Parallel Scale Space Construction using SIMD Hypercube

Anukul Chandra Panda, Hunny Mehrotra and Banshidhar Majhi
Department of Computer Science and Engineering,
National Institute of Technology Rourkela,
Odisha 769008, INDIA
Email: {pandaa,hunny,bmajhi}@nitrkl.ac.in

***Abstract*—This paper proposes parallel scale space construction of Scale Invariant Feature Transform (SIFT) using SIMD hypercube. The parallel SIFT approach is used for iris feature extraction. The input iris images and Gaussian filters are mapped to each processor in the hypercube and convolution takes place in each processor concurrently. The time complexity of parallel algorithm is $O(N^2)$ whereas sequential algorithm performs with complexity of $O(lsN^2)$, where l is the number of octaves, s is the number of Gaussian scale levels within an octave for N^2 sized iris image.**

I. Introduction

Biometrics is the process of recognising an individual using physiological and behavioral characteristics. Among various available biometric traits, iris is most reliable trait due to its robustness [1]. A generic iris biometric system involves preprocessing, feature extraction and matching. Feature extraction involves simplifying the amount of information required to describe an iris image. The purpose is real time, high confidence recognition of an individual's identity by mathematical analysis of the random patterns that are visible within the iris from some distance. There are several feature descriptors like Scale Invariant Feature Transform (SIFT) [2], PCA–SIFT [3], Speeded Up Robust Features (SURF) [4], etc. which have served brilliantly in describing the features within the iris. Among available feature extraction techniques, SIFT outperforms its counterparts due to high accuracy, reliability and invariance to blurring [5].

SIFT has three different phases namely, keypoint detection, keypoint descriptor assignment and keypoint pairing. In keypoint detection phase, there are various steps like detection of scale space extrema, keypoint localization and orientation assignment. It has been observed that there exists an inherent parallelism during scale space construction. A parallel algorithm for this computationally intensive step has been developed using SIMD hypercube parallel architecture. An asymptotic time complexity is obtained for serial and parallel implementation. The sequential phases involved in SIFT are discussed in Section II. Section III describes the proposed parallel scale space construction algorithm. The comparative asymptotic analysis is discussed in Section IV. Conclusions are given at the end of the paper.

II. Conventional SIFT

SIFT is a keypoint detector and feature descriptor scheme. Owing to its advantages SIFT could find its applicability in biometrics [6], [7]. The local features from an iris image are computed using cascade filtering approach that minimizes the feature extraction cost by applying more expensive operations at locations that pass an initial test. The feature vector is generated by performing the phases as outlined in the following subsections:

A. Keypoint Detection

The keypoint detection begins with finding potential keypoints that are invariant to scale and orientation. For each detected keypoint the location and scale is determined. The steps involved in keypoint detection are outlined with explanation in the following subsections.

1) Detection of Scale Space Extrema: The first step of keypoint detection is to find the positions and scales that can be iteratively assigned under different viewing of same object. Identification of locations consistent to scale change of the image can be found by exploring stable features across different scales using a continuous function of scale known as *scale space* [2]. The only possible scale space mask is the Gaussian function. To define the scale space, input iris image (I) is convolved with Gaussian kernel $G(x, y, \sigma)$ as defined by

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{1}$$

where $*$ is the convolution operation and σ defines the width of Gaussian filter. The smoothed images, $L(x, y, \sigma)$ is obtained by convolving the Gaussian kernel with the iris image. The Difference of Gaussian (DOG) images are computed from two nearby scales differentiated by constant multiplicative factor k

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{2}$$

The scale space is divided into various *octaves*. An octave is represented by a series of smoothed image, L which is obtained by convolving the iris image with Gaussian with different values of σ varied by a constant factor. The total number of Gaussian scale levels within an octave are denoted by s. A constant number of Gaussian scale levels are found in each octave. The subsequent octave is obtained by downsampling

978-1-4673-4805-8/12/$31.00 ©2012 IEEE

the input iris image size by half and generating the Gaussian scale levels using equation (1). This step is iterated for l such octaves. The serial scale space construction algorithm is given in Algorithm 1. This technique is found to be conducive for annular iris images since the size of iris is viable to change owing to pupil contraction and expansion. The parallelism can be exploited both within octave (Gaussian scale level computation) and across octave simultaneously.

Algorithm 1: Serial Scale Space Construction

```
Result: Serial Computation of Smoothed Image $L$
1 for each octave $l$ do
2     for each scale $s$ do
3         Convolve input image with Gaussian kernel
4     Downsample the input image by half
```

Firstly, the calculation of the smoothed image, L (Gaussian scale levels) within the octave are found with changed values of σ. Each Gaussian scale level is independent of the other since the smoothed image is found for varied values of σ and hence, L is found in parallel using different processors. The same operation is carried out for change in scales within all l octaves. Secondly, the parallelism can also be introduced across octaves. The present octave always receives the iris image with half the image size of the previous octave. This dependency can also be avoided by performing parallel computation for l octaves.

2) Keypoint Localization: DOG iris images which are found in different octaves are used to detect landmark points with the help of local maxima or minima across different scales. Each sample point in DOG image is compared to eight neighbors in current image and nine neighbors in the scales (above and below). The sample point is chosen as a candidate keypoint that is either a local maxima or minima in $3 \times 3 \times 3$ regions at current and adjacent scales.

3) Orientation Assignment: In the previous step, an effort has been made to make features invariant to scale. However, in order to mitigate the effect of rotation, orientation assignment is used. In this step, local image gradient is used to assign one or more orientations to the keypoints. A gradient orientation histogram is computed in the neighbourhood of keypoint to determine keypoint orientation. Once the histogram is obtained, the orientations corresponding to the highest peak and local peaks that are within 80% of the highest peaks are assigned to the keypoint.

B. Keypoint Descriptor

Once orientation has been selected, the feature descriptor vector for each keypoint is computed such that the descriptor is highly distinguishing and partly consistent even under change in illumination, viewpoint, etc. Initially, a set of orientation histograms on 4×4 pixel neighborhoods are created. The orientation histograms are relative to the keypoint orientation. The histogram contains 8 bins each and each descriptor contains an array of 16 histograms around the keypoint. This generates SIFT feature descriptor of $4 \times 4 \times 8 = 128$ elements.

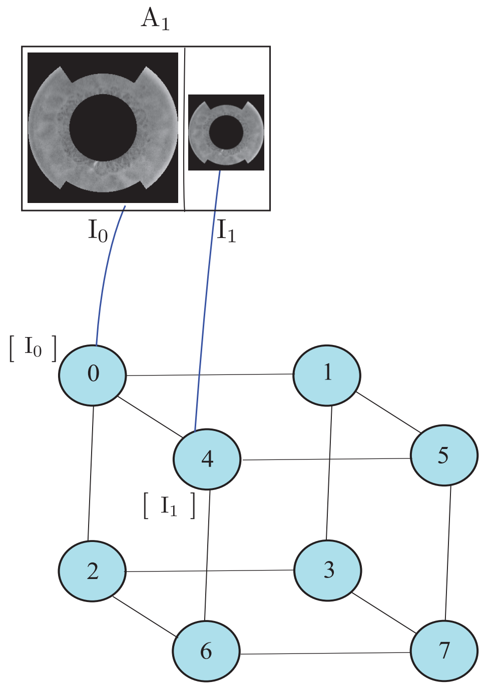
III. Parallel Scale Space Construction using SIMD Hypercube

During scale space creation, there are various octaves which divides the scale space. Each octave contains different Gaussian scale levels. Each Gaussian scale level produces smoothed image, L by the convolution of iris with Gaussian kernel at a particular scale (σ). For instance, the L_i^{th} smoothed iris image is obtained by the convolution of input iris image (I_j) with Gaussian kernel (G_i) at scale (σ_i), where $0 \leq i \leq s-1$ and $0 \leq j \leq l-1$. This same operation is iterated across different octaves to obtain L . Octave$_{j+1}$ receives the input image size that is half the input iris image size of Octave$_j$. The size of iris image and the number of octaves is always known a priori to execution. So, the inter–dependency between the octaves is eliminated by storing the different sizes of iris image in an array.

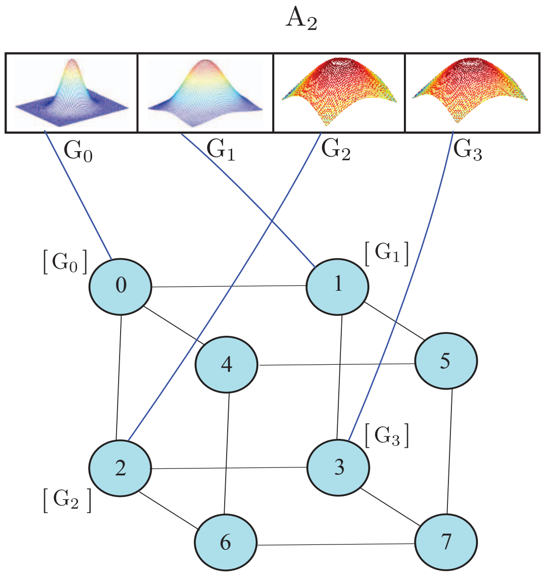
The $(p+1)^{th}$ position stores half the iris image size stored in p^{th} position of the array (A_1) and the Gaussian kernels for change in σ are stored in array (A_2). So, A_1 stores l versions of iris images based on the number of octaves. Similarly, s Gaussian filters with different σ values are stored in A_2 based on the number of Gaussian scale levels (s). The states of array A_1 and A_2 for two octaves and four Gaussian scale levels within the octave are shown in Fig. 1.

The scale space representation with s Gaussian scale levels for l octaves can be constructed using an ls–processor hypercube (assume, $ls = 2^x$). The application of hypercube facilitates in achieving both the parallelism simultaneously. The iris image labeled as j is mapped to the $((j+1)s-s)^{th}$ processor of the hypercube. Similarly, i^{th} Gaussian filter is mapped to the i^{th} processor of the hypercube. Fig. 2 shows the mapping of iris image and Gaussian filters for two octaves and four Gaussian scale levels within octave. The content of $((j+1)s-s)^{th}$ processor of the hypercube is broadcast to processors spanning from $((j+1)s-(s-1))$ to $((j+1)s-1)$ using recursive doubling. The broadcast takes $\log_2 s$ steps. Similarly, i^{th} Gaussian filter present in i^{th} processor is broadcast to the processor with empty local memory having label of the processor as i after performing the operation $i.e.$, $(i \ \% \ s)$. The broadcast takes $\log_2 l$ steps using recursive doubling mechanism.

The broadcast of images and Gaussian kernels among the processors are shown in Fig. 3. The 0^{th} iris image is broadcast to 1^{st} processor as shown in Fig. 3(a). Later 0^{th} and 1^{st} processor concurrently broadcast their content to 2^{nd} and 3^{rd} processors as shown in Fig. 3(b). The same communications can be performed for other images simultaneously in other octaves. The communication of Gaussian filters across different processors can performed in similar way as shown in Fig. 3(c). These communications across processors take 2 ($i.e.$, $\log_2 4$) and 1 ($i.e.$, $\log_2 2$) steps for broadcasting images and Gaussian kernels respectively. Fig. 3(d) shows the final configuration of the processors. Since each processor contains iris image, I



(a) Iris images



(b) Gaussian filters

Fig. 2. Mapping of iris images and Gaussian filters to the hypercube

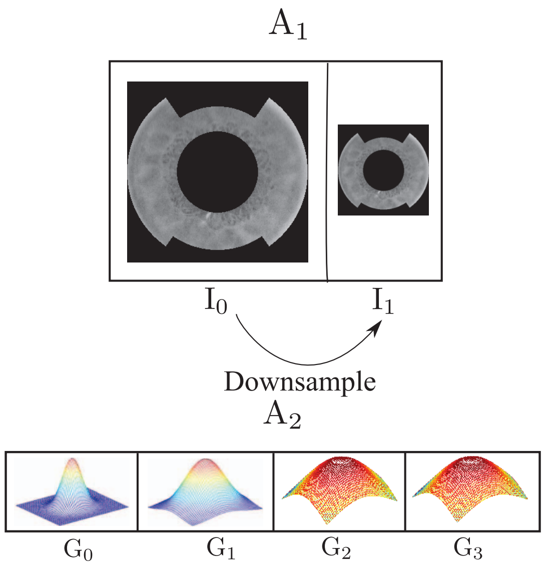


Fig. 1. A_1 and A_2 storing iris images with different sizes and Gaussian filters with different σ values, respectively

and the Gaussian filter, G, hence, the convolution take place in each processor concurrently yielding different smoothed image L. After obtaining the smoothed images they are stored in an $l \times s$ matrix and from which the DOG images are found. Algorithm 2 outlines the steps involved in parallel scale space construction.

Algorithm 2: Parallel Scale Space Construction

```
   Result: Parallel Computation of Smoothed Image, $L$
1  Store the iris images in array $A_1$
2  Store the Gaussian filters in array $A_2$
3  Map $j^{th}$ iris image to $((j+1)s - s)^{th}$ processor of the hypercube
4  Map $i^{th}$ Gaussian filter to $i^{th}$ processor of the hypercube
5  if $y \leq ((j+1)s - 1)$ and $y \geq (j+1)s - (s+1)$ then
6      Perform broadcast of iris image at $((j+1)s - s)^{th}$ to $y^{th}$ processor using recursive doubling
7  if $(i \bmod s) == i$ then
8      Perform broadcast of the $i^{th}$ Gaussian filter to the $(i \bmod s)^{th}$ processor using recursive doubling
9  Convolve iris image with Gaussian filter in parallel in the hypercube
10 Store the calculated smoothed image, $L$ in a $l \times s$ matrix
```

IV. Asymptotic Analysis

A. Serial Scale Space Construction

The construction of the octaves involves the following steps:

1) The calculation of smoothed iris image, L involves a convolution at a value σ. The convolution generally takes a time complexity of $O(N^2)$, where $N \times N$ is the size of the input iris image.
2) The calculation of s Gaussian smoothed iris image, L take place at different values of σ which is changed by a constant factor. So, the time complexity now becomes

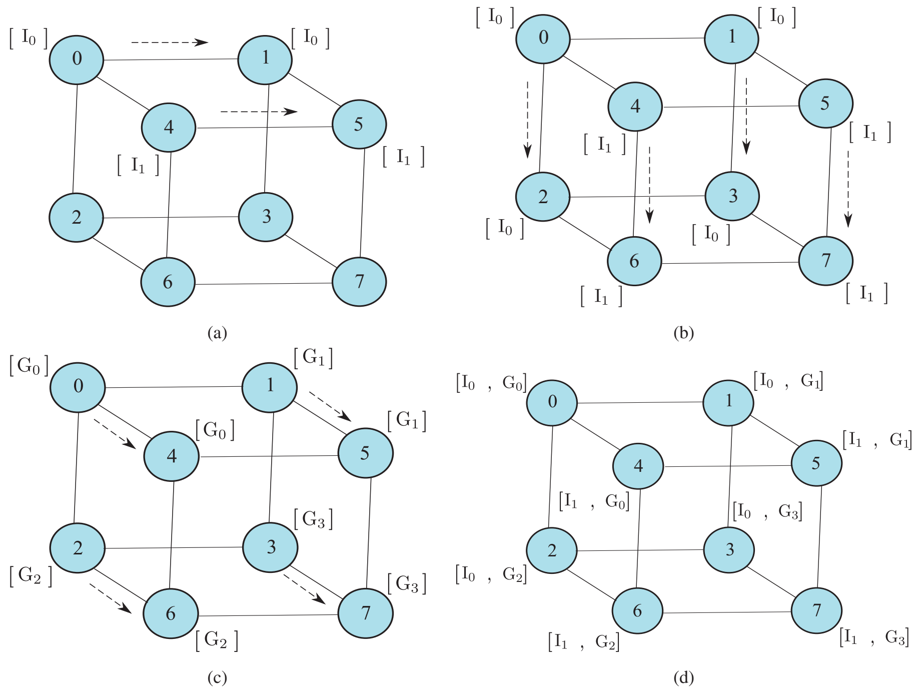


Fig. 3. (a)-(b) Broadcast of different versions of iris images (c) Broadcast of Gaussian kernels, (d) Final configuration of hypercube

$O(sN^2)$.

3) The above two operations are iteratively applied across l octaves.

Hence, the total time complexity of these serial operations denoted as T_{serial} is $O(lsN^2)$ i.e., $T_{serial} = O(lsN^2)$

B. Parallel Scale Space Construction

The time complexity of parallel octave creation with l octaves and s scale levels using SIMD hypercube is given as

$$\begin{aligned} T_{parallel} &= \log_2 l + \log_2 s + N^2 \qquad (3) \\ &= O(N^2) \end{aligned}$$

The various terms of the above equation are,

1) $\log_2 l$ is number of computations to broadcast the iris images
2) $\log_2 s$ is the number of computations to broadcast the Gaussian function
3) N^2 denotes the number of computations to perform convolution operation in each of the hypercube in parallel

Speedup is given as,

$$speedup = \frac{T_{serial}}{T_{parallel}} = \frac{O(lsN^2)}{O(N^2)}$$

V. Conclusions

In this paper, SIFT scale space construction has been made parallel for iris feature extraction using ls–processor SIMD hypercube where l denotes the number of octaves and s represents the number of Gaussian scale levels within each octave. This is possible due to the independence of different scale levels within the octave and the independence of various octaves. This has enabled a speedup of ls between parallel and serial scale space construction. Further, efforts can be made to minimise time required to detect keypoints using SIFT.

Acknowledgment

The authors are thankful to the members of Image Processing and Cluster Computing Lab, National Institute of Technology Rourkela, India for providing critical reviews to enhance the quality of this work.

References

[1] J. Daugman, "The importance of being random: statistical principles of iris recognition," *Pattern Recognition*, vol. 36, no. 2, pp. 279 – 291, 2003.
[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[3] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors." in *Computer Vision and Pattern Recognition (2)*, 2004, pp. 506–513.
[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008.
[5] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615 –1630, oct. 2005.
[6] H. Mehrotra, B. G. S., B. Majhi, and P. Gupta, "An efficient iris recognition using local feature descriptor," in *IEEE International Conference on Image Processing (ICIP)*, Egypt, November 2009, pp. 1957–1960.
[7] B. G.S. and P. Gupta, "Palmprint verification using SIFT features," in *First Workshops on Image Processing Theory, Tools and Applications*, nov. 2008, pp. 1–8.