IJCV manuscript No.
(will be inserted by the editor)

SPHORB: A Fast and Robust Binary Feature on the Sphere

Qiang Zhao · Wei Feng · Liang Wan[†] · Jiawan Zhang

Received: date / Accepted: date

Abstract In this paper, we propose SPHORB, a new fast and robust binary feature detector and descriptor for spherical panoramic images. In contrast to state-of-the-art spherical features, our approach stems from the geodesic grid, a nearly equal-area hexagonal grid parametrization of the sphere used in climate modeling. It enables us to directly build fine-grained pyramids and construct robust features on the hexagonal spherical grid, thus avoiding the costly computation of spherical harmonics and their associated bandwidth limitation. We further study how to achieve scale and rotation invariance for the proposed SPHORB feature. Extensive experiments show that SPHORB consistently outperforms other existing spherical features in accuracy, efficiency and robustness to camera movements. The superior performance of SPHORB has also been validated by real-world matching tests.

Keywords Spherical image · Binary feature · Panoramic image matching · Geodesic grid

Q. Zhao
School of Computer Science & Technology, Tianjin University
E-mail: qiangzhao@tju.edu.cn

W. Feng
School of Computer Science & Technology, Tianjin University
E-mail: wfeng@tju.edu.cn

L. Wan[†] (Corresponding Author)
School of Computer Software, Tianjin University and Tianjin Key Lab for Advanced Signal Processing, Civil Aviation University of China
E-mail: lwan@tju.edu.cn

J. Zhang
School of Computer Software, Tianjin University
E-mail: jwzhang@tju.edu.cn

The final publication is available at Springer via http://dx.doi.org/10.1007/s11263-014-0787-4.

1 Introduction

The past decade has witnessed the increasing trend of spherical panoramic images with wide-angle (up to 360°) field-of-view being more and more easily obtained for common users. This mainly attributes to the development of image stitching techniques and the maturity of economic panoramic imaging systems, such as the Ladybug cameras. By providing fields-of-view far beyond the conventional (planar) images, spherical panoramic images have been successfully applied in a number of recent new applications, including online street-level virtual navigation (Anguelov et al, 2010; Zhao et al, 2013), city-scale change detection (Taneja et al, 2013), scene recognition and view detection (Xiao et al, 2012), and 3D scene reconstruction (Micusik and Kosecka, 2009).

However, the largely increased volume and wide applications of spherical panoramic images are being confronted with the dilemma that feature detection and matching, a fundamental problem for many computer vision tasks, are far less studied than for the planar images. In the literature, some robust feature and matching algorithms designed for planar images have been directly applied on the unfolded latitude-longitude map (Valgren and Lilienthal, 2007) or on the piecewise perspectives (Micusik and Kosecka, 2009). These intuitive methods more or less suffer from deformation problems associated to the underlying spherical parameterizations. A recent notable stream attempts to build SIFT-like features on the spherical domain (Hansen et al, 2007, 2010; Hadj-Abdelkader et al, 2008; Cruz-Mota et al, 2012). Since Gaussian filtering on the sphere can be performed as a diffusion process through spherical Fourier transform, they construct scale-spaces on the sphere using the spherical harmonics. Theoretically, these features are plausible and can be effectively invariant to changes of camera pose and position. However, spherical harmonics usually require costly computation and suffer from inherent bandwidth limitation, which significantly weaken their feasibility in handling large-scale matching problems.

In contrast to the hysteretic development of spherical panoramic features, plenty of successful progresses in fast and robust feature detection and matching for planar images have been contributed by the computer vision community in recent years. One representative and exciting development is the binary features, like ORB (Rublee et al, 2011), BRISK (Leutenegger et al, 2011), and FREAK (Alahi et al, 2012), which simply leverage local neighborhood intensity comparison to construct binary strings to describe feature points. The balanced good performance makes binary features quickly gain prevalence in state-of-the-art image matching applications. The major desirable advantages of binary features are *speediness* (much faster than state-of-the-art, e.g. SIFT (Lowe, 2004) and SURF (Bay et al, 2008)), *compactness* (much shorter codes leading to much less memory consumption) and *accuracy* (comparable or even better matching precision to SIFT and SURF (Rublee et al, 2011; Alahi et al, 2012)). Their success inspires us to think about an interesting question: is it possible to rapidly detect and describe robust keypoints for high-definition spherical panoramic images using a binary feature? Clearly, all those attractive properties of planar binary features are certainly required in the spherical domain.

In this paper, we give a positive answer to this question by proposing a fast and robust binary feature, namely SPHORB, on the sphere. Our approach, for the first time (to the best of our knowledge), directly detects and describes binary keypoints in the uniformly parameterized sphere domain. It relies on the *geodesic grid*, a nearly equal-area hexagonal grid representation of the sphere, which has been successfully used as a global Earth reference in climate modeling (Randall et al, 2002). Based on this hexagonal spherical grid, we reinvent the FAST algorithm (Rosten and Drummond, 2006; Rosten et al, 2010) and the ORB feature (Rublee et al, 2011) in the spherical 6-neighborhood system, which results in SPHORB, a new spherical binary feature detector and descriptor. We further study how to achieve scale and rotation invariance for SPHORB. In summary, our main contributions are:

– The framework of using an equal-arc subdivision-based geodesic grid to build robust spherical features; and the analysis of its geometric properties, based on which the intensity comparison, the core of binary features, becomes valid and feasible.
– The construction of a scale-invariant spherical FAST corner detector, and the evaluation of its properties.
– The construction of an adapted ORB-like descriptor on the sphere, including the determination of spherical keypoint's orientation and the efficient computation of oriented spherical binary descriptors.

Extensive experiments have validated the superior performance of our approach. As evident in the results, the proposed SPHORB are much faster and more robust than Spherical SIFT (Cruz-Mota et al, 2012). Particularly, SPHORB consistently outperforms the state-of-art methods, including Spherical SIFT, and those applying planar feature matching methods on unrolled panoramic images, in accuracy, efficiency and robustness to camera movements. We also apply our methods to real-world matching tasks, i.e. the matching between two spherical panoramic images, and the matching between spherical panoramas and planar images. The encouraging results positively support the promising potential of the proposed SPHORB feature and the underlying spherical feature framework.

2 Related Work

2.1 Planar Feature Extractors

To extract features from planar images, we need a feature detector and a feature descriptor. The earliest well-known feature detector is Harris corner (Harris and Stephens, 1988), which is invariant to image rotation. Scale invariance is achieved in (Lowe, 2004) using DoG filter. Bay et al. (2008) used the determinant of the Hessian matrix to detect the key points. Rosten and Drummond (2006) proposed FAST detector based on intensity comparison, which is many times faster than previous detectors.

For the descriptors, the SIFT (Lowe, 2004) is one of the highest quality approaches under peer evaluation (Mikolajczyk and Schmid, 2005). However, its computing complexity and high dimensionality imposes a large computational burden. There is a series of descriptors like SURF (Bay et al, 2008) and CARD (Ambai and Yoshida, 2011) proposed to speed up SIFT, but they may not be fast enough for large-scale problems and real-time applications. As an alternative to SIFT, the BRIEF (Calonder et al, 2010) descriptor computes binary strings using intensity comparison, achieving a 100$\times$ speed-up over SIFT, however it is very sensitive to image rotation and scale changes. To address this problem, the ORB (Rublee et al, 2011) descriptor, combining a FAST corner detector with a rotation-invariant version of BRIEF, is developed. Another two similar binary features with scale invariance and rotation invariance are BRISK (Leutenegger et al, 2011) and FREAK (Alahi et al, 2012). In these two methods, the sampling patterns are composed of sampling points equally spaced on concentric circles surrounding the detected key point. Trzcinski et al (2013) developed a learning-based method that finds a low-dimensional yet highly discriminative binary descriptor. In our work, we port ORB to the spherical domain due to its efficiency and simplicity, while our method provides a framework in which other planar feature extractors can be embedded as well.

2.2 Spherical Feature Extractors

The feature matching algorithms designed for planar images have been applied on the unfolded latitude-longitude map (Valgren and Lilienthal, 2007). Since it is not geometrically correct, severe deformation near the polar regions will penalize the detection and description performance. Some researchers projected the spherical image into several perspective images on which planar detector and descriptor, e.g. SURF, are then applied (Micusik and Kosecka, 2009).

A recent research trend is carrying out spectral analysis of spherical panoramic images (Hadj-Abdelkader et al, 2008; Hansen et al, 2007, 2010; Cruz-Mota et al, 2012), with spherical harmonics as the basis functions. Then the Gaussian smoothing can be performed as a diffusion on the sphere through spherical Fourier transform (Bulow, 2004). Based on it, Hadj-Abdelkader et al. (2008) proposed a spherical Harris corner detector. Hansen et al. (2007) obtained spherical scale-space images, and extracted features by reprojecting back to the planar images. Recently, Cruz-Monta et al. (2012) presented an algorithm that both detects and describes the keypoints based on the spherical harmonics, which is referred as Spherical SIFT in our paper. It is noted that these SIFT-like algorithms usually suffer from inherent bandwidth limitation and heavy computational burden of spherical harmonics.

Another work closely related to ours describes a combination of Harris corner and planar SIFT descriptor on a bisection-based geodesic division of the sphere (Qin and Li, 2012). It adopts a complex storage format for the geodesic division, and the spherical image is reprojected to a tangential plane for the SIFT feature description. In contrast, in our work we build the geodesic grid by applying equal-arc subdivision, and directly detect and describe keypoints on the geodesic grid, enabling very fast running performance and high description quality.

In addition, there exist methods that compute the scale-space for large field-of-view images based on Riemannian geometry (Puig and Guerrero, 2011; Arican and Frossard, 2012). They are able to compute features directly on the input omnidirectional images without resampling. However they are designed for catadioptric images, which are limited to a hemispherical field-of-view. Our method can deal with all types of large field-of-view images, provided that the images are mapped to a sphere.

3 Overview of Our Method

As is well known, the binary descriptors of ORB, BRISK and FREAK are generated by intensity comparison [1] in pixel neighborhoods. For planar images, the intensity comparison asserts two assumptions: the pixels are uniformly distributed on the plane, and pixel neighborhoods share the same structure over the planar domain. The two assumptions, however, may not hold for the spherical case. For instance, when the spherical panorama is represented as the latitude-longitude map, the pixels distribute more densely in polar regions. One possible straightforward method to alleviate this deformation problem is locally projecting spherical neighborhood of each pixel into its tangential plane. Because lots of local patches should be projected, as demonstrated in our experiments, such strategy degrades the speed performance.

Our method supports directly detecting and describing keypoints on the sphere without extra interpolations. We build our method on a close hexagonal grid parameterization of the sphere, namely geodesic grid (Randall et al, 2002), which is widely used in climate modelling (Sec. 4). As discussed in Sec. 4.2, the geodesic grid well supports the two aforementioned assumptions. Based on the grid representation, we adapt FAST detection algorithm to locate keypoints (Sec. 5.1). In the description stage (Sec. 5.2), we build an ORB-like descriptor by respecting the hexagonal structure of the geodesic grid. We finally discuss how to achieve rotation invariance (Sec. 5.3).

4 Geodesic Grid for Spherical Features

In this section, we first introduce the structure of the geodesic grid, then analyze its properties relative to other common spherical parametrizations, and show that geodesic grid is suitable for our application. In the end, we describe how to conduct computation on the geodesic grid.

4.1 Grid Generation

To construct a geodesic grid, we begin with an icosahedron inscribed inside a unit sphere, which has 20 triangular faces and 12 vertices (Figure 1(a)). Each face is subdivided into finer resolutions; the new vertices are projected onto the unit sphere (Figure 1(c)). Next, Voronoi cells centered on the vertices are constructed, with each cell consisting of the neighboring points that are nearest to the centered vertex (Figure 1(d)). We finally get a geodesic grid composed of N_c grid cells, among which 12 cells are pentagons corresponding to the vertices of the icosahedron and the rest $N_c - 12$ cells are all hexagons.

It is noted that the grid structure is specifically determined by the subdivision scheme employed. In our work, we apply *equal-arc* subdivision (Williamson, 1968) to allow a broad variety of possible grid resolutions (Figure 1(b)). To be specific, each arc connecting the vertices of the icosahedron is subdivided into n segments, and the new vertices

[1] There exist binary descriptors that are not generated by intensity comparison, e.g. LDAHash (Strecha et al, 2012).

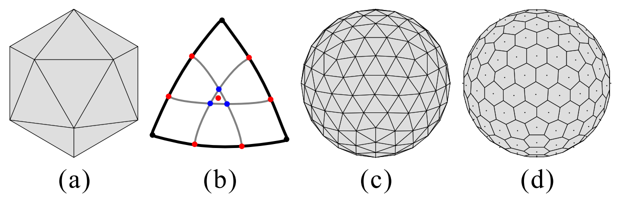


Fig. 1 Geodesic grid construction. (a) Icosahedron. (b) Equal-arc subdivision scheme generates a spherical triangle, whose centroid is taken as the sampled vertex. (c) The generated mesh with the subdivision level $n = 4$. (d) The geodesic grid that is dual to the mesh in (c).

are connected pairwise with great circles. Note that the corresponding three great circles do not intersect at the single point. The centroid of the resulting triangle is computed and taken as the sampled vertex. By this way, we can know that the number of grid cells N_c is $N_c = 10 \times n^2 + 2$.

4.2 Grid Property

The geodesic grid has nice geometric properties that facilitate its application in binary keypoint detection on the sphere. Firstly, all the grid cells at the same subdivision level have very similar solid angles, which means those cells are equally important. Secondly, the centers of neighboring cells are separated with similar geodesic distances when the grid resolution is fine enough. Therefore, we can safely regard the local neighborhood as being planar, and compute the planar distances between cells directly on the flatted geodesic grid rather than their geodesic distances on the sphere. Furthermore, the geodesic grid is rather regular: the grid cells are all hexagons except 12 pentagons. By omitting the pentagons, we can perform the same binary test on the entire sphere.

Since spherical panoramic images are usually represented as the latitude-longitude map and the cube map, we make a quantitative comparison to show the advantage of the geodesic grid. Specifically, we evaluate the first two properties in terms of the standard deviation of cell's area and in-between distance, respectively. The latitude-longitude map paramterizes the sphere by equally subdividing latitudes and longitudes. In the cube map, the sphere is projected onto a cube, with each side defining a 90° view frustum. Figure 2 clearly demonstrates the advantage of the geodesic grid, which has the smallest values at all the resolutions.

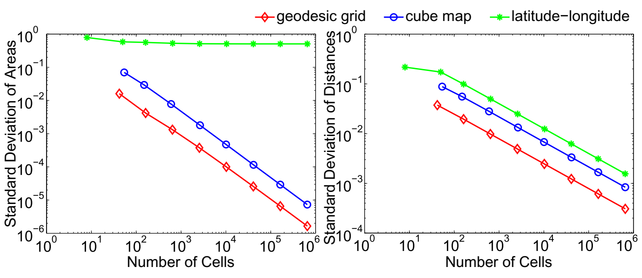


Fig. 2 Grid property of the geodesic grid, latitude-longitude map and cube map.

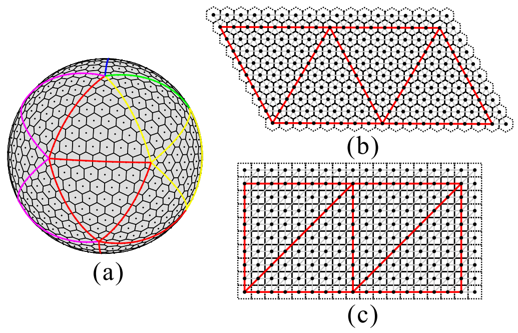


Fig. 3 Illustration of grid structure. (a) The geodesic grid is separated into five sets of four triangles. (b) Computation grid. (c) Storage grid.

4.3 Computation Grid & Storage Grid

Given a spherical image represented by geodesic grid, we have to know how to conduct computation on it and how to store it in physical memory. In this paper, we adopt the data structure proposed in (Randall et al, 2002). The geodesic grid is partitioned into five sets of four triangles, which have the same structure and can be made coincident with any other through rotation (Figure 3(a)). We then roll out the five parts to the plane and get five parallelograms composed of hexagons (Figure 3(b)). Thanks to the small distance deviation of geodesic grid as aforementioned, we will detect and describe the features based on this flattened hexagonal grid, which we term as *computation grid*. Each parallelogram is readily packed to a rectangle fitting the physical memory. We will fetch and index the pixels on this rectangular grid, which we term as *storage grid* (Figure 3(c)).

4.3.1 Neighborhood Fetching

Based on the storage grid, we define the neighbors of a pixel as the following set,

$$\mathcal{N}(r) = \{[s,t]^{\mathrm{T}} | -r \le t \le r, \\ \max(-r, -r-t) \le s \le \min(r, r-t)\}, \tag{1}$$

where r is the neighborhood radius; $[s,t]^{\mathrm{T}}$ are the coordinate offsets of the neighboring pixels with respect to the centered pixel. The number of neighbors is $|\mathcal{N}(r)| = 3r^2 + 3r + 1$. The neighbors of one pixel with different radius form regular concentric hexagons in the computation grid, and distorted ones in the storage grid (Figure 4). To find the neighbors of the boundary cells, we extend the storage grids for convenience (see Sec. 5.4.1 for details).

4.3.2 Coordinate Transform Between Two Grids

Indexing the pixels on the storage grid is based on an orthogonal coordinate system, shown in red color in Figure 4(b).

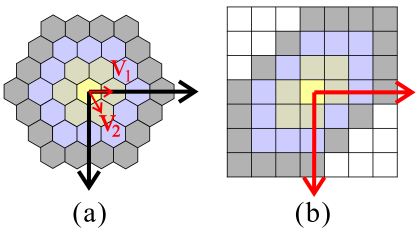


Fig. 4 Illustration of pixel neighborhood and local coordinates on the (a) computation grid and (b) storage grid.

This is equivalent to indexing the pixels on the computation grid based on a skewed coordinate system. However when detecting and describing the features, we should operate the pixels based on the orthogonal system defined on the computation grid, shown in black color in Figure 4(a). With simple mathematics, we know that the basis vectors of the skewed system with the coordinates defined in the orthogonal system are $\mathbf{v}_1 = [1,0]^\mathrm{T}$ and $\mathbf{v}_2 = [\frac{1}{2}, \frac{\sqrt{3}}{2}]^\mathrm{T}$. Then given the storage coordinates of a pixel, its computation coordinates can be found using the following transformation matrix,

$$\mathbf{T} = [\mathbf{v}_1, \mathbf{v}_2] = \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} \end{bmatrix}. \quad (2)$$

It should be noted that geodesic grid is still subjected to small nonuniform distortions. Towards the edges, the grid is more distorted. It seems that a non-constant transformation matrix helps to address the distortion. However it is not an easy task, because there will be many different non-constant transformation matrices, and they still rely on local planar assumption suffering from approximation errors. On the other hand, as shown in our experiments a constant $\mathbf{T}$ is sufficient to reach high speed performance and decent matching accuracy.

5 SPHORB: The Algorithm

In this section, we present how to construct spherical FAST detector and spherical rBRIEF descriptor on the geodesic grid, with attention to rotation invariance.

5.1 Spherical FAST

The FAST detector compares the intensities of the central pixel and those in a circular ring around it. In the geodesic grid, we consider the outer ring of the 3^{rd}-level neighborhood of pixel $\mathbf{k}$, which contains 18 pixels as shown in Figure 5(a). The pixel $\mathbf{k}$ is classified as a corner if there exist $n_f \in [10, 18]$ consecutive pixels that are either sufficiently brighter or darker than $\mathbf{k}$, i.e. $|I(\mathbf{x}, \mathbf{k}) - I(\mathbf{k})| > t$, where $I(\mathbf{k})$ is the intensity of pixel $\mathbf{k}$, $I(\mathbf{x}, \mathbf{k})$ is the intensity of neighboring pixel of $\mathbf{k}$ with $\mathbf{x} \in \mathcal{N}(3) \setminus \mathcal{N}(2)$, t is a threshold. After the corners are detected, the FAST score is computed as the maximum value of t that still detects the point as a corner. Non-maximal suppression is then applied to filter unstable corners.

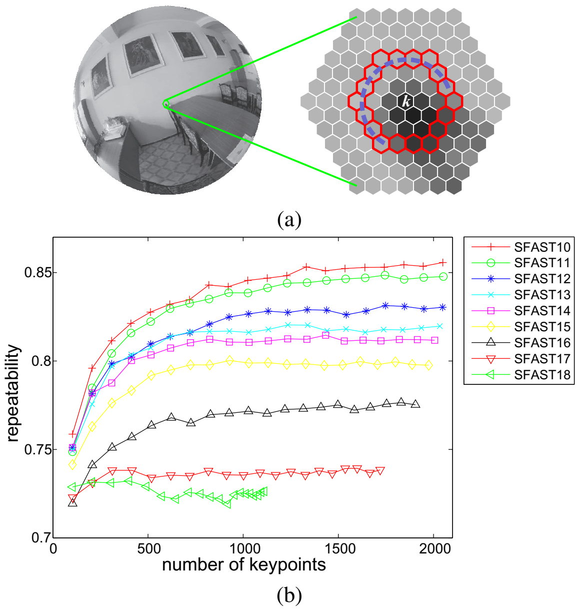


Fig. 5 Spherical FAST: (a) The corner detection mask, (b) Among different mask choices, SFAST-10 gives the best performance. The repeatabilities are averaged over 100 images under 90° camera rotation with Gaussian noise 10.

The choice of n_f defines different masks and leads to different detectors. To train an optimal spherical FAST detector, which we term as SFAST, we select 200 spherical images from the SUN360 database (Xiao et al, 2012), including 110 indoor images and 90 outdoor images from different scenes. We evaluate the detection performance in terms of detection repeatability (its definition is given in Section 6.1.1) and computation speed. Simply speaking, for a given similar number of detected keypoints, the repeatability is proportional to the number of corresponding keypoints. In Figure 5(b), we extract different number of keypoints with SFAST detectors from $n_f = 10$ to 18, and plot the average repeatability values among another 100 panorama images. We can see that SFAST-10 has the best detection accuracy. We also note when n_f is approaching 18, the actually detected keypoints may not reach the expected number, hence the curve becomes shorter as compared to those from a small n_f. As for the computation speed, SFAST-10 is no doubt the fastest, hence it is used in all of our experiments.

Scale-space construction. To achieve scale invariance, we build a scale pyramid of the spherical image. The pyramid contains m octaves and $l(m-1)$ intra-octaves. Typically we set $m = 3$, and $l = 2$. The octaves and intra-octaves are formed by subdividing the icosahedron at different levels. Given an initial level $n = 2^8$ (corresponding to the original panoramic image), other octaves are progressively four times smaller in resolution. In-between two adjacent octaves we create l intra-octaves, each with a down-scaling factor $2^{\frac{1}{l+1}}$. After the construction of the scale pyramid, we produce SFAST features at each scale level, the union of which is used as the features of the spherical panoramic image.

5.2 Spherical rBRIEF

Given a detected key point $\mathbf{k}$, the descriptor can be constructed from a set of intensity comparisons on the neighborhood (patch) $\mathcal{P} = \mathcal{N}(r)$ of $\mathbf{k}$ with neighborhood radius $r = 15$. We define the intensity comparison τ as

$$\tau(\mathbf{k};\mathbf{x},\mathbf{y}) = \begin{cases} 1, & \text{if } I(\mathbf{x},\mathbf{k}) < I(\mathbf{y},\mathbf{k}), \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where $\mathbf{x},\mathbf{y} \in \mathcal{P}$ are neighboring pixels of $\mathbf{k}$, $I(\mathbf{x},\mathbf{k})$ and $I(\mathbf{y},\mathbf{k})$ are their pixel intensities respectively. It is noted that to suppress noise the neighborhood of $\mathbf{k}$ should be smoothed in the computation grid. The smoothing can be done by using the hexagonal Gaussian kernel given by

$$g(\mathbf{x}) = \exp\left\{-\frac{(\mathbf{Tx})^{\mathrm{T}}(\mathbf{Tx})}{2\delta^2}\right\},\ s.t.\ \mathbf{x} \in \mathcal{P}. \tag{4}$$

Given a sampling pattern $\mathcal{S} = \{(\mathbf{x}_i,\mathbf{y}_i)|i = 1,\dots,N_s,\mathbf{x}_i \in \mathcal{P},\mathbf{y}_i \in \mathcal{P}\}$, each element of which is a test pair for intensity comparison, the feature vector can be constructed as an N_s–dimensional bit-string,

$$F(\mathbf{k}) = \sum_{1\le i\le N_s} 2^{i-1}\tau(\mathbf{k};\mathbf{x}_i,\mathbf{y}_i),\ s.t.\ (\mathbf{x}_i,\mathbf{y}_i) \in \mathcal{S}. \tag{5}$$

Following a similar training scheme in (Rublee et al, 2011), we train an optimal sampling pattern $\mathcal{S}$ from the image set used in the SFAST training. The optimal $\mathcal{S}$ contains $N_s = 256$ test pairs. This results a bit-string of length 256 for each feature vector, which takes 32-byte memory.

5.3 Rotation Invariance

Although spherical images capture up-to 360° field-of-view environment, the rotation invariance for spherical features is indispensable, at least for two situations. For the matching between planar images and spherical panoramas, the planar images may be located in any position or orientation in the sphere. Secondly, in sparsely sampled spherical panoramic sequences, such as in Google StreetView, camera motion may make an object severely change its spherical position (and scale).

5.3.1 Orientation Estimation

Since a small neighborhood on the geodesic grid can be regarded as being planar, we directly estimate the orientation of keypoints on the computation grid. Given one keypoint $\mathbf{k}$, its orientation is assigned as:

$$\theta = \arctan 2(m_{01}, m_{10}), \tag{6}$$

where arctan 2 is the quadrant-aware version of arctan; m_{01} and m_{10} are the moments of the patch $\mathcal{P}$ centered at $\mathbf{k}$, defined as:

$$m_{pq} = \sum_{\mathbf{x}=[s,t]^{\mathrm{T}}\in\mathcal{P}} s_c^p\, t_c^q\, I(\mathbf{x},\mathbf{k}), \tag{7}$$

Here, $\mathbf{x} = [s,t]^T$ denote the relative coordinates of a neighboring pixel on the storage grid. The corresponding coordinates on the computation grid are computed as

$$[s_c,\ t_c]^{\mathrm{T}} = \mathbf{T}[s,\ t]^{\mathrm{T}}. \tag{8}$$

The orientation θ is the rotation angle with respect to the local horizontal axis at $\mathbf{k}$ (see the short red arrow in Figure 6). Since the local horizontal axis varies in direction on the sphere, a reasonable way is choosing an universal reference on the sphere for orientation computation. In our work, we select the south pole as the universal reference, and take into account the inherent spherical angle (ϕ) between the arc connecting the key point to the south pole and the local horizontal axis projected on the sphere (see the yellow angle). The resulting orientation angle becomes

$$\theta' = \theta - \phi. \tag{9}$$

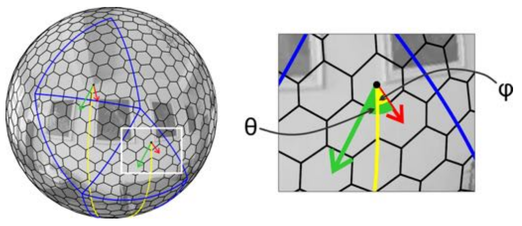


Fig. 6 Illustration of the orientation.

5.3.2 Rotation of the Sampling Pattern

To achieve rotation invariance, we should rotate the sampling pattern to the orientation of the key point. However, we can not rotate the sampling pattern directly, since it is trained on the storage grid. Instead we first transform the coordinates of test pairs in the sampling pattern to the computation grid, yielding

$$\mathcal{S}_c = \{(\mathbf{Tx}_i,\mathbf{Ty}_i)|(\mathbf{x}_i,\mathbf{y}_i) \in \mathcal{S}\}. \tag{10}$$

Next we rotate the pattern based on the key point orientation θ'. The resulted pattern is relative to the universal reference. Then the pattern is transformed back to the computation grid, and further to the storage grid for intensity accessing. The final sampling pattern becomes,

$$\mathcal{S}_f = \{(\mathbf{T}^{-1}\mathbf{R}_\phi\mathbf{R}_{\theta'}\mathbf{T}\mathbf{x}_i, \mathbf{T}^{-1}\mathbf{R}_\phi\mathbf{R}_{\theta'}\mathbf{T}\mathbf{y}_i)|(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{S}\}. \quad (11)$$

The rotation invariant feature vector will be

$$F(\mathbf{k}) = \sum_{1\le i\le N_s} 2^{i-1}\tau(k;\tilde{\mathbf{x}}_i,\tilde{\mathbf{y}}_i),\ s.t.\ (\tilde{\mathbf{x}}_i,\tilde{\mathbf{y}}_i)\in\mathcal{S}_f. \quad (12)$$

In our current implementation, nearest point sampling is used to access the pixel intensity when computing feature vectors. Note that Eq. 11 can be simplified by substituting Eq. 9, and we get

$$\mathcal{S}_f = \{(\mathbf{T}^{-1}\mathbf{R}_\theta\mathbf{T}\mathbf{x}_i, \mathbf{T}^{-1}\mathbf{R}_\theta\mathbf{T}\mathbf{y}_i)|(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{S}\}. \quad (13)$$

This implies that the reference angle ϕ can be safely ignored when describing the feature. However when we show the key points on the unfolded spherical images and display their orientation angles with arrows, the reference angles must be accounted in determining the orientation of arrows.

5.4 Implementation Details

5.4.1 Boundary Extension

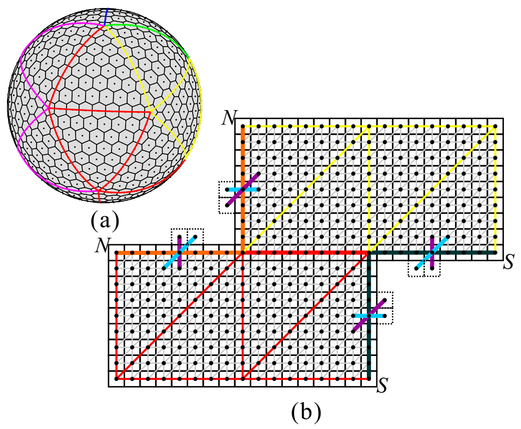


Fig. 7 The illustration of how to extend the boundary of the storage grid. See text for details. The letters N and S represent the north pole and south pole, respectively.

We now discuss in details about boundary extension for the geodesic grid. As shown in Figure 7, each storage grid has two neighbors (marked in different colors), and shares three edges with every neighbor as illustrated in Figure 7(b). Specifically, the top left edge, the top right edge and the right edge of the red storage grid are coincident with the left edge, the bottom left edge and the bottom right edge of the yellow grid, respectively. These three pairs of edges are shown as bolded lines in orange, overlapped red and blackish green.

According to the pixels' neighboring relationships on the sphere, we can determine their neighboring relationships on the storage grid. For the top left edge of the red grid, the horizontal neighbors become vertical neighbors in the yellow grid, while the vertical ones become diagonal and the diagonal neighbors become horizontal. Hence the extended pixels can be got from the yellow grid in the diagonal direction. For the overlapped red edge between two grids, the neighborhood patterns in the two grids are normal. The right edge has the similar property as the top left edge, and we should index the extended pixels in the diagonal direction. So far, we discussed how to handle three edges of a storage grid. The rest three edges can be processed in a similar way. Taking the pentagons into consideration, there will be some holes in the extended storage grid (see Figure 8).

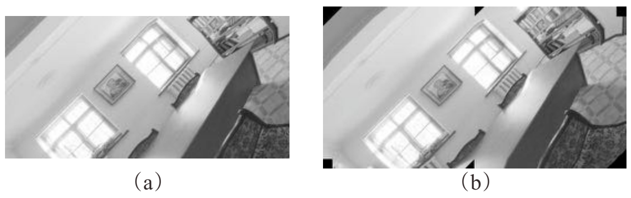


Fig. 8 Samples for the (a) original and (b) extended storage grids.

5.4.2 Dealing With Pentagons

Recall that the geodesic grid contains 12 pentagonal cells located at the vertices of the icosahedron. Due to the shape difference, the pentagons have different neighborhood patterns with hexagons, as shown in Figure 9. If there is a pentagon located on the n-th layer neighborhood, the difference of cell number between layer r and $r-1$ is 6, when $2\le r\le n$; however if $r>n$, the difference will be 5. Hence, given a neighborhood radius r, the number of neighbors can be computed as follows,

$$|\mathcal{N}(r,n)| = \begin{cases} 3r^2+3r+1, & r\le n, \\ (3n^2+3n+1)+\frac{(7n+5r+5)(r-n)}{2}, & r>n, \end{cases} \quad (14)$$

where n is the layer index on which the pentagon appears.

Considering the trade-off between computing complexity and effectiveness, we discard these pixels whose neighbors contain pentagons in our current work [2]. The next issue is how to determine the pixels to be discarded. Instead of checking the neighborhood for each pixel, we adopt a logically equivalent strategy: finding the neighbors for the 12 pentagons with a given neighborhood radius. For instance,

[2] A direct workaround may be rotating the panorama so that the pentagons cover different regions of the original panorama and the transformed one. Then the keypoints from the transformed panorama are taken as the complements after removing the reduplicative keypoints.

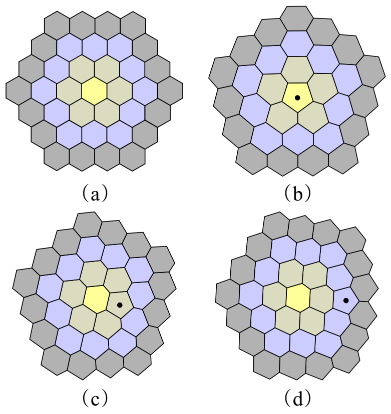


Fig. 9 Illustration of the pixel neighborhood. (a) There is no pentagon in the neighborhood. The pentagon appears on the (b) 0^{th}, (c) 1^{st}, and (d) 2^{nd} layer neighborhood.

we use a neighborhood radius of 15 for feature description, and 3 for Gaussian smoothing, then the number of discarded pixels will be $12 \times |\mathcal{N}(17,0)| = 9192$. Since the geodesic grid to represent the spherical image has a subdivision level 256, the ratio of discarded pixels is $\frac{9192}{10\times256^2+2} \approx 1.4\%$. Although the ratio becomes larger when the subdivision level decreases, we find the multi-scale SPHORB detects more keypoints than the single scale version, along with higher matching accuracy (as shown in Table 1 and Figures 12, 13, and 14). This implies our scheme to discard the pentagon-affected pixels is feasible in practice.

6 Experiments and Discussion

To evaluate the performance of SPHORB, we compare it with previous algorithms. Firstly, we evaluate their detector behaviour and matching precision on a dataset of spherical images undergoing synthetic camera rotation and added Gaussian noise. We then report their performance on real-world data with camera movement. The ratio matching strategy (Mikolajczyk and Schmid, 2005) is used in the evaluation with the threshold usually set as 0.75. The timing performance is given at the end.

6.1 The Performance under Camera Rotation and Noise

To complement the training image set, we select another 100 spherical images from SUN360 database (Xiao et al, 2012). Each image is rotated around the principal axis of the camera and corrupted with additive Gaussian noise. To be specific, we use 36 rotation angles evenly distributed in [0°, 350°], and 6 noise levels evenly distributed in [0, 25]. Hence, we get all together $100 \times 36 \times 6 = 21,600$ testing image pairs.

SPHORB is compared with three methods: Spherical SIFT (the authors' code[3]), planar ORB (OpenCV 2.4.2), and planar SIFT (OpenCV 2.4.2). Planar ORB and planar SIFT are applied on spherical images unfolded in the latitude-longitude parameterization. Since in this experiment there are no scale changes, we also test the performance of SPHORB and planar ORB with single scale, indicated with prefix 'S-'. We tune the parameters of different methods such that they extract approximately a similar number of keypoints (about 1600 in our experiment).

[3] https://sites.google.com/site/javicm/software

6.1.1 Evaluation of Detector Behavior

The detector behavior is evaluated in terms of the repeatability score (Mikolajczyk et al, 2005). It is calculated as the ratio between the number of corresponding keypoints and the minimum number of detected keypoints in the original and transformed images. The correspondences are identified by checking the spherical distance between a keypoint in one image and the projection of its closest keypoint from the other image. We set the distance threshold to be 2 pixels with respect to the circumference of a great circle (1280 pixels in our experiment).

Figure 10(b) shows the averaged repeatability of these algorithms under different angles of rotation with Gaussian noise of 10. The algorithms that take the spherical geometry into consideration have higher repeatability. As the rotation around the principal axis induces a complex image transformation going beyond the capability of planar features, planar ORB and planar SIFT give relatively poor performance. Figure 10(c) shows the repeatability under 90° rotation with different noise levels. Among the six methods, SPHORB is most robust to Gaussian noise, while retaining the highest repeatability. Spherical SIFT has similar repeatability as SPHORB when the noise level is 0, but falls gradually when the noise level increases. Another interesting thing is that the multi-scale version of SPHORB and planar ORB are more robust than their single scale version. This is because the single scale version will detect more keypoints on the first level, which may contain many less distinctive keypoints.

6.1.2 Evaluation of Matching Precision

The descriptor performance is evaluated in terms of precision, which is the ratio between the number of true matches and the number of all matches. Figure 10(d) shows the averaged precision of these algorithms under different angles of rotation with Gaussian noise of 10. We can see that SPHORB and Spherical SIFT have similar precision values. The precision of planar SIFT is relatively low, but higher than that of planar ORB. We also observe that the single scale version of SPHORB and planar ORB are better than their multi-scale versions. However for real data matching, multi-scale SPHORB achieves better results, as later demonstrated in Table 1. Figure 10(e) plots the precision of SPHORB and

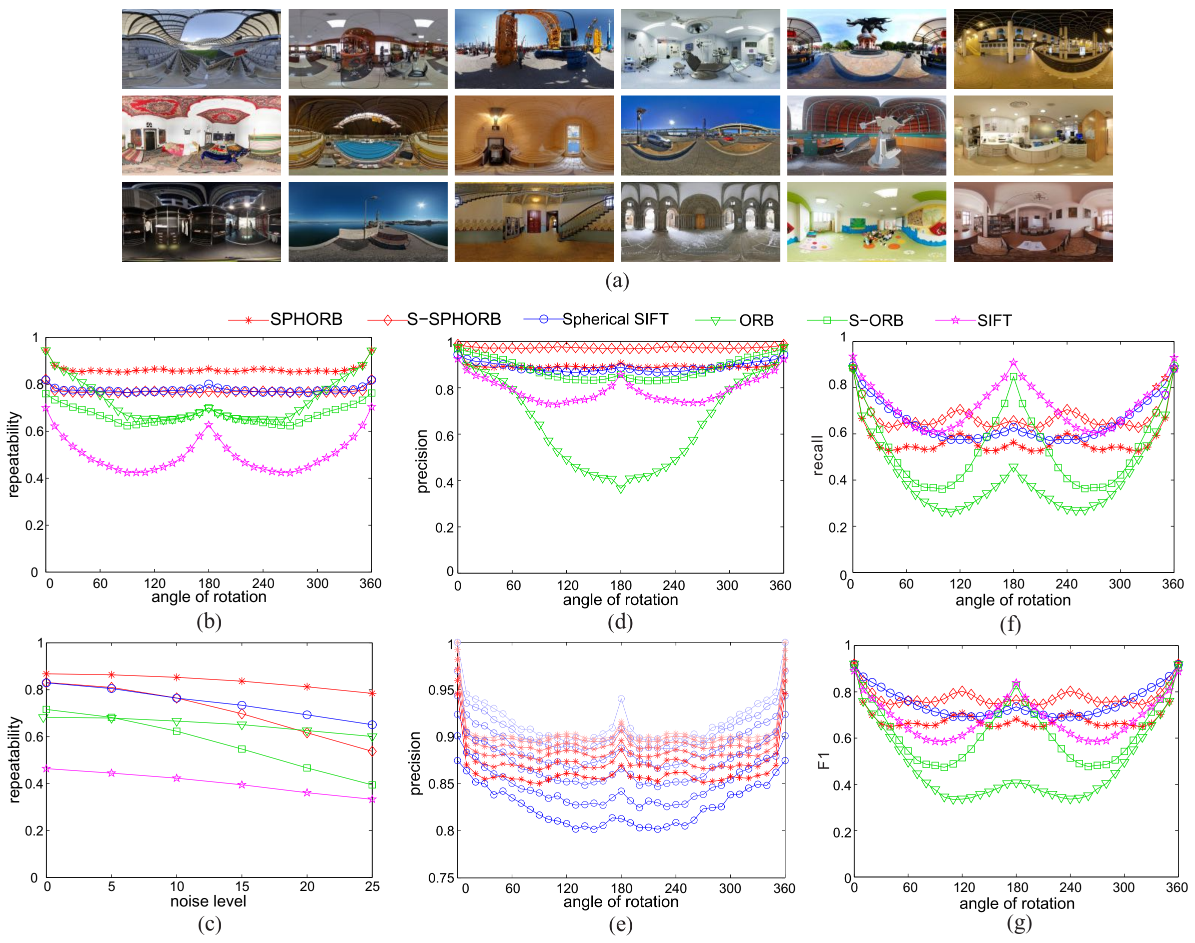


Fig. 10 Averaged performance under camera rotation and Gaussian noise over 100 images. (a) Some exemplar spherical panoramic images. (b) The repeatability evaluation of planar SIFT, planar ORB, planar S-ORB, Spherical SIFT, SPHORB, and S-SPHORB under synthetic rotations with Gaussian noise of 10. (c) The repeatability under 90° rotation with different noise levels. (d) The precision under different rotations with Gaussian noise of 10. (e) The precision of Spherical SIFT and SPHORB under camera rotation with different noise levels 0, 5, 10, 15, 20 and 25. Higher color saturation represents more noise. (f) and (g) The recall and F1 score under different rotations with Gaussian noise of 10.

Spherical SIFT under camera rotation with different noise levels 0, 5, 10, 15, 20 and 25. SPHORB is relatively immune to Gaussian image noise, obtaining at least 85% precision values for noise level 25.

Another criterion is recall value, which is the number of true matches with respect to the number of expected corresponding keypoints between two images. We plot the averaged recall of the six algorithms under different angles of rotation with Gaussian noise of 10. From Figure 10(f) we can see that the ranking of recall values is similar to that of precision values, except for SIFT. This is because SIFT finds much less corresponding keypoints between two images, which helps to give high recall values.

Considering the above two criterions together, we also plot the averaged F1 score in Figure 10(g), which is a harmonic mean of precision and recall. We can see that the F1 performance of the spherical features are better than the planar features in comparison.

6.1.3 SPHORB v.s. WTA Hash on Geodesic Grid

As pointed out in (Ziegler et al, 2012), BRIEF is a locality sensitive hashing scheme on Kendall's tau metric; other descriptors based on intensity comparisons are dimensionality reduction schemes on Kendall's tau. To verify whether other hashing schemes give better performance, we adapt WTA hash (Yagnik et al, 2011) as another descriptor. WTA hash (Yagnik et al, 2011) is an embedding method that transforms an input feature space into sparse codes. It is defined by a sequence of permutations Θ^n, $1 \leq n \leq N$, of an input feature vector $\mathbf{p}$. Each code $f_n(\mathbf{p})$ of the resulting vector is the index of the maximum value in the first K entries of the feature vector permutated by Θ^n. Specifically,

$$f_n(\mathbf{p}) = \underset{0 \leq i \leq K-1}{\arg\max}\,\{\Theta_i^n(\mathbf{p})\}, 1 \leq n \leq N, \tag{15}$$

where $\Theta_i^n(\mathbf{p})$ is the i-th entry of the permutated feature vector. As $f_n(\mathbf{p})$ is represented by $\lceil \log_2 K \rceil$ bits, each resulting feature vector is compactly represented by $N\lceil \log_2 K \rceil$ bits.

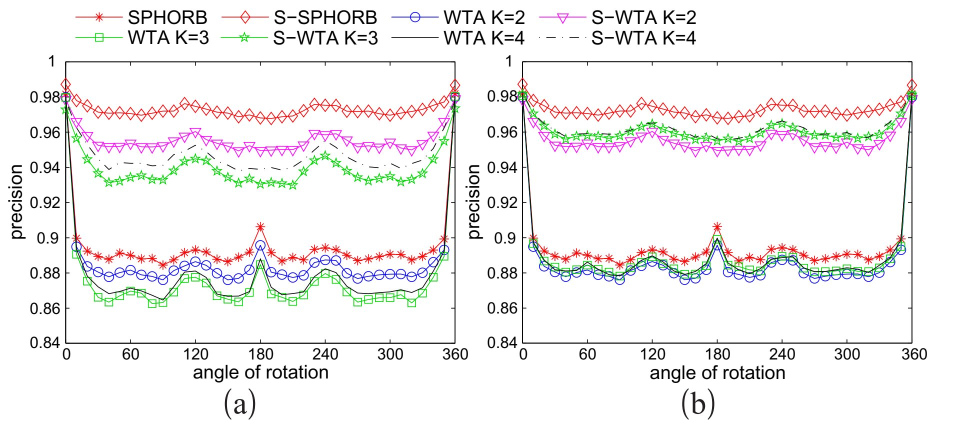


Fig. 11 Precision comparison between SPHORB and WTA hash under different rotations with Gaussian noise of 10: (a) the same byte setting (32 bytes) and (b) the same length setting (length=256).

	Macau		EPFL		MTS Lab		SE Lab		CUHK1		CUHK2	
Planar SIFT	64	78.13%	75	66.67%	76	80.26%	89	79.78%	76	78.95%	68	64.71%
Planar ORB	254	79.53%	182	80.77%	335	82.09%	332	85.54%	403	80.65%	319	73.04%
Spherical SIFT	203	35.96%	179	39.11%	320	40.31%	258	32.17%	340	30.88%	354	20.62%
Cube ORB	247	75.30%	185	80.00%	358	88.55%	336	85.71%	395	78.48%	316	71.52%
Lat-Local ORB	257	82.88%	204	**91.67%**	320	90.63%	367	91.28%	402	78.61%	310	76.13%
Geo-Local ORB	246	75.61%	202	89.60%	346	88.15%	355	85.35%	377	81.70%	320	**85.94%**
WTA Hash K=2	271	76.01%	254	75.98%	445	82.47%	427	83.84%	390	76.15%	354	70.06%
WTA Hash K=4	292	61.64%	223	58.30%	493	71.60%	408	72.06%	428	63.79%	378	50.26%
S-SPHORB	113	88.50%	29	55.17%	90	83.33%	45	73.33%	103	72.82%	53	54.72%
SPHORB	244	**90.98%**	183	84.15%	357	**92.44%**	341	**92.08%**	407	**90.66%**	314	81.53%

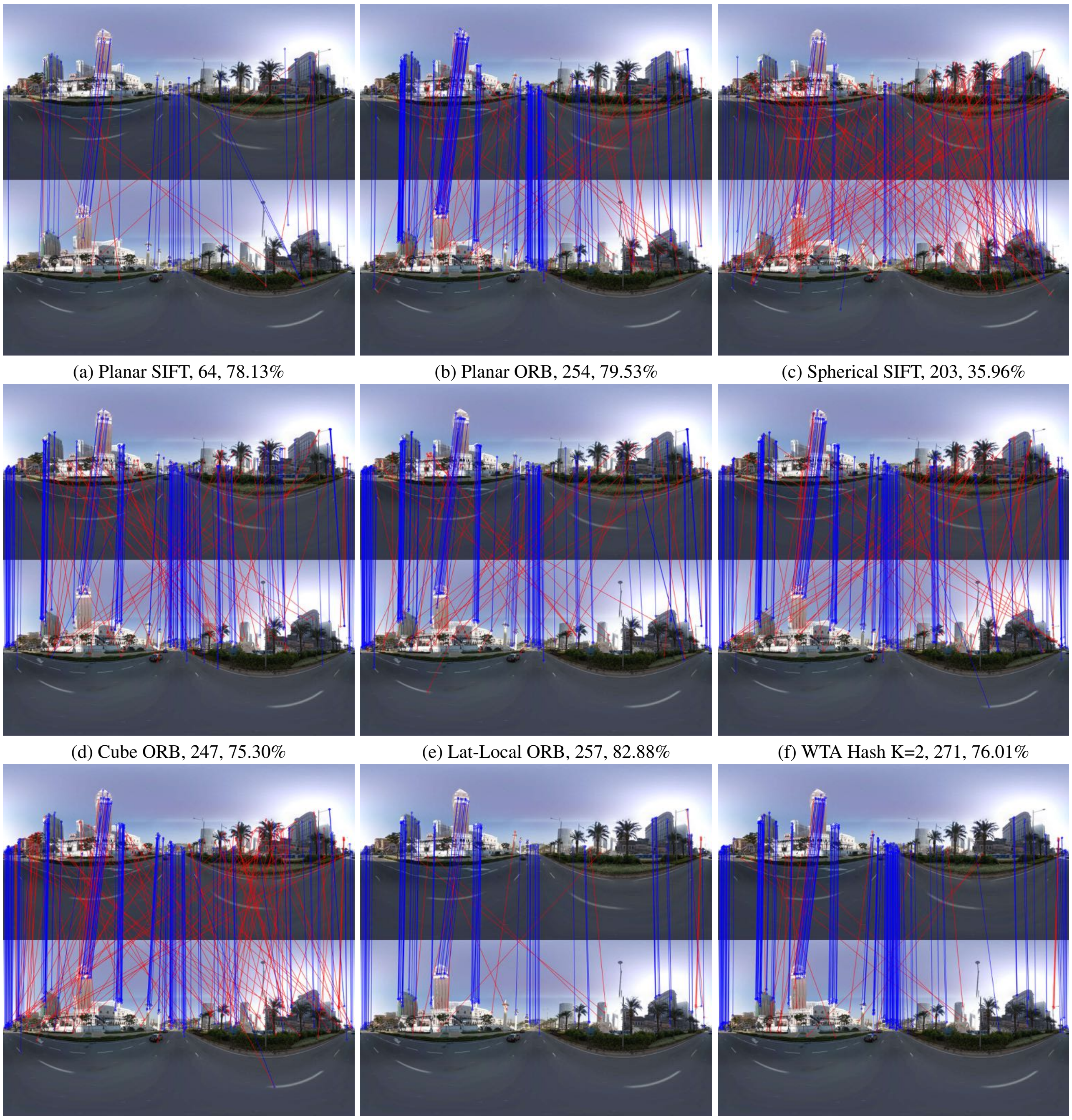
Table 1 Matching performance for real world spherical image pairs. In each cell, the first number is the number of matches, and the second is the inlier ratio. For each image pair the highest ratio is bolded.

In our experiment, we use the pixel intensities of the smoothed image patch around a keypoint (on the geodesic grid) as the input feature vector. We construct WTA hash descriptors with different K, e.g. $K = 2,3,4$, and make hash vectors have the *same byte* or the *same length* as the SPHORB descriptor, which is a bit-string of length 256 and takes 32-byte memory. Note each code of the hash vector takes 1 bit for $K = 2$, and 2 bits for $K = 3$ or 4. In the same byte setting, the length of the hash vector is 256 for $K = 2$, and 128 for $K = 3,4$. In the same length setting, each hash vector uses 32 bytes for $K = 2$, and 64 bytes for $K = 3,4$.

Figure 11 shows the precision of SPHORB and WTA hash under different rotations with Gaussian noise of 10. We can see that SPHORB gives best results in both settings either among multi-scale family or among single scale family. Compared with WTA hash $K = 2$, the two cases $K = 3$ and $K = 4$ give worse performance in the same byte setting, while a little better performance in the same length setting. This observation is as expected, since in the same byte setting the length of hash vector for $K = 3$ and $K = 4$ is shorter than that of $K = 2$, which captures less information. We also observe that in both settings the performance of WTA hash $K = 4$ is slightly better than that of $K = 3$. This is because $K = 4$ fully explores the potential of 2 bits. For the recall values, we can get similar curves, which is not included.

6.2 Matching between Two Spherical Panoramas under Camera Movement

In this experiment, we collected six groups of spherical images covering both indoor and outdoor scenes. The two images in each group are subjected to camera movement, hence may contain obvious scale changes. For quantitative evaluation, we extract keypoints from two images and identify the inliers and outliers by manual intervention after matching. Besides the methods compared in Sec. 6.1, we consider another three possible solutions. In the first solution *Cube ORB*, we represent spherical images in cube map parameterization, then the planar ORB is applied on cube faces. In the other two solutions, the spherical image is locally projected to the tangential plane of the sphere, which generates a projected patch for each sampling point. We then detect and describe the keypoints on the projected patches with the planar ORB. Finally the duplicated keypoints are filtered out. This projection-based scheme relies on different sampling strategies of spherical images. We adopt the latitude-longitude sampling and form one solution, denoted


(a) Planar SIFT, 64, 78.13% (b) Planar ORB, 254, 79.53% (c) Spherical SIFT, 203, 35.96%

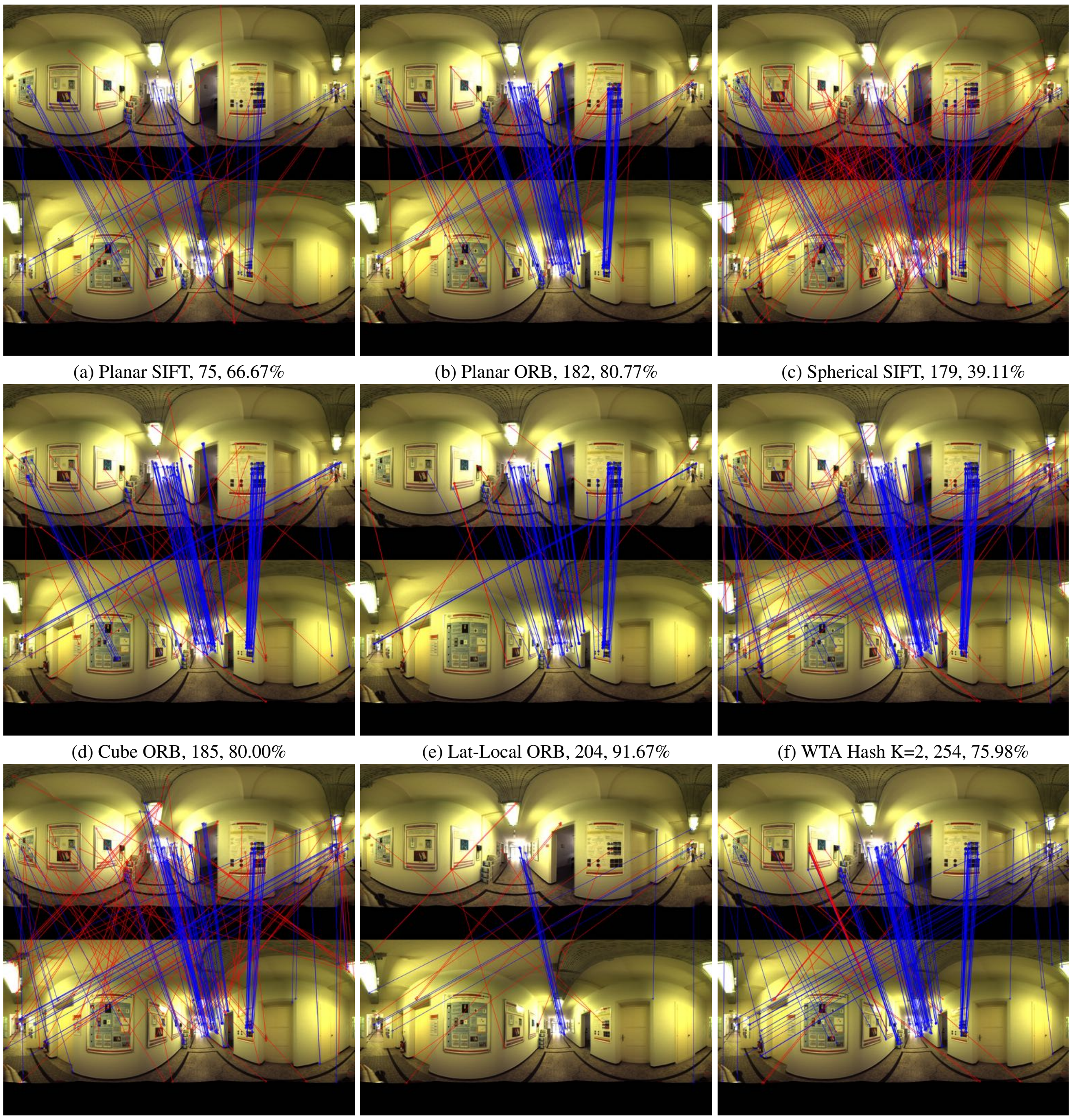
(d) Cube ORB, 247, 75.30% (e) Lat-Local ORB, 257, 82.88% (f) WTA Hash K=2, 271, 76.01%

(g) WTA Hash K=4, 292, 61.64% (h) S-SPHORB, 113, 88.50% (f) SPHORB, 244, 90.98%

Fig. 12 Matching between spherical panoramic images captured in Macau for different methods. The number of matches and the inlier ratio are listed for each method.

as *Lat-Local ORB*, and the geodesic grid sampling forming another solution denoted as *Geo-Local ORB*.

Table 1 presents the number of matches, and the inlier ratio of different methods. In terms of the inlier ratio, SPHORB achieves the best performance in most cases, while its single scale version finds much less matches and gets a relatively lower inlier ratio. This verifies that the multi-scale version of SPHORB is encouraged when processing the real world spherical images. The performance of WTA hash in the same byte setting with $K = 2$ is better than that of $K = 4$, which is the same result as in the camera rotation and noise experiment. As for Spherical SIFT, we unexpectedly find that it performs rather poor with the code provided by the authors. Since Spherical SIFT has good performance for camera rotation and additive noise, we think it may be the inher-



(a) Planar SIFT, 75, 66.67% (b) Planar ORB, 182, 80.77% (c) Spherical SIFT, 179, 39.11%

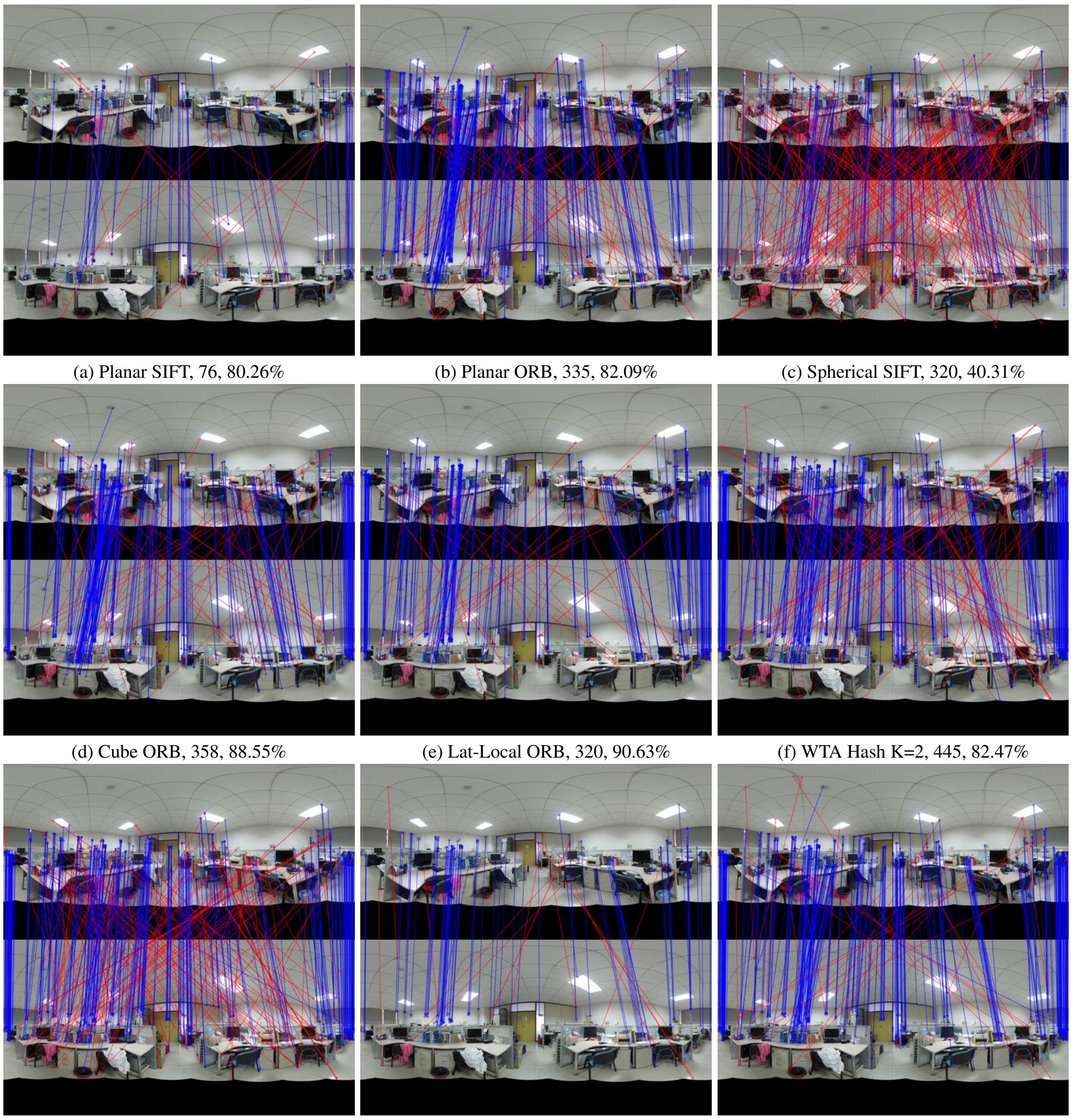
(d) Cube ORB, 185, 80.00% (e) Lat-Local ORB, 204, 91.67% (f) WTA Hash K=2, 254, 75.98%

(g) WTA Hash K=4, 223, 58.30% (h) S-SPHORB, 29, 55.17% (i) SPHORB, 183, 84.15%

Fig. 13 Matching between spherical panoramic images captured in EPFL for different methods. The number of matches and the inlier ratio are listed for each method.

ent bandwidth limitation of spherical harmonics that causes this phenomenon.

When the distinctive image structure of some spherical images is mainly located in the equatorial region, which is less deformed, these images can be regarded as local planar images. Due to this reason, planar ORB and planar SIFT, although detecting features on the plane, have acceptable performance. Cube ORB has a little better performance than planar SIFT, and close to planar ORB. It is because the cube-map samples spherical images more uniformly. As for Lat-Local ORB and Geo-Local ORB, since they are based on local projection and suffer less from sampling distortion, we expect they will have higher performance than SPHORB. However, they just get similar matching precision, and they are very computing intensive as shown in Table 2. Figures 12, 13 and 14 show the matching results for the first three cases


(a) Planar SIFT, 76, 80.26% (b) Planar ORB, 335, 82.09% (c) Spherical SIFT, 320, 40.31%

(d) Cube ORB, 358, 88.55% (e) Lat-Local ORB, 320, 90.63% (f) WTA Hash K=2, 445, 82.47%

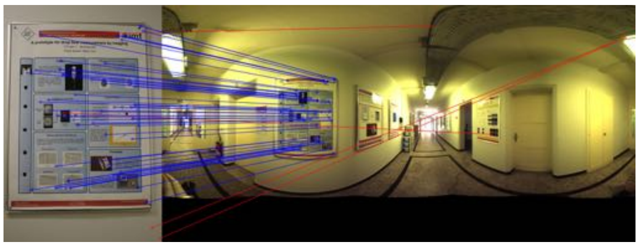
(g) WTA Hash K=4, 493, 71.60% (h) S-SPHORB, 90, 83.33% (i) SPHORB, 357, 92.44%

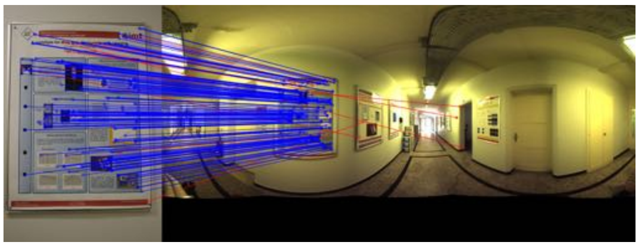
Fig. 14 Matching between spherical panoramic images captured in MTS Lab for different methods. The number of matches and the inlier ratio are listed for each method.

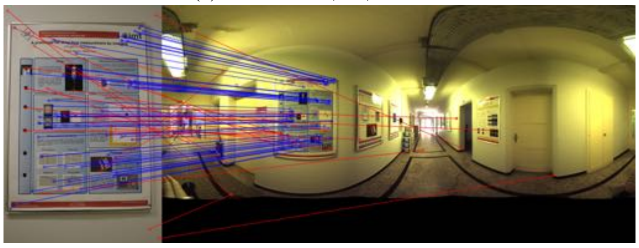
in Table 1, with the inlier correspondences ploted in blue color and the outliers in red color.

6.3 Matching between Planar and Spherical Images

Besides the matching between two spherical images, there are increasing applications that require to match a planar image to a spherical one, such as image based localization (Zamir and Shah, 2010) and scene recognition (Xiao et al, 2012). In (Cruz-Mota et al, 2012), a regular SIFT descriptor is computed on a planar approximation of the region around each interest point, and the extracted local planar descriptors are matched with a preexisting database of SIFT descriptors computed on planar images. A similar strategy is adopted in (Qin and Li, 2012). However more often we may need to match a planar image to a set of spherical ones in the applications


(a) Planar SIFT, 61, 93.44%


(b) Planar ORB, 220, 93.18%


(c) Spherical SIFT, 90, 67.78%

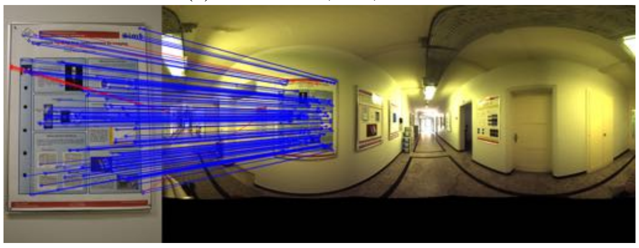
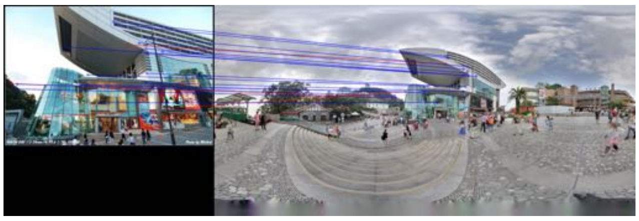
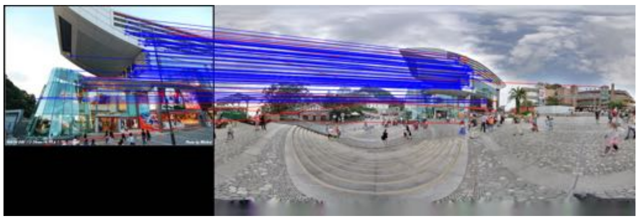
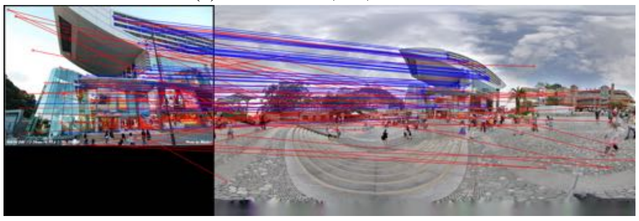

(d) SPHORB, 223, 96.41%

Fig. 15 Matching between a planar image and a spherical image captured in EPFL for planar SIFT, planar ORB, Spherical SIFT, and SPHORB. The number of matches and the inlier ratio are listed for each method.


(a) Planar SIFT, 19, 84.21%


(b) Planar ORB, 126, 88.10%


(c) Spherical SIFT, 105, 45.71%

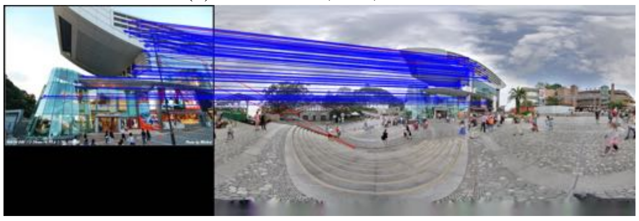

(d) SPHORB, 122, 95.90%

Fig. 16 Matching between a planar image and a spherical image captured in HK for planar SIFT, planar ORB, Spherical SIFT, and SPHORB. The number of matches and the inlier ratio are listed for each method.

mentioned above. Instead of projecting the keypoint regions on the tangential plane to extract planar descriptors, we simply render the planar image to the equatorial region of a spherical image. Then spherical image feature matching is carried out to match the planar image to the spherical one. Because the focal length and the field of view of a planar image are often unavailable, we assume that the field of view of the planar image along the longer axis (horizontal or vertical) is 90° and the focal length is the half length of that axis. The planar image is attached to the front face of a cube map, which is then converted to a spherical panorama.

Figure 15 and 16 show the matching results between a planar image and a spherical one using the above strategy. In Figure 15, since the query poster is located in the equatorial region in the reference spherical image, planar descriptors work well in this case. Yet, our method still has less false matches. In Figure 16, the query region is in the upper hemisphere, and SPHORB outperforms the other methods.

6.4 Timing Performance

We implemented our method on a computer installed with Intel(R) Core (TM) i7-2600 CPU @ 3.40GHz and NVIDIA GeForce GT 640. It runs in single thread and extracts features on 7 scale levels, with the first level having 655362 grid cells. To make a fair comparison, we make the number of the actual pixels compatible for different methods. Specifically, the input images for planar ORB and planar SIFT are resized to 1144×572; the spherical images for Spherical SIFT are resized to 808×808; the face size of Cube

ORB is 330×330; the sampling densities of Lat-Local ORB and Geo-Local ORB are the same as that of planar ORB and SPHORB respectively. Since Lat-Local ORB and Geo-Local ORB are based on local projection, we generate projected patches through hardware rendering without using time consuming trigonometric functions.

Table 2 presents the timing concerning both feature detection and description on the same image, with the values averaged over 10 runs. Note that the timing of Spherical SIFT accounts only for the construction of spherical scale-spaces, which is implemented in C++. The remaining steps, implemented in Matlab, are not taken into account. Given the above settings, SPHORB is an order of magnitude faster than planar SIFT, over three orders faster than Spherical SIFT. Compared with SPHORB, Cube ORB takes a little more time to extract the features, because the procedure converting the spherical images to cube maps is not accelerated in our current implementation. As for Lat-Local ORB and Geo-Local ORB, they spend a rather long time to project the spherical images, and hence are almost impractical.

	Point number	Total time (ms)	Time per point (ms)
Planar SIFT	1697	209.32	0.123
Planar ORB	1697	22.62	0.013
Spherical SIFT	1345	28,686.4	21.33
Cube ORB	1699	141.49	0.083
Lat-Local ORB	1631	3.00×10^6	1.8×10^3
Geo-Local ORB	1679	3.02×10^6	1.8×10^3
SPHORB	1710	104.88	0.061

Table 2 Computation timing for different algorithms.

We also record the timings for each step of our method. The results are shown in Table 3. The most time-consuming part is constructing the scale space, which needs to convert the spherical images to the geodesic grid representation. Although we have accelerated this procedure using lookup tables, it still takes much time for our unoptimized code. Also note that this time is constant, which means if there are more keypoints detected, the time for this procedure will be same.

Time (ms)	Scale Space	detection	description
SPHORB	58.141	11.271	35.468

Table 3 The timing for each part of our method (1710 keypoints detected)

7 Conclusions

In this paper, we have proposed a fast and robust binary spherical feature, namely SPHORB, for detecting and describing keypoints in spherical panoramic images. Unlike existing methods relying on costly spherical harmonics to build SIFT-like features, our method exploits a near hexagonal sphere parameterization, and reinvents ORB-like features by respecting the hexagonal neighborhood structure. As evident in the extensive experiments, the proposed SPHORB outperforms state-of-the-art methods in terms of computing efficiency, detection accuracy, and robustness to camera movements. The encouraging results are also reported in real-world matching tests, including the matching between spherical panoramic images, and the matching between spherical panoramas and planar images.

Our method can serve as a general framework to port planar binary keypoint methods to the spherical domain. Accordingly, near future work includes incorporating other binary keypoint methods, such as BRISK (Leutenegger et al, 2011) or FREAK (Alahi et al, 2012) in our framework. Another near future work is constructing a benchmark dataset for spherical features evaluation. In addition, the implementation of SPHORB can be further optimized so as to serve real-time or time-demanding applications.

Acknowledgements The authors thank all reviewers and the associate editor for their valuable comments. The authors also thank Chu Han, Wuyao Shen, and Xiangyu Mao for capturing CUHK dataset used in Table 1. The work was supported by the National Natural Science Foundation of China (61100122 and 61100121), New Century Excellent Talents in University (NCET-11-0365), the National Science and Technology Support Project (2013BAK01B01 and 2014BAK09B04), Tianjin Science Foundation for Youth (12JCQNJC00100), and the research Fund from The Tianjin Key Lab for Advanced Signal Processing, Civil Aviation University of China (TJKLASP-2012-2).

References

Alahi A, Ortiz R, Vandergheynst P (2012) FREAK: Fast retina keypoint. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 510–517

Ambai M, Yoshida Y (2011) CARD: Compact and real-time descriptors. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 97–104

Anguelov D, Dulong C, Filip D, Frueh C, Lafon S, Lyon R, Ogale A, Vincent L, Weaver J (2010) Google Street View: Capturing the world at street level. Computer 43(6):32–38

Arican Z, Frossard P (2012) Scale-invariant features and polar descriptors in omnidirectional imaging. IEEE Transactions on Image Processing 21(5):2412–2423

Bay H, Ess A, Tuytelaars T, Gool LV (2008) Speeded-up robust features (SURF). Computer Vision and Image Understanding 110(3):346 – 359

Bulow T (2004) Spherical diffusion for 3D surface smoothing. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(12):1650–1654

Calonder M, Lepetit V, Strecha C, Fua P (2010) BRIEF: Binary robust independent elementary features. In: Proceedings of the European Conference on Computer Vision (ECCV), vol 6314, pp 778–792

Cruz-Mota J, Bogdanova I, Paquier B, Bierlaire M, Thiran JP (2012) Scale invariant feature transform on the sphere: Theory and applications. International Journal of Computer Vision 98(2):217–241

Hadj-Abdelkader H, Malis E, Rives P (2008) Spherical Image Processing for Accurate Visual Odometry with Omnidirectional Cameras. In: The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras - OMNIVIS

Hansen P, Corke P, Boles W, Daniilidis K (2007) Scale-invariant features on the sphere. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1–8

Hansen P, Corke P, Boles W (2010) Wide-angle visual feature matching for outdoor localization. The International Journal of Robotics Research 29(2-3):267–297

Harris C, Stephens M (1988) A combined corner and edge detector. In: Proc. of Fourth Alvey Vision Conference, pp 147–151

Leutenegger S, Chli M, Siegwart R (2011) BRISK: Binary robust invariant scalable keypoints. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 2548–2555

Lowe D (2004) Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2):91–110

Micusik B, Kosecka J (2009) Piecewise planar city 3D modeling from street view panoramic sequences. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2906–2912

Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(10):1615–1630

Mikolajczyk K, Tuytelaars T, Schmid C, Zisserman A, Matas J, Schaffalitzky F, Kadir T, Gool L (2005) A comparison of affine region detectors. International Journal of Computer Vision 65(1-2):43–72

Puig L, Guerrero J (2011) Scale space for central catadioptric systems: Towards a generic camera feature extractor. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1599–1606

Qin X, Li S (2012) Finding scale-invariant corner feature from full-view image based on discrete spherical model. In: International Conference on Systems and Informatics (ICSAI), pp 1914–1918

Randall D, Ringler T, Heikes R, Jones P, Baumgardner J (2002) Climate modeling with spherical geodesic grids. Computing in Science Engineering 4(5):32–41

Rosten E, Drummond T (2006) Machine learning for high speed corner detection. In: Proceedings of the European Conference on Computer Vision (ECCV), vol 1, pp 430–443

Rosten E, Porter R, Drummond T (2010) Faster and better: A machine learning approach to corner detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(1):105–119

Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: An efficient alternative to sift or surf. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 2564–2571

Strecha C, Bronstein A, Bronstein M, Fua P (2012) LDAHash: Improved matching with smaller descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence 34(1):66–78, URL http://cvlab.epfl.ch/research/detect/ldahash

Taneja A, Ballan L, Pollefeys M (2013) City-scale change detection in cadastral 3D models using images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 113–120

Trzcinski T, Christoudias M, Lepetit V, Fua P (2013) Boosting binary keypoint descriptors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2874–2881

Valgren C, Lilienthal AJ (2007) SIFT, SURF and seasons: Long-term outdoor localization using local features. In: Proceedings of the European Conference on Mobile Robots (ECMR), pp 253–258

Williamson DL (1968) Integration of the barotropic vorticity equation on a spherical geodesic grid. Tellus 20(4):642–653

Xiao J, Ehinger K, Oliva A, Torralba A (2012) Recognizing scene viewpoint using panoramic place representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2695–2702

Yagnik J, Strelow D, Ross D, Lin RS (2011) The power of comparative reasoning. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 2431–2438

Zamir A, Shah M (2010) Accurate image localization based on google maps street view. In: Proceedings of the European Conference on Computer Vision (ECCV), vol 6314, pp 255–268

Zhao Q, Wan L, Feng W, Zhang J, Wong TT (2013) Cube2Video: Navigate between cubic panoramas in real-time. IEEE Transactions on Multimedia 15(8):1745–1754

Ziegler A, Christiansen E, Kriegman D, Belongie S (2012) Locally uniform comparison image descriptor. In: Advances in Neural Information Processing Systems 25, pp

1–9