Accelerating Local Feature Extraction Using Two Stage Feature Selection and Partial Gradient Computation

Keundong Lee, Seungjae Lee, Weon-Geun Oh

ETRI, Daejeon, Rep. of Korea

Abstract. In this paper, we present a fast local feature extraction method, which is our contribution to ongoing MPEG standardization of compact descriptor for visual search (CDVS). To reduce time complexity of feature extraction, two-stage feature selection, which is based on the feature selection method of CDVS Test Model (TM), and partial gradient computation are introduced. The proposed method is examined on SIFT and compared to SIFT and SURF extractor with the previous feature selection method. In addition, the proposed method is compared to various feature extraction methods of the current CDVS TM 11 in CDVS evaluation framework. Experimental results show that the proposed method significantly reduces the time complexity while maintaining the matching and retrieval performance of previous work. For its efficiency, the proposed method has been integrated into CDVS TM since 107^{th} MPEG meeting. This method will be also useful for feature extraction on mobile devices, where the use of computational resource is limited.

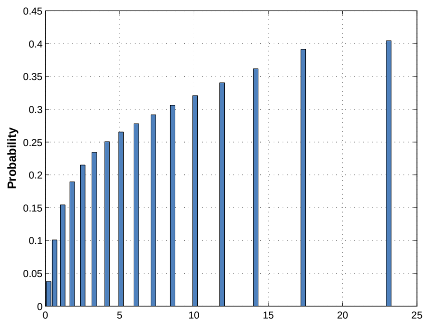
1 Introduction

Local features such as SIFT [14] and SURF [3] are widely used in visual search, object recognition and image classification for their robustness against scale, rotation, illumination changes and affine transformation.

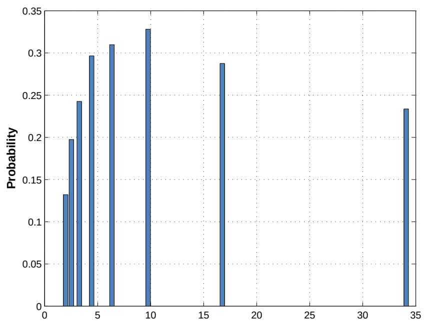
However, these local features are not feasible in mobile visual search or large-scale image retrieval due to large size and high computational complexity. To handle this problem, MPEG-7 is working on the standardization of compact descriptors for visual search (CDVS) [11]. They study on feature selection, feature compression and searching method for mobile visual search. In feature selection, it selects more helpful features for correct match based on probabilistic model [7] instead of using all the detected features to reduce computational burden in descriptor extraction and achieve compactness of descriptor. However, the complexity of feature extraction is still high.

In this paper, the efficient feature extraction with considering feature selection is proposed. To reduce computational complexity, two-stage feature selection and partial gradient computation are introduced. Experimental results verify the effectiveness of the proposed method.

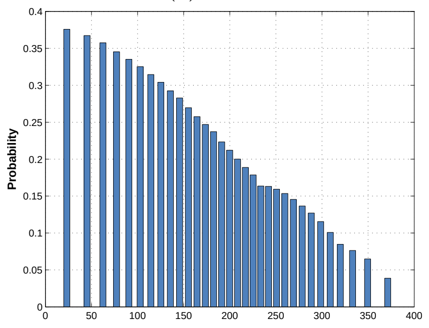
This paper is organized as follows: Section 2 describes the feature selection method of CDVS TM. Section 3 introduces the proposed method in detail.



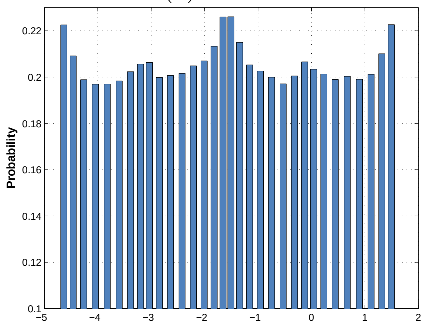
(a) Peak



(b) Scale



(c) Distance from center



(d) Orientation

Fig. 1. Conditional Probabilities of correct match on SIFT given the peak, scale, distance from center and orientation.[1]

Experimental results on uncompressed SIFT framework and CDVS evalation framework are presented in Section 4. Finally, we conclude in Section 5.

2 Feature Selection for Compact Descriptor

Feature selection is the method to select more important feature among the detected features. G. Francini et al. [7] proposed feature selection based on the characteristics of detector to generate compact descriptor with specific size such as 512B, 1KB, and this method has been adopted in CDVS TM. They estimated conditional probability of correct match as a function of four characteristics of the feature: scale, orientation, peak response (e.g., Difference-of-Gaussian (DoG) response), and distance from the center of image as shown in Fig. 1.[1] For each feature, the relevance, the probability of correct match, is assigned by multiplying four probabilities.

[1] This figure was reproduced by the author with the permission of G. Francini et al. [7] using the program and dataset which they provided during the MPEG meetings. Note that this figure is not just taken from [7], the larger datasets [4, 8, 17, 18] were used to obtain the results compared to that of [7].

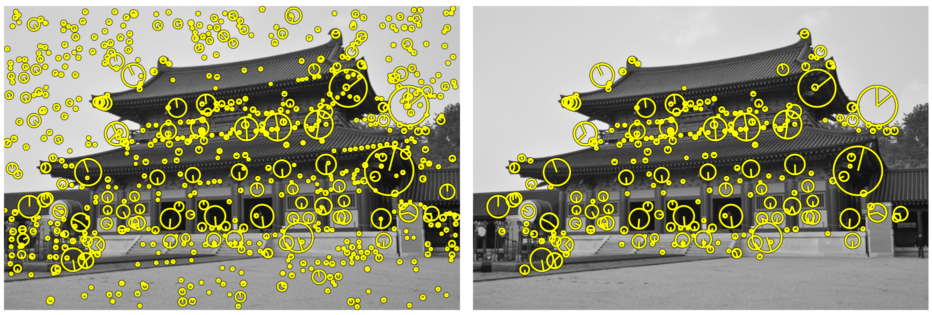

(a) Before feature selection (b) After feature selection

Fig. 2. An example of feature selection [7] with DoG detector [14].

And then, only a small number of features with high relevance are selected to satisfy its target descriptor length. An exemplar result of feature selection is shown in Fig. 2.

Consequently, they also reduced time complexity of descriptor extraction by selecting features before descriptor extraction. However, there are still rooms to be improved in view of time complexity.

In this paper, we propose more efficient feature extraction method with feature selection and show that improvement in time complexity can be achieved.

3 Proposed Method

We propose a fast feature extraction method based on SIFT descriptor which significantly reduces the time complexity while maintaining the matching and retrieval accuracy of [7]. This was achieved by two-stage feature selection, and partial gradient computation. Two methods are described in the following subsections.

3.1 Two-stage Feature Selection

In [7], keypoints are detected for input image, and orientation is computed for each keypoint, after that the relevance of feature is assigned by its peak, scale, location and orientation, and then, N most relevant features are selected to extract descriptor. Even though time complexity is reduced by extracting descriptor only for the selected features, it is still computational burden that orientation is computed for every keypoint.

To handle this problem, we divide the feature selection method into two stages: upright feature selection and feature selection. We introduce the following definition for better understanding:

- Upright Feature : Keypoint having detectors output such as peak (e.g. DoG response), scale and location (coordinate) except orientation

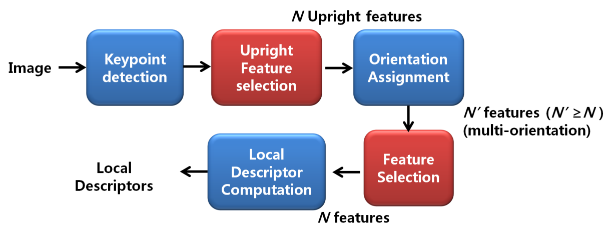


Fig. 3. The proposed feature extraction pipeline based on two-stage feature selection.

– Feature : Keypoint having peak, scale, location and orientation after orientation assignment

As shown in Fig. 3, upright feature selection stage is inserted before orientation assignment. In this step, first, the relevance of upright feature is computed by its peak, scale, location (Eq.(1)), and then, top N upright features with high relevance are selected to compute orientation. With this approach, time complexity can be reduced by selecting relevant upright features in advance of orientation assignment.

The relevance of i_{th} upright feature, $r(u_i)$ is presented by the following equation:

$$r(u_i) = c(p_{u_i})c(s_{u_i})c(l_{u_i}) \tag{1}$$

where p, s, l are peak, scale, and location of upright feature, respectively, and $c(x)$ is the probability of correct match corresponding to x as in Fig. 1.

In orientation assignment stage, N upright features produce N' features. N' is usually bigger than N because there are features with different orientations at the same location and scale. After computing orientation, the relevancies of features are updated for each feature by multiplying the relevance computed in upright feature selection stage and the relevance corresponding to its orientation (Eq.(2)).

$$r(f_{i,j}) = r(u_i)c(o_{i,j}) \tag{2}$$

where j is positive integer, and $f_{i,j}$ is feature u_i with orientation $o_{i,j}$, and $r(f_{i,j})$ is the relevance of $f_{i,j}$.

Finally, N features are selected to extract descriptors. The detail of procedure is presented in Algorithm 1. As we expected from the low variance of probabilities according to orientation shown in Fig. 1, the selected features with the proposed method is very similar to the features selected by [7].

To verify the effect of two-stage feature selection in computational complexity, the average number of upright features, for which orientation is computed, is compared to that of the previous feature selection [7] for 250 images of VGA resolution according to the target number of selected features in Table 1. When the target number of selected feature is 150, orientation is computed only for the

Algorithm 1 Two-stage Feature Selection.

```
Input: $\mathbb{U}$, $N$ ($\mathbb{U}$: detected upright features , $N$: target number of selected features)
Output: selected features $\mathbb{F}$
 1: for each upright feature $u_i \in \mathbb{U}$ do
 2:     $r(u_i) \leftarrow$ relevance of upright feature in Eq.(1)
 3: end for
 4: sort the relevancies of upright features in descending order
 5: $\mathbb{U}_s \leftarrow$ select top $N$ upright features with high relevance in $\mathbb{U}$
 6: for each selected upright feature $u_i \in \mathbb{U}_s$ do
 7:     compute orientations $\mathbb{O}_i$
 8:     for each orientation $o_{i,j} \in \mathbb{O}_i$ do
 9:         $f_{i,j} \leftarrow$ feature $u_i$ with orientation $o_{i,j}$
10:         $r(f_{i,j}) \leftarrow$ relevance of feature in Eq.(2)
11:     end for
12: end for
13: sort the relevancies of features in descending order
14: $\mathbb{F} \leftarrow$ select top $N$ features with high relevance
```

Table 1. Comparison of the average number of upright features in orientation assignment stage.

Target # of selected features	Feature selection [7]	Two-stage Feature selection	Computational improvement
150	913.8	150.4	83.54%
300	913.8	300.3	67.14%
500	913.8	492.1	46.15%
650	913.8	621.4	32.00%

16.46% of keypoints in the proposed method which means large computational saving can be achieved.

3.2 Efficient Partial Gradient Computation

In this subsection, we present an efficient partial gradient computation method. In SIFT, Gaussian smoothed image patch around each keypoint are extracted, and gradient magnitude and angle of each pixel in this patch are used to compute orientation or descriptor. The patches used in orientation and descriptor computation are called orientation patch and description patch, respectively. Orientation patch belongs to descriptor patch.

Considering the complexity of gradient computation, previous SIFT implementations are inefficient. In OpenCV [16], gradients are directly computed for orientation and description patches, so there are unnecessary multiple calculations at the same pixel, because orientation patch belongs to description patch, and usually this patch is overlapped with other features patch. In Vlfeat [19], gradient map is used, which computes the gradients in advance for all pixels of

Table 2. Comparison of complexity in gradient computation.

Target # of selected features	FS+GM	FS+DGC	FS+PGC	TFS+DGC	Proposed
150	1	0.48	0.28	0.26	0.11
300	1	0.67	0.33	0.50	0.20
500	1	0.90	0.39	0.79	0.31
650	1	1.05	0.44	0.98	0.38
Average	1	0.78	0.36	0.63	0.25

scale space images to avoid multiple calculations at the same pixel. However, there are unused areas in gradient map for descriptor computation.

To tackle this problem, proposed method combines two approaches for efficient gradient computation. Gradients are calculated only for the pixels used in orientation and descriptor computation, and updated in the gradient map to avoid multiple calculations. The procedure is as follows:

1. Initialize gradient map to -1
2. When the gradients are required, read the values of gradient magnitude in the map
   (a) If the value is -1, compute gradients and update the map
   (b) Otherwise, use the values

It is obvious that multiple calculations at the same pixel will not occur, because once gradient magnitude is computed, it will have non-negative value, which means that it cannot be eqaul to -1. Its effectiveness is enhanced with two-stage feature selection, which significantly reduces the regions where gradients computation is required for orientation compuation as shown in Fig. 4 (e).

To show the efficiency of partial gradient computation and compare the two-stage feature selection with [7] in terms of computational complexity, we compared five different implementations of SIFT extractor:

- FS+GM : Feature selection [7] and Gradient map
- FS+DGC : Feature selection [7] and Direct gradient computation
- TFS+DGC : Two-stage Feature selection and Direct gradient computation
- FS+PGC : Feature selection [7] and Partial gradient computation
- Proposed : Two-stage Feature selection and Partial gradient computation

For each implementation, the average number of pixels, for which gradients are computed, is counted for 250 images of VGA resolution according to the target number of selected features, and the relative computational complexity with respect to FS+GM is summarized in Table 2. Comparing FS+DGC and FS+PGC, there are many unnecessary multiple calculation at the same pixel in direct gradient computation. Partial gradient computation greatly reduced unnecessary calculation in this case. Comparing FS+PGC and proposed method, the complexity is further reduced when the two-stage feature selection is combined with

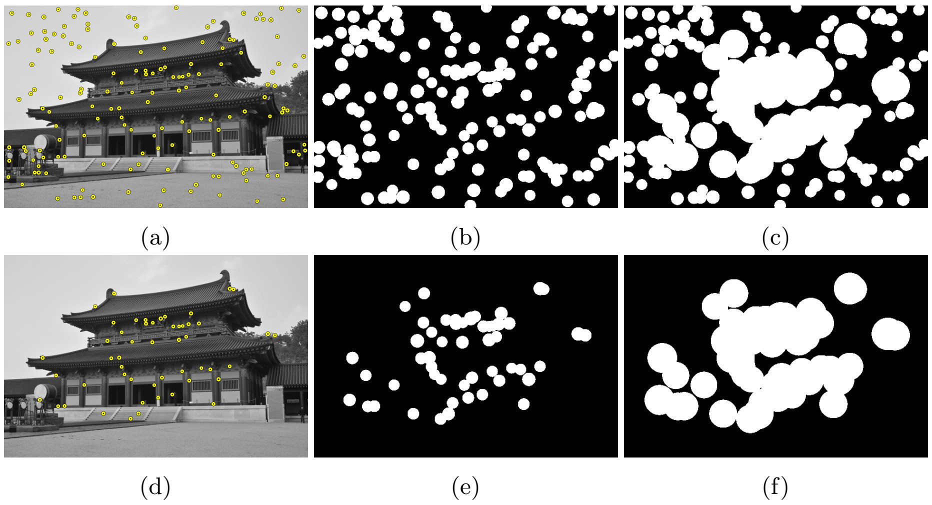


Fig. 4. Partial gradient computation applied to (a)-(c) the previous feature selection [7] and (d)-(f) the two-stage feature selection in 2nd layer of 1st octave. (a) 159 detected upright features by DoG detector. (b) Orientation patches. (c) Orientation and description patches. (d) 52 pre-selected upright features with two-stage feature selection scheme. (e) Orientation patches (f) Orientation and description patches.

partial gradient computation. This is visualized in Fig.4. FS+PGC and proposed method case is shown in Fig.4 (a)-(c), and (d)-(f), respectively.

For simple comparison, only one layer of scale space is considered in this example. By DoG detector, 159 upright features are detected in this layer as shown in Fig.4 (a). Corresponding to these upright features, the white regions in Fig.4 (b) represent orientation patches where gradients are calculated. When 300 features are selected in whole layer by [7], the gradients in description patch are computed and updated for the 61 selected features in this layer as shown in Fig.4 (c). When two-stage feature selection is combined, computational complexity can be further reduced as shown in Fig.4 (d)-(f).

4 Experimental Results

The proposed method is evaluated on uncompressed SIFT framework, and CDVS framework (where descriptors are compressed). The experimental results are presented in Section 4.1 and 4.2, respectively. MPEG CDVS datasets [9] were used in both experiments as described in Table 3. These datasets can be divided into two datasets: 2D planar object (books, CDs, and paintings) dataset and 3D non-planar object (buildings or common 3D objects) dataset.

Table 3. CDVS Dataset used in the experiments.

Name	Categoty	# of images	# of matching pairs	# of non-matching pairs
2D	Graphics	2500	3000	30000
	Paintings	455	364	3640
	Video frames	500	400	4000
3D	Buildings	14935	4005	48675
	Objects	10200	2550	25500

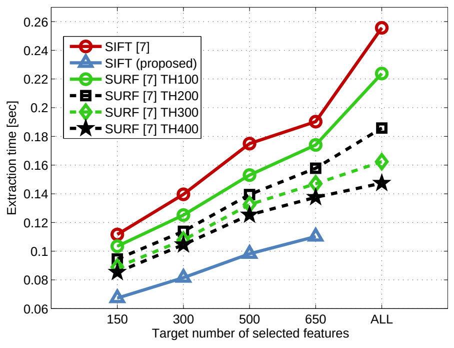
4.1 Evaluation on SIFT Framework

To evaluate the proposed method, we compared the time complexity and matching performance of the proposed method with that of [7] on the SIFT framework. Target number of selected features is varied from 150 to 650 to examine the effect of the proposed method. Moreover, the feature selected SURF based on [7] was examined for comparison. Vlfeat 9.14 [19] and OpenCV 2.4.6 [16] were used to extract SIFT and SURF respectively and several training sets in [4, 8, 17, 18] were used to estimate the conditional probabilities of correct match for each descriptor as a function of scale, orientation, peak, and distance from image center. As peak threshold of SURF influences the number of features, time complexity, and matching performance, SURF with various peak threshold were compared.

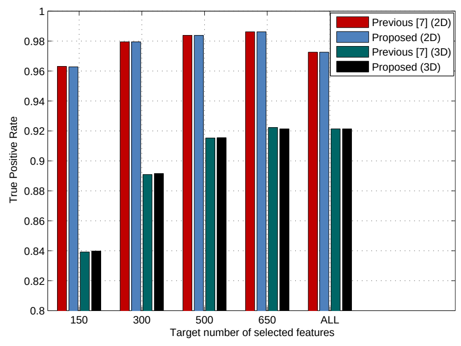
Time Complexity. Time complexity was measured in terms of average extraction time including feature detection and descriptor extraction time. For fair comparison, image loading and resizing time was excluded because different libraries were used for SIFT and SURF, respectively. Average extraction time was measured with i7-2600 @ 3.4GHz PC on 250 randomly selected images with a resolution of VGA. All of the results were measured on single core not to consider speed-up by parallel processing of different implementation.

The results are shown in Fig.5 (a). The extraction time is increased according to the target number of selected features. The SIFT with the proposed method is more than 1.7 times faster compared to SIFT with [7], and 1.4 times faster on average compared to SURF based on [7] with various peak response threshold. When compared to SIFT with all features, the speed-up factor is up-to 3.8 and on average 2.96. While SURF with [7] extracts 150 features, SIFT with proposed method can extract about 500 features. These results tell us that the proposed method significantly improved the time complexity.

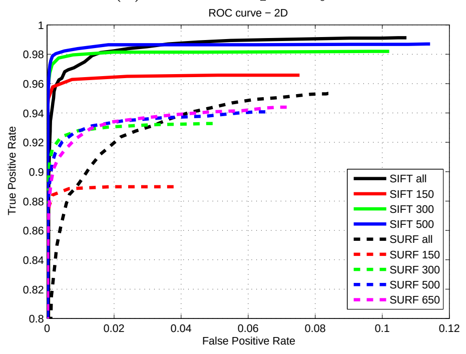
Matching Performance. To verify the influence of the proposed method on matching performance, receiver operating characteristic (ROC) curve and true positive rate (TPR) at 1% of false positive rate (FPR) were compared. Nearest neighbor distance ratio matching scheme based on Euclidean distance was used



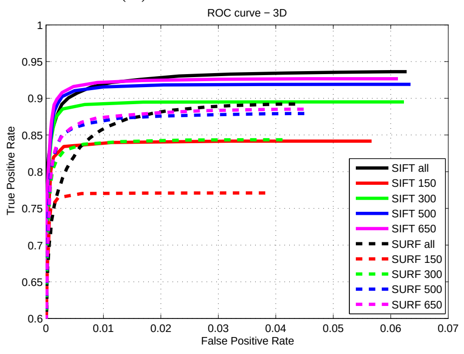
(a) Time complexity



(b) TPR at 1% FPR



(c) ROC curves on 2D dataset



(d) ROC curves on 3D dataset

Fig. 5. Performance Comparisons.

with the ratio of 0.85, and geometrical consistency was checked by DISTRAT [13] with the Chi-square percentile of 95.

As shown in Table 4, peak response threshold of SURF influences the number of features, and affects the time complexity and matching performance. Considering the trade-off between complexity and performance, 200 was used in the following experiments for the threshold of SURF.

In Fig.5 (b), the proposed method and [7] on SIFT are compared in terms of TPR at 1% FPR for 2D and 3D dataset according to the target number of selected features. The performance of the proposed method is almost the same as that of [7]. It is noticeable that the proposed method significantly improved the time complexity without affecting the matching performance.

ROC curves of feature selected SIFT and SURF are compared in Fig.5 (c-d) These curves were drawn by changing the threshold of the number of the inlier matching pair in DISTRAT [13]. From the results, it is shown that the matching performance of 300 selected features for 2D and 500 selected features for 3D is comparable to that of all features both on SIFT and SURF, and even outperform at low FPR. This is because the irrelevant features from background or clutters degrade the matching performance while increasing descriptor length.

Table 4. The influence of peak response threshold of SURF.

	Threshold	# of features	Extraction time (sec)	TPR in 3D @ 1% FPR
SIFT	0	1115.5	0.2556	0.921
SURF	100	1575.5	0.2238	0.872
	200	1229.0	0.1858	0.862
	300	1025.3	0.1624	0.844
	400	886.3	0.1474	0.831

Moreover, it is noticeable that SIFT with 150 features for 2D dataset and 300 features for 3D dataset outperform SURF with all features. This tells us that SIFT with the proposed scheme for 2D dataset is 2.8 times (2.3 times for 3D) faster in extraction time than SURF while using 25% (40% for 3D) of descriptor length and still outperforming in the matching performance compared to SURF.

4.2 Evaluation on MPEG CDVS Framework

The proposed method has been integrated into CDVS TM as an fast mode of feature extractor for its efficiency since 107th meeting. In current CDVS TM11 [10, 11], there are four different modes on feature extraction. A Low-degree Polynomial (ALP) detector [6] and SIFT descriptor are baseline of feature extraction, and its three variants :

- ALP low memory mode [1] : ALP detector with block-based processing and spatial domain filtering
- ALP fast mode [12] : fast extraction mode of ALP detector (Proposed method)
- ALP BF [5] : ALP detector with block-based processing and frequency domain filtering

ALP detector approximates the LoG filtering by polynomials to find keypoints of image.

In Table 5, four different feature extractors are compared. For ALP, ALP low memory, ALP BF cases, the previous feature selection method [7] is used. However, features are selected after SIFT descriptor extraction in ALP low memory, ALP BF for their block-based processing to acheive low memory usage.

For gradient computation, ALP and ALP low memory mode use gradient map, which computes the gradients in advance for all pixels, and ALP BF compute gradient directly for each detected keypoint where multiple calculations at the same pixel can exist.

In fast mode of ALP, the proposed method was integrated into ALP detector and SIFT descriptor. Because the proposed method speed up the computation after keypoints are detected, it could be easily integrated into ALP detector framework.

Table 5. Comparison of TM11 feature extractors.[2]

	ALP	ALP Low memory	ALP Fast mode	ALP BF
Feature selection	[7]	[7]	TFS	[7]
Gradient Computation	GM	GM	PGC	DGC
Average extraction time (sec)	0.1385	0.2689	0.0915	0.1374
Speed-up Factor	1	0.5149	1.5130	1.0076
Max Memory usage (512B)	28.2 MB	21.0 MB	24.3 MB	21.2 MB
Max Memory usage (16KB)	28.3 MB	22.2 MB	25.7 MB	21.0 MB

In this section, we compare the time complexity, memory usage, and mathcing and retrieval accuracy of the proposed method (ALP fast mode) with three other modes of feature extractor on MPEG CDVS framework.

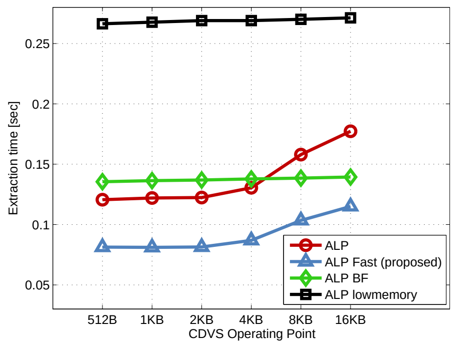
Experimental Setup. Evaluation procedure follows the evaluation framework for CDVS [9]. Descriptors are extracted at six operating point (512B, 1KB, 2KB, 4KB, 8KB, and 16KB), and the target number of selected features are 250 (512B, 1KB, 2KB), 300 (4KB), 500 (8KB), and 650 (16KB), respectively. For feature selection, the conditional probabilities of correct match in TM11 were used. MPEG CDVS datasets [9] were used in matching and retrieval experiments as described in Table 3. In retrieval experiment, 7814 images were queried to retrieve relevant images in a dataset of 1 million images including distractor set collected from Flicker.

Time Complexity. Time complexity was measured for each operating point of CDVS. The other condition is the same of Section 4.1 Time Complexity, but extraction time measures whole processing time including image loading, resizing and descriptor encoding in this case, because all of TM11 feature extractors are built on the same framework.

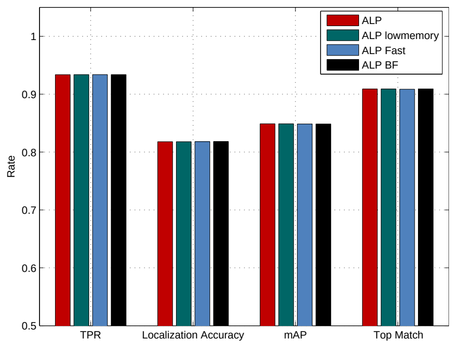
The results are shown in Fig.6 (a), and average extraction time for six operating points and speed-up factor compared to ALP are shown in Table 5.

For ALP BF and ALP low memory cases, the extraction times over six operating points are nearly constant because that features are selected after SIFT extraction for their block-based processing to achieve low memory usage. For

[2] TFS : Two-stage feature selection, GM : Gradient map.
PGC : Partial gradient computation, DGC : Direct gradient computation.



(a) Time complexity



(b) Matching and Retrieval accuracy

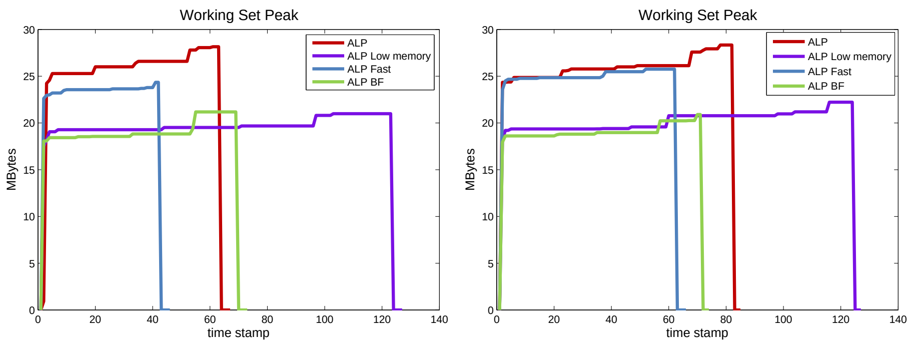
Fig. 6. Comparison of feature extractors of CDVS TM11.

ALP and the proposed method (ALP Fast mode), the extraction time is increased according to the target number of selected features (250 features are selected at 512B, 1KB, 2KB).

The proposed method outperforms the others in terms of extraction time at every operating point. Especially, the proposed method at 16KB, where 650 features are selected, is faster than other methods at 512B, where only 250 features are selected. On average, the proposed method is more than 1.5 times faster compared to ALP and ALP BF, and 2.94 times faster compared to ALP low memory mode. From these results, it is verified that the proposed method significantly improved the time complexity of feature extraction in CDVS TM. Morover, in [2], they examined ALP, ALP low memory and ALP fast mode (proposed method) on ARM based device (LG Nexus5) and reported the proposed method is not only faster than the other methods, but also consumes the lowest energy (about 0.75 $\sim$ 0.95 J per image) because of its shorter extraction time.

Memory Usage. In this section, memory usages of the proposed method (ALP fast mode), and other ALP variants of TM11 in feature extraction are examined by profiling with their binaries. All the binaries of four different feature extractor were compiled in 64bit release mode, and Visual Studio 2010 performance wizard in CPU sampling method was used as profiling tool. While extracting feature of 250 randomly selected VGA images at 512B and 16KB operating point, the actual memory used by the extraction binaries were measured by Working Set Peak memory counter on i7-2600 @ 3.4GHz / 16GB RAM PC. The definition of Working Set Peak is as follows from [15]:

– Working Set : The current number of bytes in the working set of the process. The working set is the set of memory pages touched recently by the threads in the process. It includes both shared and private data.
– Working Set Peak : The maximum size, in bytes, in the working set of the process at any point in time.



(a) 512B (b) 16KB

Fig. 7. Memory Profiling Results.

Profiling results are shown in Fig. 7. In this figure, both of extraction speed and memory usage can be compared. Memory usage of ALP BF and ALP Low memory was lower than the other methods at 512B, 16KB operating point both. This is because two methods are based on block-wise processing. The maximum of working set peak in this figure is presented in Table 5. The proposed method uses 3.1 MB and 4.7 MB more than ALP BF at 512B and 16KB, respectively. Even though, the proposed method is based on whole-image processing, the memory usage difference with that of feature extractor based on block-wise processing is not huge. It is because two-stage feature selection is helpful to keep less features in feature extraction process.

Matching and Retrieval Performance. Matching and retrieval performance of the proposed method, and other feature extractors of TM11 are compared on MPEG CDVS evaluation framework. For pair-wise matching performance, true positive rate (TPR) at a fixed false positive rate (1%), and localization accuracy were measured. Mean average precision (mAP) and success rate for top match were measured for retrieval performance. All of measures were measured for each dataset at each operating point, and their average values are compared.

Experimental results are shown in Fig.6 (b). It shows that the performance is similar for all methods.

5 Conclusion

In this paper, a fast feature extraction method based on two-stage feature selection and efficient partial gradient computation is proposed. SIFT with the proposed method is compared to SIFT and SURF with [7]. With the proposed scheme, feature selected SIFT is not only much faster than SURF, but also outperforms in terms of matching accuracy.

Also, the proposed method applied to ALP detector is compared to other feature extraction methods of TM11 on CDVS evaulation framework. Experimental results show that the proposed method significantly reduces the time complexity while maintaining the matching and retrieval performance of TM11. Moreover, the proposed method are compared to ALP and ALP low memory on ARM-based device (LG Nexus5) in [2]. From the results of [2], the proposed method is not only faster than the other methods, but also consumes the lowest energy (about $0.75 \sim 0.95$ J per image) because of its shorter extraction time.

From the results, we can conclude that the proposed method significantly reduces the time complexity of feature extraction without affecting the matching and retrieval accuracy, and this method will be useful for fast extraction of the high accuracy feature on mobile devices, where the use of computational resource is limited.

For further work, the proposed method can be easily extended to SURF framework, and time complexity can be further reduced when combined with parallel processing or GPU implementation. Memory usage can be also improved considering block-based processing.

Acknowledgement. This work was supported by the ICT R&D program of MSIP/IITP. [2014(R2012030111), Development of The Smart Mobile Search Technology based on UVD(Unified Visual Descriptor)]

References

1. Balestri, M., Francini, G., Lepsøy, S., Lee, K. D., Na, S. I., Lee, S. J.: CDVS: ETRI and TI's response to CE1 - "An invariant low memory implementation of the ALP detector with a simplified usage interface". 107th MPEG Meeting, M31987 (2014)
2. Ballocca, G., Mosca, A., Fiandrotti, A., Mattelliano, M.: CDVS: TM10 Extraction Evaluation on ARM Architectures. 109th MPEG Meeting, M34086 (2014)
3. Bay, H., Ess, A., Tuytelaars, T., Van G. L.: Speeded-up robust features (SURF). Computer Vision and Image Understanding **110** (2008) 346–359
4. California Institute of Technology: Pasadena Buildings 2010 dataset. `http://www.vision.caltech.edu/archive.html` (accessed 2014)
5. Chen, J., Duan, L. Y., Huang, T., Gao, W., Kot, A. C., Balestri, M., Francini, G., Lepsøy, S.: CDVS CE1: A low complexity detector ALP BFLoG. 108th MPEG Meeting, M33159 (2014)
6. Francini, G., Balestri, M., Lepsøy, S.: CDVS: Telecom Italia's response to CE1 - interest point detection. 106th MPEG Meeting, M31369 (2013)
7. Francini, G., Lepsøy, S., Balestri, M.: Selection of local features for visual search. Signal Processing: Image Communication **28** (2013) 311–322
8. Hervé, J., Matthijs, D., Cordelia, S.: Hamming embedding and weak geometric consistency for large scale image search. European Conference on Computer Vision **1** (2008) 304–317
9. ISO/IEC JTC1/SC29/WG11: Evaluation Framework for Compact Descriptors for Visual Search. 97th MPEG Meeting, N12202 (2011)
10. ISO/IEC JTC1/SC29/WG11: Study text of ISO/IEC DIS 15938-13 Compact Descriptors for Visual Search. 109th MPEG Meeting, N14681 (2014)

11. ISO/IEC JTC1/SC29/WG11: Test Model 11: Compact Descriptors for Visual Search. 109th MPEG Meeting, N14682 (2014)
12. Lee, K. D., Na, S. I., Lee, S. J., Balestri, M., Francini, G., Lepsøy, S.: CDVS: ETRI and TI's Response to CE1 - A fast feature extraction based on ALP detector. 107th MPEG Meeting, M31991 (2014)
13. Lepsøy, S., Francini, G., Cordara, G., de Gusmao, P. P. B.: Statistical modelling of outliers for fast visual search. IEEE International Conference on Multimedia and Expo (ICME) (2011) 1–6
14. Lowe, D. G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60** (2004) 91–110
15. Microsoft: Counters in Process performance object. `http://msdn.microsoft.com/en-us/library/ms804621.aspx` (accessed 2014)
16. OpenCV Library: `http://opencv.org/` (accessed 2014)
17. Telecom Italia: 201 Books, InternetArchive and DistractorPairs dataset. `http://pacific.tilab.com/` (accessed 2013)
18. University of Oxford: The Oxford Buildings Dataset. `http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/` (accessed 2014)
19. VLFeat Library: `http://www.vlfeat.org/` (accessed 2014)