Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

AnyFeature-VSLAM: Automating the Usage of Any Chosen Feature into Visual SLAM

Alejandro Fontan[1], Javier Civera[2], and Michael Milford[1]
Queensland University of Technology[1]
University of Zaragoza[2]

***Abstract*—Feature-based SLAM heavily relies on the specific type of visual features employed, as illustrated in Fig. 1. The most effective feature in some conditions may perform worse or not be suitable for other ones, leading to significant performance variability. Seamlessly switching to the most effective visual feature is a desirable quality for SLAM, but, currently, this involves a cumbersome manual task that demands substantial parameter tuning efforts and expert knowledge.**

In this paper, we present AnyFeature-VSLAM, an automated visual SLAM pipeline capable of switching to a chosen type of feature effortlessly and without manual intervention. The tuning of parameters associated with visual features is performed automatically to achieve the best performance. We built AnyFeature-VSLAM on top of ORB-SLAM2, one of the most popular and widely used feature-based visual SLAM implementations. Through extensive experiments across various benchmark datasets, we demonstrate that AnyFeature-VSLAM consistently delivers good results irrespective of the chosen visual feature, outperforming baseline implementations. Specifically, our paper includes a quantitative assessment of trajectory estimation involving seven different keypoint and descriptor combinations across thirty sequences spanning four distinct publicly available datasets. Furthermore, we showcase the enhanced flexibility of our system by subjecting it to four additional challenging datasets. Code publicly available at: https://github.com/alejandrofontan/AnyFeature-VSLAM.

I. Introduction

Simultaneous Localization and Mapping (SLAM) aims at estimating the trajectory of a mobile robot while concurrently constructing a representation of the environment [10]. Feature-based visual SLAM (VSLAM) refers to the modality that utilizes RGB video as the main sensory input, and specifically multi-view correspondences between 2D salient image features (see an illustration of such features in the first two columns of Figure 1). VSLAM has witnessed remarkable advancements, progressing from early filtering methods [20, 4, 17, 36] to contemporary pipelines rooted on parallel tracking and mapping [50, 26, 49]. Similarly, the evolution of feature detection and description spans from image patches [65] to corner detectors and descriptors [55, 56, 45, 11, 57, 40, 2, 5, 43, 3], and more recently, to methods heavily reliant on deep learning [19, 54].

Ideally, VSLAM features should capture distinctive local patterns and exhibit robustness under illumination and viewpoint changes, rolling shutter effects, sensor asynchrony, and calibration errors. They should also present invariance to rotation and perspective effects, facilitating wide baseline matching and consequently enhancing the accuracy of Bundle Adjustment (BA) [75] and facilitating Visual Place Recognition (VPR) [63].

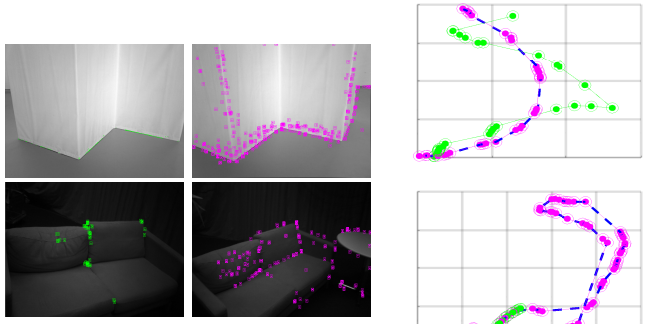


Fig. 1: Illustration of the effectiveness of our automated method for changing features. **Left column:** Frames from challenging sequences from 2 public datasets and ORB-SLAM2 tracks (in green). Note the small amount of features tracked, that lead to failure in both sequences. **Center column:** Features tracked (in purple) for the same 2 frames by our **AnyFeature-VSLAM** with R2D2 keypoints. Without any manual parameter tuning, we leverage the distinctiveness of R2D2 compared to ORB, leading to superior performance. The **right column** shows the trajectories estimated by ORB-SLAM2 (in green) and our approach (in purple). Note how AnyFeature-VSLAM (R2D2) successfully estimates the entire trajectory, while ORB-SLAM2 experiences catastrophic failure.

Lastly, the quick extraction and comparison of features are crucial for real-time execution, ensuring seamless integration into real-time pipelines for use cases within augmented reality (AR), virtual reality (VR), and robotics.

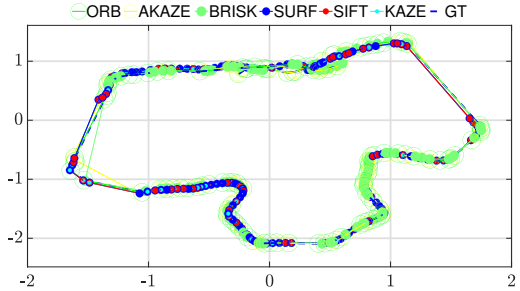


Fig. 2: **AnyFeature-VSLAM** results for arbitrary features. We achieve comparable trajectory accuracy for six different state-of-the-art features without manual tuning.

		Size	Scale Inv.	Rot. Inv.	Comput. Speed
BRIEF [11]	Binary	32 x 8U	✗	✗	$\sim$ 10 ms
ORB [57]	Binary	32 x 8U	✗	✔	$\sim$ 10 ms
BRISK [40]	Binary	48† x 8U	✔	✔	$\sim$ 10 ms
AKAZE [2]	Binary	61 x 8U	✔	✔	$\sim$ 35 ms
SURF [5]	Non-bin	64 x 32F	✔	✔	$\sim$ 35 ms
SIFT [43]	Non-bin	128 x 8U	✔	✔	$\sim$ 50 ms
KAZE [3]	Non-bin	64 x 32F	✔	✔	$\sim$ 150 ms
SuperPoint [19]	Non-bin	256 x 32F	✔	✔	-
R2D2 [54]	Non-bin	128 x 32F	✔	✔	-

TABLE I: **Characteristics of Different Descriptors**. This table provides a comparison of various descriptors that could be used in VSLAM applications. It outlines key characteristics such as scale invariance, rotation invariance, and computational speed. It should be noticed that only ORB and BRISK attain the necessary speed for the real-time operation of VSLAM. †BRISK is described by Leutenegger et al. in [40] as a 64-byte descriptor; however, the author's implementation has 48 bytes.

Reference	Acronym	Keypoint/Descriptor
Fontan et al. [24]	SID SLAM	BoW with AKAZE
Tang et al. [71]	GCNv2	Gcnv2 modification of ORB-SLAM2
Deng et al. [18]	SuperPoint SLAM	Superpoint modification of ORB-SLAM2
Schops et al. [62]	BAD SLAM	BoW with BRIEF
Qin et al. [52]	VINS-MONO	Corner features
Gao et al. [28]	LDSO	BoW with ORB
Lee et al. [39]	LCSD SLAM	ORB for geometric BA
Mur et al. [49]	ORB-SLAM	ORB tracking, mapping and LC
Lim et al. [42]		Implementation with FAST + BRIEF
Leutenegger et al. [41]	OKVIS	Implementation with BRISK
Song et al. [67]		Implementation with ORB
Concha et al. [14]	RGBDTAM	BoW with ORB
Strasdat et al. [68] [69]	Large scale SLAM	Loop Closure Detection with SURF
Klein and Murray [36] [37]	PTAM	FAST + Cross corr.
Davison et al. [17]	MonoSLAM	Image patches + Cross corr.

TABLE II: **Visual SLAM Baselines and associated Keypoint/Descriptor.** This table provides an overview of relevant SLAM pipelines and their respective keypoint and descriptor configurations. It offers insights into the choices made for robust simultaneous localization and mapping. Note that all VO/VSLAM systems listed in the table share a common aspect: their implementation commits to one specific feature, although the methodologies proposed should not be restricted to these particular features.

Each specific VSLAM implementation heavily depends on a particular visual feature, to the extent of trademarking the entire pipeline in the ORB-SLAM series case [50, 49, 12]. Refer to Table II for an illustrative summary of various SLAM pipelines from the literature and the keypoints and descriptors they use in an exclusive manner. However, a particular type of visual feature should not constraint the rest of the VSLAM pipeline, that should be kept generic. While the high computational demands of certain features can indeed restrict their use in real-time VSLAM, we argue that **it is not only these computational constraints but also a lack of flexibility in the design of SLAM pipelines what limits the full utilization of the capabilities offered by different types of features [10].**

In this work, our aim is achieving seamless feature interchangeability within a VSLAM pipeline, eliminating the need for manual tuning and avoiding performance degradation (see Figure 2). Specifically, our contribution involves automating six feature-dependent processes commonly found in VSLAM implementations. We present a comprehensive set of experimental ablations to underscore the enhanced flexibility of our system compared to conventional feature-dependent methodologies. We conduct a quantitative assessment of trajectory estimation performance, involving seven state-of-the-art keypoint and descriptor combinations, across 30 sequences drawn from four distinct publicly available datasets. Additionally, we extend the capabilities of ORB-SLAM2 [49] to introduce AnyFeature-VSLAM, which, to the best of our knowledge, stands as the first adaptable and feature-interchangeable VSLAM pipeline. Our novel framework facilitates the seamless interchangeability of keypoints and descriptors with zero tuning efforts, thereby fostering flexibility and innovation in the realm of visual SLAM applications.

II. Related Work

Keypoints and Descriptors. Among the most popular visual features in the literature [47, 1, 34], briefly summarized in Table I, Lowe et al.'s SIFT [43], Bay et al.'s SURF [5] and Alcantarilla et al.'s KAZE [3] stand out due to their high descriptive capability and robustness to illumination and viewpoint changes. Both SIFT and SURF leverage the concept of Gaussian scale space to enable multiscale detection and description. KAZE adopts a locally adaptive approach to blurring through the utilization of nonlinear scale spaces.

These keypoints and descriptors have achieved a remarkable success in various applications, such as Structure from Motion (SfM) [75, 64, 32, 60, 61]. Nonetheless, despite ongoing research efforts aimed at enhancing their computational efficiency, such as exploiting GPU devices [66], these methods still fall short in terms of computational speed when compared to the more modest yet faster feature alternatives currently available [72]. This is particularly true in the context of real-time systems, such as VO or VSLAM [10], as well as on low-end platforms like small robots and AR/VR glasses [16].

Among the methods for finding keypoints, Rosten et al.'s FAST [55, 56] and Mair et al.'s AGAST [45] are efficient and find reasonable corner keypoints. Calonder et al.'s BRIEF [11] is a feature descriptor that uses simple binary tests between pixels in a smoothed image patch. Its performance is similar to SIFT in many respects, including robustness to lighting, blur, and perspective distortion. However, it is highly sensitive to image rotation and scale changes, which limits its application to general tasks.

Rublee et al.'s ORB [57] and Leutenegger et al.'s BRISK [40] speed up feature detection and description by combining the strengths of the FAST corner detector and BRIEF binary descriptors, incorporating scale and rotation invariance. Notably, these extraction methods offer significantly enhanced computational efficiency when compared to SIFT, SURF and KAZE while maintaining a comparable performance, particularly in scenarios characterized by small image transformations. Alcantarilla et al.'s AKAZE [2] shows a compromise between speed and performance, using embedded numerical schemes to speed-up feature detection in nonlinear scale spaces.

Learned keypoints detectors and descriptors in the literature [29, 76, 58, 15] include DeTone et al.'s SuperPoint [19], which contributed a fully-convolutional neural network architecture for keypoint detection and description. They have demonstrated a competitive performance for small appearance variations and significantly outperform handcrafted descriptors in low-texture scenarios and high appearance variations. Revaud et al.'s R2D2 [54] proposed a new learning-based feature extraction method that jointly detects and describes keypoints in images, resulting in sparse, repeatable, and reliable keypoints.

As these keypoint detectors and descriptors have shown excellent performance for matching tasks [33, 44, 51, 73, 74], it is reasonable to believe that they can be used to tackle all visual data-association tasks in 3D computer vision problems like SLAM and SfM, and that a learning-based Visual SLAM front-end will enable more robust applications in robotics and augmented/virtual reality.

Feature-based SLAM. The taxonomy of modern VO/SLAM, divided into feature-based, direct, and semi-direct (or hybrid) categories, has been extensively addressed in previous works [77, 24]. Here, we focus only on feature-based methods, i.e., those that extract and match a sparse set of salient image features using partially invariant descriptors to robustly recover both camera motion and structure [26].

The initial solutions for monocular SLAM filtered every frame sequentially, in a single processing thread, to jointly estimate the map feature locations and the camera pose [17, 13]. Among them, Davison et al. [17] detected salient image regions using the detection operator of Shi and Tomasi [65]. The map points of PTAM by Klein and Murray [36] [37] correspond to FAST corners matched by patch correlation. Strasdat et al. [68] presented a large scale monocular SLAM system with a front-end based on optical flow, followed by FAST feature matching and motion-only BA and using SURF features for loop detection.

The visual odometry of Song et al. [67] uses ORB features for tracking and a temporal sliding window BA back-end. Lim et al. [42] uses BRIEF features for tracking, mapping and loop detection, however it limits the system to in-plane trajectories. Leutenegger et al.'s OKVIS [41] used BRISK keypoints and descriptors and Qin et al.'s VINS-Mono [52] a Shi and Tomasi detector [65] for their visual-inertial odometries.

Mur et al. developed ORB-SLAM [50], the first fully complete SLAM pipeline to perform real-time, accurate tracking, mapping, loop detection, and loop closing, all with ORB features.

Forster et al.'s SVO [25, 26] combined photometric alignment for tracking and pixel triangulation, and feature-based joint optimization of structure and motion. Similarly, Lee et al.'s LCDO [39] combines photometric bundle adjustment of the local structure and motion [22] with feature-based bundle adjustment for larger optimization windows [49]. Numerous approach have been built upon ORB-SLAM, given its high degree of robustness and versatility [49, 12, 6, 38].

Typically, dense approaches add to a direct VO thread a (feature-based) bag-of-words loop closure [27]. For instance, Concha et al.'s RGBDTAM [14] and Gao et al.'s LDSO [28] search for ORB correspondences, Schops et al.'s BAD-SLAM [62] utilizes BRIEF, and Fontan et al.'s SID-SLAM [24] looks for AKAZE. Although all these works benefit from both point types, their loose coupling limits their performance compared to using the same landmarks in tracking, mapping, and relocalization tasks [50, 12].

All previous VO/VSLAM systems have in common that their implementation commits to one specific feature, although the methodologies proposed are not restricted to these particular features. Approaches such as Den et al.'s SuperPoint SLAM [18] or Tang et al.'s GCNv2 SLAM [71] constitute the first attempts to use learned features in real-time SLAM pipelines, but again they are tied to one single specific type of feature. Differently from all of them, our **AnyFeature-VSLAM**, is the first fully automated system enabling high-performance usage of any feature in VSLAM.

Algorithm 1 Automatic Tuning of Keypoint Extractor

function AUTOMATIC TUNING (I, keypoint type) ▷ I : grayscale image
  $\#\boldsymbol{f} \leftarrow$ Detect FAST (I, t_{fast} = 7) ▷ $\#(\cdot)$: operator 'number of $(\cdot)$'
  detector $\leftarrow$ createDetector (keypoint type)
  $t_s \leftarrow$ detector.getNominalThreshold() ▷ t_s : detection threshold at step s
  $t_{s-1} \leftarrow t_s + \delta t$ ▷ $\delta t = 1.1 \cdot t_s$
  $\#\boldsymbol{k}_{s-1} \leftarrow$ detector.detectKeypoints (I, t_{s-1})
  $e \leftarrow$ Initialize error ()
  for s = 1 : numMaxSteps **do** ▷ numMaxSteps = 200
    $\#\boldsymbol{k}_s \leftarrow$ detector.detectKeypoints (I, t_s)
    $e \leftarrow (\#\boldsymbol{k}_s - \#\boldsymbol{f})$
    If ($|e| < tol$) **then** break ▷ $tol = 0.01 \cdot \#\boldsymbol{f}$
    $\frac{\partial t}{\partial \#\boldsymbol{k}} \leftarrow \frac{t_s - t_{s-1}}{\#\boldsymbol{k}_s - \#\boldsymbol{k}_{s-1}}$ ▷ Estimate derivative
    $t_{s-1} \leftarrow t_s$ ▷ Update thresholds
    $t_s \leftarrow t_{s-1} + \frac{\partial t}{\partial \#\boldsymbol{k}} \cdot e$
    $\#\boldsymbol{k}_{s-1} \leftarrow \#\boldsymbol{k}_s$
  end for
  return t_s
end function

III. Automatic Usage of Any Feature into Visual SLAM

In this section, we provide a detailed explanation of our method for seamlessly switching the type of features within a VSLAM implementation. We offer technical insights into the modifications implemented to enable VSLAM to adapt to any feature without the need for manual tuning.

A. Keypoint and Descriptor Extraction

The quality and quantity of keypoints, defined by their distinctive 2D positions in the image domain, are crucial for feature-based VSLAM, impacting the system's accuracy, robustness, and speed. The extraction process typically begins with detecting keypoints using a minimum edge threshold t, followed by a selection process to spread their distribution over the image for accuracy, robustness and speed. Figure 3 illustrates how the number of keypoints varies for each type as the detection threshold $t_r = t/t_0$ deviates from the nominal value t_0 proposed in each implementation, underscoring the challenge in selecting an appropriate threshold.

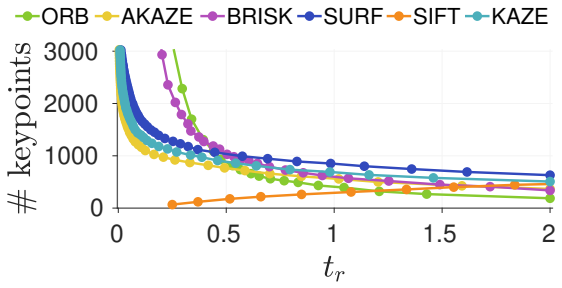


Fig. 3: Illustration of the number of detected keypoints (#) in an image for different values of the extracion threshold t. We plot in the *x axis* the normalized threshold $t_r = t/t_0$ with respect the nominal value from their respective implementations. Note the very different response between descriptors, not only in the absolute number of keypoints extracted for the same value but also for the different grow ratios. Moreover, the behavior of SIFT is inverted with respect to the other features, increasing the number of keypoints when the threshold increase.

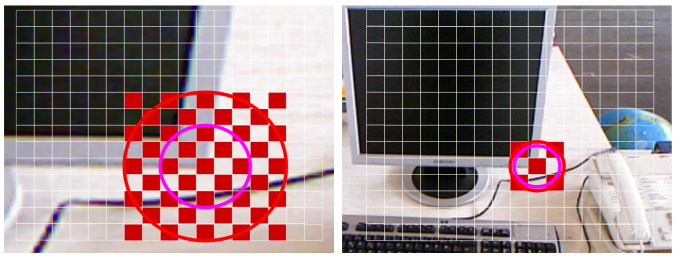


Fig. 4: The **size** of a keypoint (red circle) defines the area of pixels used for estimating its 2D image position. The **covariance** of a keypoint (purple circle) defines the uncertainty of its 2D image position. While an underlying relationship exists between the size of a keypoint and its covariance, there are no models that generalizes to all descriptors.

Our automatic tuning mechanism employs the count of FAST keypoints detected in the first image of a sequence as a proxy to approximate a functional threshold for any given feature. A preset extraction threshold $t_{fast} = 7$ effectively balances the computational demands with the complexity of correlating the number of keypoints to the texture of the image. We achieve convergence to the desired threshold through a gradient descent method, incrementally adjusting the extraction threshold until the number of detected keypoints aligns with the reference quantity of FAST keypoints, as detailed in Algorithm 1. This method ensures a robust initialization of the extractor, regardless of the detector type or the specific characteristics of the RGB input.

B. Keypoint Size

The size of a keypoint, commonly referred to as its radius in pixels, defines the area used for estimating the 2D image position of a salient point, as depicted by the red area in Figure 4. In VSLAM pipelines, keypoints of varying sizes facilitate matching across different views [50].

Given fixed camera intrinsics and an estimation of the motion

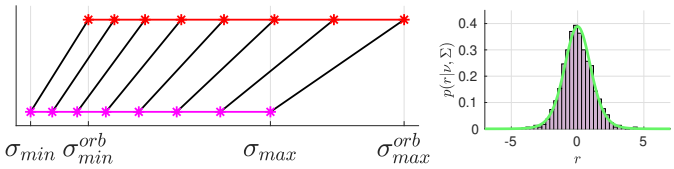


Fig. 5: **Left:** The figure provides a simplified representation of our approach to enable the utilization of the nominal size of keypoints for estimating their covariance (see Eq 1). This scaling **allows for the relative weighting of reprojection residuals** based on keypoint size while **ensuring the proper functioning of both outlier rejection and matching processes**. **Right:** In order **to taylor the outlier thresholds to the uncertainty associated with each specific descriptor** we utilize a 95% probability level from the **fitting of a *t location-scale distribution*** with ν degrees of freedom and scale Σ to the reprojection residuals r from the local Bundle Adjustment.

of the camera, it is feasible to predict transformations of image patches when observed from varying viewpoints. Specifically, the expected size s_i of a keypoint $\boldsymbol{k}_i$ in image i can be derived from its size in a reference image j using $s_i^* \approx \frac{z_j}{z_i} s_j$, assuming that the perspective distortion of the keypoints is predominantly caused by their distances z_i, z_j to the 3D point [23].

This predictive model facilitates the estimation of the size of a potential matching keypoint, thus enabling the selection of candidates, $\forall \boldsymbol{k}_i \in K_i$ such that $\frac{s_i^*}{\alpha} < s_i < \alpha s_i^*$, with a certain tolerance $\alpha \geq 1$ ($\alpha = 1.2$ in our experiments).

There exist various implementations for computing the nominal size of a keypoint (e.g. image diffusivity [2] [3] and resolution [57] [49]) which yield to different scales. Given the proportionality of our approach, we can use keypoints with any input size irrespective of its absolute scale.

C. Keypoint Covariance

The estimated positional uncertainty (σ) of a keypoint is used in VSLAM for weighting reprojection residuals, filtering outliers and actively defining search areas for matching. Typically, keypoints detected at the finest discretized resolution are assigned an initial value (σ_0) of 1 pixel. As resolutions change, it is adjusted proportionally $\hat{\sigma}(l) = f^l \cdot \sigma_0$, being l the level of the keypoint in the resolution pyramid and f the scaling factor between levels.

We propose a two-step method to estimate the functional covariances of keypoints (Figure 5). Initially, we address the sensitivity of the matching and outlier rejection processes to covariance scaling by adjusting the estimated covariances to align with the nominal parameters of ORB-SLAM2:

$$\sigma = \sigma_{min}^{orb} + \frac{\sigma_{max}^{orb} - \sigma_{min}^{orb}}{\sigma_{max} - \sigma_{min}} (\hat{\sigma} - \sigma_{min}), \tag{1}$$

where σ_{max} and σ_{min} are the maximum and minimum size values provided by the new descriptor, and σ_{max}^{orb} and σ_{min}^{orb} denote the maximum and minimum for ORB in ORB-SLAM2.

We fit a *t location-scale distribution* to the reprojection residuals after the joint optimization of camera poses and 3D

landmarks in the local Bundle Adjustment [35] [31]. The cost function is defined as the difference between the observations $\mathbf{k}_{ij} \in K$ of a set of keypoints and their predictions:

$$\mathbf{r}_{ij} = (r_u, r_v)_{ij}^T = \mathbf{k}_{ij} - \Pi(\mathbf{R}_j \cdot \mathbf{p}_i + \mathbf{t}_j), \tag{2}$$

where $\Pi(\cdot)$, determined by the intrinsic camera parameters, projects the point $\mathbf{p}_i \in \mathbb{R}^3$ into the image domain. $\mathbf{R}_j \in \mathrm{SO}(3)$ and $\mathbf{t}_j \in \mathbb{R}^3$ are the rotation and translation of frame j.

We assume an isotropic distribution for the residuals $r \in \mathbf{r}_u, \mathbf{r}_v$ in both image directions, to estimate the distribution parameters: degrees of freedom (ν) and scale (Σ). The estimation is performed by minimizing the negative log-likelihood of the distribution:

$$\{\nu, \Sigma\}^* = \underset{\nu,\Sigma}{\operatorname{argmin}} \{ - \sum_{r \in \mathbf{r}_u, \mathbf{r}_v} \log p(r|\nu, \Sigma) \}. \tag{3}$$

We identify outliers when their Mahalanobis residuals exceed a χ^2 threshold [49]. We dynamically adjust this threshold to tailor the uncertainty to each descriptor and scene conditions, scaling it according to the estimated distribution scale,

$$\mathbf{r}^T \sigma^{-2} \mathbf{r} > \Sigma^2 \cdot \chi^2. \tag{4}$$

This approach effectively manages positional uncertainty of keypoints, enhancing residual weighting, outlier rejection, and matching processes.

D. Feature Matching Thresholding

Feature matching involves establishing correspondences between individual keypoints, denoted as $\mathbf{k}_i \in K_i$, within a source image I_i, and their corresponding keypoints, denoted as $\mathbf{k}_j \in K_j$, in a target image I_j. This process ensures that each correspondence accurately represents the multi-view of the same 3D point. To expedite and enhance the matching, the candidates are reduced to a subset of keypoints $\mathbf{k}_j \in \overline{K}_j \in K_j$, localized within the vicinity of its 2D projection.

The matching of keypoints entails the comparison of their associated descriptors, achieved through the computation of the distance between them, expressed as $d_{i,j} = f(\mathbf{D}_i, \mathbf{D}_j)$. The choice of distance metric is contingent upon the descriptor type employed. For binary descriptors $\mathbf{D} \in \mathbb{I}^n$, such as those utilized in ORB, AKAZE, or BRISK, the Hamming distance is the conventional choice. In contrast, continuous descriptors $\mathbf{D} \in \mathbb{R}^n$, as exemplified by SURF, SIFT, or KAZE, are compared using the L2 norm.

If the computed distance between two descriptors falls below a predefined threshold d_{th}, their corresponding keypoints are marked as a match. As we show in Section V, the suitability of d_{th} extends beyond the choice of keypoints and descriptors, encompassing factors such as camera intrinsics, photometric calibration, and environmental conditions.

We propose an approach for its initial estimation and continuous update. Specifically, we maintain records of pairs consisting of the number of candidates $\overline{K}_j$ and the minimum computed distance $d_{min} = \min(f(\mathbf{D}_i, \mathbf{D}_j), \mathbf{k}_j \in \overline{K}_j)$. The threshold is determined as the most likely distance computed as the weighted average of the distances, with the weighting inversely proportional to the number of candidates:

$$d_{th} = \frac{\sum_{\mathbf{k}_i \in K_i} \#\overline{K}_j|_i}{(\#K_i)^2} \sum_{\mathbf{k}_i \in K_i} \frac{d_{min}}{(\#\overline{K}_j)}|_i. \tag{5}$$

This method enables the estimation of a valid threshold for any given arbitrary keypoint-descriptor pair without the need for a priori knowledge or fine-tuning, thus accommodating the evolving nature of incoming frames in a continuous video stream.

Dataset				Resolution
ETH	[62]	Indoor	Handheld	739 x 458
RGBD TUM	[70]	Indoor	Handheld	640 x 480
EuRoC	[7]	Indoor	Drone	752 x 480
KITTI	[30]	Outdoor	Car	1241 x 376
HAMLYN	[48, 53]	Intracorporeal	Endoscope	640 x 480
Mono TUM	[21]	Indoor/Outdoor	Handheld	640 x 480
Minimal Texture	[24]	Conceptual	Handheld	1280 x 720
MADMAX	[46]	Outdoor	Mobile Robot	1032 x 772

TABLE III: **Datasets** we used for the comparisons between vanilla ORB-SLAM2 [49] and our AnyFeature-VSLAM.

E. Redundancy Detection in Keyframes

A *survival of the fittest* strategy, typically used in visual SLAM, involves creating keyframes as frequently as possible to maximize its robustness, which is critical under challenging conditions [50]. However, as the complexity of bundle adjustment increases and to prevent the unbounded growth of keyframes, it becomes necessary to implement a policy for detecting and removing redundant keyframes.

Heuristics based on visual redundancy, such as the number of shared matches between keyframes, heavily rely on the repeatability of the selected keypoint-descriptor. Moreover, many of these approaches adopt greedy strategies for keyframe removal due to computational constraints. This can lead to the removal of keyframes from previously visited areas, transforming explored regions into unexplored ones, consequently impairing re-localization and map reuse, critical aspects of VSLAM.

In our AnyFeature-VSLAM, we have introduced a new policy for keyframe removal based on spatial constraints. We maintain a positional record for all created keyframes, both active and removed. Under this policy, a keyframe is eligible for removal if all positions in the record, closer than a certain distance ($0.1/d$ in our experiments, with d the median depth of the scene), are in proximity to at least one other active keyframe. This approach reduces the dependency of the removal policy on specific descriptors and ensures the presence of a distributed set of active keyframes across all visited areas.

IV. Experimental Results

In this section, we show the versatility of AnyFeature-VSLAM, and our novel seamless interchangeability of very different descriptors. To demonstrate this capability, we have implemented R2D2 [54], a learned keypoint detector grounded in self-supervised neural network training. Remarkably, without

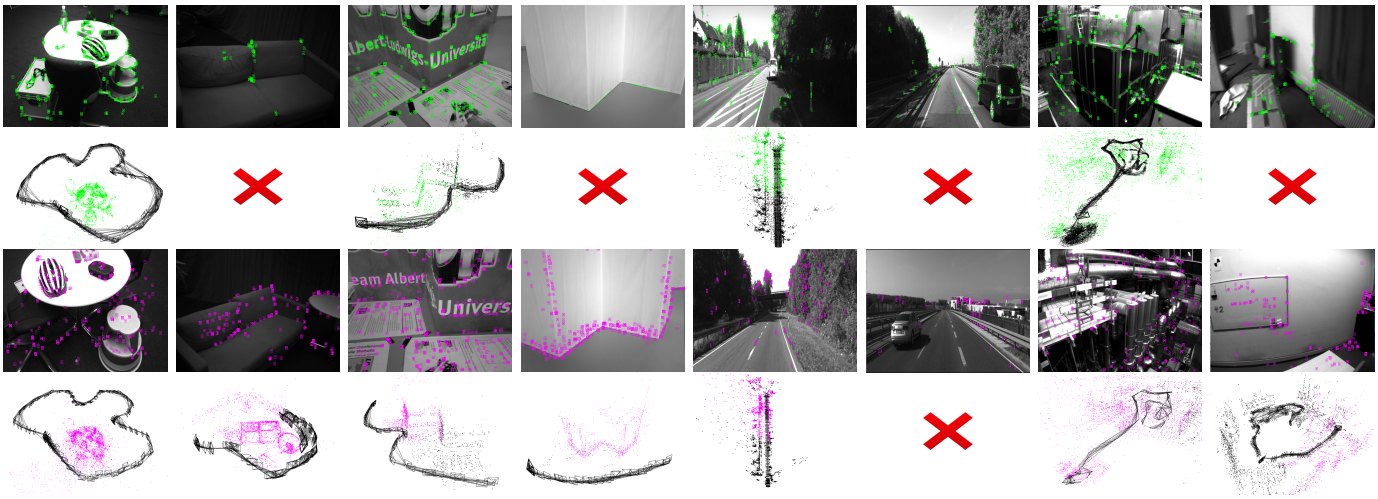


Fig. 6: **AnyFeature-VSLAM (R2D2)**. This figure showcase RGB sample frames with the active keypoints along with the estimated trajectories for vanilla ORB-SLAM2 (green upper rows) and AnyFeature-VSLAM (R2D2) (purple lower rows). In sequences with rich texture and structure, our AnyFeature-VSLAM(R2D2) **with non-additional fine-tuning**, performs competitively with vanilla ORB-SLAM2, thanks to our automated process for feature interchange. In challenging scenes where R2D2 is able to extract more keypoints (notice, for example, the sofa scene or the white wall corner), our system outperforms ORB-SLAM2. Red crosses stand for tracking failure.

		Tracked Frames (#)		ATE	
Dataset	Sequence	Or*	R2D2	Or*	R2D2
ETH	Table 3 (e)	1170	**1180**	**0.5**	**0.5**
ETH	Einstein 1 (d)	470	**480**	14.7	**1.9**
ETH	Large Loop 1 (d)	456	**1064**	2.6	**1.1**
ETH	Ceiling 1 (d)	-	**1490**	-	**1.8**
RGBD TUM	Fr1 xyz (e)	**785**	778	**1.0**	**1.0**
RGBD TUM	Fr2 Desk (e)	**2954**	2942	**0.7**	0.8
RGBD TUM	Fr3 Long Office (e)	**2559**	2557	**1.1**	**0.9**
RGBD TUM	Fr3 Str Tex Near (e)	**103**	**103**	1.3	**1.0**
RGBD TUM	Fr3 Str NoTex Far (d)	15	**790**	-	**1.2**
RGBD TUM	Fr3 Str NoTex Near (d)	10	**107**	-	**9.6**
EUROC	MH 02 easy (e)	3007	**3037**	**5.9**	**5.9**
EUROC	MH 03 medium (d)	**2639**	2560	6.0	**5.5**
EUROC	V1 02 medium (d)	1602	**1614**	1.1	**1.0**
KITTI	00 (e)	**4541**	**4541**	**7.0**	7.4
KITTI	01 (e)	-	-	-	-
KITTI	04 (e)	**271**	269	**0.6**	3.96
KITTI	07 (e)	**1099**	1098	**2.0**	**2.0**

TABLE IV: **Trajectory Accuracy Comparison**: The table compares the trajectory estimation quality between the original ORB-SLAM2 [49] (Or*) and our AnyFeature-VSLAM (R2D2) implementation. In *easy* (e) sequences, our AnyFeature-VSLAM (R2D2) and ORB-SLAM2 attain similar accuracy, while in *difficult* (d) sequences, AnyFeature-VSLAM (R2D2) performs substantially better. We conduct this evaluation across 17 sequences drawn from four distinct datasets: ETH , RGB-D TUM, KITTI, and EuRoC. The Absolute Trajectory Error (ATE) [70] is in centimeters for ETH, RGB-D TUM, and EuRoC, and in meters for KITTI. Values in **bold** indicate the **best performance**, and "-" stands for tracking failure.

engaging in any manual tuning efforts with respect to the ORB configuration, we are able to run AnyFeature-VSLAM (R2D2) across eight distinct datasets (refer to Table III).

A. *Easy vs. Difficult Sequences*

In our experiments, we employed sequences of varying difficulty levels. The inclusion of easy sequences enabled a direct comparison with vanilla ORB-SLAM2. As shown in Table IV, our approach with R2D2, without any additional manual tuning, achieves accuracy results comparable to those in the *easy* sequences, while notably surpassing it in the *difficult* ones.

Figure 6 compiles RGB images displaying the active keypoints for both vanilla ORB-SLAM2 and AnyFeature-VSLAM (R2D2), alongside the estimated trajectories. Observe how, in sequences characterized by rich texture and structure, our AnyFeature-VSLAM (R2D2), without any fine-tuning, performs comparably to vanilla ORB-SLAM2, thanks to our automated process for feature interchange. Furthermore, in challenging scenes where R2D2 can extract more keypoints (as evident in, for instance, the sofa scene or the white wall corner), our AnyFeature-VSLAM (R2D2) outperforms ORB-SLAM2.

B. *Challenging datasets*

We extend our evaluation to datasets from heterogeneous sources to showcase the versatility provided by our automatic tuning. This versatility extends beyond indoor and outdoor scenes, encompassing intracorporeal environments, planetary analog settings, and even conceptual data. Figure 7 provides examples of RGB frames and camera trajectories estimated by AnyFeature-VSLAM (R2D2) in these challenging scenarios.

In Table V, we show a qualitative analysis comparing the performance of vanilla ORB-SLAM2 with AnyFeature-VSLAM (R2D2) in these scenarios. It is noteworthy that the number of tracked frames indicates a similar performance

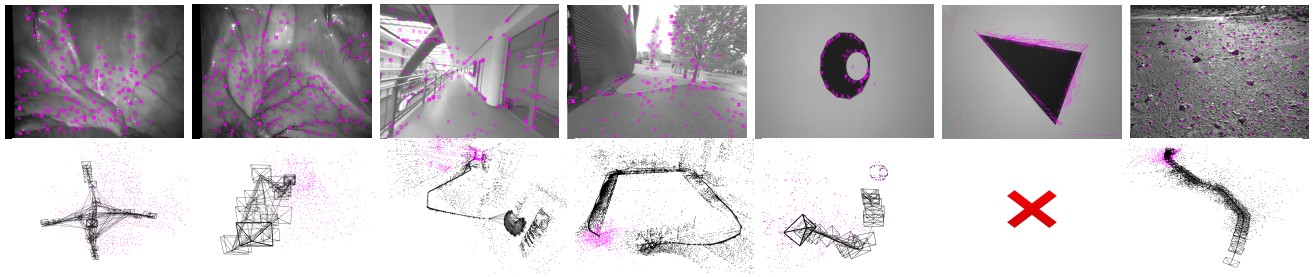


Fig. 7: **AnyFeature-VSLAM (R2D2) in challenging scenarios**. Our automated system, without any manual tuning, demonstrates robust performance across a wide spectrum of challenging scenarios: indoor and outdoor environments, intracorporeal sequences, planetary analog scenes, and even conceptual data. The red cross stands for tracking failure.

				Tracked Frames (#)		Keyframes (#)		Points $\times 10^3$ (#)		Observations $\times 10^3$ (#)		Observations (#) / Point (#)	
Dataset	Sequence			Or*	R2D2	Or*	R2D2	Or*	R2D2	Or*	R2D2	Or*	R2D2
MONO TUM	Sequence 17	Indoor	Handheld	**2942**	2937	**421**	276	**15.6**	13.1	**122.3**	89.5	**7.9**	6.8
MONO TUM	Sequence 23	Outdoor	Handheld	2970	**2974**	**693**	435	**27.0**	21.6	**202.4**	131.8	**7.5**	6.1
HAMLYN	Rectified 01	Intracorporeal	Endoscope	777	**922**	**40**	32	**2.3**	2.1	**16.0**	13.7	**6.9**	6.5
HAMLYN	Rectified 06	Intracorporeal	Endoscope	**1136**	1079	**40**	20	**2.0**	1.4	**14.2**	7.7	**7.1**	5.4
HAMLYN	Rectified 08	Intracorporeal	Endoscope	**512**	464	**71**	18	**2.2**	0.7	**13.7**	3.2	**6.2**	4.7
Minimal Texture	Circ. Dodec. 1	Conceptual	Handheld	-	862	-	108	-	3.5	-	33.9	-	9.6
Minimal Texture	Triangle 1	Conceptual	Handheld	-	-	-	-	-	-	-	-	-	-
MAD MAX	A-0	Outdoor	Mobile Robot	2025	**2479**	**378**	229	**22.6**	19.9	**156.8**	111.7	**6.9**	5.6

TABLE V: **Statistics of the SLAM Graph**: The presented table provides a qualitative comparison of key graph metrics within the SLAM system. The number of tracked frames between vanilla ORB-SLAM2 and our implementation indicates similar performance. Crucially, the final graph size for AnyFeature-VSLAM (R2D2) is significantly smaller than that of ORB-SLAM2, aligning with the expectation that R2D2 identifies more repeatable and reliable keypoints compared to ORB. AnyFeature-VSLAM (R2D2) effectively leverages this advantage without any prior fine-tuning, achieving comparable performance while utilizing a reduced yet more reliable dataset size. Biggest value **bold** and smallest underlined. We refer as Or* to the original implementation of ORB-SLAM2.

between vanilla ORB-SLAM2 and our AnyFeature-VSLAM (R2D2) without any manual tuning. Furthermore, the size of the final graph for AnyFeature-VSLAM (R2D2) is considerably smaller when compared to ORB-SLAM2. This is consistent with the expectation that R2D2 keypoints are more repeatable and reliable than ORB ones, allowing AnyFeature-VSLAM to utilize a smaller yet more reliable amount of data to achieve similar performance.

V. Implementation Details and Sensitivity Analysis

In this section, we present implementation details pertaining to AnyFeature-VSLAM and conduct a sensitivity analysis for our approach. We evaluate the performance of six widely recognized detector-descriptor pairs (as listed in Table I): ORB, AKAZE, BRISK, SURF, SIFT, and KAZE. Our ablation studies encompass sequences from four diverse datasets: RGBD-TUM, ETH, EUROC, and KITTI (refer to Table III for details).

Trajectory Estimation and SLAM Graph Analysis: We assess the accuracy of trajectory estimation using metrics defined in [70]: the Absolute Trajectory Error (ATE) and the Relative Pose Error (RPE). Additionally, we consider the number of tracked frames and the count of loop closures. Our evaluation includes metrics concerning the final SLAM graph, allowing for a qualitative comparison of descriptor performance.

Table XI shows how AnyFeature-VSLAM, facilitates consistent SLAM graphs across all six descriptor combinations. This result aligns with expectations, as these descriptors are anticipated to exhibit similar behavior under conditions of minimal motion and favorable environmental circumstances. Table X shows how all descriptor combinations achieve comparable trajectory estimation errors, reinforcing the consistency of their performance. Notably, in most occasions, different combinations outperform the original ORB-SLAM2 [49], illustrating the relevance of feature interchangeability and validating our initial research hypotheses and our work.

Feature Matching Threshold Estimation. The determination of the feature matching threshold d_{th} (as discussed in Section III-D), relies on the analysis of candidate counts $\#\overline{K}_j$ and the minimal descriptor distances d_{min} observed within the local keyframe window as implemented in ORB-SLAM2 [49].

Real-time threshold updates based on a partial map can pose a risk, especially in challenging conditions like textureless areas, potentially leading to suboptimal parameter estimates. To reduce this risk, we store all computed thresholds and update the parameter using their median value.

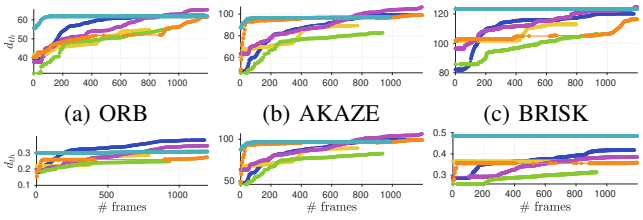
In Figure 8, we present the evolution of d_{th} for six different descriptors across distinct sequences captured using various cameras. Notably, we observe a transition from initially conservative values to stable thresholds over time. It is remarkable that even when employing the same descriptor in different sequences, as indicated in Table VII, different threshold values

	Keypoint Detection - Descriptor Extraction (ms / frame)							Complete Extraction (ms / frame)							Automatic Tuning (s)					
	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka	Or	Ak	Br	Su	Si	Ka
ETH - 30 Hz	4-3	4-2	18-17	9-4	16-14	24-20	82-44	**7**	**6**	35	**13**	31	44	126	0.2	0.8	0.3	0.5	0.5	4.1
RGBD TUM - 30 Hz	5-3	6-2	18-15	8-4	19-14	23-18	84-40	**8**	**9**	32	**13**	34	41	124	0.2	1.2	0.3	0.5	0.5	5.3
KITTI - 10 Hz	8-3	11-2	25-19	10-5	26-10	34-26	95-53	**11**	**14**	**45**	**15**	**37**	60	169	0.2	2.2	0.4	0.8	0.7	6.8
EUROC - 20 Hz	6-3	7-2	18-16	8-4	18-13	25-19	90-42	**8**	**10**	34	**12**	32	44	133	0.2	1.2	0.3	0.5	0.5	5.1

TABLE VI: **Keypoint Extraction Profiling**. We report the median extraction times for keypoint detection and descriptor extraction across each dataset and feature. In **bold**, we highlight the values where the duration necessary for the complete extraction process supports real-time operation (less than half the frequency (Hz) of the respective image stream). Our findings indicate that only the ORB and Brisk descriptors achieve speeds suitable for real-time applications across all evaluated datasets. Notation: **Or*** denotes the ORB extractor used in ORB-SLAM2 [49]. We utilize the OpenCV [8] implementations for ORB (**Or**), AKAZE (**Ak**), SURF (**Su**), SIFT (**Si**), and KAZE (**Ka**). For BRISK (**Br**), we adopt the latest author's implementation [40], noting our experiments show that the OpenCV version significantly slows during keypoint detection. The procedure for automatic threshold adjustment for extraction, detailed in Section III-A, is time-consuming but is executed only once at the start of each sequence, thereby not impeding real-time performance.

	Feature Matching Threshold (d_{th})						
	Or*	Or	Ak	Br	Su	Si	Ka
1a	62.4	62.4	99.4	120.1	0.4	164.8	0.4
2a	54.8	54.8	89.3	113.0	0.3	145.1	0.4
2e	65.3	65.3	106.5	124.7	0.3	156.3	0.4
2g	55.1	55.1	82.7	106.4	0.2	140.1	0.3
3a	61.5	61.5	99.1	116.3	0.3	132.3	0.4
4a	61.8	61.8	97.2	123.3	0.3	115.2	0.5

TABLE VII: **Feature Matching Threshold Estimation**. This table presents the final estimations for the matching threshold d_{th}. Note that the same descriptor in distinct sequences results in varying threshold values. Descriptors: ORB [49] (**Or***), ORB (**Or**), AKAZE (**Ak**), BRISK (**Br**), SURF (**Su**), SIFT (**Si**), KAZE (**Ka**).



(a) ORB (b) AKAZE (c) BRISK

(d) SURF (e) SIFT (f) KAZE

Table_3 (ETH), Fr1. xyz (RGBD TUM), Fr2. desk (RGBD TUM), Fr3. Str Tex Far (RGBD TUM), MH_01_easy (EuRoC), 00 (KITTI)

Fig. 8: **Feature Matching Threshold Tuning**: Evaluation of feature matching threshold convergence across the initial frames for six descriptors in six distinct sequences captured by different cameras. Noteworthy is the observation that even for the same descriptor applied in different sequences, distinct threshold values are obtained. Furthermore, it is evident that during the initialization phase, the thresholds tend to exhibit a more conservative behavior.

are observed accounting for factors such as camera intrinsics and environmental conditions.

t Location-Scale Distribution Fitting in Local Bundle Adjustment. In the context of ORB-SLAM2 [49], we undertake the process of fitting a *t location-scale distribution* to the reprojection residuals generated during the local Bundle Adjustment phase. Specifically, we accumulate all reprojection residuals that emerge from the joint optimization of 3D landmarks and camera poses. The fitting procedure is initiated once the keyframe window attains a certain baseline, normalized by the median depth of points within the scene (in our experimental setup, denoted as $b = 0.5/z$). The outcomes, including the distribution parameters ν, Σ, and the threshold employed for outlier rejection $\chi^2_{95\%}$, are presented in Table VIII.

Implementations for Keypoint Detection and Description. Within AnyFeature-VSLAM, we assess the detectors using their implementations provided by OpenCV [8]. Surprisingly, the BRISK implementation experiences a significant slowdown when the extraction threshold is decreased. Consequently, we have opted for the most recent author's implementation of BRISK [40] in our experiments.

Table VI offers a detailed time profiling for keypoint detection and description. This analysis reveals that only ORB and BRISK demonstrate suitability for real-time operation, maintaining a 30 Hz video stream. Notably, as anticipated, KAZE and SIFT prove to be the most time-consuming options.

Additionally, Table VI provides insights into the time required for our automatic tuning process, juxtaposed with the extraction time per frame. Although the tuning process takes approximately one second, which might seem slow relative to the extraction rate, it's important to note that this process occurs only once, during the first frame of the sequence.

Real-time Operation and Multi-threading: The practice of separating tracking, mapping, and loop closure tasks into distinct threads plays a pivotal role in achieving real-time performance. In our implementation, we have introduced a modification to the keyframe creation policy originally employed in ORB-SLAM2.

In the conventional ORB-SLAM2 approach, when the necessity to create a new keyframe arises (e.g., due to a reduction in the number of matches), it traditionally halts

	ORB			AKAZE			BRISK		
	$\chi^2_{95\%}$	ν	Σ	$\chi^2_{95\%}$	ν	Σ	$\chi^2_{95\%}$	ν	Σ
1a	2.9	2.7	0.3	6.0	1.9	0.2	5.2	2.1	0.2
2a	5.9	-	-	5.9	-	-	5.9	-	-
2e	2.9	2.9	0.3	4.2	2.6	0.3	3.8	2.5	0.3
2g	1.7	2.8	0.2	2.0	2.3	0.2	2.5	2.2	0.2
3a	2.9	3.0	0.3	3.4	2.3	0.2	3.4	2.9	0.3
4a	2.2	2.7	0.2	2.4	1.9	0.1	3.3	1.8	0.2

	SURF			SIFT			KAZE		
	$\chi^2_{95\%}$	ν	Σ	$\chi^2_{95\%}$	ν	Σ	$\chi^2_{95\%}$	ν	Σ
1a	5.9	2.4	0.3	5.9	1.9	0.3	5.5	2.2	0.3
2a	5.9	-	-	5.9	-	-	5.9	-	-
2e	4.8	3.0	0.4	3.2	3.6	0.4	4.4	2.6	0.3
2g	3.7	2.1	0.2	2.2	1.4	0.2	2.8	2.1	0.2
3a	3.7	3.2	0.3	4.4	2.4	0.1	3.9	2.9	0.3
4a	3.7	2.2	0.2	2.6	1.6	0.2	5.6	1.7	0.2

TABLE VIII: **t Location-Scale Distribution Fitting Analysis**. In this table, we present the shape ν and scale Σ, of the best *t location-scale distribution* fit for the reprojection residuals from ORB-SLAM2's local Bundle Adjustment. We also estimate a threshold $\chi^2_{95\%}$, corresponding to a 95% confidence level, used for outlier removal. In the case of sequence *2a*, the baseline remains smaller than the minimum, preventing the fitting of the distribution. Consequently, the original outlier threshold is retained.

Code	Dataset	Sequence	# RGB
1a		table 3	1182
1b		table 4	1018
1c		table 7	1677
1d		cables 1	1182
1e	ETH	repetitive	1968
1f		einstein 1	489
1g		planar 2	632
1h		planar 3	903
1i		plant scene 1	742
1j		large loop 1	1513
2a		RGBD dataset freiburg1 xyz	800
2b		RGBD dataset freiburg1 desk	615
2c		RGBD dataset freiburg1 desk2	642
2d	RGBD TUM	RGBD dataset freiburg2 xyz	3671
2e		RGBD dataset freiburg2 desk	2967
2f		RGBD dataset freiburg3 long office household	2587
2g		RGBD dataset freiburg3 structure texture far	940
2h		RGBD dataset freiburg3 structure texture near	1101
3a		00	4543
3b		01	1103
3c		02	4663
3d		03	803
3e	KITTI	04	273
3f		05	2763
3g		06	1103
3h		07	1103
3i		08	4073
3j		09	1593
4a		MH 01 easy	3684
4b		MH 02 easy	3042
4c		MH 03 medium	2702
4d		MH 04 difficult	2035
4e		MH 05 difficult	2275
4f	EuRoC	V1 01 easy	2914
4g		V1 02 medium	1712
4h		V1 03 difficult	2151
4i		V2 01 easy	2282
4j		V2 02 medium	2350
4k		V2 03 difficult	1924

TABLE IX: **Sequences** for trajectory accuracy comparison.

the local Bundle Adjustment optimization process. While this alteration contributes to improved real-time operation, it may come at the expense of system robustness. In our approach, we prioritize system stability by delaying the insertion of a new keyframe until after the optimization process is completed. This deliberate trade-off enhances system resilience at the potential cost of a temporary slowdown in processing speed.

In essence, even if a significant number of matches introduced by a new descriptor temporarily slows down the optimization process, our system remains functional, thereby avoiding catastrophic failures and enhancing overall flexibility.

Visual Place Recognition: In AnyFeature-VSLAM, we enable the visual place recognition (VPR) module with any descriptor by making modifications to the ORB-SLAM2 [49] and DBoW2 library [27] modules for loop closure and map merging. However, this has key limitations. Firstly, it requires a carefully assembled vocabulary every time a new descriptor is tested. Secondly, the reliance on a training dataset (such as Bovisa 2008-09-01 [7] in the case of ORB-SLAM2) restricts its applicability. For example, if the training data consists of outdoor images and we intend to use the system indoors, these limitations can significantly curtail the system's flexibility and adaptability. Future work should address newer VPR approaches, such as HBST [59], where a binary tree of feature descriptors is built online and hence adapts to domain changes.

VI. Conclusion

AnyFeature-VSLAM is an automated visual SLAM system that seamlessly adapts to different visual features without any manual tuning. Built over ORB-SLAM2 [49], it consistently delivers competitive performance across utterly different features and datasets, offering flexibility and innovation in visual SLAM for robotics and computer vision. The code is publicly available for broader accessibility at: https://github.com/alejandrofontan/AnyFeature-VSLAM.

Future Work. The methodologies proposed in this work are not restricted to monocular cameras and can be extended to stereo, RGB-D and visual-inertial setups. We will research solutions to overcome limitations to seamless feature interchange imposed by current bag-of-words methods. AnyFeature-VSLAM allows for automating the usage of any feature into visual SLAM, future research should focus on optimization to seamlessly obtain the highest performance.

Acknowledgments

The authors would like to thank Mur et al. for the contribution of ORB-SLAM2 [49] and for allowing its academic use for the benefit of the community.

This research was partially supported by funding from ARC Laureate Fellowship FL210100156 to Michael Milford and the QUT Centre for Robotics, the Spanish Government under Grant PID2021-127685NB-I00 and TED2021-131150B-I00 and the Aragon Government under Grant DGA_FSET45_20R.

References

[1] Henrik Aanæs, Anders Lindbjerg Dahl, and Kim Steenstrup Pedersen. Interesting interest points: A comparative study of interest point performance on a unique data set. *International Journal of Computer Vision*, 97:18–35, 2012.

[2] Pablo F Alcantarilla and T Solutions. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell*, 34(7):1281–1298, 2011.

[3] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI 12*, pages 214–227. Springer, 2012.

[4] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE robotics & automation magazine*, 13(3):108–117, 2006.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*, pages 404–417. Springer, 2006.

[6] Berta Bescos, José M Fácil, Javier Civera, and José Neira. Dynaslam: Tracking, mapping, and inpainting in dynamic scenes. *IEEE Robotics and Automation Letters*, 3(4): 4076–4083, 2018.

[7] Andrea Bonarini, Wolfram Burgard, Giulio Fontana, Matteo Matteucci, Domenico Giorgio Sorrenti, Juan Domingo Tardos, et al. Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets. In *In proceedings of IROS*, volume 6, page 93, 2006.

[8] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[9] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10):1157–1163, 2016.

[10] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

[11] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 778–792. Springer, 2010.

[12] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37 (6):1874–1890, 2021.

[13] Javier Civera, Andrew J Davison, and JM Martinez Montiel. Inverse depth parametrization for monocular slam. *IEEE transactions on robotics*, 24(5):932–945, 2008.

[14] Alejo Concha and Javier Civera. RGBDTAM: A cost-effective and accurate RGB-D tracking and mapping system. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2017.

[15] Gabriela Csurka and Martin Humenberger. From handcrafted to deep local invariant features. *arXiv preprint arXiv:1807.10254*, 2(1), 2018.

[16] Andrew J Davison. Futuremapping: The computational structure of spatial ai systems. *arXiv preprint arXiv:1803.11288*, 2018.

[17] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007.

[18] Chengqi Deng, Kaitao Qiu, Rong Xiong, and Chunlin Zhou. Comparative study of deep learning based features in slam. In *2019 4th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, pages 250–254. IEEE, 2019.

[19] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018.

[20] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[21] Jakob Engel, Vladyslav Usenko, and Daniel Cremers. A photometrically calibrated benchmark for monocular visual odometry. *arXiv preprint arXiv:1607.02555*, 2016.

[22] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017.

[23] Alejandro Fontan, Laura Oliva, Javier Civera, and Rudolph Triebel. Model for multi-view residual covariances based on perspective deformation. *IEEE Robotics and Automation Letters*, 7(2):1960–1967, 2022.

[24] Alejandro Fontan, Riccardo Giubilato, Laura Oliva, Javier Civera, and Rudolph Triebel. Sid-slam: Semi-direct information-driven rgb-d slam. *IEEE Robotics and Automation Letters*, 2023.

[25] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE, 2014.

[26] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2016.

[27] Dorian Gálvez-López and Juan D Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[28] Xiang Gao, Rui Wang, Nikolaus Demmel, and Daniel Cremers. Ldso: Direct sparse odometry with loop closure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2198–2204. IEEE, 2018.

[29] Steffen Gauglitz, Tobias Höllerer, and Matthew Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *International journal of computer vision*, 94:335–360, 2011.

[30] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[31] Ruben Gomez-Ojeda, Francisco-Angel Moreno, and Javier Gonzalez-Jimenez. Accurate stereo visual odometry with gamma distributions. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1423–1428. IEEE, 2017.

[32] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[33] Kun He, Yan Lu, and Stan Sclaroff. Local descriptors optimized for average precision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 596–605, 2018.

[34] Jared Heinly, Enrique Dunn, and Jan-Michael Frahm. Comparative evaluation of binary features. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II 12*, pages 759–773. Springer, 2012.

[35] Christian Kerl, Jürgen Sturm, and Daniel Cremers. Robust odometry estimation for rgb-d cameras. In *2013 IEEE international conference on robotics and automation*, pages 3748–3754. IEEE, 2013.

[36] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007.

[37] Georg Klein and David Murray. Improving the agility of keyframe-based slam. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part II 10*, pages 802–815. Springer, 2008.

[38] Jose Lamarca, Shaifali Parashar, Adrien Bartoli, and JMM Montiel. Defslam: Tracking and mapping of deforming scenes from monocular sequences. *IEEE Transactions on robotics*, 37(1):291–303, 2020.

[39] Seong Hun Lee and Javier Civera. Loosely-coupled semi-direct monocular SLAM. *IEEE Robotics and Automation Letters*, 4(2):399–406, 2018.

[40] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. Ieee, 2011.

[41] Stefan Leutenegger, Paul Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. Keyframe-based visual-inertial slam using nonlinear optimization. *Proceedings of Robotis Science and Systems (RSS) 2013*, 2013.

[42] Hyon Lim, Jongwoo Lim, and H Jin Kim. Real-time 6-dof monocular visual slam in a large-scale environment. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 1532–1539. IEEE, 2014.

[43] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.

[44] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Contextdesc: Local descriptor augmentation with cross-modality context. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2527–2536, 2019.

[45] Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II 11*, pages 183–196. Springer, 2010.

[46] Lukas Meyer, Michal Smíšek, Alejandro Fontan Villacampa, Laura Oliva Maza, Daniel Medina, Martin J Schuster, Florian Steidle, Mallikarjuna Vayugundla, Marcus G Müller, Bernhard Rebele, et al. The madmax data set for visual-inertial rover navigation on mars. *Journal of Field Robotics*, 38(6):833–853, 2021.

[47] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10): 1615–1630, 2005.

[48] Peter Mountney, Danail Stoyanov, and Guang-Zhong Yang. Three-dimensional tissue deformation recovery and tracking. *IEEE Signal Processing Magazine*, 27(4): 14–24, 2010.

[49] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.

[50] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[51] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *Proceedings of the IEEE international conference on computer vision*, pages 3456–3465, 2017.

[52] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[53] David Recasens, José Lamarca, José M Fácil, JMM Montiel, and Javier Civera. Endo-depth-and-motion:

Reconstruction and tracking in endoscopic videos using depth networks and photometric constraints. *IEEE Robotics and Automation Letters*, 6(4):7225–7232, 2021.

[54] Jerome Revaud, Philippe Weinzaepfel, César Roberto de Souza, and Martin Humenberger. R2D2: repeatable and reliable detector and descriptor. In *NeurIPS*, 2019.

[55] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*, pages 430–443. Springer, 2006.

[56] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):105–119, 2008.

[57] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[58] Ehab Salahat and Murad Qasaimeh. Recent advances in features extraction and description algorithms: A comprehensive survey. In *2017 IEEE international conference on industrial technology (ICIT)*, pages 1059–1063. IEEE, 2017.

[59] Dominik Schlegel and Giorgio Grisetti. Hbst: A hamming distance embedding binary search tree for feature-based visual place recognition. *IEEE Robotics and Automation Letters*, 3(4):3741–3748, 2018.

[60] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[61] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[62] Thomas Schops, Torsten Sattler, and Marc Pollefeys. BAD SLAM: Bundle Adjusted Direct RGB-D SLAM. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[63] Stefan Schubert, Peer Neubert, Sourav Garg, Michael Milford, and Tobias Fischer. Visual place recognition: A tutorial. *arXiv preprint arXiv:2303.03281*, 2023.

[64] Stephen Se, David Lowe, and Jim Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *The international Journal of robotics Research*, 21(8):735–758, 2002.

[65] Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[66] Sudipta N Sinha, Jan-Michael Frahm, Marc Pollefeys, and Yakup Genc. Gpu-based video feature tracking and matching. In *EDGE, workshop on edge computing using new commodity architectures*, volume 278, page 4321. Citeseer, 2006.

[67] Shiyu Song, Manmohan Chandraker, and Clark C Guest. Parallel, real-time monocular visual odometry. In *2013 ieee international conference on robotics and automation*, pages 4698–4705. IEEE, 2013.

[68] Hauke Strasdat, J Montiel, and Andrew J Davison. Scale drift-aware large scale monocular slam. *Robotics: science and Systems VI*, 2(3):7, 2010.

[69] Hauke Strasdat, Andrew J Davison, JM Martìnez Montiel, and Kurt Konolige. Double window optimisation for constant time visual slam. In *2011 international conference on computer vision*, pages 2352–2359. IEEE, 2011.

[70] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[71] Jiexiong Tang, Ludvig Ericson, John Folkesson, and Patric Jensfelt. Gcnv2: Efficient correspondence prediction for real-time slam. *IEEE Robotics and Automation Letters*, 4(4):3505–3512, 2019.

[72] Shaharyar Ahmed Khan Tareen and Zahra Saleem. A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*, pages 1–10. IEEE, 2018.

[73] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 661–669, 2017.

[74] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. Sosnet: Second order similarity regularization for local descriptor learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11016–11025, 2019.

[75] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*, pages 298–372. Springer, 2000.

[76] Yusuke Uchida. Local feature detectors, descriptors, and image representations: A survey. *arXiv preprint arXiv:1607.08368*, 2016.

[77] Nan Yang, Rui Wang, Xiang Gao, and Daniel Cremers. Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect. *IEEE Robotics and Automation Letters*, 3(4):2878–2885, 2018.

	Tracked Frames $\times 10^2$ (#)							ATE (units)							RPE (units/s)							Loop Closures (#)						
	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka
1a	11.7	11.7	11.7	2.7	11.7	2.6	**12.3**	-	0.5	**0.3**	-	0.6	-	-	186.2	0.8	**0.5**	44.3	0.9	185.0	187.0	0	0	**1**	0	**1**	0	0
1b	10.1	10.1	10.1	10.1	10.1	10.4	**10.5**	0.9	0.8	1.0	1.0	1.6	**0.8**	0.8	1.2	**1.1**	1.3	1.2	2.0	1.2	1.2	0	0	0	0	0	0	0
1c	16.4	15.5	**16.7**	7.9	14.0	15.4	15.6	-	6.7	**0.5**	1.9	-	0.5	6.9	36.2	7.2	**0.7**	1.8	19.1	34.9	36.1	0	0	**1**	0	0	0	0
1d	**11.7**	8.3	11.7	6.5	8.7	8.5	11.4	-	2.3	**0.5**	8.7	-	0.5	0.5	46.1	2.0	0.6	7.4	14.0	2.0	**0.6**	0	0	0	0	0	0	0
1e	10.8	10.6	**16.7**	15.0	2.5	15.0	2.6	1.1	**0.5**	1.9	4.6	4.9	0.5	1.0	1.2	**0.6**	2.4	4.3	69.1	70.5	1.2	0	0	0	0	0	0	0
1f	4.7	4.4	4.8	4.6	**4.8**	4.6	4.5	-	-	0.4	-	0.5	-	**0.4**	25.9	21.3	**0.6**	21.6	0.8	20.4	0.7	0	0	0	0	0	0	0
1g	4.7	2.6	5.0	**5.0**	4.1	4.8	4.5	0.2	0.3	0.1	0.3	0.1	0.3	**0.1**	0.3	0.4	0.2	0.3	0.2	**0.2**	0.2	0	0	0	0	0	0	0
1h	8.4	8.5	7.6	8.5	7.9	**8.6**	8.0	1.0	0.7	0.6	0.7	0.3	0.7	**0.3**	1.2	0.9	0.7	0.9	**0.4**	0.9	0.4	1	1	1	1	1	**1**	1
1i	2.6	0.7	**7.0**	2.1	6.9	2.2	6.7	1.2	0.7	1.0	0.5	0.8	**0.5**	0.8	1.9	2.4	1.8	**1.0**	2.0	1.9	2.5	0	0	0	0	0	0	0
1j	4.6	4.4	**15.0**	4.8	0.7	4.5	4.9	-	2.4	**1.7**	1.8	2.2	1.8	2.3	164.9	3.0	2.6	2.4	38.0	**2.4**	37.6	0	0	**1**	0	0	0	0
2a	**7.8**	7.8	7.8	7.8	7.6	7.3	7.5	1.0	1.1	**0.8**	1.1	1.1	1.1	1.1	1.5	1.5	**1.2**	1.5	1.5	1.6	1.5	0	0	0	0	0	0	0
2b	4.6	**6.1**	4.4	0.3	3.8	4.8	3.6	**1.5**	1.7	9.6	4.1	1.6	1.7	-	2.7	2.8	12.4	14.5	**2.3**	12.2	14.5	0	0	0	0	0	0	0
2c	0.3	0.5	0.5	**0.8**	0.4	0.5	0.5	5.9	4.5	3.4	**2.2**	3.0	4.7	4.4	6.8	18.8	4.3	3.6	**3.4**	3.8	3.5	0	0	0	0	0	0	0
2d	36.5	36.3	36.5	36.6	36.6	**37.8**	34.9	0.3	0.2	0.2	**0.2**	0.2	0.2	0.3	1.0	0.9	1.0	1.0	1.1	1.0	**0.9**	0	0	0	0	0	0	0
2e	29.6	29.5	29.5	29.5	28.2	**29.7**	28.5	1.3	0.7	0.8	**0.5**	1.2	1.2	1.1	3.3	**2.8**	3.1	3.0	3.0	3.3	3.4	1	1	1	1	1	**1**	1
2f	25.5	25.6	25.6	15.8	25.6	24.4	**26.0**	1.6	1.8	1.2	-	**1.1**	1.3	1.6	2.8	2.8	2.3	75.7	**2.2**	2.9	2.9	1	1	1	0	1	1	**1**
2g	9.1	9.0	9.1	9.1	9.1	9.2	**9.2**	0.9	1.0	0.9	0.9	**0.7**	1.0	1.0	1.5	1.6	1.4	1.6	**1.4**	1.6	1.5	0	0	0	0	0	0	0
2h	10.3	10.3	10.3	10.3	**10.3**	9.9	10.1	1.2	1.3	1.0	1.1	**0.9**	1.1	1.0	1.9	1.9	1.7	1.6	1.6	**1.5**	2.0	0	0	0	0	0	0	0
3a	**45.4**	42.7	39.9	42.0	41.5	40.6	39.0	11.9	**6.7**	11.0	33.3	8.7	34.7	32.5	15.3	**9.4**	13.4	46.5	11.3	11.6	9.8	4	3	2	**5**	3	5	2
3b	**11.0**	2.5	1.1	2.1	1.1	2.6	2.5	383.0	2.2	**0.5**	0.8	0.5	2.1	0.6	445.5	3.1	**0.6**	1.4	0.7	1.4	0.7	0	0	0	0	0	0	0
3c	46.6	0.1	0.1	0.1	0.0	**48.6**	0.0	22.9	0.0	**0.0**	0.0	100.0	23.0	0.0	24.1	**0.0**	0.0	0.1	100.0	23.8	0.0	2	0	0	0	0	**2**	0
3d	8.0	8.0	8.0	8.0	8.0	**8.2**	7.9	2.5	1.2	1.4	1.2	**1.0**	2.6	1.2	3.2	**1.7**	1.8	1.8	1.7	1.8	1.7	0	0	0	0	0	0	0
3e	2.7	2.7	2.7	2.7	2.7	**2.7**	2.7	0.9	0.3	0.3	0.3	0.3	0.3	**0.2**	1.2	**0.7**	0.7	0.7	0.8	0.7	0.7	0	0	0	0	0	0	0
3f	27.6	27.6	27.6	27.6	27.6	**28.1**	27.1	7.2	6.0	6.9	7.2	**5.5**	5.7	7.5	9.7	8.2	9.4	10.4	**7.0**	8.2	9.5	3	3	3	3	3	**3**	3
3g	11.0	11.0	0.4	11.0	11.0	**11.6**	10.5	15.4	19.2	**0.0**	16.0	18.4	16.7	18.3	19.1	22.2	**0.1**	17.4	21.4	21.9	21.1	1	1	0	2	1	**2**	1
3h	11.0	11.0	8.2	11.0	11.0	**11.5**	10.5	3.0	2.6	16.6	4.0	2.1	**2.1**	3.1	4.3	3.6	21.9	5.1	**2.9**	4.4	20.9	1	1	0	1	1	**1**	1
3i	40.7	35.4	36.2	40.7	40.7	39.2	**42.3**	**42.2**	58.3	68.0	79.8	77.9	58.1	57.0	**51.2**	70.7	83.9	94.4	91.8	94.4	94.2	0	0	0	0	0	0	0
3j	**15.7**	7.7	7.7	7.7	12.6	7.8	7.3	46.2	**6.7**	9.0	8.3	31.0	6.7	6.7	57.0	8.6	11.1	10.2	39.0	**8.3**	8.6	0	0	0	0	0	0	0
4a	36.8	36.8	36.8	36.8	36.8	**38.2**	35.2	7.3	7.8	6.8	7.1	7.3	6.9	**6.5**	625.1	610.1	642.2	641.1	609.9	652.7	**581.7**	0	0	0	0	0	0	0
4b	30.4	30.4	30.0	30.0	30.4	31.1	**31.9**	-	5.9	**5.9**	8.7	6.2	5.9	6.0	663.8	649.9	664.1	642.8	656.7	**634.3**	670.2	0	0	0	0	0	0	0
4c	26.4	0.1	2.6	0.2	**26.5**	2.7	26.3	6.4	0.0	0.5	0.1	5.5	**0.0**	0.1	540.0	0.7	8.5	4.6	566.3	8.6	**0.7**	0	0	0	0	0	0	0
4d	19.9	19.9	19.9	0.2	2.5	**20.4**	19.7	6.4	-	-	**0.4**	1.1	-	-	940.7	896.9	909.7	**8.3**	9.1	876.9	873.8	0	0	**1**	0	0	0	0
4e	21.9	21.8	21.5	0.2	21.7	22.5	**22.8**	-	6.3	6.3	**0.2**	6.7	6.3	6.6	878.4	827.6	893.5	8.2	933.4	**8.2**	863.7	2	2	2	0	2	2	**2**
4f	28.0	28.0	27.8	28.0	28.0	28.4	**29.2**	-	-	-	**-**	-	-	-	368.2	362.8	364.2	371.5	371.9	362.2	**357.4**	0	0	0	0	0	0	0
4g	16.0	13.7	15.8	13.7	16.2	16.4	**16.6**	-	9.9	-	9.9	-	-	**9.7**	**320.0**	333.0	345.8	338.4	330.2	359.2	353.8	1	**2**	**2**	1	1	1	1
4h	19.9	16.9	20.0	14.4	**20.0**	16.4	16.4	-	-	-	**9.5**	9.8	-	-	284.4	280.8	293.9	310.4	293.8	294.0	**270.1**	**1**	**1**	**1**	0	**1**	1	1
4i	20.8	20.8	20.8	20.8	20.8	21.0	**21.5**	9.1	8.8	**8.5**	8.8	-	8.8	8.9	312.3	314.7	**306.7**	309.5	309.1	318.2	316.6	0	0	0	0	**1**	0	0
4j	22.8	0.1	22.1	19.5	22.8	19.1	**23.3**	9.7	0.0	9.7	9.4	9.9	-	**0.0**	323.9	**0.9**	317.9	327.2	322.5	317.7	337.3	1	0	1	1	**2**	1	1
4k	**16.9**	14.3	9.1	10.5	10.9	9.5	10.8	-	-	-	-	-	**9.8**	-	389.3	365.9	326.5	**316.4**	364.2	349.6	366.0	**1**	**1**	0	0	**1**	0	0

TABLE X: **Trajectory Accuracy Comparison**: The table shows the trajectory estimation accuracy between the original ORB-SLAM2 [49] and our AnyFeature-VSLAM, utilizing different descriptors. **Notably, the performance across all descriptors exhibits remarkable similarity**. We conduct this evaluation across 39 sequences drawn from four distinct datasets: ETH (1) [62], RGB-D TUM (2) [70], KITTI (3) [30], and EuRoC (4) [9]. For metrics, we employ the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) [9], expressed in centimeters for ETH, RGB-D TUM, and EuRoC, and meters for KITTI. Values in **bold** indicate the **best performance**. Descriptors: ORB [49] (**Or***), ORB (**Or**), AKAZE (**Ak**), BRISK (**Br**), SURF (**Su**), SIFT (**Si**), KAZE (**Ka**). Sequence names and dataset characteristics can be found in Tables IX and III.

	Keyframes (#)							Points $\times 10^3$ (#)							Observations $\times 10^3$ (#)							Observations (#) / Point (#)						
	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka	Or*	Or	Ak	Br	Su	Si	Ka
1a	**259**	165	173	69	201	62	154	9.9	7.0	9.5	2.0	**11.2**	1.8	8.5	62.9	48.2	58.5	11.0	**68.1**	9.8	52.0	6.4	**6.9**	6.1	5.5	6.1	4.9	5.5
1b	124	153	156	176	**189**	168	138	5.8	6.9	9.1	7.9	**10.7**	7.5	6.2	43.5	45.4	53.8	50.5	**63.9**	49.7	39.8	**7.5**	6.6	5.9	6.4	6.0	7.2	5.0
1c	231	233	201	128	**308**	153	288	8.3	9.8	10.2	5.5	**13.5**	6.6	12.1	67.7	66.2	67.3	34.4	78.1	41.1	**81.8**	8.1	6.8	6.6	6.2	5.8	7.5	**8.4**
1d	**284**	91	136	119	110	260	84	**13.4**	3.8	6.2	5.4	5.6	12.3	4.3	**86.4**	24.6	39.1	35.8	33.6	79.2	25.7	6.5	6.5	6.3	**6.6**	6.0	5.9	4.6
1e	79	124	**286**	240	52	108	86	3.7	5.3	**12.0**	9.7	2.7	4.6	4.0	27.5	35.3	**72.0**	58.7	15.3	30.8	30.0	7.5	6.7	6.0	6.1	5.7	5.8	**8.2**
1f	67	89	62	**100**	69	53	71	3.1	4.6	4.6	**5.1**	4.8	2.5	3.7	20.6	28.0	26.2	**30.6**	29.1	16.2	22.5	**6.5**	6.1	5.7	6.0	6.1	5.1	4.9
1g	66	48	63	**81**	51	74	59	3.9	3.1	5.2	**5.9**	4.2	5.4	3.8	27.1	18.2	32.0	**38.8**	27.4	35.4	22.5	7.0	5.8	6.2	6.6	6.6	6.0	**7.2**
1h	109	123	107	**146**	119	122	113	5.8	6.3	7.2	**8.5**	6.8	7.0	6.4	40.9	41.8	43.8	**55.9**	41.9	43.1	39.9	**7.0**	6.7	6.1	6.6	6.2	6.4	5.9
1i	59	13	87	67	**98**	11	14	1.9	0.2	**5.0**	2.4	4.9	0.2	0.2	12.1	1.1	30.0	15.8	**30.3**	0.9	1.1	6.2	4.6	6.0	**6.7**	6.2	4.1	4.9
1j	48	60	**291**	82	23	63	259	2.5	3.5	**17.1**	4.2	1.6	3.6	15.2	13.3	19.6	**98.4**	23.5	8.3	20.6	87.5	5.4	5.7	5.8	5.7	5.2	**6.0**	5.1
2a	37	34	39	42	**51**	27	41	2.0	1.7	2.4	2.4	**2.7**	1.4	2.1	12.9	10.6	14.1	15.2	**18.6**	8.5	14.8	6.6	6.2	5.9	6.2	**6.9**	5.0	5.5
2b	94	**131**	95	23	65	21	101	4.6	**5.5**	4.3	0.8	3.4	0.7	4.2	33.0	**35.3**	24.9	3.9	19.5	3.5	27.2	**7.2**	6.5	5.8	5.0	5.8	4.4	5.0
2c	21	24	24	**46**	20	20	39	0.9	0.8	1.1	**2.2**	0.9	0.7	1.9	5.2	4.8	5.5	**12.2**	4.5	3.9	10.4	**5.7**	5.7	5.2	5.5	4.9	4.6	4.8
2d	40	38	**49**	36	41	38	34	1.6	1.4	2.3	1.7	**2.7**	1.8	1.6	13.1	10.4	**17.3**	12.3	16.4	13.3	11.5	**8.4**	7.6	7.5	7.1	6.1	5.7	6.6
2e	185	262	237	258	**273**	213	264	7.0	10.1	12.2	11.4	**14.7**	8.2	11.6	54.0	68.8	74.3	76.7	**92.1**	55.9	78.4	**7.7**	6.8	6.1	6.7	6.3	5.5	6.9
2f	209	263	**289**	234	288	233	214	9.8	11.9	16.2	11.4	**17.0**	11.0	9.7	75.4	78.7	97.4	72.7	**106.2**	84.2	64.1	7.7	6.6	6.0	6.4	6.3	**8.5**	5.4
2g	39	37	42	**45**	41	39	34	2.2	2.0	3.2	2.8	**3.5**	3.0	1.8	17.3	13.7	19.3	19.6	**21.1**	18.1	12.5	**7.8**	6.8	6.0	6.9	6.1	5.6	6.2
2h	56	65	81	**87**	79	65	87	3.5	3.8	5.6	5.2	5.6	4.6	**6.2**	22.7	23.6	31.5	32.1	32.2	26.5	**35.5**	**6.5**	6.3	5.6	6.2	5.7	4.7	6.3
3a	2874	2554	2013	3276	1970	**3934**	2512	236.9	162.9	129.7	231.1	151.5	**277.5**	160.2	1573.7	973.9	726.8	1392.3	788.8	**1672.0**	957.7	6.6	6.0	5.6	6.0	5.2	**7.2**	5.9
3b	**664**	185	72	201	59	218	194	**42.1**	11.4	4.6	14.7	5.0	13.5	14.2	**277.5**	59.8	24.1	81.4	22.6	70.4	78.7	**6.6**	5.2	5.2	5.5	4.6	6.2	5.3
3c	3944	9	9	8	0	**4651**	4479	356.4	0.8	1.0	0.9	0.0	**420.3**	404.8	2225.3	3.3	3.3	3.0	0.0	**2624.1**	2527.4	6.2	4.0	3.5	3.3	NaN	**7.4**	7.1
3d	537	439	369	**678**	340	489	344	44.8	27.0	24.3	**50.6**	26.8	30.1	21.2	317.4	164.9	143.5	**332.8**	150.7	183.7	129.3	**7.1**	6.1	5.9	6.6	5.6	6.8	4.8
3e	227	212	172	**264**	148	254	234	18.8	13.2	9.7	**20.6**	11.4	19.9	14.6	**119.8**	72.2	49.8	115.3	51.9	111.0	79.6	**6.4**	5.5	5.2	5.6	4.6	5.4	6.0
3f	1812	1697	1424	**2132**	1291	2069	1292	**146.8**	94.2	83.6	145.7	92.8	114.9	71.7	**999.7**	577.5	466.6	918.3	490.2	704.2	439.7	6.8	6.1	5.6	6.3	5.3	**7.5**	4.7
3g	717	717	18	**933**	577	659	676	59.2	40.0	1.3	**64.1**	43.1	36.8	37.7	**389.2**	235.1	6.0	384.5	220.8	216.1	221.5	**6.6**	5.9	4.6	6.0	5.1	5.4	5.5
3h	737	729	441	**868**	557	577	427	**61.3**	42.9	23.6	59.1	40.9	33.9	22.8	**408.8**	256.2	129.6	364.8	213.6	202.8	125.5	**6.7**	6.0	5.5	6.2	5.2	4.7	5.3
3i	3345	2598	2245	**3683**	2143	2231	3068	**285.0**	160.2	129.5	266.7	158.4	128.7	189.2	**1864.2**	905.8	705.8	1618.0	815.7	701.3	1069.5	6.5	5.7	5.5	6.1	5.1	5.4	**6.7**
3j	**1402**	612	493	709	745	611	614	**122.1**	31.8	23.4	43.8	46.0	29.0	37.9	**772.3**	185.0	124.0	260.2	224.7	153.8	225.2	6.3	5.8	5.3	5.9	4.9	**6.6**	5.1
4a	**336**	287	243	320	236	212	258	12.8	11.4	14.7	**14.9**	13.9	12.9	9.9	**114.7**	81.7	90.0	102.7	87.3	78.5	88.1	**8.9**	7.1	6.1	6.9	6.3	5.3	6.9
4b	331	306	255	386	252	**394**	271	13.2	11.5	13.5	**15.7**	13.7	15.7	14.7	105.5	87.1	80.9	**125.7**	88.7	125.6	95.3	8.0	7.6	6.0	8.0	6.5	**9.5**	7.0
4c	**275**	6	26	14	227	17	5	**10.9**	0.1	0.8	0.4	10.3	0.4	0.1	**93.9**	0.4	4.0	1.8	64.5	2.1	0.3	**8.6**	4.1	5.2	4.8	6.3	5.6	3.2
4d	329	359	299	21	24	24	**370**	14.0	14.3	14.9	0.7	1.1	0.8	**18.4**	**115.5**	102.9	86.9	3.5	5.9	4.0	107.4	**8.3**	7.2	5.8	5.2	5.2	5.9	7.2
4e	363	306	324	13	330	298	**367**	14.8	12.7	17.0	0.3	**17.0**	12.4	15.3	**125.4**	83.9	100.6	1.3	102.7	81.7	100.7	**8.5**	6.6	5.9	3.8	6.0	6.4	7.9
4f	239	263	232	**273**	206	158	199	10.5	10.6	**13.7**	13.0	13.0	10.0	11.8	87.5	80.1	87.7	**95.8**	83.7	64.2	75.3	**8.3**	7.6	6.4	7.4	6.4	4.9	5.5
4g	**268**	257	260	237	243	207	215	10.8	9.0	**12.8**	9.7	11.9	10.2	7.5	**90.2**	64.5	81.5	65.9	70.2	64.9	53.9	**8.4**	7.2	6.3	6.8	5.9	5.0	6.0
4h	348	362	323	332	275	312	**406**	13.4	12.1	13.1	12.1	12.5	12.6	**14.8**	**104.7**	84.5	84.8	78.9	82.4	82.0	96.4	7.8	7.0	6.5	6.5	6.6	6.3	**8.0**
4i	206	241	227	243	230	**297**	248	10.8	11.4	15.4	13.3	14.6	16.3	**16.8**	78.4	77.3	96.1	87.1	88.7	**106.6**	104.8	7.2	6.8	6.2	6.5	6.1	**8.0**	6.8
4j	347	8	**403**	361	332	342	395	13.3	0.3	**17.0**	14.3	16.1	13.1	16.7	**111.1**	1.3	103.9	94.0	98.6	109.6	101.9	**8.4**	4.6	6.1	6.6	6.1	8.2	6.0
4k	**426**	315	156	220	254	263	154	**16.2**	11.6	6.2	8.2	12.1	12.5	6.1	**118.6**	77.3	37.7	50.9	72.2	74.6	37.3	**7.3**	6.7	6.1	6.2	6.0	6.2	6.0

TABLE XI: **Characterization analysis of the SLAM Graph**: The table shows a comparison of key SLAM graph metrics between the original ORB-SLAM2 [49] and our AnyFeature-VSLAM implementation, utilizing various descriptors. Notably, the performance across all descriptors exhibits remarkable similarity. This evaluation encompasses 39 sequences from four distinct datasets: ETH (1) [62], RGB-D TUM (2) [70], KITTI (3) [30], and EuRoC (4) [9]. The metrics include the number of keyframes and 3D points in the final graph, the count of active observations (2D keypoints), and the observations per 3D point. **Bold** values highlight the **highest**, while underlined values denote the lowest. Descriptors are denoted as ORB [49] (**Or***), ORB (**Or**), AKAZE (**Ak**), BRISK (**Br**), SURF (**Su**), SIFT (**Si**), and KAZE (**Ka**). Sequence names and dataset characteristics can be found in Table IX and Table III.