

[14] Tewodros Habtegebrial, Christiano Gava, Marcel Rogge, Didier Stricker, and Varun Jampani. SOMSI: Spherical Novel View Synthesis with Soft Occlusion Multi-Sphere Images. In *IEEE/CVPR*, pages 15725–15734, 2022. 2

[15] Industrial Light and Magic. *Reading and Writing OpenEXR Image Files with the IlmImf Library*, September 2006. 3

[16] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey Kurenkov, Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. iGibson 2.0: Object-Centric Simulation for Robot Learning of Everyday Household Tasks. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 455–465. PMLR, 08–11 Nov 2022. 2, 3

[17] Wenbin Li, Sajad Saeedi, John McCormac, Ronald Clark, Dimos Tzoumanikas, Qing Ye, Yuzhong Huang, Rui Tang, and Stefan Leutenegger. InteriorNet: Mega-scale Multi-sensor Photo-realistic Indoor Scenes Dataset. In *BMVC*, 2018. 1, 2, 3

[18] Yuyan Li, Zhixin Yan, Ye Duan, and Liu Ren. PanoDepth: A Two-Stage Approach for Monocular Omnidirectional Depth Estimation. In *3DV*, pages 648–658, 2021. 2

[19] Zhengqi Li and Noah Snavely. MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In *CVPR*, 2018. 8

[20] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004. 4, 5

[21] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019. 2

[22] Matterport. Matterport camera. https://matterport.com/. Retrieved June, 2023. 2

[23] Marius Muja and David G. Lowe. *FLANN - Fast Library for Approxmate Nearest Neighbors*, January 2013. 6

[24] Semih Orhan and Yalin Bastanlar. Semantic Segmentation of Outdoor Panoramic Images. *Signal, Image and Video Processing*, 2021. 2, 3

[25] Alain Pagani, Christiano Gava, Yan Cui, Bernd Krolla, Jean-Marc Hengen, and Didier Stricker. Dense 3D Point Cloud Generation from Multiple High-Resolution Spherical Images. In *VAST*, Prato, Italy, October 2011. 2

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NEURIPS*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[27] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning Feature

Matching with Graph Neural Networks. In *CVPR*, 2020. 6

[28] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard A. Newcombe. The Replica Dataset: A Digital Replica of Indoor Spaces. *CoRR*, 2019. 2, 3

[29] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. In *NeurIPS*, 2021. 2

[30] Jiexiong Tang, Hanme Kim, Vitor Guizilini, Sudeep Pillai, and Rares Ambrus. Neural Outlier Rejection for Self-Supervised Keypoint Learning. In *ICLR*, Addis Ababa, Ethiopia, April 2020. OpenReview.net. 4, 5

[31] Changhee Won, Jongbin Ryu, and Jongwoo Lim. OmniMVS: End-to-End Learning for Omnidirectional Stereo Matching. In *ICCV*, pages 8986–8995, 2019. 2

[32] Changhee Won, Jongbin Ryu, and Jongwoo Lim. Sweepnet: Wide-Baseline Omnidirectional Depth Estimation. In *IEEE/ICRA*, pages 6073–6079, 2019. 2

[33] Changhee Won, Jongbin Ryu, and Jongwoo Lim. End-to-End Learning for Omnidirectional Stereo Matching with Uncertainty Prior. *IEEE/PAMI*, 2020. 2

[34] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing Scene Viewpoint Using Panoramic Place Representation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2695–2702. IEEE, 2012. 2, 3

[35] Cheng Zhang, Zhaopeng Cui, Cai Chen, Shuaicheng Liu, Bing Zeng, Hujun Bao, and Yinda Zhang. DeepPanoContext: Panoramic 3D Scene Understanding With Holistic Scene Context Graph and Relation-Based Optimization. In *ICCV*, pages 12632–12641, 2021. 1, 2

[36] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3D: A Large Photo-Realistic Dataset for Structured 3D Modeling. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, volume 12354 of *Lecture Notes in Computer Science*, pages 519–535. Springer, 2020. 1, 2, 3

[37] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image. In *CVPR*, pages 2051–2059, 2018. 1, 2