Proceedings of the
2007 IEEE Intelligent Vehicles Symposium
Istanbul, Turkey, June 13-15, 2007

A Parallelism Analysis Pattern for Real-time Vehicle Recognition Algorithm

Chunyang Yang, Huai Yuan, Nan Wang, Chunmin Zhang and Bobo Duan

Abstract—**With the rise of many-core, parallelization is expected to be the mainstream to speedup the computation. This paper presents the study of classic vision based algorithms applied to vehicle recognition in DAS domain from the point of view of building a practicable real-time system. The workloads of the algorithm is illustrated and analyzed. After summarized the difficulties prevent current algorithm from achieving real-time, especially considering recent many-core tide, this paper brings out a parallel analysis pattern for vehicle recognition, our prime work indicates it can effectively help us to most exploit and express the parallelism of the algorithm.**

I. Introduction

VEHICLE recognition is one of the most important applications for Drive Assistant System (DAS) . It's the foundation for DAS to make correct control decisions so that it can react to dangerous situation in time. The real-time requirement of vehicle recognition algorithms is definitively a critical index. Modern vehicle recognition algorithm involves the up to date research fruits of computer vision, AI, sensor fusion and image processing areas, and tends to employ more complicated statistic/machine learning algorithms, introduces more kinds of and more numbers(channels) of sensors and achieve more accurate and robustness detection rate in a fusion way. Considering the complexity of the working environment of vehicle recognition, this is a natural and reasonable way, whereas, it brings a "sensors × channels × scenarios × targets" computational workloads [1] and aggravates the difficulties in achieving real-time.

With the risen of many/multi-core tides, parallelism has been the mainstream for increasing the computing power, and the traditional free lunch for software to speed up is out of date [2]. So, for vehicle recognition algorithm which tends to exhibit large amounts of computation and parallelism inherent, the research of parallelization of algorithm, or in another words, how to most exploiting its parallelism and best mapping it to the coming many-core hardware is the key to the issue of the ability of being real-time.

Industry has brought out some specialized chips aiming to provide support real-time solution of these complex algorithms, Ex, Kerneltron [3] for SVM algorithm, IMAP [4] for image processing, and some other Streaming process oriented chips etc. Most of these chips provide high computing capability through adapt advanced IC techniques and SIMD architecture, which are very suitable for speed up the dense regular computing operations. However, the algorithms that vehicle recognition involves is much more complicated, not only does it include the algorithms suitable for speed up with the means of SIMD, but also algorithms which are composed of lots of irregular computing or dynamic scheduling, and some of the algorithms are computing sensitive whereas some other algorithms are memory sensitive. This makes the above-mentioned pure SIMD chips less effective and can not best utilize the computing capability of the hardware. We start with an analysis of a vision based hybrid vehicle recognition algorithm that we developed, conclude the importance and the difficulties of parallel analyzing for vehicle recognition algorithm, and then we propose a pattern for analysis, discover and represent the parallelism of the algorithm, and our applying it to the parallelism analysis for vehicle recognition shows that it is an effective methodology.

This paper is organized as follows. Section II describes the target vision based hybrid algorithm of vehicle recognition and the classical sub algorithms it employs. A prime workload analysis is present in this section too, while Section III gives an analysis from the parallelism analysis point of view, and summarized the difficulties for speed up the algorithms as well as the aim and the motivation of the analysis pattern. In Section IV, after introduce the pattern; we put importance on the description of our applying it to parallelization analysis of vehicle recognition algorithm. Section V concludes it and point out our future work.

II. Workload Analysis for Classical Vision Based Vehicle Recognition Algorithm

A. Target Classical Vehicle Recognition Algorithm

Fig.1 [5] below shows a sketch map of our current vehicle recognition algorithm. Like other classical algorithms, it integrates the detection and tracking, the algorithm will find new object to be tracked and start/stop object tracking according to the result of detection. The detector adopts a

Manuscript received Jan.15th.2007. This work was supported by Intel Funding Project entitled "Research Image Processing Programs Written in Data Parallel Language Primitives on Multi-Core Platform."

Chunyang Yang is with Software Center, Northeastern University, Shenyang, 110004, China (yangcy@neusoft.com).

Huai Yuan is with Shenyang Neusoft Co., Ltd. AAC. Shenyang 110179, China. (yuanh@neusoft.com).

Nan Wang is with Shenyang Neusoft Co., Ltd. AAC. Shenyang 110179, China. (wangnan@neusoft.com)

Chunmin Zhang is with Shenyang Neusoft Co., Ltd. AAC Shenyang 110179, China. (zhang.cm@neusoft.com)

Bobo Duan is with Software Center, Northeastern University, Shenyang 110004, China. (duanbb@neusoft.com)

hybrid algorithm, integrates the knowledge based method and machine learning method (Wavelet for feature extraction and SVM for classification, 3788 feature sets), the tracer adopts motion estimation and light-weight algorithm in good match and pyramid template matching when occlusion and mismatch. Tracking parts enhanced the robustness of the algorithm and shorten the process time. A prototype has been development on an Intel C2.8G PC, it has achieved 95% detection rate and less than 5% false detection rate for a test set composed of 37 video segments of various road and traffic

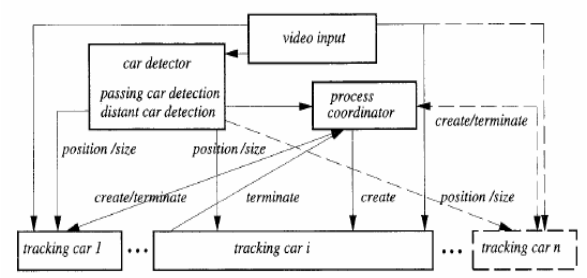

Fig. 1 Framework of classic vision based vehicle recognition algorithm. This is a classical program structure of the algorithm, and our target prototype also adopt this kind of structure as a representation for analysis

situation, taken in Japan and China, from a camera mounting on a moving vehicle. The size of the input video is 720 × 480 pixels, and the average process time is 60ms/f, whereas the peak process time has yielded 200ms/f, we consider this prototype as a typically implementation of most modern vision based vehicle recognition algorithm and use it as target for analysis.

B. A Prime Workload Analysis

Understanding the characteristics of workload is very important in building real-time system [6], workload analysis and characterization will indicate under what circumstance or which part of the algorithm tends to bring out performance problems from point of view of result analyzing, and what's more, for certain algorithm it will also indicate the ineffective cases such as most part of its rules are ineffective whereas only a few rules works, this will lead to optimization and improvement from architecture to algorithm aspects.

1) Necessary of Algorithm Parallelization

Our primary analysis about the workload firstly indicates the necessary of parallelization of the algorithm. We can find that the peak process times are most likely to happen at:

a) Complicated scenarios: more vehicles in the picture, confused background or vehicle appearance such as occlusion.

b) Close vehicle: large object in the picture and the motion is relative obvious at this time.

These scenarios are just those driver will have trouble in making quick and accurate judgment and need the supporting of DAS, so we can't use the common engineering methods such as discard frames, output delay etc to smooth down the peak time, whereas, we must rely on parallel to ensure the real time without sacrificing the accuracy and the sensitivity.

2) Bottlenecks and character of the Algorithm

Our major aim of the workload analysis is to find out the bottlenecks of the algorithm at peak time and their computation models, especially memory access models, so as to guide the future tuning and parallelization. Whereas, because of the hybrid algorithm it adopts, the workload of the vehicle recognition algorithm is somewhat input depended and hard to describe. The various input road situation

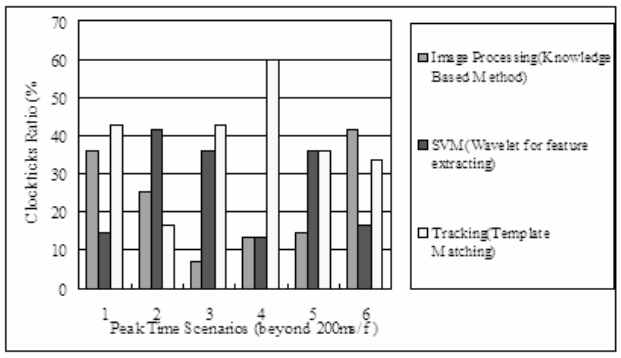

Fig. 2 Peak time Scenarios (>200ms/f among the 37 sample video segments) and the related Workload. The snapshot of each scenario is listed in appendix

(background, vehicle appearance, occlusion relationship, light-condition, motion-relationship and history information etc) will cause the hybrid algorithm to predicate the most possible strategy and select best fit sub algorithms (features) combination dynamically. So, the workload of the peak time may be a various combination of groups of sub algorithms.

Although an exhaustively enumeration of the peak time workloads is difficult, but our prime analysis indicates, in most cases, as described in figure 2, the peak time are composed of the following workloads, and they're also the most probable parts tend to be the bottlenecks in most modern vehicle recognition algorithms.

a) Image processing for Knowledge based Method: This group of knowledge based sub algorithms involves serials of classical image process algorithms such as difference and projection as basics, and it depends on the extracted geometry or statistic features to classify the vehicle object. The major factor of the time consuming of this part can be approximately consider to be the spending on memory access of image data, and the most frequently used memory access pattern are GO (Global Operation), GeO(Geometry Operation), SO(Statistical Operation) and LNO(local neighbor operation) [7].

Unlike the global image process such as Sobel (for edge extraction) to the whole image which can be easily take the advantage of data parallel with SIMD, whereas, knowledge based method are more likely a looping for paired statistic and judgment operation to see if certain feature exists. It's more sensitive to memory access and because the image process and statistic mostly restrict to certain ROI, pure SIMD optimization seems not so effective. In case of many vehicles in surroundings, especially when its appearance features such as corner, layer, symmetry etc which are not

ideal, this part tends to be a bottleneck of the algorithm.

b)SVM: Machine learning plays a key role in vehicle recognition algorithm, and SVM(supported vector machine) is one of the widely used algorithm, its major way is to compute a functional relationship between input vector (the extracted features of a ROI) and the model(pre-trained feature set). Our current SVM pre-trained model contains 3788 features, and for each ROI, the computation complex is 1024×3788. In case of there are a lot of vehicles, there is obvious time consuming increase and become a bottleneck of the algorithm.

$$Decision\_value = (\sum_{k=1}^{k=3788} Z_k \times Coef_k) - rho \qquad (1)$$

$$Z_k = e^{(-gamma)\times(\sum_{m=1}^{1024}(X-Y_m)^2)}$$

Most other adopted machine learning algorithms have a similar form like SVM and are mainly computation sensitive, like showed by equation (1), the final classification result (viz. *Decision_value*) involves a serials of multiplication/addition operations which often suitable for data parallel.

c) Tracking: Tracking module is another bottleneck of the algorithm especially when traced object is at image boundary or in the shadow or occluded which often result in mismatch and unstable tracking with the light weight tracking algorithm and in these cases, template matching is expected to be the valid method.

In theory, the computing of template tracking is to find a location with minimize motion energy of the specified template in the searching window, for example, like in equation (2), the location with the minimize normalized correlation coefficient(NC) is consider to be has the minimized motion energy and T is the specified template while F is the searching area.

$$NC(i,j) = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N} T(m,n)\times F(m+i,n+j)}{\sqrt{\sum_{m=1}^{M}\sum_{n=1}^{N} T^2(m,n)\times F^2(m+i,n+j)}} \qquad (2)$$

Instead of computing each possible location offset, in actual implement, the minimized motion energy is often be computing in some form of optimization, such as predicated diamond search or other motion estimation method. If we don't take the motion estimation into consideration, this part is obvious computing sensitive, while the most motion estimation methods are some kind of data depended or even recursive algorithm which adds extra complication to the computation model.

III. DIFFICULTIES AND MOTIVATION

A. Speedup and Parallelization Difficulties

In Section II above, the major bottleneck of the vehicle recognition algorithm has been recognized and its computation model has been characterized to be a more complicated one comparing with ordinary image processing domain, and the following difficulties will prevent traditional mainstream data driven parallel analysis method from to be applicable.

1) The Complexity of the Algorithm

Because the running environment of the vehicle recognition algorithm is very complicated, there's no single algorithm (orients to certain features) which fits all the situations, an ideal way is to combine hybrid algorithms and select best fit algorithms for current scenarios. Current vehicle algorithm consider the quantitative output of segment, classification and tracking parts as the evidence of B-N network to predicate the most suitable scenario/strategy and feed it back for a more proper algorithm adapt. It's a dynamic and complex procedure and make the mainstream function level parallel analysis method become more difficult.

2) Data Couple

The data coupling makes it difficult to divide the computing into in-depended tasks and thus can increase the parallelism.

3) Various Memory Access Model

For parallel algorithm, effective memory access plays a determined role in achieving real-time performance. But as the workload analysis showed, current vehicle recognition algorithms involves various of memory access model, e.g., some are computing sensitive and some others are memory sensitive. The inherent feature of streaming application leads to easily cache out of date and the irregular computing operations aggravates the ineffective memory access.

4) Various and Uncertain Hardware

Traditional parallel algorithm often has a close relationship with hardware; programmer often must to answer some hardware related questions from the beginning of the parallel algorithm development, such as the grain of the parallel, how much the processors available and how many divisions to be made and correspondingly communication and synchronous low-level questions. So, the uncertainty of the hardware also makes it difficult for the traditional parallel analysis.

B. Aim and Motivation

Considering these factors, we hope we can bring out a systemically parallel analysis pattern which can:

1) Most exploiting the parallelism of the algorithm

The only restriction is the producer/consumer relationship explicit or in-explicit defined by the algorithm itself, unlike with traditional parallel analysis in some forms of parallel languages which often involves unnecessary restrict of low-level implement details (such as insert barriers when using OpenMP [8]) it should have nothing to do with the low-level parallel model and the parallel languages.

2) Provide more and rich opportunities for parallelization

Unlike pure SIMD and pure task parallel model (it can't handle current vehicle recognition algorithm's complexity and hybrid computation model), it must be able to effectively handle both task parallel and data parallel target to Many/Multi-core architecture at the same time and in a uniform manner.

3) Support both fine and coarse grains parallel modeling and Hardware in-depended

It must be also easy to model the parallel algorithm in different grains, and can map to future specified hardware conveniently.

IV. PARALLELIZATION ANALYSIS

Traditional parallel analysis often starts with data dependency analyzing. Data and the producer/consumer relationship correspondingly are analyzed function by function [9]. Functions without producer/consumer restriction will be organized into independent tasks. And if necessary, more parallelism can be exploited through data duplicate and with the support of inter-task data communicate, the correct execution sequence is ensured by synchronize. Whereas, as we have pointed out above, for the target vehicle recognition algorithm, the various input surroundings, hybrid algorithm it adopted and the complicated execution path makes the traditional parallel analysis become difficult, and because traditional parallel analysis method often has a close relationship with certain concrete hardware and execution model, some potential parallelism of algorithm may be lost. So, this paper proposed a top-down parallel analysis pattern to facilitate the parallelization analysis of vehicle recognition algorithm.

A. Modeling and Expressing Parallelism

Basically, there're three levels of parallel: task parallel, data parallel and instruction level parallel. We mainly concerns the first two levels of parallel now, and for a top-down parallel analysis, the first work is to model and express the parallelism of the algorithm. Here, we choose TStreams [10] for modeling and high level parallelism analyzing. TStreams is brought out by HP labs, it is a new general parallel programming model, simple yet powerful. It is optimistic (assumed able to be executed in parallel unless it is explicitly constrained, and the only constrains are the producer/consumer relations in algorithm itself), can express all kinds of parallelism equally well and it don't make any assumptions about the hardware and other low-level details. [11]

The kernel of TStreams is composed of Items, Steps and Tags. Items encapsulate the data, Steps encapsulate the computation (conceptually atomically) and Tags encapsulate unique identifier for Step and Items. Tags play an important role, it describes what/when computations are executed and what data object exits, it is also the key clues in future mapping and scheduling.

1) Task Parallelism

Fig. 3 describes the top level modeling of vehicle recognition algorithm using TStreams. The initial tag space is "frame tag" at the top; "π" here denotes there's a 1-1 mapping from "frame tag" tag space to step "Segment frame", step "Segment frame" will dynamically generate ROI tags and correspondingly ROI data (items), once a certain ROI tag is generated and added into the ROI tag space, it indicates the correspondingly recognition step to the ROI is "available"

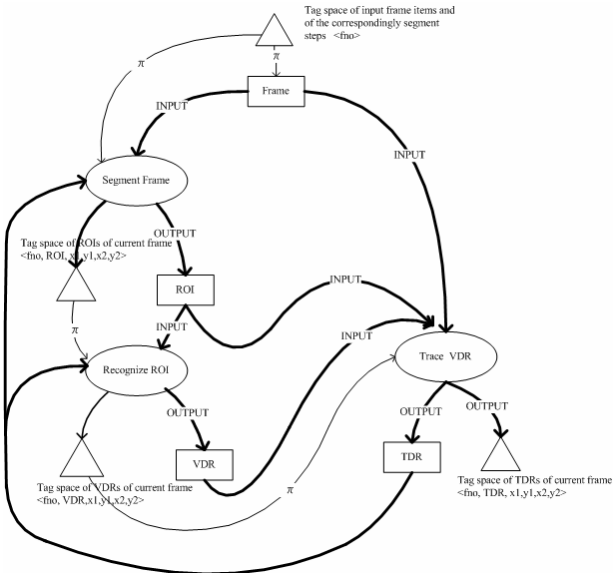

Fig. 3 The parallel model of top level vehicle recognition algorithm, TStreams Form expression of the algorithms described in Fig. 1, VDR is the abbr. of vehicle detection result, and TDR is the abbr. of tracer detection result.

and can be executed, and hence construct a producer/consumer relationship between this two steps. In similar way, when a VDR (vehicle detection result) tag is generated, the represent track step is available and can be executed.

With the model in Fig. 3, the top-level task parallel has been fully described, and the only constrains are the above

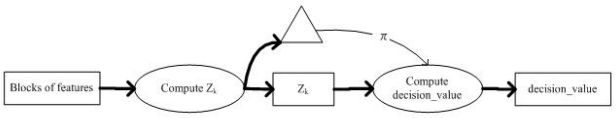

Fig. 4 The parallel model of the SVM predication algorithm. TStreams Form expression of equator (1)

mentioned producer/consumer relationship. There're no assumption about the executing order between segment for different frames and recognize/trace for certain objects. All the latent parallelism including pipeline parallelism has been exploited and expressed.

Now, at top-level, the Steps are considered as an atomic computation, whereas, of course they're much complicated actually, so, the same modeling procedure can be applied to each Step recursively until we're satisfied with a certain grains.

2) Data Parallelism

Fig.4 shows the parallel model we got after applying the same method to SVM algorithm. Feature data set is divided into blocks and computed in parallel.

With method above, we can model both the task and data parallel in a uniform way and express all the potential parallelism of the algorithm. The dynamic generation of tag and items and the abstract execution model are suitable for

modeling and analyzing the complicated dynamic scheduling and running behaviors of the vehicle recognition algorithm. So, the top-down analysis with the help of TStreams can effectively help to most exploit the parallelism of the algorithm and leads to a demandable grain of division which providing rich opportunities for parallelization.

B. Expressing and Abstraction the Data Parallelism

Although the above analysis can be proceed recursively till we got all the detailed parallel divisions and repressions in theory, but actually, the above analysis often can be stopped when we get an "enough" clear parallel division. For example, while analyzing the data parallel of SVM algorithm, the model in Fig.4 has described the data parallel clear "enough" to be understand. So, there's no need to future divide for the step "computing Z_k" and "computing decision_value" here we can consider it as an atomic processing for convenient. And the internal data parallelisms as well as the data parallel we have modeled are often concrete algorithm related, we introduces an abstract and well-designed data parallel primitives to represent the parallelism.

The introduction and the design of the data parallel primitives utilize the other work which is mainly focus on the data parallelism on GPU [12]. The primitives are composed of a serial of operation including element-wise, reduce, prefix, permute, gather, scatter, vector-scalar, and multi-prefix, multi-reduce etc. These primitives are functional semantics, and sheltered the complexity and the lower-level parallel implementation details. Because of the complexity of vehicle recognition algorithm, there're not only regular data parallel operations but also a lot of irregulars. The primitives adopts nested vector, operation with masks and custom element-wise operations to improve and facilitate the expression of irregular data parallelism. With the support of data parallel primitives, we can express both regular and irregular data parallel effectively in a clear and hardware independent expression.

C. Mapping and Execution Model

Basically, our applying the analysis pattern above to vehicle recognition has mainly answered the question: How the data and the computation task should be divided? When and how the data and computation task can be processed? For parallel analysis, the only major question left unanswered is where of the data and the computation task should be? The effective mapping can be various depending on the concrete hardware, such as different number of processors, parallel architectures or other actual low-level conditions with comprehensive consideration of computation character and balance. After the concrete mapping, from the point of view processor, each of them will has its own view of the parallel model. And each processor will loop to check/update his tag space and pick the available steps to execute in parallel.

V. CONCLUSION

In this paper, we first briefly present a prototype of vehicle recognition algorithms which employs some classical algorithms and adopt hybrid algorithm architecture. This prototype is considered to be a representation of modern vision based vehicle recognition algorithms. And a prime workload analyzing to it shows that vehicle recognition algorithm is a complicated algorithm involves different type of computation characteristics, and the dynamic algorithm adapting and scheduling required by the hybrid algorithm aggravates the complexities and difficulties for traditional parallel analysis.

We proposed a new top-down parallelism analysis pattern for the parallel analysis of the vehicle recognition algorithm. It adopts TStreams for modeling and expressing the parallelism (both task and data parallel) of the algorithm and introduces a set of data parallel primitives support to express the rich irregular data parallel in vehicle recognition algorithm. TStreams based high-level modeling provides an effective way to describes and exploit the parallelism of the algorithms and at the same time it doesn't constrain to concert low-level details, and benefits from the functional semantics and the data decoupling it supported, we can exploit and express the parallelism of the algorithm utmost.

We're now going to parallelization the algorithm and implement a prototype on board target at many-core architecture platforms. And two most possible options are many-core SIMD as co-processors, and chip symmetric multi-processors. Although the architecture and the hardware details of the two optional platforms are different in many ways, but with the help of the parallelism analysis pattern we proposed, we can focus on the algorithm itself all through, modeling and analyzing the parallelism of it in a more essential way and the analysis result is ready for easily mapping to the future platforms. Our prime work of applying this pattern indicates that it is an effective parallelism analysis pattern for vision based hybrid vehicle recognition algorithm.

APPENDIX

Fig. 5 below lists the snapshots of scenarios which bring out peak times in Fig.2.

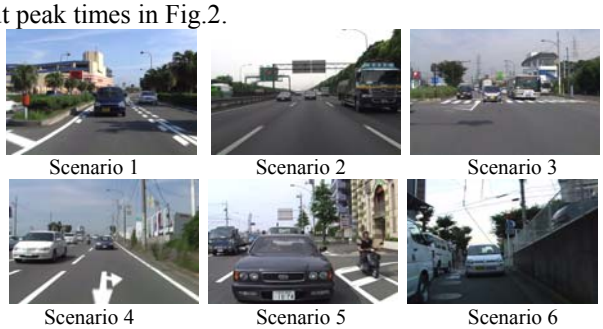

Fig. 5 Snapshots of peak time scenarios in Fig. 2, the shadow in scenario 1 and 6,the unobvious vehicle feature in scenario 2 and 6, the noise of whit-line in scenarios 1,3,4,5 and big and occlusion vehicle in scenario 5 are the major factors which caused the algorithm to bring out peak process time.

REFERENCES

[1] Shorin Kyo, Shin'ichiro Okazaki, and Tamio Arai, "An Integrated Memory Array Processor Architecture for Embedded Image Recognition Systems", *ACM SIGARCH Computer Architecture News*, Volume 33,Issue 2, pp.134-145,May 2005.

[2] Sutter, Herb, “A Fundamental Turn Toward Concurrency in Software”. *Dr. Dobb's Journal*. Vol. 30, no. 3, pp. 16-20, 22. Mar. 2005.

[3] Roman Genov, and Gert Cauwenberghs, “Kerneltron: Support Vector Machine in Silicon*,” Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machine.* pp. 120-134, 2002.

[4] Shorin Kyo, Takuya Koga and Shin'ichiro Okazaki,"IMAP-CE: a 51.2 GOPS video rate image processor with 128 VLIW processing elements." *ICIP (3),*pp.294-297,2001.

[5] Margrit Betke, Esin Haritaoglu and Larry S.Davis, “Multiple Vehicle Detection and Tracking in Hard Real-Time,” *Intelligent Vehicle Symposium, Proceedings of the 1996 IEEE*, pp.351-356, 1996.

[6] Lizy Kurian John, Purnima Vasudevan and Jyotsna Sabarinathan, “Workload characterization: motivation, goals and methodology”, *Workload Characterization: Methodology and Case Studies*, pp.3-14, 1998.

[7] P. P. Jonker, “Architectures for Multidimensional Low-and Intermidiate Level Image Processing,” *Proc. Of IAPR Workshop on Machine Vision Application (MVA'90)*, pp.307-316, 1990.

[8] “OpenMP Application Program Interface”, 2005, Specification Available at http://www.openmp.org/drupal/mp-document/spec25.pdf.

[9] Sean Rul,Hans Vandierendonck and Koen De Bosschere, “Function Level Parallelism Driven by Data Dependencies”, Workshop on Design and Simulation of Chip Multi-Processors, 2006

[10] Kathleen Knobe,Carl D.Offner, “TStreams: How to Write a Parallel Program”, *Technical report HPL-2004-193,2004*. Available at http://www.hpl.hp.com/techreports/2004/HPL-2004-193.pdf.

[11] Kathleen Knobe and Carl D. Offner. “TStreams: A Model of Parallel Computation (Preliminary Report).” *Technical report, HP Labs Technical Report HPL-2004-78*, 2004. Available at http://www.hpl.hp.com/techreports/2004/HPL-2004-78.html.

[12] David Tarditi, Sidd Puri, and Jose Oglesby, “Accelerator: simplified programming of graphics processing units for general-purpose use via data parallelism”, *Technical Report MSR-TR-2005-184*, 2005. http://research.microsoft.com/research/pubs/view.aspx?type=technical%20report&id=1040