GPU Implementations of Object Detection using HOG Features and Deformable Models

Manato Hirabayashi, Shinpei Kato, Masato Edahiro
Dept. of Information Engineering
Nagoya University

Kazuya Takeda
Dept. of Media Science
Nagoya University

Seiichi Mita
Research Center for Smart Vehicles
Toyota Technological Institute

Visione-based object detection using camera sensors is an essential piece of perception for autonomous vehicles. Various combinations of features and models can be applied to increase the quality and speed of object detection. A well-known approach uses histograms of oriented gradients (HOG) with deformable models to detect a car in an image. A major challenge of this approach can be found in computational cost introducing a real-time constraint problem in the real world. In this paper, we present an implementation technique using graphics processing units (GPUs) to accelerate computations of scoring similarity of the input image and the pre-defined models. Our implementation considers not only the algorithm part but also the entire program structure for practical use. We also apply the presented technique for the real-world car detection program and demonstrate that our implementation using commodity GPUs can achieve speedups of 1.5x to 3x in frame-rate over sequential and multithreaded implementations using traditional CPUs.

***Index Terms*—GPGPU; Computer Vision; Object Detection**

I. Introduction

Grand challenges of cyber-physical systems (CPS) include a high computational cost of understanding the physical world. Object detection is one of compute-intensive tasks for CPS. For example, an autonomous vehicle needs to detect and track other vehicles by itself. Current autonomous driving technologies [6], [12], [20] tend to rely on active sensors such as GPS, RADAR, and LIDAR [10], [18] together with very accurate pre-configured maps, but the use of passive camera sensors is becoming more practical due to recent advances in computer vision [2]–[4]: vision-based object detection can be applied for various ranges and orientations. In particular, histograms of oriented gradients (HOG) [2] features provide reliable high-level representations of an image underlying many state-of-the-art object detection algorithms [3], [5], [17], [19], [21]. However, a major concern of HOG-based object detection remains in computational cost.

Previous work on the implementation of HOG-based object detection are limited to either hardware implementations [7], [8], [11] or specific parts of HOG algorithms [1], [16]. There is even no quantitative investigation of what implementation issues could prevent HOG-based object detection from being deployed in real-world applications. Given recent innovations in commodity hardware technology such as multicores and graphics processing units (GPUs), it is worth exploring if the current state of the arts meets computational requirements of cutting-edge object detection implementations.

Contribution: This paper presents GPU implementations of HOG-based object detection in consideration of real-world applications using deformable part models [3]. While this is a popular vision-based object detection approach, what remains an open question is a genelized programming technique and a quantification of performance characteristics for practical use. We begin with an analysis of traditional CPU implementations to find fundamental performance bottlenecks of HOG-based object detection. This analysis reasons about our approach to GPU implementations where we parallelize compute-intensive blocks of the object detection program using the GPU step by step to minimize its makespan. The experimental results obstained from a real-world car detection program using a commodity GPU show that the GPU outperforms the CPU by 1.5x to 3x in frame-rate, while another 2x improvement would be needed at least to deploy in the real world.

Organization: The rest of this paper is organized as follows. Section II describes the assumption behind this paper. Section III presents an analysis of HOG-based object detection and our GPU implementation technique. Section IV evaluates the performance benefit of our technique over traditional CPU implementations. This paper concludes in Section V.

II. Assumption

We consider the system composed of a multicore CPU and commodity GPU. They communicate with each other via the PCIe bus. We use CUDA [14] for GPU programming, whose development environment can be downloaded from NVIDIA's website [15]. Input images are loaded from pre-captured JPEG files, since we focus on a high computational cost of image processing. Systemized coordinations of computations and I/O devices are outside the scope of this paper. The use of multiple GPUs is also not in consideration.

We follow the object detection method presented by Felzenszwalb *et. al.* [3], where objects are represented by HOG features [2] and the detectors is composed of a "root" filter plus a set of "parts" filters that allow visual appearance to be modeled at multiple scales. This is one of the most recognized approach to object detection. See [3] for the detail.

Object detection often requires a machine learning phase to construct the object models. We assume that this learning phase has already been done a priori and the object models are stored in the system. Particularly we restrict our attention to vehicle detection in this paper, utilizing the vehicle models provided by prior work [13]. Although these models achieve a high detection rate, the computational cost of scoring similarity of an imput image and the models using HOG features is very expensive. Specifically they include 2 root filters and 12 part filters, each of which needs to be scored against 32 resized images. The scoring could be conducted for every 8×8 or 4×4 pixels independently. In consequence, there are approximately 100 billion computational blocks for a single high-definition image, while the frame-rate needs to meet 10 to 20 frames per second (FPS) for practical use. This data-parallel compute-intensive nature of HOG-based object detection motivates the use of GPUs in this paper.

The CPU implementation of HOG-based vehicle detection has already been developed in prior work [13]. It leverages the POSIX *pthread* to parallelize the scoring per filter on a multicore CPU. While we use this multicore implementation for a performance comparison as it is, we also serialize it to execute on a single core so that we can compare our GPU implementation to two variants of the CPU implementation.

III. GPU Implementation

This paper presents GPU implementations of the existing object detection program using a pupular computer vision technique [13]. Our contribution is distinguished from prior GPU implementations work [1], [16] in that we analyze the performance characteristics of the object detection program to figure out what part of the program should be accelerated using the GPU and our implementations build on this analysis optimizing performance.

Note that this paper focuses on vehicle detection but the presented GPU implementations and our technical contribution can be applied for other object detection methods using HOG features and deformable models.

A. Analysis

B. Approach

C. Programming

IV. Evaluation

V. Conclusion

In this paper, we have presented GPU implementations of HOG-based object detection and their performance evaluation. Unlike preceding work that highly stressed on performance improvements, our implementations are based on an analysis of performance bottlenecks posed due to an introduction of the deformable models in HOG-based object detection. This approach ensures that the GPU truly accelerates approapriate computational blocks. Our evaluation using a commodity GPU showed that our GPU implementation can speed up the existing HOG-based vehicle detection program tailored to the deformable models by 1.5x to 3x over traditional CPU implementations. Given that this performance improvement is obtained from the entire program runtime rather than particular algorithm parts of the program, our contribution is useful and significant for real-world applications of vision-based object detection.

In future work, we plan to complement this work with systemized coordinations of computations and I/O devices. Since real-world applications require camera sensors to obtain input images while GPUs are compute devices off the host computer, the data I/O latency could become a bottleneck upon data buses. In this scenario, we need enhanced system support such as zero-copy approaches [9] to minimize the data latency raised between camera sensors and GPUs. We also plan to augment our GPU implementations using multiple GPUs in order to meet the real-time and real-fast requirement of real-world CPS applications.

References

[1] Y-P. Chen, S-Z. Li, and X-M. Lin. Fast HOG Feature Computation based on CUDA. In *Proc. of the IEEE International Conference on Computer Science and Automation Engineering*, pages 748–751, 2011.

[2] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Proc. of the IEEE Conference on Compute Vision and Pattern Recognition*, pages 886–893, 2005.

[3] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[4] P. Felzenszwalb and D. Huttenlocher. Pictorial Structures for Object Recognition. *International Journal of Computer Vision*, 61(1), 2005.

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conference on Compute Vision and Pattern Recognition*, pages 3354–3361, 2012.

[6] E. Guizzo11. How Google's Self-Driving Car Works. IEEE Spectrum, 2011.

[7] R. Kadota, H. Sugano, M. Hiromoto, H. Ochi, R. Miyamoto, and Y. Nakamura. Hardware Architecture for HOG Feature Extraction. In *Proc. of the International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 1330–1333, 2009.

[8] F. Karakaya, H. Altun, and V. Cavuslu. Implementation of HOG algorithm for real time object recognition applications on FPGA based embedded system. In *Proc. of the IEEE Signal Processing and Communications Applications Conference*, pages 508–511, 2009.

[9] S. Kato, J. Aumiller, and S. Brandt. Zero-Copy I/O Processing for Low-Latency GPU Computing. In *Proc. of the IEEE/ACM International Conference on Cyber-Physical Systems*, 2013 (to appear).

[10] A. Kirchner and C. Ameling. Integrated Obstacle and Road Tracking using a Laser Scanner. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 675–681, 2000.

[11] M. Komorkiewicz, M. Kluczewski, and M. Gorgon. Floating Point HOG Implementation for Real-Time Multiple Object Detection. In *Proc. of the International Conference on Field Programmable Logic and Applications*, pages 711–714, 2012.

[12] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J.Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun. Towards Fully Autonomous Driving: Systems and Algorithms. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 163–168, 2011.

[13] H. Niknejad, T. Kawano, M. Simizu, and S. Mita. Vehicle detection using discriminatively trained part templates with variable size. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 766–771, 2000.

[14] NVIDIA. CUDA Documents. http://docs.nvidia.com/cuda/, 2013.

[15] NVIDIA. CUDA TOOLKIT 5.0. http://developer.nvidia.com/cuda/cuda-downloads, 2013.

[16] V. Prisacariu and I. Reid. fastHOG – a Real-Time GPU Implementation of HOG. Technical Report 2310/09, Department of Engineering Science, University of Oxford, 2009.

[17] P. Rybski, D. Huber, D. Morris, and R. Hoffman. Visual Classification of Coarse Vehicle Orientation using Histogram of Oriented Gradients Features. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 921–928, 2010.

[18] D. Streller, K. Furstenberg, and K. Dietmayer. Vehicle and Object Models for Robust Tracking in Traffic Scenes using Laser Range Images. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 118–123, 2002.

[19] F. Suard, A. Rakotomamonjy, A. Bensrhair, and A. Broggi. Pedestrian Detection using Infrared Images and Histograms of Oriented Gradients. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 206–212, 2006.

[20] C. Urmson, J. Anhalt, H. Bae, D. Bagnell, C. Baker, R. Bittner, T. Brown, M. Clark, M. Darms, D. Demitrish, J. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, M. Likhachev, B. Litkouhi, A. Kelly, M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y-W. Seo, S. Singh, J. Snider, J. Struble, A. Stentz, M. Taylor, W. Whittaker, Z. Wolkowicki, W. Zhang, and J. Ziglar. Autonomous Driving in Urban Environments: Boss and the Urban Challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[21] Q. Zhu, M-C. Yeh, K-T. Cheng, and S. Avidan. Histograms of Oriented Gradients for Human Detection. In *Proc. of the IEEE Conference on Compute Vision and Pattern Recognition*, pages 1491–1498, 2006.