K-054

Task Level SURF Keypoint Extraction Parallelization in AR Application

H.A. Dang†, Pao Sriprasertsuk†, Wataru Kameyama†

1. INTRODUCTION

In the last several decades, following the development of technology, information producing and consuming have become easier than ever before. With the incredible speed of information grown, we will soon face a new issue of information flood in which Augmented Reality (AR) is one of the most promising solutions to resolve the problem.

Limited by computation resource, large amount of common AR systems are featured by simple algorithms to meet the requirement of lightweight and real time image processing (marker recognition is a typical example). For that reason, AR application is usually constrained within managed environment.

On the other hand, advanced image recognition algorithms such as SIFT [1] (Scale-invariant feature transform) or SURF [2] (Speed-Up Robust Feature) require enormous processing capability to carry out similar tasks as an extra cost for flexibility.

Reached the barrier of material technology, hardware innovation in recent years turns into another direction that promotes the use of parallel computing. Therefore, real-time (low latency) object recognition by traditional implementation of SIFT or SURF is still not possible, even with the latest hardware.

1.1 SIFT and SURF

SIFT is the most well-known local descriptor algorithm ever since its publication in 1999. Inspired by SIFT, SURF was published by Herbart Bay in 2008 with greater performance compared to SIFT in both accuracy and computational efficiency. Even though the methodology implemented in SURF is totally different from SIFT's, they are sharing the same principle of scale space construction.

1.2 Task Parallelization and Expectation

Mentioned above, SURF requires constructing multiple scale spaces in order to identify and extract keypoint at different scale. Scale spaces in SURF are constructed by up-scaling box filter, which results lower and constant computation cost (compared to SIFT, in which scale spaces are constructed by recursively down-sampling original image).

Searching for keypoints across multiple scale spaces increases in-variance effectiveness of SURF. However, as the down side of this process, it results more complexity for parallelization. Especially on task level parallelism, where smaller parts of image are processed simultaneously, keypoints distortion and loss are expected on lower scale space, around slice's boundaries. Extra safe margin can only preserve keypoints on top levels of scale space.

† GITS, Waseda University

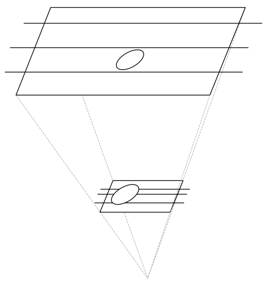

Figure 1: Keypoints are Loss at Smaller Scale Space

Figure 1 demonstrates how keypoints are loss in this parallelization methodology. As smaller parts of original image are processed separately by parallel processer, keypoints cannot be extracted form smaller scale space if they lie over slice's boundaries. Even in some case, slice width is not wide enough to extract descriptor.

In AR application, keypoints extracted from query images (usually a camera feed) are compared with a pre-set database. For that reason, it's not essential to preserve keypoints from smaller scale space of query images.

2. ANALYSIS

As keypoint distortion and loss is expected at smaller scale spaces. The focuses of this analysis are keypoint consistency and system performance. A set of 50 SVGA size images (800x600) has been used as sample dataset. Parallel process is done by extract SURF keypoints from smaller slices of the image simultaneously.

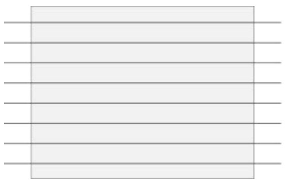

Figure 2: Image Slice Map

In this analysis, each sample image is sliced vertically as shown in Figure 2. The number of slices is varying from 2 to 16. 30px wide margin is applied to preserve top scale-space level keypoints.

2.1 Keypoint Consistency

To verify the keypoint consistency, two sets of keypoints extracted by non-parallel and parallel process of each image are compared together to determine the number of missing keypoints and keypoint distortion.

The percentage of loss keypoints is easily calculated by the following equation:

Copyright © 2011 by Information Processing Society of Japan and
The Instiute of Electronics, Information and Communication Engineers
All rights reserved.

$$p_j = \frac{1}{n}\sum_{i=1}^{n}(1 - \frac{k_{ji}}{k_{1i}}) \quad with \quad 2 \le j \le l \qquad (1)$$

Where p_j is percentage of keypoint loss as the result of j parallel process, n is the number of samples, k_{ji} is number of keypoint extracted by j parallel processes from sample i, and j is total number of sample.

Besides keypoints loss, there are also a significant number of keypoint distortions, due to the process of octaves construction in each scale space. A simple matching process between two keypoint sets can determine whether a keypoint is distorted and the level of distortion. Thus, the percentage of distorted keypoint is calculated by similar formula as below:

$$o_j = \frac{1}{n}\sum_{i=1}^{n}(1 - \frac{d_{ji}}{k_{1i}}) \quad with \quad 2 \le j \le l \qquad (2)$$

Where o_j is the percentage of distorted keypoints and d_{ji} is the number of distorted keypoint of image i resulted by j parallel process. From (1) and (2), keypoint consistency is calculated by the following formula:

$$kpc_j = 1 - o_j - p_j \qquad (3)$$

Where kpc_j is the average keypoint consistency of sample data set extracted by j parallel processes.

No. of Processes	2	3	4	5	6	7	8	9	10	11	12
KP loss (%)	2	4	6	8	9	11	12	13	14	15	15
KP Distortion (%)	30	44	49	52	55	57	59	61	62	63	65
Acceptable Distortion (%)	29.9	43.8	48.2	51.7	54.3	55.5	57.5	60.1	60.7	62.0	63.8
KPC (%)	70	56	51	48	45	43	41	39	38	37	35
Acceptable KPC (%)	98	95	94	92	91	88	86	87	85	84	84

Table 1: Keypoint Consistency Experiment Result (%)

However, as Table 1 demonstrated, the keypoint consistency (KPC) is remarkably low as the number of processes (or slices) is increased. It is apparently contradicted to actual experiment which archived high accuracy under different environment conditions. A further study on keypoint distortion was conducted by constructing a frequency histogram of distance between distorted keypoint and original keypoint.

As shown in Figure 2, the degree of keypoint distortion is acceptable in most cases. A threshold of 0.1 is chosen to determine whether a distorted keypoint is acceptable. As the result, the acceptable KPC is remarkably higher than absolute-matched KPC and suitable for AR application implementation.

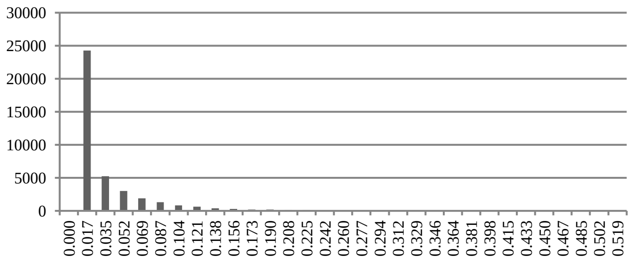

Figure 3: Keypoint Distortion Distribution (16 parallel processes)

It is necessary to note that the result of this experiment is only true for described sample size (SVGA). Other size of sample and slicing method will produce different KPC.

2.2 System Performance

System performance analysis is relatively simpler than KPC analysis. The process is done by record processing time of extraction process and each sub-process. The machine used to conduct this test is equipped with i7-920CPU (4 physical cores) running at 2.66GHz and 3.88GHz with 12GB RAM. Even though parallel processing test is feasible on multicore processor, as all cores are sharing same cache, the performance improvement is expected to be lower as the number of parallel process reach the number of real cores.

	Processes	1	2	3	4	5	6	7	8	9	10	11	12
2.66	Observed	694	390	287	233	215	189	173	173	167	168	164	160
	Total Time	694	667	649	665	714	742	747	836	895	1020	1018	1034
3.88	Observed	459	257	186	149	136	117	114	108	103	104	97	98
	Total Time	459	439	421	433	428	450	466	492	517	540	569	590

Table 2: Running Time of Single and Parallel Process

Since test system has only 4 physical cores, the result is fairly accurate only from 2 to 4 parallel processes. Result of performance testing is shown on Table 2. The total processing time of all parallel process also calculated to contrast the efficiency of real parallel process with time-division process.

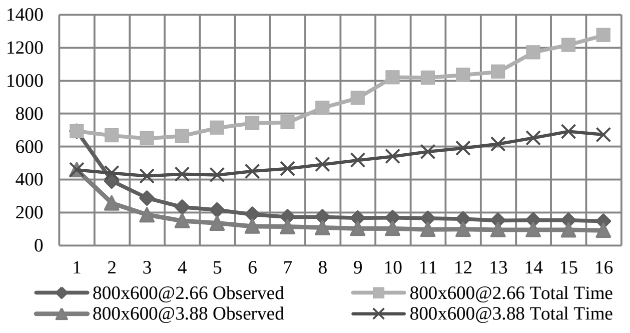

Figure 4: Parallel Processing Time

As shown in Figure 4, the processing time is only improved within 2 and 4 parallel processes. No significant improvement observed as the number of parallel process excessing 4. This result shows that in most cases, the optimum number of parallel process should be equal or less than the number of real core.

3. CONCLUSION

This paper has introduced SURF keypoint extraction parallelization at task level. Because of keypoint loss and distortion at smaller scale space, this methodology is only appropriate for query image's keypoint extraction in application like AR. Pre-set database must be processed by standard process or better parallelization methodology to assure the scale in-variant nature of SURF. Furthermore, constructing extra scale space for keypoint extraction on sample data set would improve accuracy at multi scale.

Finally, both KPC and performance analysis results in this paper cannot be used to determine optimum number of parallel process for other system, dataset and input. Separate analysis is required to determine an optimum solution for specific circumstance.

REFERENCE

[1]. Lowe, David G. "Object recognition from local scale-invariant features". Proceedings of the International Conference on Computer Vision. 2. pp. 1150–1157, 1999

[2]. Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008

Copyright © 2011 by Information Processing Society of Japan and
The Instiute of Electronics, Information and Communication Engineers
All rights reserved.