IJICIS, Vol.21, No.2, 124-139
DOI: 10.21608/ijicis.2021.63324.1073

International Journal of Intelligent Computing and Information Sciences

https://ijicis.journals.ekb.eg/

Binary Descriptors for Dense Stereo Matching

Hanaa I.F Ibrahim*
Computer Systems Department,
Faculty of Computer and Information Sciences, Ain Shams University,
Governorate, Egypt
hanaa_ibrahim@cis.asu.edu.eg

H. Khaled
Computer Systems Department,
Faculty of Computer and Information Sciences, Ain Shams University,
Governorate, Egypt
heba.khaled@cis.asu.edu.eg

Noha A. Seada
Computer Systems Department,
Faculty of Computer and Information Sciences, Ain Shams University,
Governorate, Egypt
noha_sabour@cis.asu.edu.eg

H. M. Faheem
Computer Systems Department,
Faculty of Computer and Information Sciences, Ain Shams University,
Governorate, Egypt
hmfaheem@cis.asu.edu.eg

Abstract: *Dense local stereo matching is traditionally based on initial cost evaluation using a simple metric followed by sophisticated support aggregation. There is a high potential of replacing these simple metrics by robust binary descriptors. However, the available studies focus on comparing descriptors for sparse matching rather than the dense case of extracting a descriptor per each pixel. Therefore, this paper studies the design decisions of well-established binary descriptors such as BRIEF (Binary Robust Independent Elementary Features), ORB (Oriented FAST and rotated BRIEF), BRISK (Binary Robust Invariant Scalable Keypoints) and FREAK (Fast Retina Keypoint) to decide their suitablility for the dense matching case. The results shows that support agregation is required for use with binary descriptors to handle edges in local dense matching. Also, BRIEF produced the smoothnest disparity map if geometric transformations is not present. Whereas, FREAK and BRISK achieved the least overall error percentage across all regions. The lastest Middlebury Stereo benchmark is utilized in the experiments.*

Keywords: *Local Stereo Matching, Dense Matching, Binary Descriptor, Descriptor Matching, Computer Vision*

1. Introduction

* Corresponding author: Hanaa I.F Ibrahim
Computer Systems Department, Faculty of Computer and Information Sciences, Ain Shams University, Governorate, Egypt
E-mail address: hanaa_ibrahim@cis.asu.edu.eg

Augmented Reality (AR) relies on understanding RGB images of the scene to guide the placement of the virtual content such as for medical applications [1]. Stereo matching, which resembles the human visual system in recovering depth from RGB images, is equally essential for AR [2], [3].

Stereo matching searches for the projections of the 3D point in two images taken of the same scene from different viewpoints. If the 3D point lies within the field of view of both cameras, disparity is calculated as the difference in the image-plane locations of these two projected pixels. The depth, which is the distance between the 3D point and the camera, can be recovered using the disparity and the camera parameters.

Dense stereo matching, which is required for AR, searches for a correspondence of every pixel in the image resulting in a dense disparity map. Sparse matching, which is beneficial for other computer vision applications such as image stitching, is performed only for a set of pixels that are surrounded by salient features. In either case, the cost of matching two pixels can be computed using functions that assess their similarity.

Dense stereo matching has been largely based on simple intensity-based metrics such as absolute intensity difference that assume brightness constancy. The intensities of very few pixels are involved in evaluating an initial matching cost [4]. Then, support is aggregated given the initial costs of neighboring pixels believed to belong to the same depth [5]. Intensity-based metrics are fast to be computed. However, they are recently getting replaced by invariant descriptors.

Per-pixel SIFT (Scale Invariant Feature Transform [6]) descriptors are compared to evaluate pixels' similarity by He et. al. [7]. Dense matching can be performed by extracting a descriptor for each pixel [8] and comparing these robust descriptors to calculate the matching cost as illustrated by descriptor-based stereo matching [5], [9], SIFT-flow [10], binary stereo matching [11], DAISY [12], [13].

Distribution-based descriptors , such as SIFT [6] and SURF (Speeded Up Robust Features [14]), have high computational complexity [15]. They encode gradient distributions extracted from the patch surrounding the pixel of interest as a real-valued vector. The binary descriptors, such as BRIEF [16] and FREAK [17] encode the patch's structure as a binary string.

Binary descriptors are less invariant to geometric transformations. However, they can be more invariant to photometric transformations than SIFT and SURF [15]. Also, FREAK can be more robust against some geometric transformations than SIFT and SURF [18]. The binary descriptors are faster to be computed and compared making them more suitable for dense matching.

Similarly, traditional dense matching weighs the contribution of each pixel to the aggregated cost using either real-valued or binary weights. Real-valued weighting requires higher processing and storage requirements but is believed to produce more accurate results than binary aggregation. Accordingly, Heinley et. al. [19] classify traditional methods into a higher memory demanding class than some binary descriptors and into the same class as distribution-based descriptors.

Deep learning (DL) results in a recent performance leap. However, engineered descriptors are still necessary [20]. Training a supervised DL model requires intensive resources which is not the case with engineered descriptors [21]. The deployment environment can be different from the training dataset which requires re-training or online unsupervised adaptation [22]. RGB images may not be available to be fed to the deep neural network [23]. Furthermore, engineered descriptors can be used with machine and deep learning [24], [25].

The research conducted to test descriptors in dense matching is very limited, despite their increased utilization and potential to replace traditional matching metrics. Hence, this paper experimentally compares the dense disparity maps resulting from employing different binary descriptors in local dense matching context. We choose the most adopted and cited descriptors whose OpenCV implementations are widely used. Also, this work was extended by further studying binary descriptors in dense matching [26], [27].

The paper is organized as follows: section 2 reviews the main directions of descriptor evaluation studies. Section 3 compares the design decisions of well-established binary descriptors and summarizes the findings of sparse evaluations. The results and the conclusion are discussed in sections 4 and 5, respectively.

2. Related Studies

Many studies are aimed to test detector-descriptor combinations for sparse matching. The sparse matching evaluations are performed in two directions. The first type of studies compares descriptors in terms of invariance and accuracy over various general-purpose datasets. The second type of studies focuses on a certain application to determine which descriptor is more suitable to the application's requirements. Other studies compared descriptors for semi-dense matching in a keypoint grid.

2.1. General-purpose sparse matching

Alshazly et. al. [15] performed a very valuable evaluation of the invariance of the state-of-the-art binary descriptors. They studied detector-descriptor combinations on the CPU and compared binary descriptors to distribution-based descriptors. Işık et. al. [28] compared seven different detector-descriptor combinations. Heinly et al. [19] studied binary descriptors along with detector couplings and extended the evaluation to include all the categories of descriptor-based matching and patch matching.

2.2. Sparse matching for a specific application

Other studies focused on sparse matching for specific applications. Cīrulis et. al. studied the descriptors and detectors for pose detection and tracking of the marker surface on which the augmented content will be overlayed in AR [29]. Malekabadi et. al. [30] tested recent combinations for the specific task of matching sparse keypoints detected on a tree and reported the best combinations for this task. Figat et. al. [31] evaluated various combinations for the specific task of indoor recognition of objects. Peng [32] compared descriptors for embedded SLAM and concluded that SIFT is more robust but computational complex which makes binary descriptors more suitable to the task. Bansal et. al. [33] compared sparse SIFT, SURF and ORB for object recognition.

2.3. Matching keypoints sampled from a grid

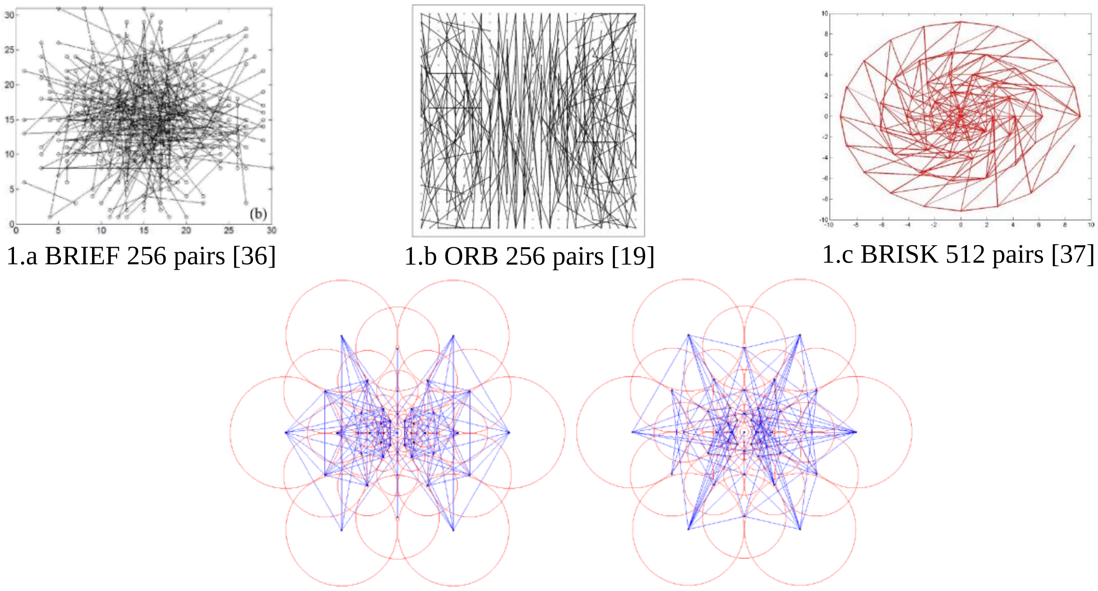
Chatoux et. al. [18] proposed a general framework to test the invariance of descriptors on a grid rather than using different detectors. They extracted and matched descriptors on three different dense keypoint grids such that the grid’s density is inversely proportional to the keypoint size. The densest grid out of the three grids had 10-pixel spacing between the keypoints to be matched. Kayım [34] extracted descriptors from a pre-computed disparity map of the object according to a 4-pixel grid. The extracted descriptors are encoded to describe the disparity image itself for object class recognition. Kayım evaluated SIFT, PCA-SIFT, BRISK, and FREAK for this purpose.

The studies described in this section do not fully answer the research questions about the applicability of binary descriptors in the extreme case of dense matching by extracting a descriptor per-pixel.

3. Comparing Binary Descriptors Design

In this section, we analyze BRIEF, ORB, BRISK, FREAK and LATCH binary descriptors to assess their suitability for generating accurate dense disparity map. The motivation is that these light-weight descriptors are not thoroughly studied and compared in the case of dense matching, despite their increasing utilization, taking BRIEF as example [11], [35].

The descriptor of a pixel is extracted from the patch centered around it by preselecting n pixel pairs surrounding the center of the patch. This arrangement is fixed for use to compute the descriptor of any pixel in the left or right images. Each pair of pixels is represented by a line in the following figure:



1.a BRIEF 256 pairs [36]

1.b ORB 256 pairs [19]

1.c BRISK 512 pairs [37]

1.d FREAK (two clusters each composed of 128 pairs) [17]

Figure 1: Pixel Pairs

The result of the binary test τ utilizing the pixel pair $< p_i, q_i >$ is calculated using the following equation [38] where i is the pair's index that ranges from 1 to n and I is the intensity function:

$$\tau(i) = \begin{cases} 1, & I(p_i) < I(q_i) \\ 0, & otherwise \end{cases} \qquad (1)$$

LATCH differs in pre-selecting n patch triplets rather than n pixel pairs to expand the spatial distribution of the samples. LATCH accordingly edits τ to use Frobenious norm on a patch-triplet $< p_{i,a}, p_{i,1}, p_{i,2} >$ such that i is the triplet's index that ranges from 1 to n. $p_{i,a}$, $p_{i,1}$ and $p_{i,2}$ are the coordinates of anchor patch, and the first and second patches, respectively [39].

$$\tau(i) = \begin{cases} 1, & \left\| p_{i,a} - p_{i,1} \right\|_F^2 > \left\| p_{i,a} - p_{i,2} \right\|_F^2 \\ 0, & otherwise \end{cases} \qquad (2)$$

For any binary descriptor, the descriptor of pixel x is the collective result of the n tests as shown by the following equation [38]:

$$desc(x) = \sum_{i=1}^{n} 2^{i-1}\,\tau(i) \tag{3}$$

A pixel x in the left image has potential correspondences x_d in the rectified right image that are left-shifted. Therefore, the location of correspondences in the right image is $x_d = x - (d, 0)$, where d is potential disparity value ranging from 0 to d_{max}. $cost(x, d)$ is the cost of matching pixel x to pixel x_d. It equals the Hamming distance between their respective descriptors as shown by the following equation [38]:

$$cost(x, d) = Hamming\_distance(x, d) = |desc(x)\ XOR\ desc(x_d)| \tag{4}$$

Binary descriptors share the descriptor computation steps described above. However, they differ in certain aspects that we summarized in the following table:

Table 1 Descriptores Comparison

<table>
<tr><th></th><th>BRIEF</th><th>ORB</th><th>BRISK</th><th>FREAK</th><th>LATCH</th></tr>
<tr><td>Sampling</td><td colspan="4">pairs</td><td>triplets</td></tr>
<tr><td>Sampling pattern</td><td colspan="2">no</td><td colspan="2">yes</td><td>no</td></tr>
<tr><td>Sampling algorithm</td><td>random sampling using a Gaussian distribution</td><td>unsupervised learning</td><td>short pairs from the pattern</td><td>learn from the pattern using ORB’s learning method</td><td>supervised learning: triplets that best describes the patch given labeled data</td></tr>
<tr><td>Pairs significance</td><td colspan="3">equal</td><td>not equal</td><td>equal</td></tr>
<tr><td>Descriptor size (bits)</td><td>256</td><td>256</td><td>512</td><td>512</td><td>256 (can be 512 bits [21])</td></tr>
<tr><td>Noise resistance</td><td colspan="4">pair-based Gaussian kernels</td><td>triplet patch-based spatial distribution</td></tr>
<tr><td>Overlapping Gaussian kernels</td><td>random</td><td colspan="2">not allowed</td><td>allowed</td><td rowspan="2">doesn’t use Gaussian kernels</td></tr>
<tr><td>Gaussian kernel size</td><td colspan="2">fixed</td><td colspan="2">adaptive</td></tr>
<tr><td>Orientation</td><td rowspan="2">none</td><td rowspan="2">expected from a detector</td><td colspan="2">calculated by the descriptor</td><td>expected from a detector</td></tr>
<tr><td>Scale</td><td colspan="2">Expected from multi-AGAST detector</td><td>none</td></tr>
</table>

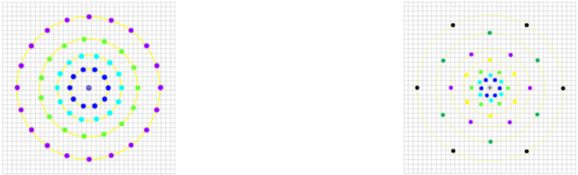
The following subsections briefly describe the similarities and differences summarized in Table 1.

3.1. Sampling

The first difference is how each descriptor selects the samples (i.e., pairs or triplets) that best describe the patch. The result of a binary test of a certain pixel-pair i, $\tau(p_i, q_i)$, is biased if tends to be the same in different patches. τ can be modeled as a Bernoulli random variable whose highest variance is reached with equal probability of getting one or zero (i.e., low-bias). The randomness of BRIEF results in this desired high variance according to PCA analysis [40].

BRIEF designers compared five methods including random points using a Gaussian distribution and sampling from a polar grid pattern. The recognition rate resulting of polar grid samples was slightly higher in the dataset with the greatest viewpoint change. In the remaining datasets, random points surpassed all the five methods and hence was adopted.

A pattern describes the relative positions between the coordinates that can be selected, whereas descriptors that do not adopt a pattern such as ORB and LATCH can select any point in the patch. The center of the pattern is the location of the pixel to be described and the pattern gets scaled according to the scale of patch. Like BRIEF's polar grid pattern, FREAK (that mimics sensors positioning on the retina) and BRISK both adopt a pattern as shown in Figure 2 where allowed sample points (represented by dots) on both patterns form concentric circles (represented by yellow circles).



2.a BRISK [41] 2.b FREAK [17]

Figure 2: Sampleing Pattern

Also, uncorrelated pairs increase descriptiveness so that each test or pair introduces new information about the patch's content. ORB and FREAK employ unsupervised learning for selecting uncorrelated pairs with the highest variance. ORB allows selection of any pair while FREAK is limited by the pattern. BRISK chooses pairs from the pattern that satisfy a certain property (i.e., short pairs). The performance of these methods is dependent on the training dataset [31].

The choice of size (i.e., number of pairs or triplets), n, is arbitrary and is proportional to the accuracy requirements. Table 1 shows the size recommended by the designers. The discussed binary descriptors consider all the pairs to be of equal significance except for FREAK that mimics the human visual system by analyzing the peripheral pairs before the pairs near the center. The 512 pairs used by FREAK are composed of two clusters each with 128 pairs (i.e., coarse-to-fine structure).

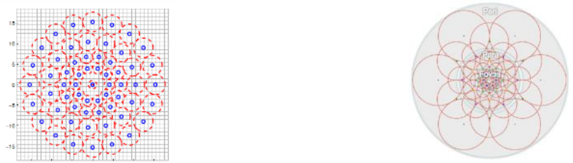
3.2. Gaussian Kernels

Except for LATCH, all the discussed binary descriptors employ a Gaussian kernel around each sampled pixel in the pair for two reasons. First, providing robustness to noise that may affect either of the pixels in the pair. Second, aggregating support by weighting the contribution of the few neighboring pixels.

Subtracting two different Gaussian Kernels (i.e., difference of Gaussian) yields an approximation of Laplacian of Gaussian which equals the second derivative. Instead, a binary descriptor takes only the sign of the second derivative resulting in robustness to illumination changes.

The choice of kernel size depends on the descriptor which is fixed for BRIEF and ORB. BRISK and FREAK varies the size with respect to the distance between the sample point and the center as shown in Figure 3, where the size of the kernel applied to a sample point is represented by the radius of the red circle centered around it.

This way the peripheral samples are characterized by lower resolution and the foveal samples are characterized by higher resolution with high acuity. Also, the kernels of FREAK are designed to overlap to mimic the overlapping receptive fields of the retinal ganglion cells.



3.a BRISK [41]

3.b FREAK [17]

Figure 3: Gaussian Kernels

On the other hand, LATCH's methodology is based on combining more pixels from different spatial locations rather than using Gaussian kernels because blurring compromises high frequency information. This triplet-sampling methodology makes LATCH more computationally complex than the other binary descriptors [18].

3.3. Invariance

Geometric invariance is dependent on the quality of the detected orientation or scale. To identify key-points' scale, FAST (Features from Accelerated Segment Test [42]) was supplemented by pyramid scheme [43]. ORB then reinforced FAST with non-maximal suppression within each pyramid level [40] which didn't prevent duplicates resulting in inferiority to scale changes [15], [19]. Therefore, BRISK applied the suppression between the levels resulting in a high degree of scale invariance [19]. Subsequently, BRISK is more time consuming than ORB [31]. FREAK and BRISK become more invariant to scale changes by increasing the key point size [18].

At the other end of the spectrum, BRIEF is the least costly and the least computationally complex [19] amongst the descriptors mentioned in this discussion with no dependence on a specific detector as shown by Table 1. BRIEF was not designed with the aim of outperforming the invariance of distribution-based descriptors but to speed up matching. Also, rotation invariance is inherently present in BRIEF if the camera orientation is known from IMU sensors attached to it.

Geometric invariance is generally required for sparse matching following a detection step. Table 1 shows whether the description algorithm expects pre-calculated orientation or scale information from the detector. Also, descriptors' invariance is generally dependent on the application. Alshazly et. al. [15] used Oxford dataset [44] to evaluate descriptors invariance. They found that BRISK is the most pure-scale and pure-rotation invariant binary descriptor and ORB is indeed inferior to pure-scale changes. In the same study to evaluate combined rotation-scale invariance, ORB performed better than BRISK in the structured scene (i.e., Boat dataset) and BRISK performed better in the textured scene (i.e., Bark dataset) shown in Figure 4.

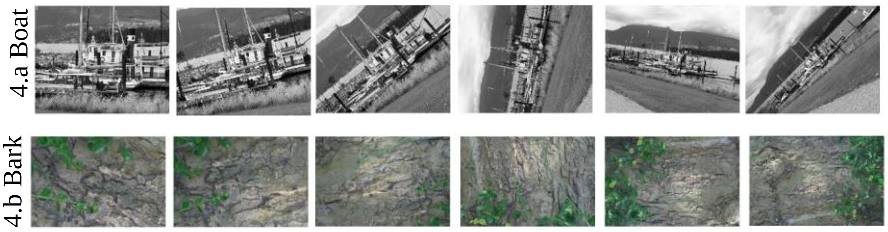


Figure 4: Combined Rotation-Scale Dataset

It is worthy to note that invariance usually comes with decreased matching quality by decreasing the descriptor's discriminative power [14], [45], [46]. Although sophisticated geometric invariant descriptors can better handle matching against rotated/scaled images, BRIEF performs is more invariant to radiometric changes in the absence of geometric transformations [15].

4. Results

This study is concerned with the applicability of sparse binary descriptors in dense matching. Middlebury benchmark is used under WSL (Windows Subsystem for Linux). Middlebury dataset [47] provides near dense ground truth, and it is the standard in dense matching literature. Our platform runs Visual Studio 2017 on an Intel i7-9750H processor with 16 GB of RAM.

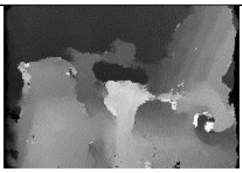
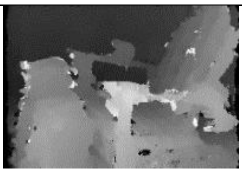
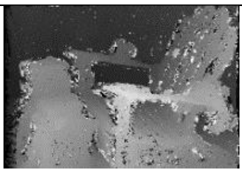
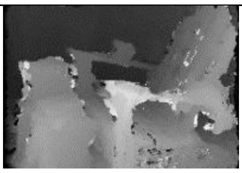
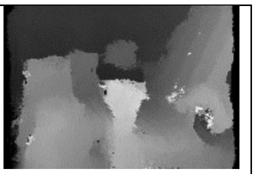
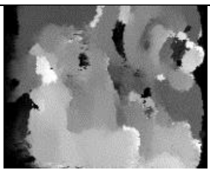
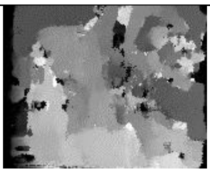
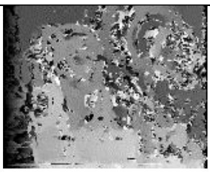
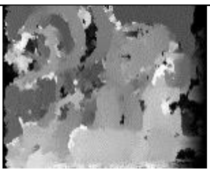
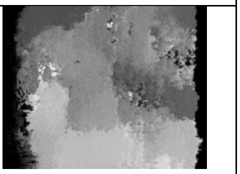
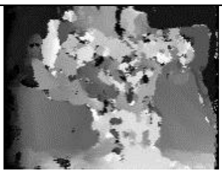
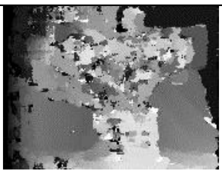
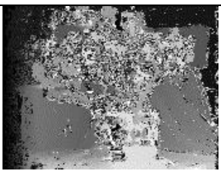
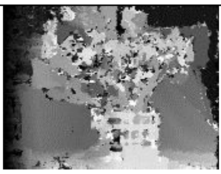
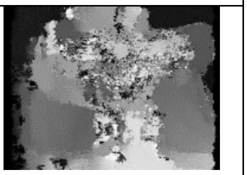
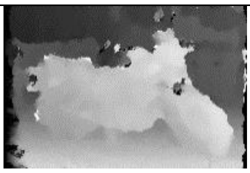
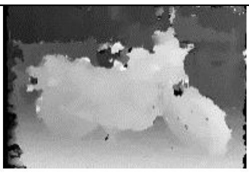
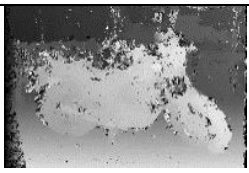
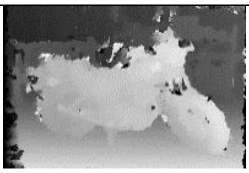
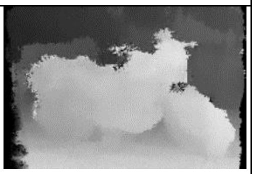
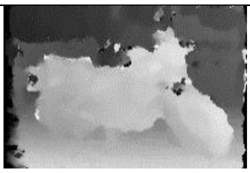
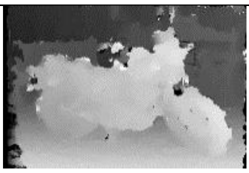
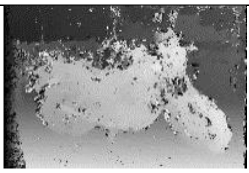
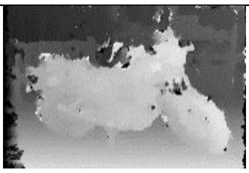
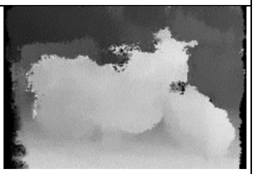
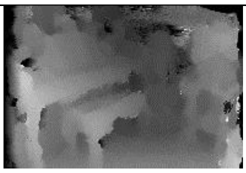
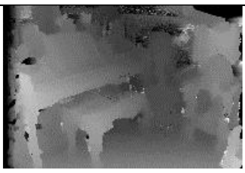
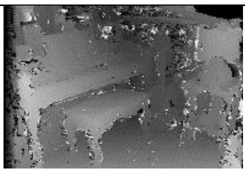
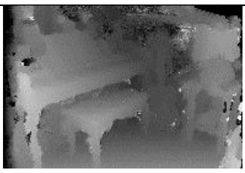
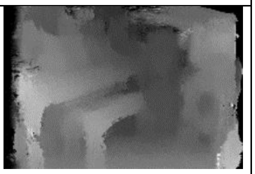
OpenCV 4.1.1 is used to compute a descriptor for every pixel in the image by passing a pre-filled keypoints array to the compute method. Instead of using a detector, the pre-filled array carries all the pixels in the image whose scale is set to one. A single octave is used for all descriptors except for FREAK for which octaves parameter is set zero because the disparity map and the error ratio are much better than setting it to one. Also, the normalization option is disabled for all descriptors. The unrequired normalization reduces FREAk’s accuracy. Apart from this, the default parameters and the sequential implementations of the descriptors are not altered. Padding is applied to the input images and different border and patch sizes are handled.

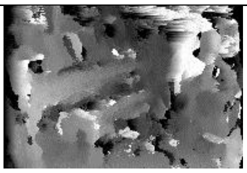
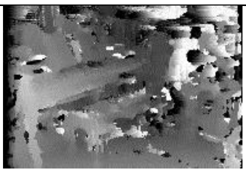
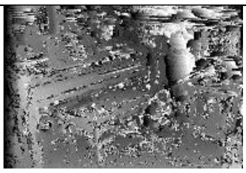
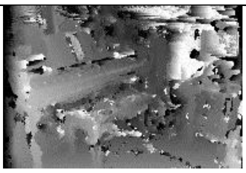
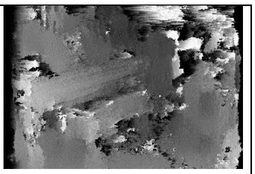
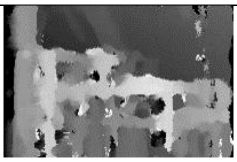
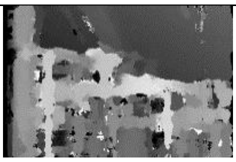
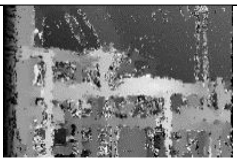
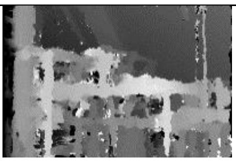
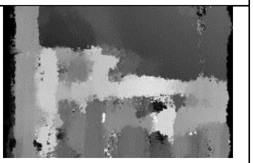
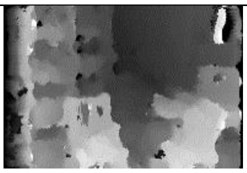
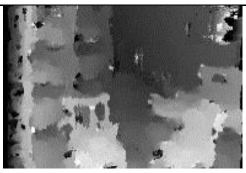
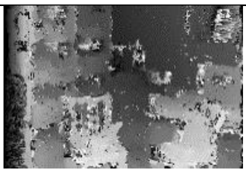
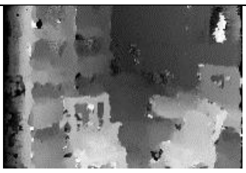
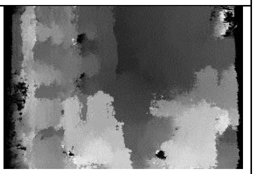
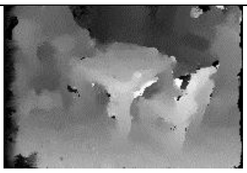
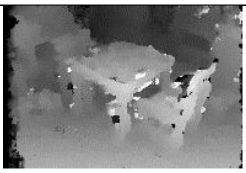
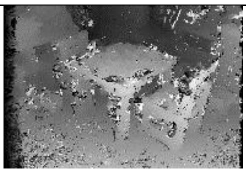
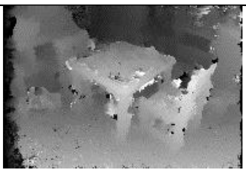
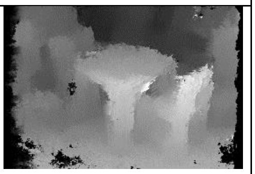
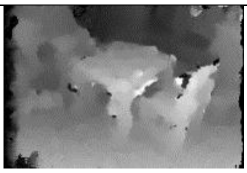
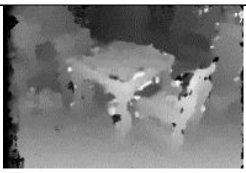
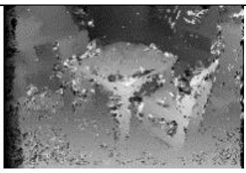
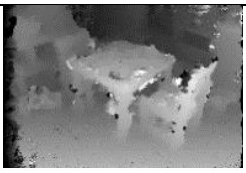
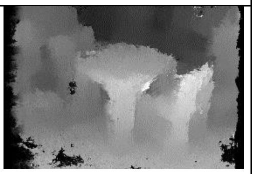
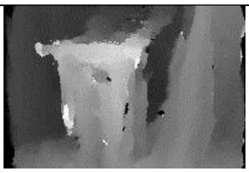
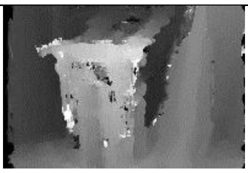
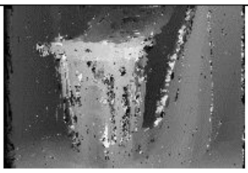
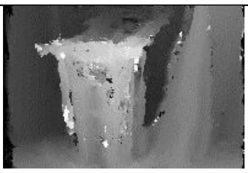
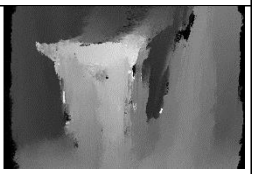
The latest version of Middlebury dataset is used. The suffixes E, L, P appended to the dataset name indicate exposure changes, illumination changes and perfect rectification, respectively. The disparity maps resulting from running each descriptor without normalization according to the above-mentioned description on Middlebury trainingQ dataset are shown in Table 2. The first column indicates the dataset name and each of the following columns represents a certain descriptor. Also, Table 2 reports the time in seconds consumed to extract all the descriptors of the two images and the percentage of pixels whose disparity error exceeds 1 pixel.

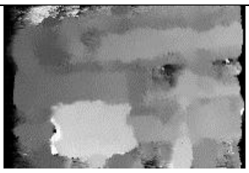
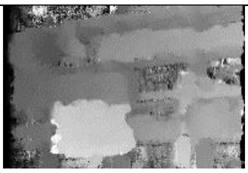
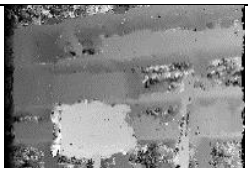
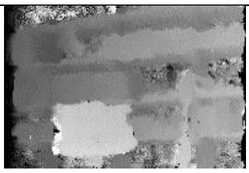
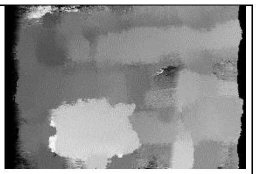
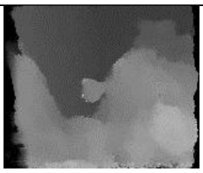
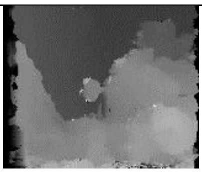
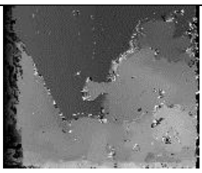
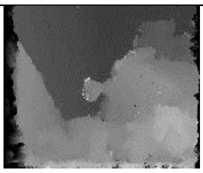
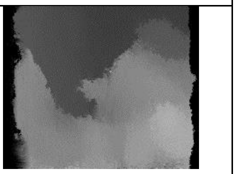
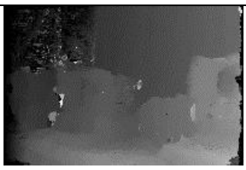
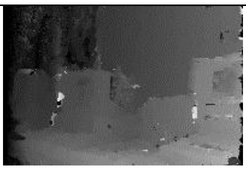
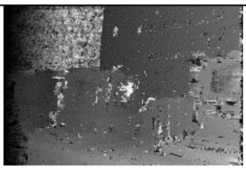
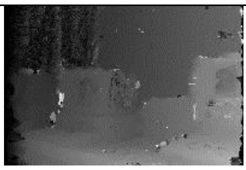
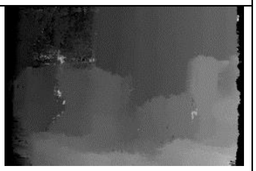
The first value of time and accuracy measurements are recorded for the displayed disparity map which is associated with the best performance. We performed other experiment with different invariance and octaves parameters and reported the measurements without displaying the map for FREAK and LATCH. The first value is the measurement without invariance and the second value is with invariance enabled. The third value of FREAK is obtained by setting number of octaves to 1 which increases the used patch size.

Table 2 Resulting Dispairty maps

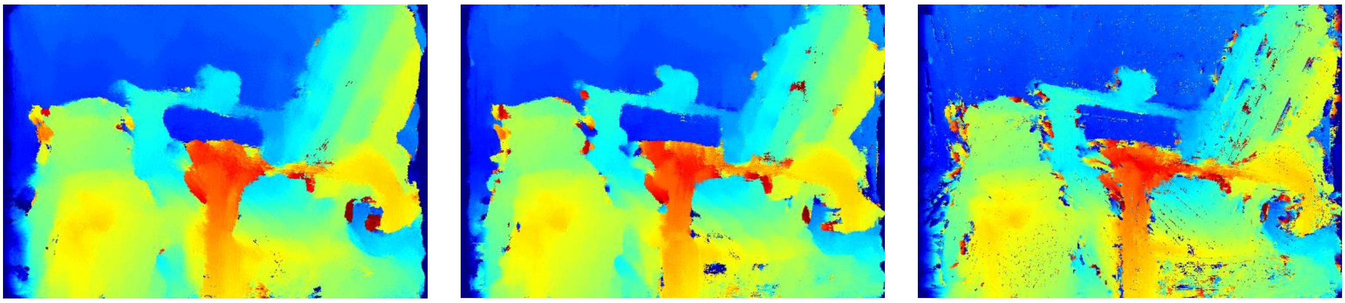
	BRIEF (256 bits)	ORB (256 bits)	BRISK (512 bits)	FREAK (512 bits)	LATCH (256 bits)

Adirondack	 time: 1.38 bad (1.0): 35.97	 time: 1.01 bad (1.0): 33.97	 time: 5.63 bad (1.0): 27.35	 time: 2.03/3.73/2.19 bad (1.0): 29.29/36.64/ 35.97	 time: 25.64/26.36 bad (1.0): 42.52/41.53
ArtL	 time: 0.39 bad (1.0): 64.54	 time: 0.25 bad (1.0): 49.78	 time: 1.63 bad (1.0): 51.91	 time: 0.58/1.06/0.61 bad (1.0): 43.77/56.69/57.62	 time: 6.9/7.14 bad (1.0): 71.08/70.04
Jadeplant	 time: 1.25 bad (1.0): 61.9	 time: 0.77 bad (1.0): 56.37	 time: 5.22 bad (1.0): 50.58	 time: 1.91/3.58/1.97 bad (1.0): 49.33/58.40/62.41	 time: 23.66/24.33 bad (1.0): 72.37/71.47
Motorcycle	 time: 1.39 bad (1.0): 38.58	 time: 0.84 bad (1.0): 33.16	 time: 5.91 bad (1.0): 25.31	 time: 2.06/3.92/2.22 bad (1.0): 23.54/31.13/33.36	 time: 26.56/27.05 bad (1.0): 47.86/46.69
MotorcycleE	 time: 1.39 bad (1.0): 39.72	 time: 0.84 bad (1.0): 34.44	 time: 5.91 bad (1.0): 25.97	 time: 2.11/3.92/2.28 bad (1.0): 23.87/33.73/34.65	 time: 26.48/27.09 bad (1.0): 48.97/47.70
Piano	 time: 1.30 bad (1.0): 53.49	 time: 0.80 bad (1.0): 46.63	 time: 5.47 bad (1.0): 34.06	 time: 2.06/3.61/2.08 bad (1.0): 36.02/42.59/47.42	 time: 24.55/25.02 bad (1.0): 63.43/62.19

PianoL	 time: 1.30 bad (1.0): 79.2	 time: 0.80 bad (1.0): 68.83	 time: 5.47 bad (1.0): 68.38	 time: 1.95/4.05/2.08 bad (1.0): 63.03/72.93/72.58	 time: 25.13/24.98 bad (1.0): 80.58/80.10
Pipes	 time: 1.34 bad (1.0): 42.26	 time: 0.86 bad (1.0): 36.13	 time: 5.64 bad (1.0): 33.2	 time: 2.05/3.77/2.14 bad (1.0): 29.44/37.74/38.47	 time: 26.55/26.19 bad (1.0): 51.31/50.25
Playroom	 time: 1.28 bad (1.0): 48.05	 time: 0.80 bad (1.0): 46.63	 time: 5.36 bad (1.0): 42.1	 time: 1.91/3.52/2.30 bad (1.0): 38.67/45.84/46.01	 time: 24.52 /24.45 bad (1.0): 57.12/56.32
Playtable	 time: 1.22 bad (1.0): 52.38	 time: 0.75 bad (1.0): 45.34	 time: 5.05 bad (1.0): 38.97	 time: 1.84/3.36/2.03 bad (1.0): 34.97/47.72/49.92	 time: 23.31/23.17 bad (1.0): 66.16/65.37
PlaytableP	 time: 1.22 bad (1.0): 51.09	 time: 0.75 bad (1.0): 43.82	 time: 5.09 bad (1.0): 36.58	 time: 1.86/3.36/1.95 bad (1.0): 31.5/45.04/47.55	 time: 23.06/23.17 bad (1.0): 65.71/64.98
Recycle	 time: 1.36 bad (1.0): 39.03	 time: 0.83 bad (1.0): 34.93	 time: 5.66 bad (1.0): 28.83	 time: 2.03/3.73/2.27 bad (1.0): 30.87/38.16/38.68	 time: 25.34 /26.14 bad (1.0): 47.05/45.81

Shelves	 time: 1.42 bad (1.0): 52.17	 time: 0.86 bad (1.0): 53.09	 time: 5.80 bad (1.0): 45.93	 time: 2.14/3.81/2.28 bad (1.0): 49.27/53.15/54.02	 time: 26.77/27.41 bad (1.0): 55.95/55.35
Teddy	 time: 0.66 bad (1.0): 30.11	 time: 0.41 bad (1.0): 26.55	 time: 2.75 bad (1.0): 21.43	 time: 1.00/1.8/1.03 bad (1.0): 21.15/27.89/29.59	 time: 12.16/12.56 bad (1.0): 37.20/36.25
Vintage	 time: 1.31 bad (1.0): 55.12	 time: 0.80 bad (1.0): 52.8	 time: 5.53 bad (1.0): 47.56	 time: 2.00/3.69/2.11 bad (1.0): 46.5/52.67/52.17	 time:25.05 /25.63 bad (1.0): 65.68/64.54

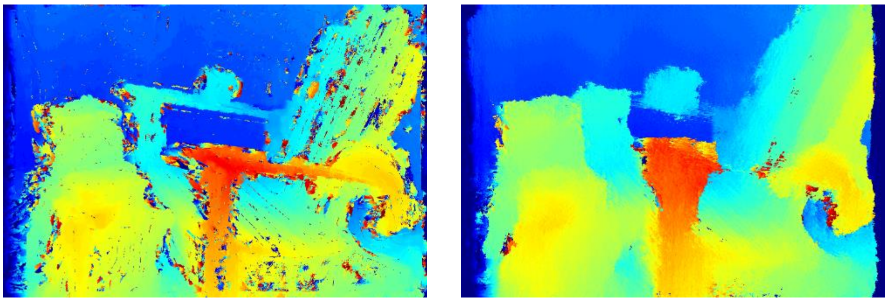
The Middlebury dataset provides images of challenging scenes. However, the Middlebury benchmark lacks metrics to quantitatively assess the important aspects of a disparity map such as smoothness of planar surfaces [48]. Instead, the Middlebury benchmark provides a number, bad 1.0, that counts mismatches over all the regions of the map. This value doesn't indicate the smoothness of the surfaces nor the preciseness near edges which are required for many applications such as AR. We observed that the error within a planar surface is minimal with the use of BRIEF. Figure 5 and Figure 6 show the color-coded disparity maps of Adirondack and Teddy datasets, respectively.



5.a BRIEF

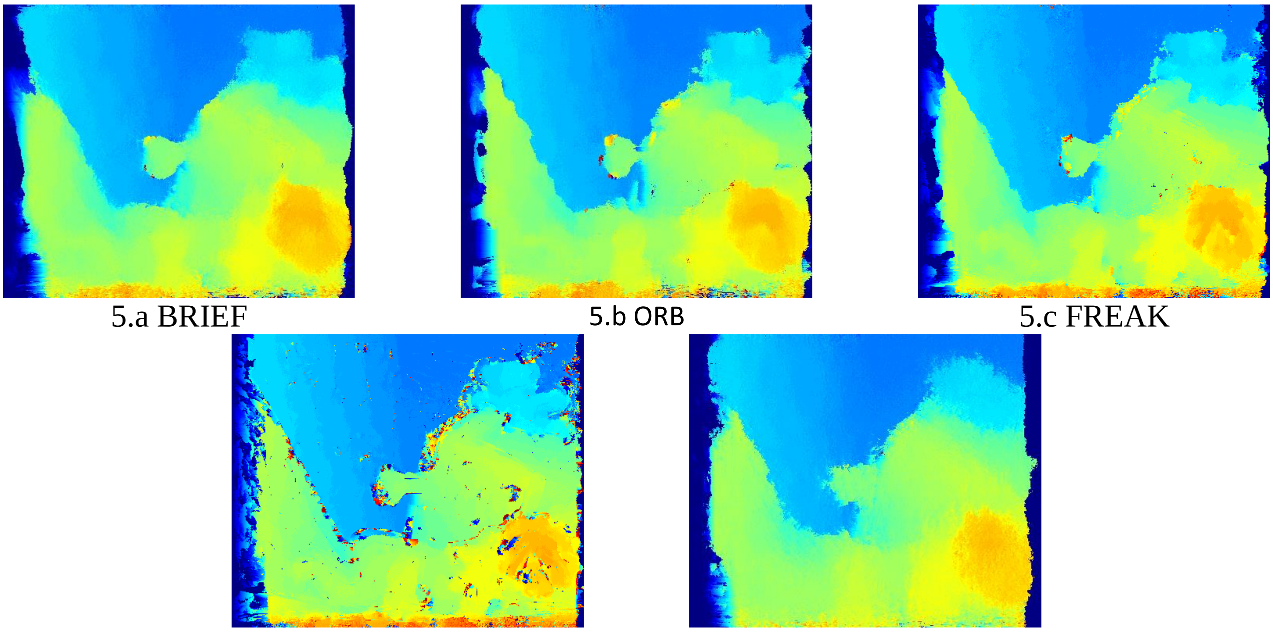
5.b ORB

5.c FREAK



5.d BRISK 5.e LATCH

Figure 5: Color-Coded Disparity Maps of Adirondack Dataset



5.a BRIEF 5.b ORB 5.c FREAK

5.d BRISK 5.e LATCH

Figure 6: Color-Coded Disparity Maps of Teddy Dataset

5. Conclusion

The results demonstrate inaccuracy near the edges in the disparity maps of all the descriptors. This problem is caused by utilizing pairs belonging to a different disparity level than the pixel to be matched. Therefore, aggregation is required for all binary descriptors and a hybrid metric can also be utilized to account for the information loss that may result from binary aggregation.

BRIEF produced the smoothest disparity map, and it suffered the most from edge fattening. The disparity map resulting from BRISK is noisy. However, BRISK excelled in preserving edges which indicates that the descriptor used a smaller patch. Enabling unrequired scale and orientation normalization options on FREAK resulted in increasing the error.

The overall error percentage does not reflect important characteristics such as smoothness of planar surfaces in which BRIEF excelled. This study would be greatly enhanced if the required masks were available to employ quantitative analysis on challenging regions such as edges and planar surfaces [48]. FREAK and BRISK achieved the least over all error. The ascending ordering of OpenCV 4.4.1 implementations according to the running time is BRISK, FREAK, BRIEF then ORB. However, parallel implementations can further speed up the descriptor [26]. On the other hand, Heinly et. al. [19] reported that BRIEF is faster than ORB and BRISK. We conclude that different implementations result in different the running time. We plan to extend this study by including other binary descriptors.

References

[1] A. Yassin, T. Elarif, and M. Hefny, ‘Augmented Reality System in Total Hip Arthroplasty Using Transverse Acetabular Ligament’, *IJICIS*, vol. 20, no. 2, pp. 79–88, Dec. 2020, doi: 10.21608/ijicis.2020.43005.1029.

[2] R. Du *et al.*, ‘DepthLab: Real-time 3D Interaction with Depth Maps for Mobile Augmented Reality’, in *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, Virtual Event USA, Oct. 2020, pp. 829–843. doi: 10.1145/3379337.3415881.

[3] Mikhail Sizintsev, Sujit Kuthirummal, Supun Samarasekera, Rakesh Kumar, Harpreet S. Sawhney, and Ali Chaudhry, ‘GPU accelerated realtime stereo for augmented reality’, 2010. Accessed: Sep. 07, 2016. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.681.4744&rep=rep1&type=pdf

[4] K. Valentín, R. Huber-Mörk, and S. Štolc, ‘Binary descriptor-based dense line-scan stereo matching’, *Journal of Electronic Imaging*, vol. 26, no. 1, p. 013004, Jan. 2017, doi: 10.1117/1.JEI.26.1.013004.

[5] Q. M. ul Haq, C. H. Lin, S.-J. Ruan, and D. Gregor, ‘An edge-aware based adaptive multi-feature set extraction for stereo matching of binocular images’, *J Ambient Intell Human Comput*, Feb. 2021, doi: 10.1007/s12652-021-02958-8.

[6] D. G. Lowe, ‘Distinctive Image Features from Scale-Invariant Keypoints’, *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[7] K. He, X. Wang, and Y. Ge, ‘Adaptive support-weight stereo matching algorithm based on SIFT descriptors’, *Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)*, vol. 49, no. 9, pp. 978–983, 2016, doi: 10.11784/tdxbz201505043.

[8] Z. Liu, E. Zavesky, D. Gibbon, and B. Shahraray, ‘Chapter 13 - Joint Audio-Visual Processing for Video Copy Detection’, in *Academic Press Library in signal Processing*, vol. 5, S. Theodoridis and R. Chellappa, Eds. Elsevier, 2014, pp. 417–455. doi: 10.1016/B978-0-12-420149-1.00013-2.

[9] Yong-Ho Kim, Jamin Koo, and Sangkeun Lee, ‘Adaptive descriptor-based robust stereo matching under radiometric changes’, *Pattern Recognition Letters*, vol. 78, pp. 41–47, Jul. 2016, doi: 10.1016/j.patrec.2016.04.015.

[10] Ce Liu, J. Yuen, and A. Torralba, ‘SIFT Flow: Dense Correspondence across Scenes and Its Applications’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 978–994, May 2011, doi: 10.1109/TPAMI.2010.147.

[11] Kang Zhang, Jiyang Li, Yijing Li, Weidong Hu, Lifeng Sun, and Shiqiang Yang, ‘Binary stereo matching’, in *Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012, pp. 356–359. Accessed: Oct. 24, 2016. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6460145

[12] E. Tola, V. Lepetit, and P. Fua, ‘DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 5, pp. 815–830, May 2010, doi: 10.1109/TPAMI.2009.77.

[13] X. Peng, A. Bouzerdoum, and S. L. Phung, ‘Efficient cost aggregation for feature-vector-based wide-baseline stereo matching’, *EURASIP Journal on Image and Video Processing*, vol. 2018, no. 1, Dec. 2018, doi: 10.1186/s13640-018-0249-y.

[14] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, ‘Speeded-Up Robust Features (SURF)’, *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, Jun. 2008, doi: 10.1016/j.cviu.2007.09.014.

[15] Hammam A. Alshazly, M. Hassaballah, Abdelmgeid A. Ali, and G. Wang, ‘An Experimental Evaluation of Binary Feature Descriptors’, in *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2017*, Cham, 2018, vol. 639, pp. 181–191. doi: 10.1007/978-3-319-64861-3_17.

[16] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, ‘BRIEF: Binary Robust Independent Elementary Features’, in *Computer Vision – ECCV 2010*, 2010, pp. 778–792.

[17] A. Alahi, R. Ortiz, and P. Vandergheynst, ‘FREAK: Fast Retina Keypoint’, in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, Jun. 2012, pp. 510–517. doi: 10.1109/CVPR.2012.6247715.

[18] H. Chatoux, F. Lecellier, and C. Fernandez-Maloigne, ‘Comparative study of descriptors with dense key points’, Dec. 2016, pp. 1988–1993. doi: 10.1109/ICPR.2016.7899928.

[19] J. Heinly, E. Dunn, and J.-M. Frahm, ‘Comparative evaluation of binary features’, in *Computer Vision–ECCV 2012*, Springer, 2012, pp. 759–773.

[20] C. Chahla, H. Snoussi, F. Abdallah, and F. Dornaika, ‘Learned versus handcrafted features for person re-identification’, *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 34, no. 04, p. 2055009, 2020.

[21] C. Parker, M. Daiter, K. Omar, G. Levi, and T. Hassner, ‘The CUDA LATCH binary descriptor: because sometimes faster means better’, Sep. 2016, pp. 685–697. [Online]. Available: http://arxiv.org/abs/1609.03986

[22] A. Tonioni, F. Tosi, M. Poggi, S. Mattoccia, and L. D. Stefano, ‘Real-Time Self-Adaptive Deep Stereo’, in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 195–204. doi: 10.1109/CVPR.2019.00028.

[23] Z. Li *et al.*, ‘Efficient parallel optimizations of a high-performance SIFT on GPUs’, *Journal of Parallel and Distributed Computing*, vol. 124, pp. 78–91, Feb. 2019, doi: 10.1016/j.jpdc.2018.10.012.

[24] A. Eltahawi, I. Mostafa, and A. Ghuniem, ‘Image De-noising Using Intelligent Parameter Adjustment’, *IJICIS*, vol. 20, no. 2, pp. 53–66, Jan. 2021, doi: 10.21608/ijicis.2020.43046.1030.

[25] A. Amin, ‘A Face Recognition System Based on Deep Learning (FRDLS) to Support the Entry and Supervision Procedures on Electronic Exams’, *IJICIS*, vol. 20, no. 1, pp. 40–50, Jun. 2020, doi: 10.21608/ijicis.2020.23149.1015.

[26] H. I. F. Ibrahim, H. Khaled, N. A. Seada, and H. M. Faheem, ‘Parallel Dense Binary Stereo Matching Using CUDA’, in *2020 15th International Conference on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, Dec. 2020, pp. 1–6. doi: 10.1109/ICCES51560.2020.9334591.

[27] H. I. F. Ibrahim, H. Khaled, N. A. Seada, and H. M. Faheem, ‘Combining BRIEF and AD for Edge-Preserved Dense Stereo Matching’, in *Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2021)*, Cham, 2021, pp. 494–504.

[28] Şahin Işık and Kemal Özkan, 'A Comparative Evaluation of Well-known Feature Detectors and Descriptors', *Int J App Math Elt Comp*, vol. 3, no. 1, p. 1, Dec. 2014, doi: 10.18100/ijamec.60004.

[29] A. Cīrulis, K. Brigmanis-Briģis, and G. Zvejnieks, 'Analysis of Suitable Natural Feature Computer Vision Algorithms for Augmented Reality Services', *BJMC*, vol. 8, no. 1, 2020, doi: 10.22364/bjmc.2020.8.1.10.

[30] A. Jafari Malekabadi, M. Khojastehpour, and B. Emadi, 'A comparative evaluation of combined feature detectors and descriptors in different color spaces for stereo image matching of tree', *Scientia Horticulturae*, vol. 228, pp. 187–195, Jan. 2018, doi: 10.1016/j.scienta.2017.10.030.

[31] J. Figat, T. Kornuta, and W. Kasprzak, 'Performance Evaluation of Binary Descriptors of Local Features', in *Computer Vision and Graphics*, Cham, 2014, vol. 8671, pp. 187–194. doi: 10.1007/978-3-319-11331-9_23.

[32] Z. Peng, 'Efficient matching of robust features for embedded SLAM', Master's Thesis, 2012.

[33] M. Bansal, M. Kumar, and M. Kumar, '2D object recognition: a comparative analysis of SIFT, SURF and ORB feature descriptors', *Multimed Tools Appl*, Feb. 2021, doi: 10.1007/s11042-021-10646-0.

[34] Guney Kayım, 'EVALUATION OF 2D LOCAL IMAGE DESCRIPTORS AND FEATURE ENCODING METHODS FOR DEPTH IMAGE BASED OBJECT CLASS RECOGNITION', Master Thesis, Bogazici University, Istanbul, Turkey, 2014.

[35] G. Eilertsen, P.-E. Forssén, and J. Unger, 'BriefMatch: Dense Binary Feature Matching for Real-Time Optical Flow Estimation', in *Image Analysis*, 2017, pp. 221–233.

[36] J. Huang and G. Zhou, 'On-Board Detection and Matching of Feature Points', *Remote Sensing*, vol. 9, no. 6, p. 601, Jun. 2017, doi: 10.3390/rs9060601.

[37] gillevicv, 'Tutorial on Binary Descriptors – part 1', *Gil's CV blog*, Aug. 26, 2013. https://gilscvblog.com/2013/08/26/tutorial-on-binary-descriptors-part-1/ (accessed Jun. 13, 2020).

[38] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, 'Brief: Binary robust independent elementary features', in *European conference on computer vision*, 2010, pp. 778–792.

[39] G. Levi and T. Hassner, 'LATCH: learned arrangements of three patch codes', in *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, 2016, pp. 1–9.

[40] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, 'ORB: An efficient alternative to SIFT or SURF', 2011, pp. 2564–2571.

[41] S. Leutenegger, M. Chli, and R. Y. Siegwart, 'BRISK: Binary robust invariant scalable keypoints', in *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011, pp. 2548–2555.

[42] E. Rosten and T. Drummond, 'Machine Learning for High-Speed Corner Detection', in *Computer Vision – ECCV 2006*, vol. 3951, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 430–443. doi: 10.1007/11744023_34.

[43] G. Klein and D. Murray, 'Parallel Tracking and Mapping for Small AR Workspaces', in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nara, Japan, Nov. 2007, pp. 1–10. doi: 10.1109/ISMAR.2007.4538852.

[44] K. Mikolajczyk *et al.*, 'A Comparison of Affine Region Detectors', *International Journal of Computer Vision*, vol. 65, no. 1–2, pp. 43–72, Nov. 2005, doi: 10.1007/s11263-005-3848-x.

[45] D.-D. Truong, C.-S. N. Ngoc, V.-T. Nguyen, M.-T. Tran, and A.-D. Duong, 'Local Descriptors without Orientation Normalization to Enhance Landmark Regconition', Cham, 2014, pp. 401–413.

[46] G. Baatz, K. Köser, D. Chen, R. Grzeszczuk, and M. Pollefeys, 'Handling urban location recognition as a 2d homothetic problem', 2010, pp. 266–279.

[47] D. Scharstein *et al.*, ‘High-resolution stereo datasets with subpixel-accurate ground truth’, in *German Conference on Pattern Recognition*, 2014, pp. 31–42.

[48] K. Honauer, L. Maier-Hein, and D. Kondermann, ‘The HCI Stereo Metrics: Geometry-Aware Performance Analysis of Stereo Algorithms’, in *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, Dec. 2015, pp. 2120–2128. doi: 10.1109/ICCV.2015.245.