Anywhere Pixel Compositor

Ruigang Yang[*]
University of Kentucky

Anselmo Lastra[†]
University of North Carolina at Chapel Hill

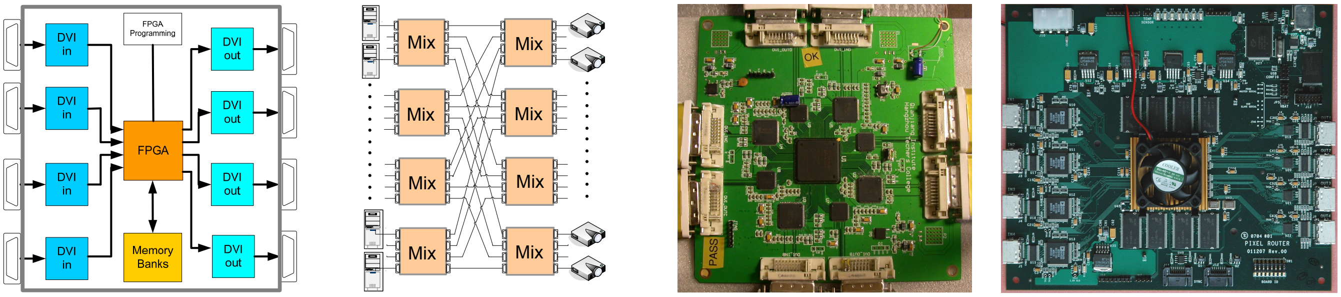

Figure 1: *From left to right (a) A high-level schematic of the compositor design with four inputs and four outputs. (b) A sample configuration to connect 16 inputs to 16 outputs. Any pixel from any input can be routed to any location in any output. (c) an early version of our compositor (d) the current version of our compositor.*

1 Introduction

Even with the recent rapid advancement in hardware, the demand from high-end graphics applications (including video games) seems to always outpace the capability that a single GPU can offer. Graphics hardware vendors are now offering dual or even quad GPU configurations (such as NVIDIA's SLI and ATI's CrossFire technology). As we migrate from a single GPU to multiple GPUs or eventually GPU clusters, how to effectively assemble the final image from these distributed rendering nodes becomes an important issue. Here we propose to develop a flexible pixel compositor to solve this problem. Our compositor is capable of performing an arbitrary mapping of pixels from any input frame to any output frame, and executing typical composition operations at the same time. Figure 1(a) shows a schematic of our design. The pixels are transmitted digitally. The core mapping and arithmetic operations are carried out by a programmable FPGA chip. It is connected to a large memory bank that stores both the mapping information and temporary frames (if necessary). A single compositor unit has four input links and four outputs. Multiple units can be arranged in a network, shown in Figure 1(b), to achieve the scalability for large clusters.

As reviewed in [Cavin et al. 2005], dedicated video compositing hardware currently exists. Examples include the Lightning-2 system [Stoll et al. 2001] and the commercially available HDVG system (http://www.orad.co.il) that also utilize digital transmissions. However, they are limited to conventional compositing tasks in which each video stream is restricted to a rectilinear region in the final output, i.e., the routing is limited to block transfer of pixels. None of the current systems can provide the ultimate flexibility envisioned here. This level of flexibility is motivated by recent advances in display technologies that demand significantly more pixels and more complex composition operations. The first is the development and commercialization of auto-stereoscopic (multi-view) displays, in particular lenticular-based displays. These 3D displays in fact display many views simultaneously and therefore require orders of magnitude more pixels to provide an observer adequate resolution. This can be achieved only by a rendering cluster. Furthermore, images from the rendering nodes typically need to be sliced and interleaved to form the proper composite image for display. The second driving application is multi-projector displays driven by a cluster. To completely remove the photometric seams or geometric discontinuities on projector boundaries, the frame-buffer content for each projector has to be warped and attenuated properly. While there are (special) projectors that can perform piecewise linear warp to the input content, they cannot be directly used in a multi-projector display that has overlaps since one projector would need part of the images from its neighborhood in the overlap region. Our compositor performs two functions in one: pixel distribution and warping. We also envision that our flexible hardware can be used for distributed GPGPU applications. It provides the random write capability missing in most current graphics hardware.

2 Work in Progress

We have designed and built several hardware prototypes shown in Figure 1 (c)-(d). The current version includes a Xilinx VIRTEX-4 FPGA core, 256MB of DDR RAM arranged on four independent buses, and 4 HDMI inputs and 4 HDMI outputs. The HDMI interface is backward compatible with the DVI interface. The FPGA core is running at 200MHz. We adopted inverse mapping for image transformation to avoid any holes in the output. The input image needs to be buffered. The most difficult part of developing the firmware is to maintain the refresh rate. To achieve the target operation at 1024×768@60Hz (most projectors operate at this mode), we need to sustain a minimum bandwidth of 1.7 GB/s, which includes at least a read, a write, and a table-look-up operation at each pixel tick. Unlike traditional composition tasks that have excellent data locality, our look-up-table-based mapping can be arbitrary. We have calculated that our current prototype can sustain 30Hz update in the worst case, i.e., cache miss all the time. In the best case (i.e., cache hit all the time), it can operate at over 60Hz.

In summary, we are developing the next generation compositor to satisfy the demanding needs from emerging applications. By providing a scalable and flexible link among a cluster of GPUs, they can efficiently work in concert to solve problems, both graphical and non-graphical, on a much larger scale.

References

CAVIN, X., MION, C., AND FILBOIS, A. 2005. Cots cluster-based sort-last rendering: Performance evaluation and pipelined implementation. In *Proceedings of IEEE Visualization*, 15–23.

STOLL, G., ELDRIDGE, M., PATTERSON, D., WEBB, A., BERMAN, S., LEVY, R., CAYWOOD, C., TAVEIRA, M., HUNT, S., AND HANRAHAN, P. 2001. Lightning-2: a high-performance display subsystem for pc clusters. In *Proceedings of SIGGRAPH 2001*, 141–148.

[*] e-mail: ryang@cs.uky.edu, this work is supported in part by US National Science Foundation grant IIS-0448185.

[†] e-mail: lastra@cs.unc.edu