IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 13, 2019, accepted September 5, 2019, date of publication September 17, 2019, date of current version October 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2940995

An Adaptive Uniform Distribution ORB Based on Improved Quadtree

JINJIN YAO[1,2], PENGCHAO ZHANG[1,2], YAN WANG[1,2], ZHAOYANG LUO[1,2], AND XIAOHUI REN[2,3]

[1]School of Mechanical Engineering, Shaanxi University of Technology, Hanzhong 723000, China
[2]The Key Laboratory of Industrial Automation of Shaanxi Province, Shaanxi University of Technology, Hanzhong 723000, China
[3]School of Electrical Engineering, Shaanxi University of Technology, Hanzhong 723000, China

Corresponding author: Pengchao Zhang (snutzpc@126.com)

This work was supported in part by the 2011 Collaborative Center Innovation Project under Grant QBXT-17-7, and in part by the Scientific Research Plan Projects of Shaanxi Education Department under Grant 18JS021.

ABSTRACT ORB (Oriented FAST and Rotated BRIEF) feature is wildly applied in visual SLAM because of its excellent computational efficiency and stability. Aiming at the problem of uneven distribution of ORB feature, and improving the calculate efficiency of feature extraction at the same time, we proposed an ORB feature extraction algorithm based on improved quadtree in this paper. The proposed algorithm will select the threshold adaptively for FAST extraction according to the gray image instead of the value set artificially. And then we set different depth of quadtree according to the expected feature number which decreases as the number of image pyramid layers increases to reduce redundancy. The remained key points selected by Harris score will distribute well in the image. The results show that the proposed algorithm can improve the uniformity of ORB feature, and reduce feature extraction time compared to the algorithm in ORB_SLAM, it has certain application value for the realization of real-time SLAM system.

INDEX TERMS Simultaneous localization and mapping, feature extraction, improved quadtree, uniform distribution.

I. INTRODUCTION

SLAM (Simultaneous Localization and Mapping) is the process through sensor to reconstruct the environment and estimate the position of robots at the same time [1], [2]. It has been catching the attention of more and more scholars in recent years. The technology upgrades faster especially in Visual SLAM because the visual sensor can get more information than other traditional sensors [3], [4]. It plays an important role in 3D reconstruction, semantic comprehension, navigation, path planning and other application. However, there are still some questions in feature uniform distribution which will bring bad impact to the subsequent work such as image matching and pose estimation. It is necessary to improve the uniformity of feature distribution and the calculate efficiency.

There are many feature extraction algorithms for visual SLAM. Scale Invariant Feature Transform (SIFT) algorithm was proposed by Lowe [5] in 1999 and improved in 2004, the features are invariant to image scale and rotation and robust in image matching. The SIFT gives a possibility for the realization of visual SLAM. But SIFT needs a lot computational resource because of the 128-dimensional feature vector. To solve the problem, Speeded-Up Robust Features (SURF) algorithm was proposed by Bay [7] in 2006.The algorithm accelerates the extraction process by relying on integral images for image convolutions, using a Hessian matrix-based measure for the detector and a distribution-based descriptor. SURF algorithm is faster than SIFT, but it still cannot meet the real-time requirement of visual SLAM. In 2001, Rublee [8] proposed ORB algorithm which is at two orders of magnitude faster than SIFT. The algorithm solved the problem of time consumption, and it can operate in real-time without GPU. However, although this algorithm improves the extraction efficiency of feature extraction, ORB feature points are still unevenly distributed in the image plane and easy to aggregation, which reduces the accuracy of subsequent feature matching and pose estimation. Therefore, in the ORB-SLAM system, Mur-Arta *et al.* [9] and Mur-Artal and Tardós [10] proposed the use of quadtree to improve the uniformity of feature distribution, which has a very obvious effect. Yu Xinyi proposed Qtree_ORB [11] algorithm on this basis, effectively

The associate editor coordinating the review of this manuscript and approving it for publication was Xinheng Wang.

eliminate the redundant feature points, but the algorithm still adopts the traditional quadtree structure, so the computational efficiency needs to be improved. Wang set the maximum depth of the quadtree for matching. The algorithm is used to adaptively segment the image into blocks by quadtree. However, it should not be same maximum depth for different pyramid layer of the image, it ought to determine by the layer. The maximum depth should be larger for low pyramid layer than the high pyramid layer's, because the number of feature points in low layer is larger than the number in high layer.

In this paper, we proposed an ORB uniform distribution algorithm based on improved quadtree. The overall image contrast is taken into account when extracting feature points, and the maximum depth of quadtree is set according to different pyramid layers, so as to eliminate redundant feature points and improve the efficiency of feature detection.

This article is organized as follows. Section II provides an overview of existing feature extracting algorithms. Section III introduces our uniform distribution ORB feature based on improved quadtree. Section IV discusses the results between the previous algorithms an ours. Finally, Section V concludes the paper.

II. RELATED WORK

Traditional ORB algorithm includes two steps, the FAST corner detection and Rotated in the descriptor computation [8]. Among them, FAST algorithm detects corner points by the difference between the selected element and the pixel gray of surrounding elements. In this paper [8], fast-9 algorithm is adopted. Taking pixel P as an example, a total of 16 pixel points are found on a circle with a center radius of 3 pixels of P. If the gray difference between 9 consecutive pixels and P in these 16 pixels is greater than the set threshold value t, P is judged to be the corner point of FAST. Computing such as:

$$N = \sum_{x \in (circle(p))} |I(x) - I(p)| \geq t \tag{1}$$

where $I(x)$ is the pixel value of any point on the circumference, $I(p)$ is the gray value of the pixel to be detected, t is the corner detection threshold, and N=9 in FAST-9. In addition, in order to eliminate the edge points, the Harris response value is used to replace the FAST response value, and in order to improve the robustness of the feature, the scale invariance is realized by constructing the pyramid, and the main direction is added to the feature point to realize the rotation invariance. In order to avoid the problem of corner concentration, it is screened by non-maximum suppression. The calculation of the direction of its feature points is as follows. The moments of a patch is defined as:

$$m_{pq} = \sum_{x,y \in B} x^p y^q I(x, y), p, q = \{0, 1\} \tag{2}$$

Next we find the centroid of the image block by the moment:

$$C = \left(m_{10}/m_{00},\ m_{01}/m_{00}\right) \tag{3}$$

Then we connect the geometric center O and the centroid C of the image block to obtain a direction vector $\overrightarrow{OC}$, and the direction of the feature is defined as:

$$\theta = \arctan\left(m_{01}/m_{10}\right) \tag{4}$$

The description of the feature points uses Rotation BFIEF, which is a binary descriptor whose description vector consists of multiple 0 and 1. The sum of the gray values of the sub-windows in the pixel area near the feature point is used as the basis for judging the descriptor, and the sub-segment function is obtained. The specific calculation is:

$$\tau(p; x, y) = \begin{cases} 1: & p(x) < p(y) \\ 0: & p(x) \geq p(y) \end{cases} \tag{5}$$

The feature is defined as a vector of n binary tests:

$$f_n(p) := \sum_{1 \leq i \leq n} 2^{i-1} \tau(p; x_i, y_i) \tag{6}$$

where p(x) and p(y) are the gray values of the pixels at point x and point y, respectively, τ is the value of the descriptor, and the gray value at x and y is the sum of the gray values of the above sub-windows. Furthermore, the direction calculated in equation (4) is added to the descriptor so that the descriptor has good rotational invariance.

III. UNIFORM DISTRIBUTION ORB

The improved algorithm firstly aims at the problem of poor anti-interference ability for FAST corner points, and replaces the artificial value by the gray value calculation of each image. Secondly, according to the method proposed by Mur-Arta *et al.* [9] and Mur-Artal and Tardós [10], we set different maximum depth for quadtree according to different pyramids to limit the over-segmentation. The maximum depth will be determined by the expected number of key-points in each pyramid layer. The purpose of different maximum depth is to reduce redundant features and improve the computational efficiency. The algorithm structure is shown in Figure 1.

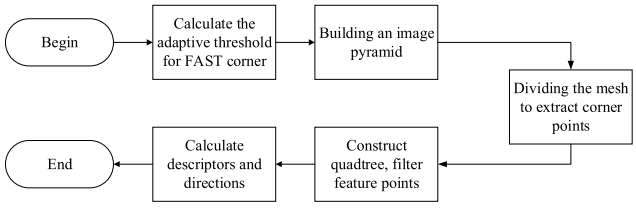


FIGURE 1. Algorithm flowchart.

The extraction of the FAST corner point is performed according to the gradation difference between the pixel to be detected and the surrounding pixel. However, the extracted threshold of FAST corner is set artificially in traditional ORB algorithm, it's usually engineering experience value. It's the same situation in Mur-Arta's algorithm. But this value does

not consider the global information of the image itself, it cannot apply to all images, especially in images with large contrast differences. Otherwise, the adaptive threshold proposed by Fan [8] still contains the scale factor set artificially. So we set a adaptive threshold based on image gray information for FAST corner detection. The calculation method of initial threshold is as shown in equation (7):

$$\text{ini}T = \left(\frac{1}{n}\sum_{i=1}^{n}\left(I\left(x_i\right) - \overline{I\left(x\right)}\right)^2\right) \Big/ \overline{I\left(x\right)} \tag{7}$$

where $I\left(x_i\right)$ is the gray value of each pixel in the image, $\overline{I\left(x\right)}$ is the average value of the image gray. And *iniT* is the calculated initial extraction threshold. By using equation (7), different comparison thresholds can be calculated according to different images, so that the algorithm has stronger anti-interference ability.

In the second step, we calculated the image pyramid in order to make the ORB feature scale invariant, an eight-layer pyramid is constructed for the image, and then we calculated the number of desired feature points required by each layer according to the scale factor. For example, we set the total number of required feature points is m, the scale factor is s, and the number of features required for the first layer is a, then:

$$a + \frac{1}{s}a + \frac{1}{s^2}a + \frac{1}{s^3}a + \ldots\ldots + \frac{1}{s^7}a = m \tag{8}$$

The third step is to divide the mesh to extract the FAST corner points. In order to make the corner points evenly distributed throughout the image, the image is meshed. Since the nodes of the quadtree use the same rectangle as the original image aspect ratio, a square is used for the mesh division, so that the division of the mesh does not coincide with the boundary of the quadtree, which reduces the effect of ignoring mesh edge corner detection on feature extraction. After the meshing is completed, the FAST corner extraction is started, and the initial extraction threshold of the corner point is set to the calculated value *iniT* of the equation (7). If the corner point is not extracted in the grid, the threshold is lowered to *minT*= *iniT/4*. And then corner detection will continue within the grid until the traversal of all the meshes in the image to complete the adaptive extraction of the FAST corners.

However, in the fourth step, since the quadtree depth is not limited, the number of segmentation times is too much, which reduces the computational efficiency of the algorithm. To solve the problem, an adaptive maximum depth is set according to the desired feature points of different pyramid layers. The relationship between the maximum depth and the number of nodes in this layer is as shown in equation (9):

$$4^{D_{\max}} \geq Num\_j \tag{9}$$

where $D_{\max}$ is the maximum depth, *Num_j* is the desired number of key-points.

Taking 500 features as an example, the maximum depth of the first layer is set to 5 to meet the requirements, and the maximum depth of the 8th layer is set to 3. So it will not split deeper sub-node when the current split depth coming to the maximum depth. The design of different maximum depths can improve the calculation efficiency. The specific algorithm flow is shown in Figure.2.

Modified Quadtree
Input: All image keypoints
output: Saved Nodes
Initialize Node
Split sub-node
while $d < Dmax$ **do**
  if $Num\_kp > 0$ **then**
    if $Num\_kp > 1$ **then**
      $Splitsub - node$
    else
      $Savesub - node$
      Num_j++
    end if
  end if
  if $Num\_j > set\_kp$ **then**
    $Calculate\ Harris\ score\ and\ filtrate$
  end if
end while

FIGURE 2. Modified quadtree splitting process about how to manage the split node.

In Figure2, d is the current quadtree depth, D_{max} is the maximum depth of the pyramid layer, it's calculated according to layers by equation (8), and the larger the value, the more key points need to be calculated, it will decide the compute efficiency. *Nun_kp* is the number of feature points in the node, *Num_j* is the number of saved nodes, there is only one key point in every saved node. *Set_kp* is the number of desired feature points of the layer, if *Num_j* is updated to *Set_kp*, the algorithm will stop splitting sub-node, it means we have got enough feature points.

The fifth step is to calculate the direction and descriptor of the feature points. This paper still uses the traditional calculation method [4], which will not be repeated here.

IV. EXPERIMENT VERIFICATION

In this section, we employ three kinds of ORB algorithms include traditional ORB, the algorithm proposed in ORB_SLAM and what we proposed in this paper to extract key points in the dataset created by K.Mikolajczyk and C.Schmid. This experiment was carried out on the Ubuntu16.04 LTS operating system, the computer CPU is i5-4258U, 8GB memory. In order to quantify the uniformity, we use the uniformity function [14] to extract the results. And in order to verify the adaptability of the algorithm to different contrast images, we performed experiments on the leuven image set. Otherwise, the bike datasets are a set of images with different degrees of blur, and the leuven datasets are a set of different contrast images, the bark datasets are

(a) (b) (c)

FIGURE 3. The result of three kinds of algorithm to extract 500 ORB key-points with img1 of the bike datasets. (a) is the result of traditional ORB algorithm, (b) is the result of Mur-Artal's algorithm used in ORB_SLAM, and (c) is the result of our algorithm.

TABLE 1. Uniformity and time-consuming comparison of three algorithms.

Image sequence	Uniformity			Time/ms		
	A	B	C	A	B	C
1	210.06	137.55	140.81	36.86	51.67	45.34
2	201.62	158.48	161.40	38.60	49.61	43.59
3	209.01	170.36	173.73	35.82	51.57	43.79
4	209.61	176.36	180.35	36.70	49.88	43.07
5	205.00	174.74	177.38	32.50	47.03	43.46
6	207.82	177.91	181.68	31.06	46.63	41.02

a set of images with a single background occupying more image, the trees datasets are a set of images with complex foreground which occupying more space, the ubc datasets are a set of images with different degrees of compression. In these datasets, the traditional ORB algorithm, the feature extraction algorithm proposed by Mur-Arta and the algorithm we proposed for comparison experiments, and without loss of generality. We extracted 500 features for each image of data set, and took the average value as the experimental result with 30 experiments. The results of three kinds of algorithm to extract ORB feature points with the same image is shown as Figure 3. And the results of uniformity and feature extraction time are shown in the Table 1.

In Table 1, A represents traditional ORB, B represents Mur-Artal's algorithm used in ORB_SLAM, C represents the our algorithm. We can see that the B and C algorithms are far better than the traditional algorithm in terms of uniformity, which is 10% to 30% better than the traditional algorithm, and the uniformity of B and C is similar. But in terms of time, C algorithm is 12.12% less than B algorithm on average.

In order to verify the adaptability of the our algorithm to different contrasts, the data set leuven was performed. We calculated the feature uniformity and extraction time of each image in leuven. The experimental results are shown in Figure 4, Figure 5 and Table 2.

Figure4 and Figure5 show the feature extraction results of the six images in the dataset leuven. The results show that our algorithm can achieve good feature uniformity even in the case of large contrast.

From Table 2, we can see that, the algorithm we proposed has the same feature distribution and degree as the Mur-Artal's algorithm, but it has a significant reduction in time consumption, the average reduction time is 10.43%. And since the extraction threshold of the FAST corner point is calculated based on the image information, the algorithm has stronger anti-interference ability for images with different contrasts.

TABLE 2. Comparison of the uniformity and time consumption of the three algorithms on leuven.

Image sequence	Uniformity			Time/ms		
	A	B	C	A	B	C
1	203.30	166.22	170.98	33.50	37.30	33.77
2	214.93	158.34	157.64	33.08	38.20	34.76
3	217.05	163.85	164.59	30.60	37.09	32.67
4	216.57	162.23	168.33	30.58	36.53	32.80
5	215.70	167.55	168.95	31.92	36.79	32.57
6	216.82	167.67	164.92	31.03	36.42	32.38

The data of the remaining groups are shown in Figure6 and Figure7. Where Old represents the traditional ORB algorithm, MR represents the algorithm used in ORB_SLAM, My represents the algorithm we proposed, and img1 represents the number of different images in the data set.

As shown in Figure 6, under three different data sets, the algorithm of Mur-Arta and ours are almost the same in uniformity, and they are much better than the traditional ORB feature extraction algorithm. While, the algorithm we proposed here is slightly better than Mur-Arta algorithm in bark and trees datasets, and the experimental results under

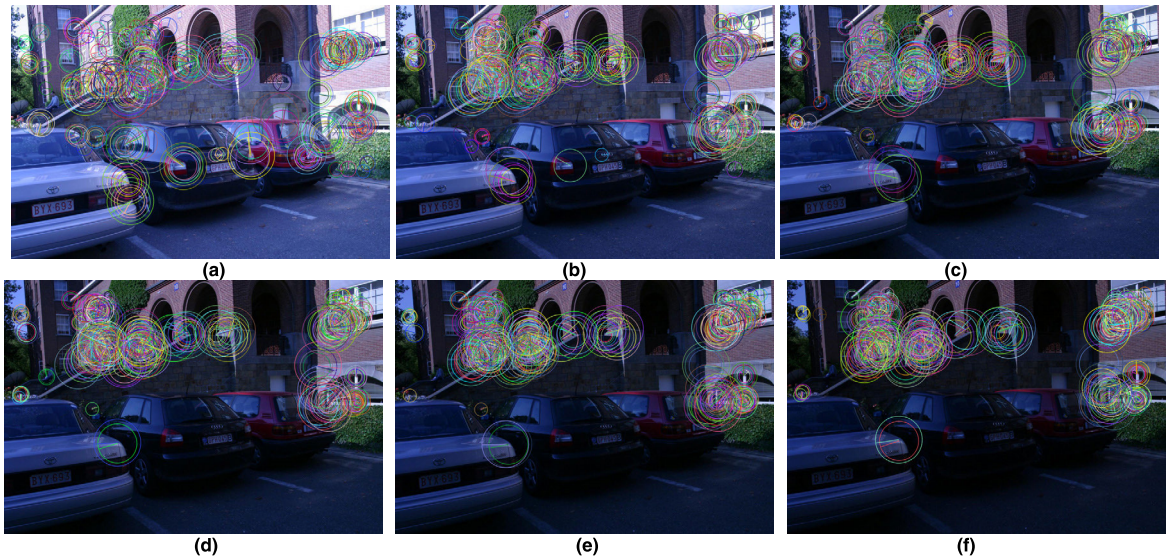


FIGURE 4. Feature extraction results of traditional ORB algorithm in leuven, (a)-(f) is image sequence of leuven. As the illumination changed, the uniformity got worse.

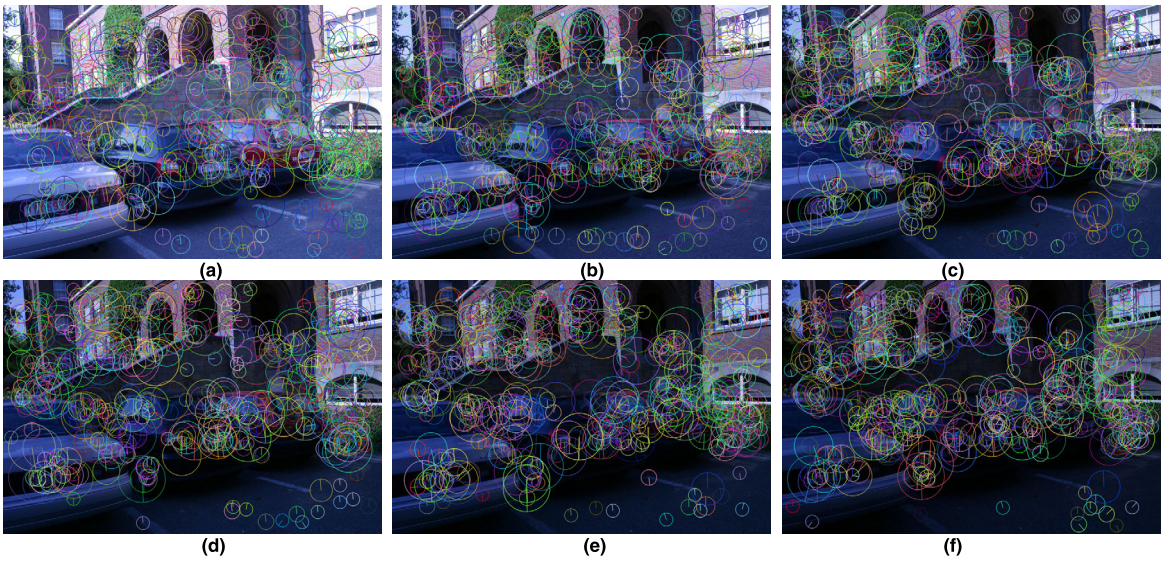


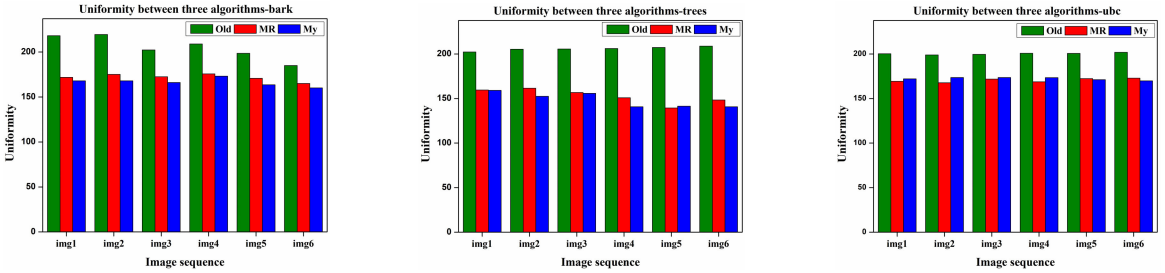
FIGURE 5. Feature extraction results of the algorithm we proposed in leuven, (a)-(f) is image sequence of leuven.

the ubc datasets have no obvious advantages, the uniformity of the two algorithms is not much different when dealing with images with higher compression.

It can be obtained from the analysis of Figure7 that under the three different datasets, the extraction time of the traditional ORB feature extraction algorithm is the fastest, while the extraction time of Mur-Arta algorithm and our algorithm is higher than the traditional ORB algorithm. It is mainly due to the calculation of increasing the uniformity. More importantly, our algorithm has a significant improvement in extraction time compared to the widely used Mur-Arta's algorithm proposed in ORB_SLAM. Under the bark data set, the algorithm extraction time is 18.10% lower than the Mur-Arta's, and the average reduction time is. 16.73% under the trees data set, the average reduction time in the ubc data set is 16.30%, and the extraction efficiency is greatly improved.

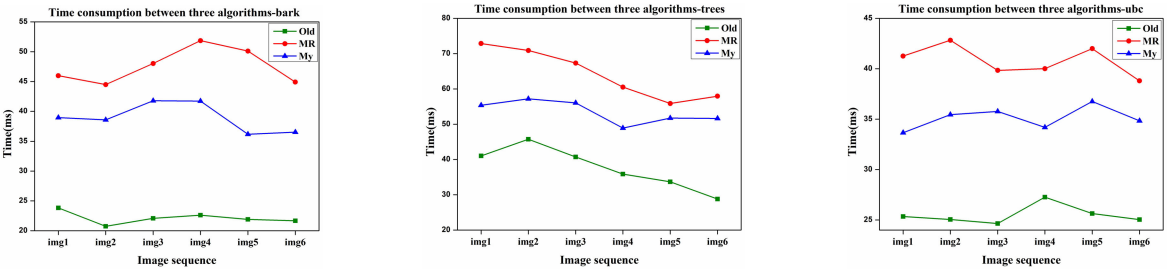
The experiment is carried out in different backgrounds, different degrees of blur, different illumination conditions, and different degrees of compression. The results show that the proposed algorithm has a great improvement over the feature uniformity of the traditional ORB algorithm. More importantly, the extraction time is reduced 10% or more.

We also did another experiment to verify the effectiveness of modified algorithms in uniformity. We used



(a) bark (b) trees (c) ubc

FIGURE 6. Uniformity comparison of three algorithms under different data sets. The traditional ORB algorithm is far less than the advanced algorithms whatever Mur-Arta's algorithm or our algorithm.



(a) bark (b) trees (c) ubc

FIGURE 7. Extract time comparisons of three algorithms under different data sets. The traditional algorithm is faster than the other algorithms. But our algorithm is faster than Mur-Arta's algorithm.

TABLE 3. Uniformity in different datasets with five methods.

Dataset	Method	Img1	Img2	Img3	Img4	Img5	Img6	Mean value
bark	SIFT	208.35	205.35	181.10	193.25	180.82	172.00	190.15
	SURF	193.05	192.66	178.53	172.16	162.70	164.22	177.22
	BRISK	212.00	207.06	211.03	204.08	197.01	201.01	205.37
	SuperPoint	159.60	197.70	190.10	132.74	181.42	157.63	169.87
	OURS	167.44	167.94	166.09	173.14	163.63	160.07	**166.39**
trees	SIFT	200.25	222.39	199.64	210.45	193.65	193.81	203.37
	SURF	214.70	215.47	222.49	214.21	210.45	195.93	212.21
	BRISK	198.69	209.20	198.90	199.25	212.74	197.48	202.71
	SuperPoint	149.38	130.02	108.16	109.30	141.75	168.14	**134.46**
	OURS	159.18	152.42	155.84	140.72	141.38	140.72	148.38
ubc	SIFT	187.39	189.53	189.18	180.87	191.02	195.47	188.91
	SURF	190.70	192.18	191.15	189.73	191.59	191.93	191.21
	BRISK	221.44	220.43	217.01	214.74	208.57	210.9	215.52
	SuperPoint	201.69	202.48	203.78	199.34	201.75	187.31	199.39
	OURS	172.04	173.44	173.56	173.43	171.13	169.86	**172.24**
bikes	SIFT	176.66	172.23	172.79	173.46	177.28	159.89	172.05
	SURF	181.63	170.63	178.91	178.84	184.07	184.57	179.78
	BRISK	199.92	185.84	194.99	197.14	198.21	197.15	195.54
	SuperPoint	141.53	189.72	197.53	185.21	188.46	186.88	181.56
	OURS	140.81	161.40	173.73	180.35	177.38	177.91	**168.60**
leuven	SIFT	192.69	202.59	208.17	210.80	212.79	215.77	207.14
	SURF	186.44	178.70	178.91	178.84	184.07	184.57	181.92
	BRISK	194.36	199.71	194.99	197.14	197.21	197.15	196.76
	SuperPoint	160.8	151.84	197.53	185.21	188.46	186.88	178.45
	OURS	170.98	157.64	164.59	168.33	168.95	164.92	**165.90**

five algorithms to extract feature points include SIFT, SURF, BRISK, SuperPoint and our algorithm. As shown in paper [18], SIFT also has good performance in feature extraction and matching. SuperPoint is a new method based on deep learning [19]. We calculate uniformity of every image in the five datasets, the uniformity mean value in all five datasets is shown in Table 3 and Figure8 (a-e).

We can see that our algorithm's uniformity is still the smallest in all five algorithms in most datasets except in trees. It means our algorithm can get better feature distribution.

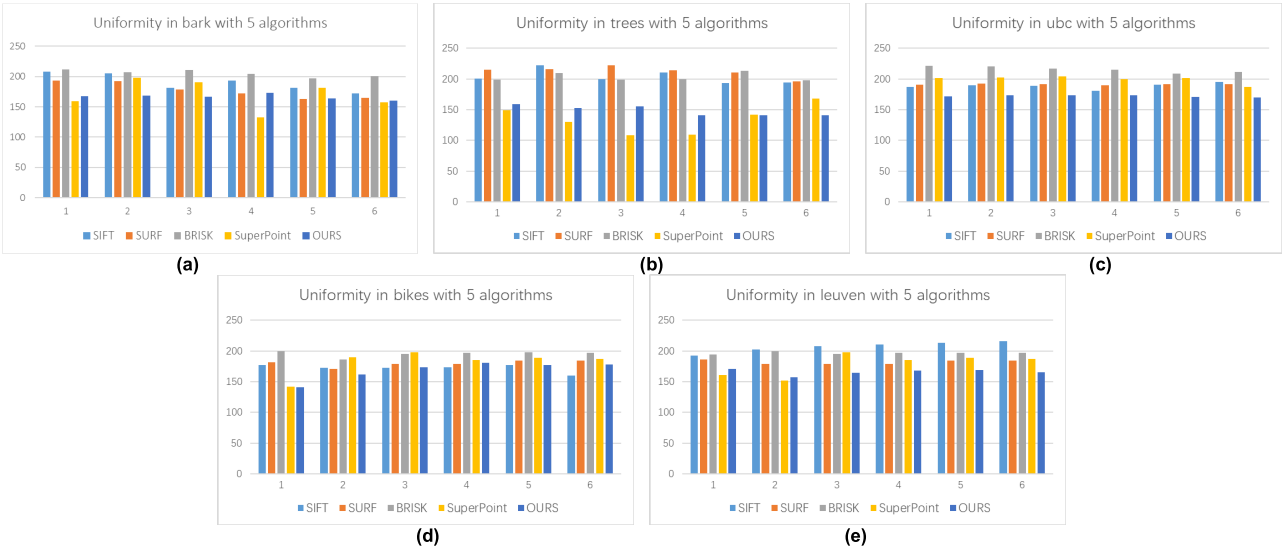


FIGURE 8. Uniformity in different datasets with five algorithms include SIFT, SURF, BRISK, SuperPoint and ours. The x-coordinate represents the sequence of images, and the y-coordinate represents the uniformity value. Our algorithm is still better than other algorithms in most cases in terms of feature distribution.

From the Fig.8, we can infer that our algorithm's uniformity is the smallest in most pictures of datasets. However, we can also see that SuperPoint also has good performance in datasets trees, the uniformity in img3 is even up to 108.16. But there is a big gap between different images in the same datasets. In trees datasets, the biggest uniformity value is 168.14, it's 50% more than the smallest one. It's not a good index, the matching algorithms maybe cannot find the real matching feature points because the large difference in uniformity. Our algorithm has a good performance in this area. The uniformity value is close to the mean value.

V. CONCLUSION

In this paper, aiming at the problem that the traditional ORB algorithm is unevenly distributed and the aggregation phenomenon is obvious, we presented an adaptive uniform distribution ORB feature extraction algorithm based on improved quadtree, which solved the problem of adaptive extraction threshold selection for FAST corner points, and the quadtree depth of each layer is set for the number of desired feature points for different pyramid layers. Finally, the public data set is used for experimental verification. The experimental results show that the algorithm has a higher uniformity than the traditional ORB algorithm, and has higher computational efficiency than the proposed algorithm by Mur-Artal. And compared with the other state-of-the-art feature extraction algorithms, our algorithm still has good performance in feature distribution. The feature extraction time is reduced by more than 10% due to the reduction of over-segmentation of quadtree. The algorithm does have certain application value for SLAM's subsequent feature matching and pose estimation.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and John J. Leonard, "Simultaneous Localization And Mapping: Present, Future, and the Robust-Perception Age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Jun. 2016.

[2] M. X. Quan, S. T. Piao, and G. Li, "An overview of visual SLAM," *CAAI Trans. Intell. Syst.*, vol. 11, no. 6, pp. 768–776, 2016.

[3] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[4] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2014, pp. 834–849.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[6] Z. Songtao, L. Chao, and L. Liqing, "An improved method for eliminating false matches," in *Proc. 2nd Int. Conf. Image, Vis. Comput. (ICIVC)*, Jun. 2017, pp. 133–137.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 346–359, 2008.

[8] E. Rublee, V. Rabaud, and K. Konolige, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2012, p. 2.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[10] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[11] X. Y. Yu, Y. A. Zhan, and F. Zhu, "Improved ORB feature extraction algorithm based on quadtree encoding," *Comput. Sci.*, vol. 45, no. 2, pp. 232–235, 2018.

[12] Z. Yi, J. Ting, J. Gang-Wu, Y. Ying, and Z. Yuan, "Uniform distributed subpixel ORB feature extraction method for high-precision SLAM," *Opt. Precis. Eng.*, vol. 26, no. 10, pp. 2575–2583, Oct. 2018.

[13] C. L. Lin, X. S. Hao, L. C. Ma, and K. C. Wang, "Partially occluded vehicle tracking based on ORB," *Comput. Eng. Des.*, vol. 37, no. 1, pp. 242–246, 2016.

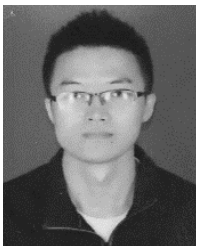
[14] H. F. Zu and C. H. Zhao, "The evaluation method of image feature point distribution uniformity," *J. Daqing Normal Univ.*, vol. 30, no. 3, pp. 9–12, 2010.

[15] Y. W. Wang, M. Yu, and H. Jiang, "An image stitching via adaptive quadtree segmentation," *J. Ningbo Univ. (NSEE)*, vol. 112, no. 4, pp. 58–64, 2018.

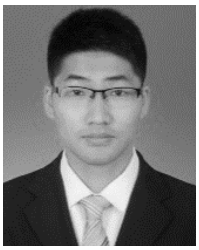
[16] H. Liu, H. Yu, and Y. Liang, "ORB feature quadtree uniform distribution algorithm," *Process Autom. Instrum.*, vol. 39, no. 441, pp. 55–57, May 2018.
[17] F. Wang, H. G. You, and X. Y. Fu, "Auto-adaptive well-distributed scale-invariant Feature for SAR image registration," *Geomatics Inf. Sci. Wuhan Univ.*, vol. 40, no. 2, pp. 159–163, 2015.
[18] K. M. Yi, E. Trulls, and V. Lepetit, "LIFT: Learned invariant feature transform," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 467–483.
[19] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Jun. 2018, pp. 224–236.

JINJIN YAO was born in Shanxi, China, in 1993. He received the B.S. degree from the Taiyuan University of Technology, China, in 2016. He is currently pursuing the M.S. degree with the School of Mechanical Engineering, Shaanxi University of Technology, Shaanxi. His current research interests include robotics and visual SLAM.

PENGCHAO ZHANG received the B.Eng. degree in automation from the Shaanxi University of Technology (SNUT), Hanzhong, China, and the M.Eng. degree in traffic control engineering from Northwestern Polytechnical University (NPU), Xi'an, China, where he is currently pursuing the Ph.D. degree. He is currently a Professor with SNUT. His current research interests include industrial robot and mobile robotics.



YAN WANG was born in Inner Mongolia, China, in 1994. He received the B.S. degree from Northwestern Polytechnical University Ming De College, China, in 2016. He is currently pursuing the M.S. degree with the School of Mechanical Engineering, Shaanxi University of Technology, Shaanxi. His current research interests include robotics and navigation.



ZHAOYANG LUO was born in Sichuan, China, in 1994. He received the B.S. degree from the North China Electric Power University Science and Technology College, China, in 2018. He is currently pursuing the M.S. degree with the School of Mechanical Engineering, Shaanxi University of Technology, Shaanxi. He is currently working on image processing.

XIAOHUI REN was born in Sichuan, China, in 1995. He received the B.S. degree from the Shaanxi University of Technology, China, in 2018, where he is currently pursuing the M.S. degree with the School of Electrical Engineering. His current research interests include robotics and neural network.