Implementation of a Highly Parameterized Digital PIV System On Reconfigurable Hardware

Abderrahmane Bennis, Miriam Leeser, Gilead Tadmor
Abennis@ece.neu.edu, mel, tadmor@coe.neu.edu
Northeastern University
Boston MA 02115

Russ Tedrake
russt@mit.edu
Massachusetts Institute of Technology
Cambridge MA 02139

Particle image velocimetry (PIV) is used in experimental fluid dynamics to obtain a detailed localized view of velocity vectors in an unsteady fluid flow. In a nutshell, the estimated velocity field is computed from local correlations of snapshot-pairs of the particle-seeded flow, obtained by high-speed cameras [1-2]. PIV is used in a wide range of fields including aerodynamics, automotive design and biomedical engineering. Despite many improvements to PIV methods over the last twenty years, PIV post-processing remains a computationally intensive task. It becomes a serious bottleneck as snapshot acquisition rates exceed 10,000 frames per second. In this research, we aim to substantially speed up PIV post-processing by implementing it in reconfigurable hardware. Furthermore, this implementation is highly parameterized, supporting adaptation to varying setups and application domains.

Our implementation is parameterized by the dimensions of the captured images as well as the size of interrogation windows and sub-areas. It is also parameterized by image quantization level (bits/pixel), the size of on-board memory and the overlap between interrogation windows. To the best of the authors' knowledge, this is the first highly parameterized PIV system implemented on reconfigurable hardware reported in the literature. For a typical PIV configuration with images of 1024×1024 pixels, 40×40 pixel interrogation windows and 32×32 pixel sub-areas, we achieved over a 100-fold speedup in hardware over a standard software implementation. Our implementation supports real-time processing for many of these setups.

Parameterized PIV

Figure 1 provides an overview of a standard 2D PIV experimental setup. Images of a planar cross section are taken, as it is illuminated by laser pulses. Velocity estimates are obtained by calculating the local displacement of particles seeded in the fluid, in successive image pairs.

While substantial stride has been made in PIV technology [2], the acceleration of the computations has received relatively little attention from the high performance computing community. The most successful projects are reported in [3, 4, 5]. The PIV algorithm is based on matching by cross-correlation which is computationally intensive but offers a high degree of parallelism. From this point of view, FPGAs promise an affordable, high performance solution. In our research, we went one step further by investigating the parameterization of such solutions.

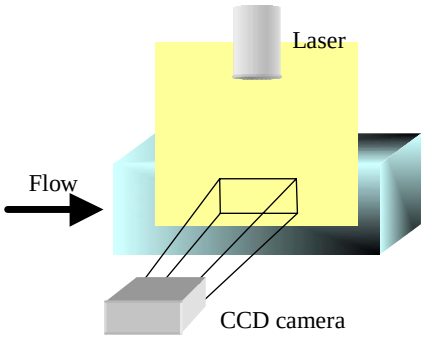

Figure1. PIV experiment setup

After examining the cross-correlation algorithm and its implementation on an FPGA board, we extracted the parameters described in Table 1. Specifying these parameters determines the behavior of every component in the PIV system.

Table 1: Parameters of PIV Circuit

Img_width	The width in pixels of the images
Img_depth	The depth in pixels of the images
Area_width	The width in pixels of the interrogation window
Area_depth	The depth in pixels of the interrogation window
Sub_area_width	The width in pixels of the sub-areas
Sub_area_depth	The depth in pixels of the sub-areas
Displacement	Number of pixels by which a sub area is moved inside an interrogation window
Pixel_bits	Number of bits that represent a pixel
RAM_width	Number of bits in each memory address
Overlap	Number of interrogation windows that overlap in an interrogation window size

We use VHDL to describe our design. VHDL has constructs that enable parameterized designs. It allows a circuit to have a parameterized number of bits and parameterized timing by using generic declarations.

PIV is a complex system to parameterize. Even though VHDL supports parameterization of small circuits, the parameterization of large circuits is more challenging and the literature reports only a limited number of parameterized circuits on FPGAs. The design of reconfigurable hardware circuits requires much more time than the design of equivalent computing in software; parameterization of such circuits adds another level of

complexity and consequently adds more time to the design. The designer must think about designing every element of the circuit in the general case so it can be parameterized. While this task is straightforward for certain circuits such as gates and adders, it is more difficult for other digital elements such as pipelined multipliers, dividers and finite state machines. Furthermore, the overall view of the design should be presented in the most general way.

Parameterization is very beneficial for both designers and consumers. Reconfigurable hardware designers use parameterized component from libraries instead of creating their own; this helps to decrease time to market. For consumers, using parameterized circuits allows the flexibility of changing the circuit parameters and then having new circuits with new parameters in a very short time. Such flexibility is highly demanded especially in research areas where intensive computing is required and appropriate computational parameters are still being investigated or simply where application parameters differ from one domain to another.

Results

The parameterized PIV system architecture is developed in VHDL and implemented on the Firebird reconfigurable computing board from Annapolis Micro Systems. The board has a Virtex-E XCV2000E. The board is programmed by a C++ interface that loads the images from the host computer to the on-board memories and load the results when they are ready from on-board memory to the host computer. We plan to move to a newer board, ADM-XRC-5LX, from Alpha-data that includes a Virtex 5 FPGA. Note the parameterization should make the transition easy.

Table 2: Parameters of different circuits

Parameters	Circuit1	Circuit2	Circuit3
Img_width	1024	1200	400
Img_depth	1024	1600	50
Area_width	40	40	50
Area_depth	40	40	50
Sub_area_width	32	32	20
Sub_area_depth	32	32	20
Displacement	1	1	1
Pixel_bits	8	8	8
RAM_width	32	32	32
Overlap	2	2	2

Table 2 provides examples of circuits implemented by our system. Note that the parameters of Circuit1 will result in 81 cross-correlations per interrogation window. Each correlation performs 1024 multiplications. Circuit2 performs the same number of cross-correlations per interrogation area as Circuit1 because they have the same interrogation window size and the same sub-area size. However, it processes larger images, resulting in more processing. The third circuit performs 961 cross-correlation per interrogation window where each correlation executes 400 multiplications.

Table 3 provides speedup results. We achieved over a 100-fold speedup in hardware over a standard software implementation in C++ on a 3 GHz Intel XEON microprocessor. The latency of the hardware is the average obtained by running the circuit one thousand times.

Table 3: Speedup of different circuits

Circuits	Hardware latency (sec)	Software latency (sec)	Speedup
Circuit1	0.025	3.21	128
Circuit2	0.027	3.76	139
Circuit3	0.00473	0.109	23

The performance results provided above are obtained using only one correlator component. Performance can be increased by duplicating the correlator unit and slightly modifying the data control unit. The first circuit can process 80 pairs of images per second by adding another correlator.

As a result of this project, a library of parameterized components has been built. The library includes basic logic elements as well as block RAMs, accumulators, multipliers, and fixed-point dividers. All the computational cores are pipelined and optimized. This library will help in rapid design of future projects and can be extended to include more components.

Specific implementations of PIV should allow for real-time processing and should enable new applications of PIV, including closing the loop on feedback control.

Acknowledgements

This research was supported in part by National Science Foundation Grant CCR-0208791 and CCR-0410246.

References

[1] M. Raffel, C. Willert and J. Kompenhans. *Particle Image Velocimetry*. Springer-Verlag, 1998.

[2] R.J. Adrian, "Twenty years of Particle Image Velocimetry." *12th International Symposium on Applications of Laser Techniques to Fluid Mechanics*, July 2004.

[3] E.B Arik, and J Carr, "Digital Particle Image Velocimetry system for real-time wind tunnel measurements." *International Congress on Instrumentation in Aerospace Simulation Facilities (ICIASF)*, pp. 267-277, September 1997.

[4] Toshihito Fujiwara, Kenji Fujimoto, and Tsutomu Maruyama, "A real-time visualization system for PIV." *Field Programmable Logic and its Applications (FPL)*, pp. 437-447, September 2003.

[5] Haiqian Yu, Miriam Leeser, Gilead Tadmor and Stefan Siegel, "Real-Time Particle Image Velocimetry for Feedback Loops Using FPGA Implementation". *Journal of Aerospace Computing, Information, and Communication*. Vol. 3 Issue 2, pp. 52–62, Feb 2006.