Performance Enhancement of Hausdorff Metric Based Corner Detection method using GPU

Ajith Padyana*, Praveen K.†, P. K. Baruah, Rachakonda. R. Sarma
Sri Satya Sai Institute of Higher Learning
Prasanthi Nilayam - 515134 Andhra Pradhesh, India
ajith.padyana@gmail.com, pravinkenator@gmail.com,
baruahpk@gmail.com,raghu.dmacs.psn@sssu.edu.in

Abstract

Corner detector algorithms are important for achieving real time performance in different computer vision applications. In this paper, we are proposing new algorithm implementations for corner detection that make use of Graphics Processing Units (GPU) provided by NVIDIA. The programmable capabilities of modern GPUs allows speeding up compared to CPU algorithms. In the case of corner detection algorithms, sizeable part of the algorithm can be easily translated from CPU to GPU. However, there are challenges for mapping the required data step to the GPU parallel computational model. This proposed implementation presents a template for implementing corner detection algorithm that runs entirely on GPU, resulting in significant speed-ups due to highly computationally intensive nature of the algorithm.

1 Introduction

With the rapid evolution of graphics card technology, GPU-based solutions have out performed corresponding sequential CPU implementations. High parallel computational capabilities, combined with an increasingly flexible programmable architecture, has turned GPUs into a valuable resource for general purpose computations [1]. In the context of this work mathematical tools such as hausdorff metric calculation, which is frequently used in image processing applications can take full advantage of the modern graphics processing power due to its computationally intensive nature.

Many applications require registration between two or more images in order to extract information from them. Two images can be registered by matching only certain points extracted from them associated with high entropy. These points are referred to as interest points and are detected using an interest point detector. A corner is one such interesting point in the image which has high variance around it. Finding the transformation between the images is then performed using these interest points.

Many different interest point detectors have been proposed with wide range definitions [6]. Some detectors find points of high local symmetry, others find areas of highly varying texture, while others locate corner points. Corner points are interesting as they are formed from two or more edges and edges usually define the boundary between two different objects or parts of the same object.

Finding the correspondence between images is used in many computer vision applications like 3D reconstruction, camera calibration, image registration, object recognition, panoramic photograph stitching, motion tracking, robot navi-

*Student Author
†Student Author

gation etc.

In this paper we are proposing to implement computationally intensive algorithm called Hausdorff Metric based Corner Detector(HMCD)[4] in the Graphical Processing Unit from NVIDIA for improving the performance we got in the sequential program.

2 Related Work

Given an image, the goal of a corner-detector algorithm is to identify the sets of pixels that faultlessly represent the corners in the image. Corner-detector algorithms are usually implemented in two separate steps: corner strength computation and non-maximum suppression. SUSAN[3], Moravec's[5], Trajkovic and Hedle's[7], Shi and Tomasi's[2] and Harris[8] are examples of this class of algorithm. In the corner strength computation step, each pixel is assigned the value of a corner response function, called cornerness value. Generally, this step ends up assigning high values for too many pixels around the vicinity of an image corner. In order to determine only one pixel to represent the position of each corner, the second step selects the pixel with the highest cornerness value in a given neighborhood. In the end, the algorithm ensures that all non-maximum values are suppressed around the considered neighborhood, and thus the pixel that more accurately represents each detected corner is retained. In this paper, we are interested in efficient corner-detector algorithms for real-time applications.

Sinha et al. [9] proposed a corner-detector algorithm accelerated by graphics hardware (GPU). However, differently from our proposal, they used the GPU only to compute the image with values of a corner response function. To suppress the non-maximum values, the image was read back to be processed on CPU. Although their approach represented a speed-up if compared to the CPU-only counterpart, the gain in performance was not enough for being used in Visual SLAM. In this paper, we propose an algorithm that will be fully implemented on GPU. As a result, our algorithm is able to detect image corners very efficiently, thus being a useful tool for real-time computer vision applications.

3 HMCD based algorithm

HMCD is one of the corner detectors developed ingeniously at the Sri Sathya Sai Institute of Higher Learning, Prashanthi Nilayam. HMCD extends the concept of hausdorff metrics to develop a corner detector. This feature extractor makes use of hausdorff distance measure for calculating the corner strength. HMCD has the capacity to pick up specific features, by considering the required feature template. The strength of the feature with respect to a template is computed using Hausdorff distance. The advantage of HMFE is that, depending on the application on hand, one can choose to have a suitable feature template. Another advantage of HMCD is its minimum sensitivity towards noise. Hausdorff distance metric (HDM) finds its place in many applications such as point pattern registration, face recognition, object tracking etc.

The HMCD corner detector was tested under all possible conditions and was benchmarked with the SUSAN and Harris corner detector. The performance of HMCD demonstrated its strength, especially in the case of blur and noise. In terms of addition of salt and pepper noise at extreme levels, the performance of HMCD is excellent with respect to False Positive Rate (FPR). In the presence of Gaussian blur when the window size increases, the HMCD still outperforms all other detectors with respect to FPR. The number of TPR in all the cases is very high and outperforms all other detectors. HMCD outperformed SUSAN in all the cases.

3.1 Requirements of corner detectors

The corner detectors must meet the following requirements:

1. All True Corners should be detected.

2. No False Corners should be detected.

3. Corner points should be well localized.

4. Detector should have a high repeatability rate (good stability).

5. Detector should be robust with respect to noise.

6. Detector should be computationally efficient.

3.2 Hausdorff Distance Calculation

Given two finite point sets $A = (a_1, a_2..., a_p)$and $B = (b_1, b_2..., b_q)$, the Hausdorff distance between the two sets A and B is defined as:

$$H(A, B) = Max(h(A, B), h(B, A))$$

Where,

$$h(A, B) = max(min||a - b||)$$

And ||.|| is some underlying norm on the points of sets A and B (e.g., the L2 or Euclidean norm).The function h(A, B) is called the one-sided Hausdorff distance or the directed Hausdorff distance from A to B. It identifies the point 'a A that is farthest from any point of B and measures the distance from 'a to its nearest neighbor in B (using the given norm ||.||). Intuitively, if h(A, B) = d, then each point of A must be within distance d of some point of B, and there also is some point of A that is exactly at distance d from the nearest point of B(the most mismatched point). The Hausdorff distance H(A, B) is the maximum of h(A, B) and h(B, A). In the subsequ So, if the Hausdorff distance is H(A, B) = d, then every point of A must be within a distance d of some point of B and vice versa. Thus, the notion of resemblance encoded by this distance is that each member of A be near some member of B and vice versa. Unlike most methods of comparing shapes, there is no explicit pairing of points of A with points of B.

3.3 The HMCD Techinique

This is a template based feature extraction technique [4], wherein a template is selected by the user. The template proposed is within an N x N window. Thus each template window has two regions, a region of low (high) intensity (R1) and a region of high(low) intensity (R2) as shown in Figure 1. C is the vertex of the template. Pixels whose N x N neighbourhood in the image exhibits the designated pattern are extracted. That is to say, with reference to the template considered, the N N neighborhood of any pixel of the image, exhibits sufficient contrast between the regions R1 and R2.

3.4 Implementation of HMCD Algorithm in GPU

In order to extract feature points that resemble the vertex C of the chosen template we perform the following: For each pixel of the given image, we create one GPU thread. For each thread in the GPU, do the following steps:

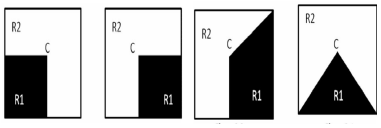

Figure 1: Feature Templates

1. Choose an N X N template.

2. Position the template centered at each pixel P of the image. The co-ordinates of P are say (Px, Py).

3. Consider the intensity values of image pixels that fall inside the region R1 of the template as Set A.

4. Consider the intensity values of image pixels that fall inside the region R2 of the template as Set B.

5. Find Hausdorff distance PH between Set A and Set B.

6. Store for each pixel P of the image, the coordinates (Px , Py) and the Hausdorff distance PH in the Feature List array.

7. Designate a pixel as a corner pixel whose PH value is above a given threshold T.

To avoid cluttering of feature points, one can impose Minimum distance criterion, which says that, once a pixel is selected as a feature point, no other pixel in its minimum neighborhood with a lesser strength would be selected as a feature point, but if it has a higher strength then the criterion would be applied to that pixel. Thresholding can be avoided by asking for some best number of points. The other forms of HDM measures, such as MinMax, MaxMax, MeanMin (Hausdorff Average), MaxMedian, MaxMean, MeanMean and so on could also be considered.

4 Results

This section shows the timing analysis for the bird image, susan blocks and cameraman image on a typical Desktop system of the sequential program for some test images which is given below and also for GPU implementations. The following figures shows the timing analysis of the different algorithms on the natural test images. Figure 5 shows that improvement in the timing of the HMCD algorithm after parallelzing the code for 4 templates. Hence, we can conclude that the computationally intensive HMCD algoritghm can be improved its performance by parallelzing the algorithm. From this we can achieve both performance.

Figure 2: Test Images

It is clearly seen from the graphs shown that the SUSAN corner detection algorithm takes the

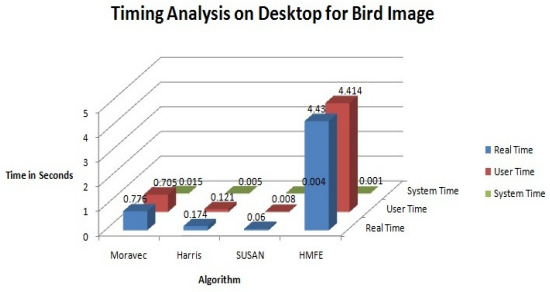

Figure 3: Timing analysis of Bird Image in a Desktop System

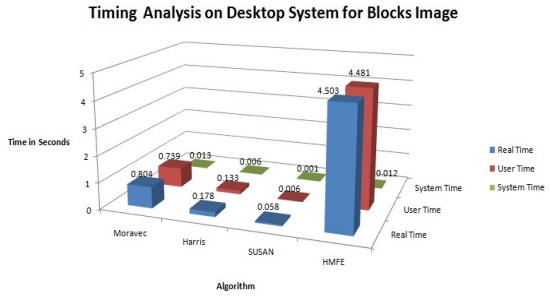

Figure 4: Timing analysis of Blocks Image in a Desktop System

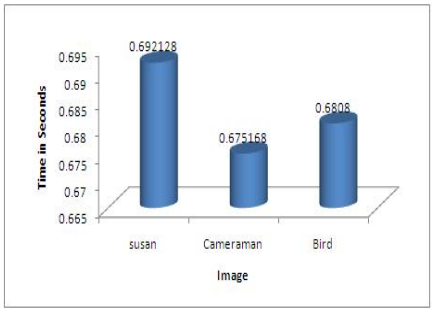

Figure 5: Timing analysis for different images like Bird, Cameraman, susan blocks for Parallel GPU system

minimalistic execution time both in the Desktop system. The HMCD corner detector takes the maximum execution time in the Desktop machine. This is quite understandable as the HMCD Corner Detection algorithm is highly computation intensive. Hence we parallelized the HMCD algorithm so that we can get bet-

ter peformance with respect to time. For each of the pixel in the image, we have calculated haussdorff distance for 4 templates. This drastically reduced the execution time considerably as shown in the figure. Another method by which the execution time can be reduced is by linearizing the hausdorff distance calculation.

5 Conclusions

The HMCD Corner Detection algorithm is highly computation intensive. For each of the pixel in the image, we have calculated the haussdorff distance in parallel for 4 templates. From the results figure we can see that the timing for the computation part has improved drastically. Hence parallelizing the algorithm is the better option for getting some good overall performance.

In future, we will extend this work to all the other possible templates and also will explore the same techinque to the image with higher resloutions and also for color images. The performance of HMCD demonstrated its strength, especially in the case of blur and noise in sequential algorithm. In terms of addition of salt and pepper noise at extreme levels, the performance of HMCD is excellent with respect to False Positive Rate (FPR). In the presence of Gaussian blur when the window size increases, the HMCD still outperforms all other detectors with respect to FPR. Since HMCD is very highly computation intensive, using accelarators will improve the computation performance resulting in overall performance with respect to other algorithms.

6 Acknowledgements

We thank NCSA for allowing us to access there GPU enabled parallel machine. Most of all, we express our gratitude to Sri Sathya Sai Baba, The Chancellor of Sri Sathya Sai Institute of higher learning, Prasanthi Nilayam, for bringing all of us together to perform this work .

References

[1] O. Fialka and M. Cadik. FFT and convolution performance in image filtering on gpu. Tenth Int. Conf. on Information Visualization, pages 609614, July 2006.

[2] J. Shi and C. Tomasi. Good features to track. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR94), Seattle, June 1994.

[3] S. Smith and J. Brady. SUSAN A new approach to low level image processing. Technical Report TR95SMS1c, Chertsey, Surrey, UK, 1995.

[4] R. Raghunath Sarma, V. Sudhir, K. S. Sreedharan, Prof. G.V.P. Rao, Sri Sathya Sai Insitute of Higher Learning, Hausdorff Metric based Feature Extractor, TENCON 2008 - 2008 IEEE Region 10 Conference, Hyderabad, 19-21, 2008.

[5] H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. In tech. report CMU-RI-TR-80-03, Robotics Institute, Carnegie Mellon University. 1980.

[6] Ambar Dutta, Avjit Kar and B N Chatterji, Corner Detection Algorithms for Digital Images in Last Three Decades ,IETE Techincal review, volume 25 Number 3, May-June 2008.

[7] M. Trajkovic and M. Hedley. Fast corner detection. Image and Vision Computing, 16, 1998.

[8] C. Harris and M. Stephens. A combined corner and edge detector. In Proceedings of The Fourth Alvey Vision Conference, pages 147152, 1988.

[9] S. Sinha, J. Frahm, M. Pollefeys, and Y. Genc. Feature tracking and matching in video using programmable graphics hardware. Machine Vision and Applications, 2007.