Toward Accurate and Efficient Order-Independent Transparency

Ethan Kerzner
Department of Computer Science
University of Iowa

Chris Wyman
Department of Computer Science
University of Iowa

Lee A. Butler
U.S. Army Research Laboratory

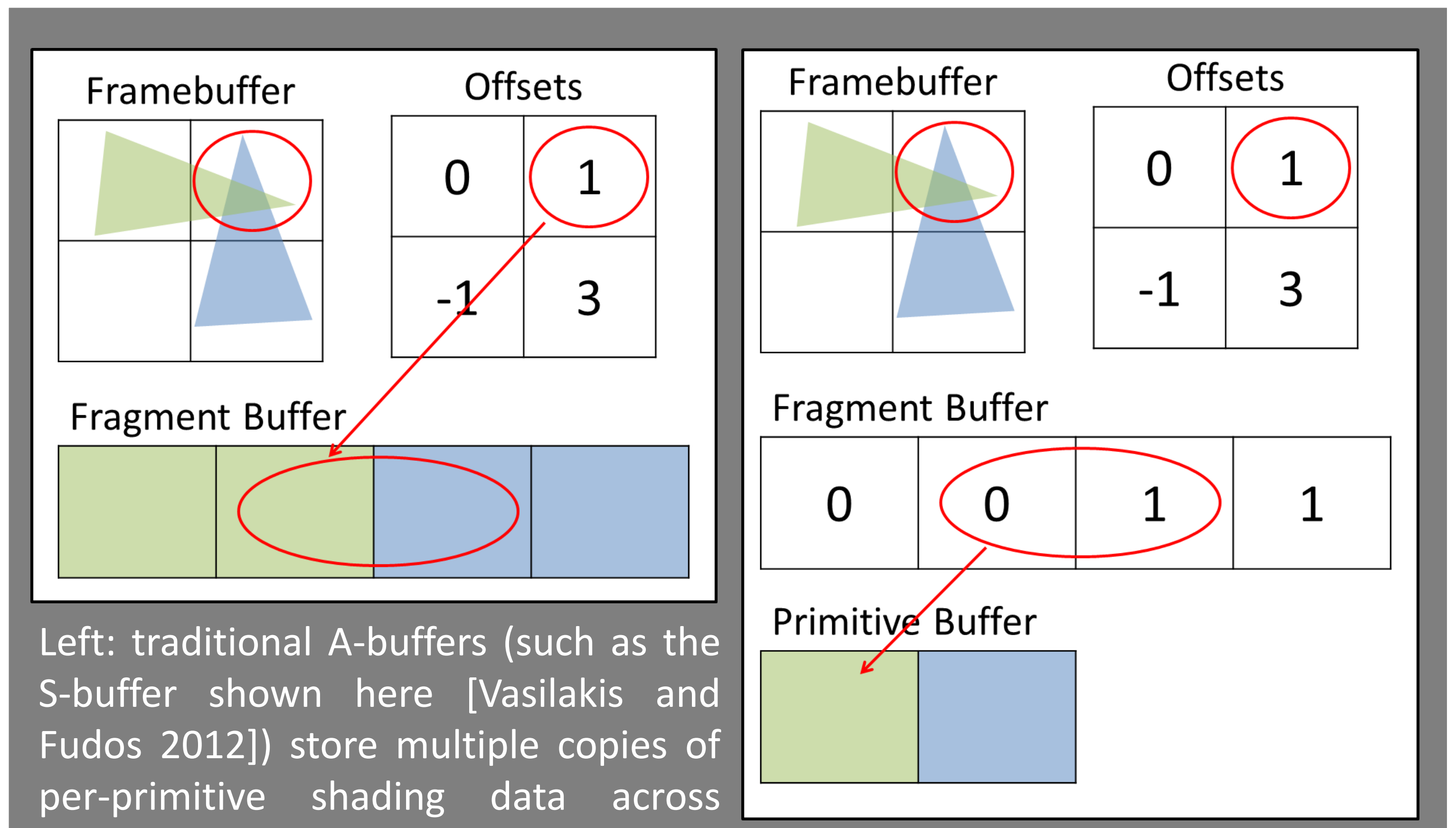
Christiaan Gribble
Applied Technology Operation
SURVICE Engineering

Overview

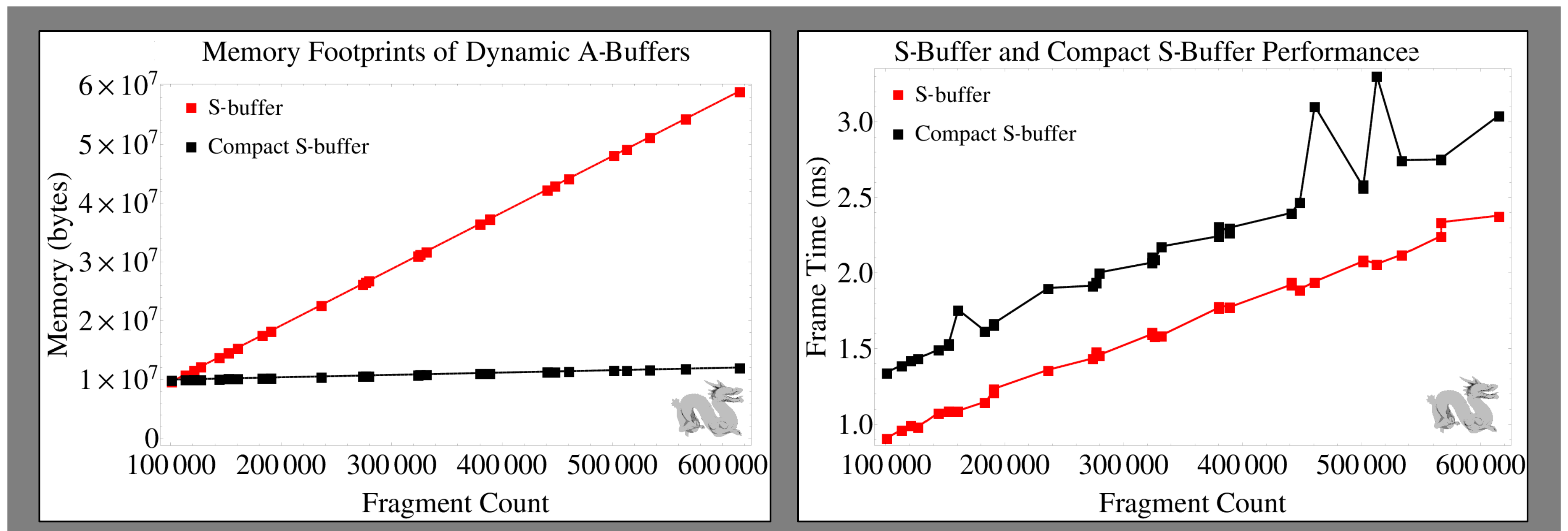
Correctly rendering multi-layered transparent geometry requires accumulating contributions from multiple fragments per pixel. Dynamic A-buffers (e.g., Yang et al's [2010] per-pixel linked lists) achieve this by storing and sorting fragments on-the-fly. We introduce an improvement to recent GPU-based interactive A-buffer techniques: we decouple visibility and shading to reduce memory demands of multi-fragment rendering.

The Compact A-buffer

Existing interactive A-buffers store shading and visibility inside fragment lists, saving per primitive shading data repeatedly for multiple pixels. Decoupling storage of primitive and fragment data in our new compact A-buffer significantly reduces memory overhead. This approach resembles the decoupling proposed by Liktor and Dachsbacher's [2012] compact G-buffer.



Left: traditional A-buffers (such as the S-buffer shown here [Vasilakis and Fudos 2012]) store multiple copies of per-primitive shading data across fragments. Right: the compact A-buffer reduces the storage of redundant primitive data.



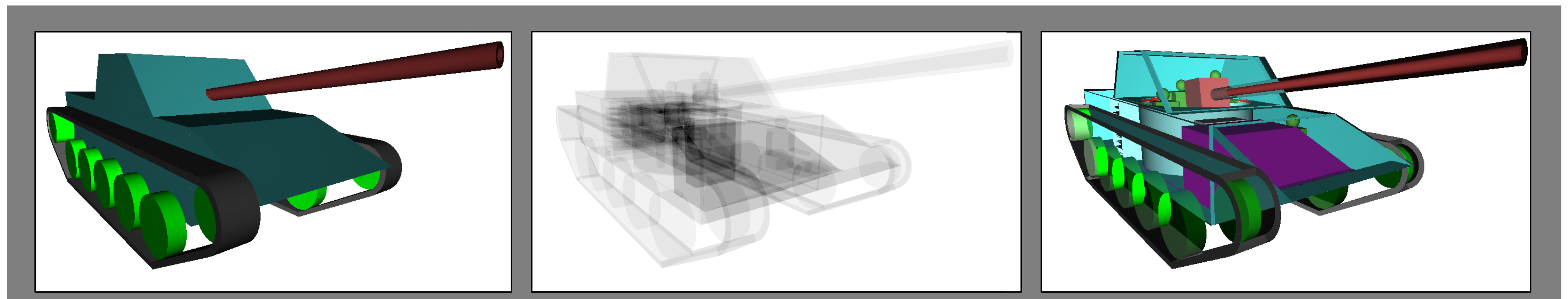
Memory usage (left) and frame time (right) of regular and compact A-buffers, computed using OpenGL 4.3 on an Nvidia GeForce 690 at 1024^2.

Performance and Memory Usage

When primitive count exceeds fragment count, our compact A-buffer has a larger memory footprint. However, our compact A-buffer scales more efficiently as fragment count increases. This scaling comes at the cost of an additional layer of indirection while accessing shading data, increasing shader execution time.

Non-Optical Rendering

Accurate and efficient OIT has applications to non-optical rendering such as ballistic simulations. Particularly, optical transparency computes the light absorbed as photons pass through the environment, whereas ballistic simulation computes the energy absorbed as projectiles pass through an object [Butler and Stephens 2007].



A tank model representative of those used in ballistic simulations rendered with the Compact A-Buffer. Shown here with flat shading (left), layer counting (center) and bullet-ray vision (right).

Future Work

Our future work may examine the performance and accuracy tradeoffs between exact and approximate raster-based transparency for non-optical rendering applications. We may also compare order-independent transparency algorithms with ray tracing for ballistic simulations.

References

[Butler and Stephens 07] L. Butler and A. Stephens. Bullet Ray Vision. In *IEEE Symposium on Interactive Ray Tracing*, 2007.
[Liktor and Dachsbacher 12] G. Liktor and C. Dachsbacher. Decoupled Deferred Shading for Hardware Rasterization. In *ACM Symposium on Interactive 3D Graphics and Games*, 2012.
[Vasilakis and Fudos 12] A. Vasilakis and I. Fudos. Sparsity Aware Multi-fragment Rendering. In *Eurographics, 2012.*
[Yang et al. 10] J. Yang, J. Hensley, H. Grün, and N. Thibieroz. Real-Time Concurrent Linked List Construction on the GPU. *Computer Graphics Forum*, 29:1297–1304, 2010.