Flexible Pixel Compositor for Plug-and-Play Multi-Projector Displays

Ruigang Yang  
Dept. of Computer Science  
University of Kentucky  
ryang@cs.uky.edu

Daniel R. Rudolf  
Sapphire Computers, Inc.  
http://www.sapphirecomp.com  
drudolf@sapphirecomp.com

Vijai Raghunathan  
ECE Department  
University of Kentucky  
vragh0@engr.uky.edu

1. Motivations

Multi-projector, visually immersive displays have emerged as an important tool for a number of applications including scientific visualization, augmented reality, advanced teleconferencing, training, and simulation. Techniques have been developed that use one or more cameras to observe a given display setup in a more casual alignment, where projectors are only coarsely aligned. Using camera-based feedback from the observed setup, the necessary adjustments needed to register the imagery, both in terms of geometry and color, can be automatically computed.

In order to create seamless imagery in such a multi-projector display the desired image (first-pass) has to be warped (second-pass) to “undo” the distortions caused by the projection process. So far this has always been done in software. The second rendering pass, while not significant, does cause some overhead. In addition, applications designed for regular displays usually have to be modified in the source code level to take advantage of the large display capability. In contrast, abutted displays (mechanically aligned projectors with no overlap), while take days or weeks to setup, can largely run any application without modification, thanks to the widely available multi-channel output even in consumer-grade PCs. Based on our prior research, this has been one of the most significant disadvantages of displays with overlaps.

2. Flexible Pixel Compositor

We have already started to address the software compatibility issue in multi-projector displays. Our solution is a flexible pixel compositor that bridges the image generator (e.g., a rendering cluster) and the projectors. Our compositor is capable of performing an arbitrary mapping of pixels from any input frame to any output frame, and executing typical composition operations (e.g., blending) at the same time. The left of Figure 8 shows a schematic of our design. The pixels are transmitted digitally via the industry standard HDMI/DVI link. 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 to achieve the scalability for large clusters. Figure 8 (right) shows a sample configuration to composite 16 DVI streams.

As reviewed in [1], dedicated video compositing hardware currently exists. Examples include the Lightning-2 system [2] 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. While there are (special) projectors that can perform piece-wise 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.

3. Preliminary Results

We have designed and built a prototype shown in Figure 2. The current version includes a Xilinx VIRTEX-4 FPGA...
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 $1024 \times 768@60$Hz (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. 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.

4. Conclusion

In summary, we are developing the next generation compositor to satisfy the demanding needs from emerging applications. It can be used beyond multi-projector displays. The first is 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. We also envision that our flexible hardware can be used for distributed general-purpose computing on graphics processor units (GPGPU). It provides the random write capability missing in most current graphics hardware. 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.

Acknowledgement  We would like to thank Anselmo Las- tra for inspiring this project. Subhasri Krishnan has worked on an early version of the hardware who is co-supervised by J. Robert Heath. This work is supported in part by University of Kentucky Research Foundation, U.S. Department of Homeland Security, U.S. National Science Foundation grant IIS-0448185, and generous dotations from NVIDIA and Xilinx.

References
